Building a Question Answering System for the Introduction to Statistics Course using Supervised Learning Techniques

Dissertation presented for the degree of Master of Science
in the Department of Statistical Sciences

University of Cape Town
July 2020

Author: Waldo Leonhardt
waldoleonhardt@gmail.com

Supervisor: Dr. Şebnem Er
sebnem.er@uct.ac.za

Co-Supervisor: Associate Professor Leanne Scott
leanne.scott@uct.ac.za

Declaration

I hereby declare that this dissertation submitted in partial fulfilment of the requirement for the Master of Science degree in Data Science is my own work except where specific reference is made to the work of others. The contents of the dissertation have not been submitted in whole or in part for a prior qualification at this university or any other university.

Abstract

Question Answering (QA) is the task of automatically generating an answer to a question asked by a human in natural language. Open-domain QA is still a difficult problem to solve even after 60 years of research in this field, as trying to answer questions which cover a wide range of subjects is a complex matter. Closed-domain QA is, on the other hand, more achievable as the context for asking questions is restricted and allows for more accurate interpretation.

This dissertation explores how a QA system could be built for the Introduction to Statistics course taught online at the University of Cape Town (UCT), for the purpose of answering administrative queries. This course runs twice a year and students tend to ask similar administrative questions each time that the course is run. If a QA system can successfully answer these questions automatically, it would save lecturers the time in having to do so manually, as well as enabling students to receive the answers immediately.

For a machine to be able to interpret natural language questions, methods are needed to transform text into numbers while still preserving the meaning of the text. The field of Natural Language Processing (NLP) offers the building blocks for such methods that have been used in this study. After predicting the category of a new question using Multinomial Logistic Regression (MLR), the past question that is most similar to the new question is retrieved and its answer is used for the new question. The following five classifiers, Naive Bayes, Logistic Regression, Support Vector Machines, Stochastic Gradient Descent and Random Forests were compared to see which one provides the best results for the categorisation of a new question. The cosine similarity method was used to find the most similar past question.

The Round-Trip Translation (RTT) technique was explored as an augmentation method for text, in an attempt to increase the dataset size. Methods were compared using the initial base dataset of 744 questions, compared to the extended dataset of 6 614 questions, which was generated as a result of the RTT technique. In addition to these two datasets, features for Bag-of-Words (BoW), Term Frequency times Inverse Document Frequency (TF-IDF), Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDiA), pre-trained Global Vector (GloVe) word embeddings and custom-engineered features were also compared.

This study found that a model using an MLR classifier with TF-IDF unigram and bigram features (built on the smaller 744 questions dataset) performed the best, with a test F1-measure of 84.8%. Models using a Stochastic Gradient Descent classifier also performed very well with a variety of features, indicating that Stochastic Gradient Descent is the most versatile classifier to use. No significant

improvements were found using the extended RTT dataset of 6 614 questions, but this dataset was used by the model that ranked eighth in position. A simulator was also built to illustrate and test how a bot (an autonomous program on a network that is able to interact with users) can be used to facilitate the auto-answering of student questions. This simulator proved very useful and helped to identify the fact that questions relating to the Course Information Pack had been excluded from the data that had been initially sourced, as students had been asking such questions through other platforms.

Building a QA system using a small dataset proved to be very challenging. Restricting the domain of questions and focusing only on administrative queries was helpful. Lots of data cleaning was needed and all past answers needed to be rewritten and standardised, as the raw answers were too specific and did not generalise well.

The features that performed the best for cosine similarity and for extracting the most similar past question were LSA topics built from TF-IDF unigram features. Using LSA topics as the input for cosine similarity, instead of the raw TF-IDF features, resolved the "curse of dimensionality". Issues with cosine similarity were observed in cases where it favoured short documents, which often led to the selection of the wrong past question. As an alternative, the use of more advanced language-modelling-based similarity measures are suggested for future study. Either, pre-trained word embeddings such as GloVe could be used as a language model, or a custom language model could be trained. A generic UCT language model could be valuable and it would be preferable to build such a language model using the entire digital content of Vula across all faculties where students converse, ask questions or post comments. Building a QA system using this UCT language model is foreseen to offer better results, as terms like "Vula", "DP", "SciLab" and "jdlt1" would be endowed with more meaning.

Acknowledgement

I would like to express my gratitude to everyone who supported me both directly and indirectly while doing this master’s degree. I am grateful to the following people:

Şebnem Er, my primary supervisor, for her attention to detail, always willing to assist where she could and her friendliness. Thank you for helping me on this journey of research writing.

Leanne Scott, my secondary supervisor, for her level-headedness, shared insights of the problem statement and steering me back on track when needed. Thank you for the guidance and helping make the data available for this study.

Stephen Marquard, the Learning Technologies Co-ordinator, who extracted the research data in all its different forms before the final subset was selected. Thank you for your patience and assistance.

The tutors of the 2019 Introduction to Statistics course (STA1000F) who helped to capture additional administrative queries consisting of frequently asked questions which students verbally ask their tutors. This helped to increase the amount of data available for this study.

Anya Kohler, who has done the proofreading for this dissertation. Thank you for all the suggestions and corrections which made a big difference. I have learned a lot from you and will continue to make use of your proofreading service.

Finally, to my family and friends who were all so understanding with the limited time I could spent with them over the last two years. I’m looking forward to making up for lost time.

Contents

List of Tables

List of Figures

List of Abbreviations

The following list of abbreviations has been used in this dissertation:

Abbreviation	Description
AI	Artificial Intelligence
API	Application Programming Interface
ASQA	Academia Sinica Question Answering
AUC	Area Under the ROC Curve
BoW	Bag-of-Words
CIOS	Course Instructor Opinion Survey
CQA	Community Question Answering
CS	Computer Science
DP	Duly Performed
Georgia Tech	Georgia Institute of Technology
GloVe	Global Vectors
IDF	Inverse Document Frequency
IR	Information Retrieval
KB	Knowledge Base
KBAI	Knowledge-Based Artificial Intelligence
LDA	Linear Discriminant Analysis
LDiA	Latent Dirichlet Allocation
LR	Logistic Regression
LSA	Latent Semantic Analysis
ML	Machine Learning
MLR	Multinomial Logistic Regression
MNB	Multinomial Naive Bayes
MOOC	Massively Open Online Courses
NB	Naive Bayes
NER	Named Entity Recognition
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
PCA	Principal Component Analysis
POS	Part-of-Speech
QA	Question Answering
RDF	Resource Description Framework
RF	Random Forests
ROC	Receiver Operating Characteristics
RSS	Really Simple Syndication or RDF Site Summary
RTT	Round-Trip Translation
SGD	Stochastic Gradient Descent
SPARQL	A recursive acronym for SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
STA1000	Introduction to Statistics course taught at the University of Cape Town
SVC	Support Vector Classifier
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TA	Teaching Assistant
TF	Term Frequency

Abbreviation	Description
TF-IDF	Term Frequency times Inverse Document Frequency
UCT	University of Cape Town
VADER	Valence Aware Dictionary for sEntiment Reasoning
VTA	Virtual Teaching Assistant

Glossary

The following list of terms has been used in this dissertation:

Term	Description
2014 Wikipedia dump	A database dump of the full English Wikipedia content, generated by Wikimedia on 6 November 2014. The pre-trained GloVe word vectors used in this study were trained on a combination of the 2014 Wikipedia dump and the Gigaword-5 dataset.
BASEBALL	One of the first QA systems which was able to answer questions about the US baseball league over a period of one year. Created in 1961 by the Massachusetts Institute of Technology. BASEBALL is an example of a system using the KB paradigm related to QA systems.
Conceptual representations	The means by which information about categories is stored and organised.
ELIZA	One of the first QA systems created to demonstrate the superficiality of communication between humans and machines, based on precompiled scripts and pattern matching. The most famous script simulated a Rogerian psychotherapist, responding with non-directional questions to user inputs or simple rephrased statements of the user inputs. Created between 1964 and 1966 at the MIT Artificial Intelligence Laboratory by Joseph Weizenbaum. ELIZA is an example of a system using the KB paradigm related to QA systems.
Episodic memory	In the field of deep neural networks (also known as deep learning) for NLP an episodic memory module is used to model the sequence of words (or sentences) and to build a memory of how words (or sentences) follow on from one another and what the relationship is between them.
Gigaword-5 dataset	A comprehensive archive of newswire text data that was acquired between 1994 and 2010 by the Linguistic Data Consortium (LDC). Also known as the English Gigaword Fifth Edition dataset. The pre-trained GloVe word vectors used in this study were trained on a combination of the 2014 Wikipedia dump and the Gigaword-5 dataset.
GloVe	Short for Global Vectors, this is an unsupervised learning algorithm for obtaining vector representations for words, developed as an open-source project at Stanford University. These vectors, also known as word embeddings, map words to a meaningful space, where the distance between words is related to semantic similarity. In other words, related words will be close to each other in the linear substructures of the word vector space.
GUS	Short for Genial Understander System and created in 1977 by the Xerox Palo Alto Research Centre. GUS was first implemented for the restricted role of a travel agent conversing with a client wanting to travel to California. GUS is an example of a system using the KB paradigm related to QA systems.
IBM Watson	A QA system developed by IBM in the field of open-domain question answering, with the goal of competing on the US quiz show *Jeopardy!*. IBM Watson uses a hybrid paradigm of NLP, IR, KB, automated reasoning, and other ML technologies. Development started in 2005, by 2008 the system was good enough to start competing with *Jeopardy!* champions, and in 2011 it became the first AI to be the final winner of *Jeopardy!*.
Inter-question similarity	A measure of the similarity between two questions, taking into account all information available for each question.

Term	Description
Intra-question similarity	A measure of the coherence of the information available for a single question. For example, if a question has a title and body, then this measure will identify the level of similarity between the title and the body of the question.
Kullback-Leibler divergence	A measure of how different two probability distributions are from one another.
Language Model	A language model aims to provide context to words by using a probability distribution over word sequences. Different model types can be used, of which the simplest is a unigram model and an example of a more complex model being a word embeddings model.
Levenshtein distance	A measure of the difference between two-character sequences. The Levenshtein distance between two words is the count of the minimum number of single-character edits required in terms of an insertion, deletion or substitution, to change one word into the other. Also known as edit distance.
Lexico-syntactic patterns	Generalised linguistic structures that indicate semantic relationships between terms and help to identify conceptual relationships in natural language text.
LUNAR	One of the first QA systems which could answer questions about the geological analysis of rocks returned by the Apollo 11 moon missions. Created in 1971 by William Woods while working at Bolt Beranek and Newman (BBN). Woods later became a professor of Computer Science at Harvard University. LUNAR is an example of a system using the KB paradigm related to QA systems.
Named-entity Recognition (NER)	The task of locating and classifying named entities mentioned in unstructured text into pre-defined categories such as names of people, organisations, locations, monetary values, quantities, percentages, date and time values, etc.
NLTK	Short for Natural Language Toolkit, is a suite of libraries for NLP, developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania.
Parse tree	An ordered, rooted tree used to represent the syntactic structure of text based on context-free grammar. Also known as parsing tree, derivation tree or concrete syntax tree.
Parser	A parser is a procedural interpretation of the formal grammar of a language. The result of parsing a natural language sentence is a parse tree, showing the syntactic relation of words to each other. It assists with grammatical divisions, for instance identifying the subject, verb and object in a sentence. Also known as syntax analysis or syntactic analysis.
Part-of-Speech (POS) tagging	The process of identifying a word in a sentence based on its definition and its context as either a noun, pronoun, verb, adjective or adverb. Also known as grammatical tagging, word classes or syntactic categories.
PubMed	An internet search engine for biomedical literature and life science journals, created in 1996 by the United States National Library of Medicine (NLM) at the National Institutes of Health (NIH). PubMed is an example of a system using the IR paradigm related to QA systems.
Resource Description Framework (RDF)	A general method, published by the World Wide Web Consortium (W3C), to model and describe information implemented in web resources. It uses the syntax notation of subject-predicate-object, known as triples, to make statements about web resources. The subject indicates the resource, and the predicate indicates the relationship between the subject and the object, as well as characteristics of the resource.
RSS feeds	A data feed that allows users and applications to access updates to websites in a standardized and computer-readable format.

Term	Description
Semantic information processing	A field of research into imparting the human capacity for common-sense reasoning to machines, thereby creating intelligent machines. Based on research done by Marvin Minsky, co-founder of the MIT AI Laboratory.
SMART	Short for System for the Mechanical Analysis and Retrieval of Text, this is an information retrieval system developed at Cornell University in the 1960s. Some important concepts developed as part of the research on the SMART system were the vector space model, relevance feedback and Rocchio classification. SMART is an example of a system using the IR paradigm related to QA systems.
SPARQL	A semantic query language for databases, which is able to retrieve and manipulate data stored in a Resource Description Framework (RDF) format.
START	The first online QA system for answering questions related to places, movies, people, dictionary definitions and other topics. Created in 1993 by Boris Katz and his associates at the InfoLab Group at the MIT Computer Science and Artificial Intelligence Laboratory. START is an example of a system using the KB paradigm related to QA systems.
Stop-words	A list of words most common in a language or a text. A single, universal list of stop-words does not exist, but libraries like NLTK, SpaCy, Gensim and TextBlob do provided pre-compiled lists of stop-words that can be used.
Structured Query Language (SQL)	A language used in computer programming for managing data stored in a Relational Database Management System (RDBMS).
Topic Coherence	A measure for topic coherence for LSA or LDiA topic models, using the four stage topic coherence pipeline as proposed by Röder, Both & Hinneburg (2015). The Gensim library was used to calculate the LSA and LDiA topic coherence scores related to this study.
Universal Ontology	The formal specification of all concepts used in any subject area, where entities, ideas and events, with all their interdependent properties and relations, are represented by a system of categories. Up until recent years a universal ontology was assumed to be infeasible in practice, but current research suggests a universal ontology could be solved.
Virtual Teaching Assistant (VTA)	An intelligent system that can be used as a personalised teaching assistant for students attending an online course. The VTA can be implemented either as a QA system or as a conversation agent (chatbot).
WordNet synsets	WordNet is a corpus reader part of the NLTK library, and supports a synsets (short for synonym sets) function of retrieving a set of synonyms for a given word, which share a common meaning.

1. Introduction

The task of using a machine to automatically answer a question posed by a human in natural language is still a difficult problem to solve, even after 60 years of research in this field. The main reason for this is the fact that what is considered natural for a human is unnatural for a machine (Montgomery, 1972). The natural languages used by humans in conversing with each other are evolving languages and can be interpreted in different ways, despite underlying universal grammar rules and structures (Berwick & Chomsky, 2016). This is natural for humans but not for machines. These natural languages were never intended to be translated into a finite set of mathematical operations, as is required for programming languages (Lane, Howard & Hapke, 2019). Context is also very important in order to enable the correct interpretation of any part of human conversation. When asking a question, the asker has some context in mind, which often forms part of a broader conversation. Humans are good at interpreting, linking or deriving context, but this is much harder for a machine to do. The language used for programming a machine is exact and quite different from the natural language used by humans (where words can have multiple meanings).

This dissertation explores how a Question Answering (QA) system, one where questions and answers are given in English text, could be built, in order to be used in the specific context of answering administrative queries asked by students taking the Introduction to Statistics course (STA1000) that is taught online at the University of Cape Town (UCT) at undergraduate level. The course runs twice a year and students tend to ask the same questions each time that the course is run. The answers for some of these questions can be found in the Course Information Pack, meaning that these questions could be avoided if students read the course outline first. On the other hand, asking questions should not be discouraged in a learning environment. Therefore, if a QA system can successfully answer questions automatically, it would save lecturers the time in having to do so manually, as well as enabling students to receive the answers immediately.

The problem statement is to determine if it would be possible to use the available data to create a QA system which can automatically answer administrative queries with reasonable results. The questions posed by this study are the following: which features will preserve the meaning of questions the best? Which classification model will perform the best in predicting the question category? Is it a good idea to apply augmentation to the base dataset to increase the number of training samples? How can the best past answer be selected for a new question? Can sentiment analysis be used to help prioritise the answering of questions? Can a simulator be built to demonstrate and test the QA system?

The scope of this study is to use a knowledge base, consisting of past questions and answers, as a QA system. First, the category of a new question will be predicted, followed by retrieving the past question falling into the predicted category and that is most similar to the new question. The answer of the selected past question will then be used as the answer for the new question. The system will not attempt automated generation of answers.

This dissertation has been structured in the following way: Chapter 2 is a review of literature relating to QA systems. The structure of QA systems is explored and three system examples relating to this study are reviewed. In Chapter 3, the data used in this study is discussed, revealing the outcome of a much smaller dataset than what was initially anticipated. This led to the exploration of a data augmentation technique relevant to text data in order to help increase the data size. The methodology for Natural Language Processing (NLP) is described in Chapter 4, which starts with the basic principles of turning words into values that can be interpreted by a machine and goes on to look at the more recent advances in working with word embeddings. Due to the small dataset available, traditional supervised learning techniques are discussed, as opposed to the deep learning techniques which would have been applicable if the study had a big dataset. In Chapter 5 the results are presented where models using the original base data are compared against those using the extra augmented data. This dissertation ends with Chapter 6, where a conclusion is provided, limitations of the study are noted and suggestions for future study are discussed.

2. Literature Review

This chapter provides a literature review for QA systems, in terms of what they are, how they are built and how they are applied in the real world. This is needed to set the foundation before aiming to construct a QA system for administrative queries asked by students of the STA1000 course, as taught online at UCT at undergraduate level.

QA is the task of automatically generating an answer to a question asked by a human in natural language. Research in the field of QA has been undertaken since 1960 (Phillips, 1960; Simmons, 1964) and the demand for QA systems is increasing by the day, due to the huge amount of digital content constantly being created and the ever-increasing need for obtaining concise and relevant information with the least amount of effort (Pudaruth, Boodhoo & Goolbudun, 2016). Soares & Parreiras (2018) have carried out a systematic literature review on QA techniques, paradigms and systems, one in which 1 842 studies related to QA were considered and finally 130 studies were reviewed for the period 2000 to 2017. This review, in combination with other academic work (Allam & Haggag, 2012; Jurafsky & Martin, 2019; Pudaruth, Boodhoo & Goolbudun, 2016) forms the foundation for related work used in this study.

QA systems are complex and consist of three main stages. The first stage is question processing, where the question is received, analysed in order to understand the type of question received and classified to determine the context of the question. The second stage is Information Retrieval (IR), where a relevant set of documents, which potentially contain the answer to the question, is retrieved. The third and last stage is answer extraction, where a final answer needs to be extracted and presented to the user. These stages form the core architecture of a QA system, but in addition to this, the domain, paradigm, algorithms and metrics needed to evaluate the reliability of answers generated need to be considered. These aspects of a QA system will be described in more detail in Section 2.1 to 2.4.

There are many QA systems already in existence and three systems related to this study have been selected for review. Two of them are virtual teaching assistants (VTAs) used in tertiary education (described in Section 2.5) and the other is used in the area of community-based QA (described in Section 2.6).

2.1 Domains and Paradigms

The first consideration when constructing a QA system is the kind of domain in which questions will be asked; this could be either an open- or a closed-domain. In an open-domain, it is possible to ask questions on any subject and this requires a universal ontology and access to information such as is available on the World Wide Web (Allam & Haggag, 2012). In a closed-domain, also known as a restricted-domain, questions are restricted to a specific domain (e.g. music, medicine, etc.) and this requires building a domain-specific ontology (Mollá & Vicedo, 2007).

The next consideration is the paradigm, or the approach to consider for the answering of the questions (Yao, 2014). The four options that can be considered are NLP, IR, the use if a Knowledge Base (KB) or a hybrid of the first three.

NLP is a general paradigm, which uses linguistic intuitions and machine learning methods to extract answers, but excludes the core methods which define the IR or KB paradigms. Instead of generating an answer this paradigm reuses past answers "as is" (Shtok et al., 2012). Note that the study by Soares & Parreiras (2018) did not enforce the same exclusion as is mentioned here. The reason for enforcing this exclusion is due to the general nature of NLP and because it is often used in combination with the other paradigms. System examples in this paradigm are the Chinese Question Answering system for Reading Comprehension (CQARC) and the rule-based Question Answering system for Reading Comprehension tests (QuARC) (Hao, Chang & Liu, 2007; Riloff & Thelen, 2000).

In the case of IR, the goal is to answer a question by finding short text segments in a collection of documents or on the World Wide Web. Examples of systems which use this paradigm are SMART, PubMed, Yahoo!, Google, Bing, etc.

The KB paradigm is used in the case where a question is mapped to a query which can be executed on a structured database. The query can be in some version of predicate calculus or a query language like SQL or SPARQL. The database can be a relational database or a set of Resource Description Framework (RDF) tuples. System examples in this paradigm are BASEBALL, LUNAR, START, ELIZA, GUS, etc.

Lastly, a hybrid paradigm, which is any combination of NLP, IR and/or KB, can be used. It is not used often however, due to its complex nature. System examples in this paradigm are the Academia Sinica Question Answering (ASQA) system and IBM Watson, which became famous in 2011 during the American quiz show *Jeopardy!*, where it became the first example of Artificial Intelligence (AI) to be a final winner (Ferrucci, 2012).

After selecting the domain and paradigm, the next consideration to take in building a QA system, is the architecture necessary for question processing and the extraction of answers. This will be discussed in the following section.

2.2 Architectures

Different architectures can be defined, but based on most studies three macro modules are needed (Allam & Haggag, 2012) namely question processing, document and passage retrieval, and lastly answer extraction.

The question processing module aims to understand the natural language question provided by turning unstructured text into structured data. The goal of this module is to classify the question and derive a query which contains the keywords of the question (using named-entity recognition, stop-word lists, Part-of-Speech tags, etc.). Five additional features can be extracted to assist with question processing, namely the question type, class, focus, answer type and reformulation.

- The question type can be one of six different types, summarised as follows (Reddy & Madhavi, 2017):
    - **Factoid** are questions which can be answered in short text by expressing a fact and which generally start or contain the words “what”, “when”, “where”, “who” or “which”. For example, “*When was the university of Cape Town founded?*” has the factual answer of “*1 October 1829*”.
    - **List** are questions which result in an answer of a list of facts or entities. For example, “*What are the names of all the South African universities?*”.
    - **Confirmation** are questions which result in a yes or no answer. For example, “*Are the University of the Western Cape and the University of Cape Town in the same city?*”.
    - **Causal** are questions which require a descriptive answer which can expand from a few sentences, to whole paragraphs or even to a whole document. Questions generally start or contain the words “why” or “how”. For example, “*How did the Cape Peninsula University of Technology come in existence?*”.
    - **Hypothetical** are questions which request information about a hypothetical event or scenario. Questions generally start or contain the words “what would happen if” or “if you would/could”. The accuracy and reliability of answers for hypothetical questions is generally low. For example, “*If you were to remember one thing only, what would it be?*”.
    - **Complex** are questions which are difficult to answer, where multiple points need to be considered, or when there is a need to combine information from multiple documents. For example, “*What are the reasons for global warming?*”.
- The question class can be derived from the question type and will be one of the WH-words (“why”, “what”, “when”, “where”, “who” or “which”).

- The focus can be obtained from the string of words in the question which can most likely be replaced with the answer. For example, in the question "*Which South African university is the oldest?*", the string "*South African university*" would be the focus and this can be replaced with "*University of Cape Town*".
- The answer type can be derived from the entity type (person, organisation, location, date, time, etc.). In the case of the previous example question, the answer type would be "place". The answer type can be derived from the question class. For example, the question class "when" will indicate an answer type of "date" and/or "time".
- For reformulation, the question keywords can be expanded using an online lexical resource like WordNet. Ontology and synonyms can be added using the WordNet synsets (Fellbaum, 2010).

The document and passage retrieval module performs queries with the aim of finding the top candidate answers based on the focus of the question. Most QA systems are based on the IR paradigm of retrieving a passage (instead of a whole document) where a passage could be a section, a paragraph or a sentence in a document. The most appropriate passages, in which the distance between the focus of the question and the candidate passages is at a minimum, are selected. Filtering of passages is also required, as only those passages which contain all the question keywords are normally desired. An issue to be aware of when using the cosine similarity method, is that it often results in the returning of passages not containing all desired keywords, as it favours short passages (Jeon, Croft & Lee, 2005a).

The last module, answer extraction, selects the most relevant passage from the set of candidate passages, and extracts and validates the final answer for the given question. The most relevant passage can be determined by using a combination of similarity, distance or raking measures.

These core modules, which form the heart of any QA system, can be implemented using a variety of techniques, algorithms and tools and these will be discussed in the following section.

2.3 Techniques, Algorithms and Tools

The top 15 techniques, algorithms and tools observed for QA systems in research papers based on the study by Soares & Parreiras (2018) are listed in Table 2.1. Note that their study did not apply mutual exclusivity to the different domains, resulting in a total count of 159 research papers, which is more than the total of 130 research papers observed.

Table 2.1: Top 15 techniques, algorithms and tools observed for QA systems

	Natural Language Processing (NLP)	Information Retrieval (IR)	Knowledge Base (KB)	Hybrid	Total
Part of Speech (POS) Tagging	14	15	11	2	**42**
Named-entity Recognition (NER)	10	14	8	2	**34**
Parser	4	4	5		**13**
Relation Finding (Similarity Distance)	5	3	4	1	**13**
Support Vector Machine (SVM)	9	4			**13**
Tokenization	6	4	3		**13**
Text Chunking	1	5	2		**8**
Stemming	2	1	1		**4**
Deep Neural Network (DNN)	2		1		**3**
Graph Based	3				**3**
Lemmatisation			3		**3**
Multi Document Summarization	2	1			**3**
Naive Bayes	2		1		**3**
Latent Semantic Analysis (LSA)	1	1			**2**
Shallow Syntactical	1		1		**2**
Grand Total	**62**	**52**	**40**	**5**	**159**

The main tools observed in order of top counts are:

- Part-of-Speech (POS) tagging, also known as grammatical tagging or word-category disambiguation, which is the process of identifying descriptive tags for a word within a text. For example, tagging a word as a noun, verb, adjective, adverb, etc.
- Named-entity Recognition (NER), also known as entity identification or extraction, which is the process of locating and classifying named entities mentioned in unstructured text into pre-defined categories such as person names, organisations, locations, date or time expressions, quantities, monetary values, percentages, etc.
- Parser, also known as syntax analysis, syntactic analysis or syntactic dependency parsing, which is the process of analysing words, conforming to the rules of a formal language grammar structure. The result is a parse tree.
- Relation finding, also known as similarity distance, which is the process of determining the similarity between two vectors. More specific examples are cosine similarity, Levenshtein distance, Euclidean distance, etc.
- Support Vector Machines (SVMs), which are supervised learning models used for classification or regression analysis.
- Tokenization, which is the process of splitting a string of text into sentences, words, sequence of words or individual characters.

Other algorithms which are of importance and mentioned often in the literature are:

- Term Frequency times Inverse Document Frequency (TF-IDF).
- Random Forests (RF).
- Radix sort to order candidate passages based on same words sequence score, distance score and missing keyword score.

It is important to be able to measure the accuracy of the answers provided by a QA system. In order to do this, evaluation metrics need to be defined and these will be introduced in the next section.

2.4 Evaluation Metrics

The metrics used to evaluate a QA system are the same as those used for classification and can best be described using a confusion matrix as shown in Table 2.2. Consider the example where a question needs to be classified as falling into the category of either "Workshops" or of "Non-Workshops" using a predictive model. The confusion matrix values would then be:

- True positive (a "Workshops" question correctly predicted as "Workshops" by the model).
- False negative (a "Workshops" question incorrectly predicted as "Non-Workshops" by the model).
- True negative (a "Non-Workshops" question correctly predicted as "Non-Workshops" by the model).
- False positive (a "Non-Workshops" question incorrectly predicted as "Workshops" by the model).

Table 2.2: Confusion matrix for two categories

		Actual Labels	
		Workshops	Non-Workshops
Predicted Labels	Workshops	True Positive	False Positive
	Non-Workshops	False Negative	True Negative

The actual labels are the pre-defined labels for the classification and the predicted labels are the labels predicted by the system. The evaluation metrics that can be considered are accuracy, error rate, precision, recall, F1-measure, area under the ROC curve (AUC) or human evaluation.

Accuracy is the percentage of all observations labelled correctly (Jurafsky & Martin, 2019). This is an intuitive measure but it is not generally used, as it does not work well when classes are unbalanced. For example, in the case of 1 000 questions where 900 have actual labels of "Non-Workshops". In the case of a dumb classifier where all questions are always labelled as “Non-Workshops”, the accuracy would be 90%. This result is misleading as the classifier would fail to find a single “Workshops” question.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + False\ Positives + True\ Negatives + False\ Negatives}$$

The inverse of accuracy is the error rate, where the percentage of all observations labelled incorrectly are calculated. Therefore, this measure has the same issue as accuracy.

$$Error\ Rate = \frac{False\ Positives + False\ Negatives}{True\ Positives + False\ Positives + True\ Negatives + False\ Negatives}$$

Precision is the percentage of observations correctly labelled as positive. In the 1 000 questions example the precision would be 0%. The precision measure is favoured for list type questions where it is important to consider the number of false positives.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Recall is the percentage of observations present in the input which were labelled correctly. In the 1 000 questions example the recall would be 0%. The recall measure is favoured for factoid type questions where it is not important how many false positives occurred.

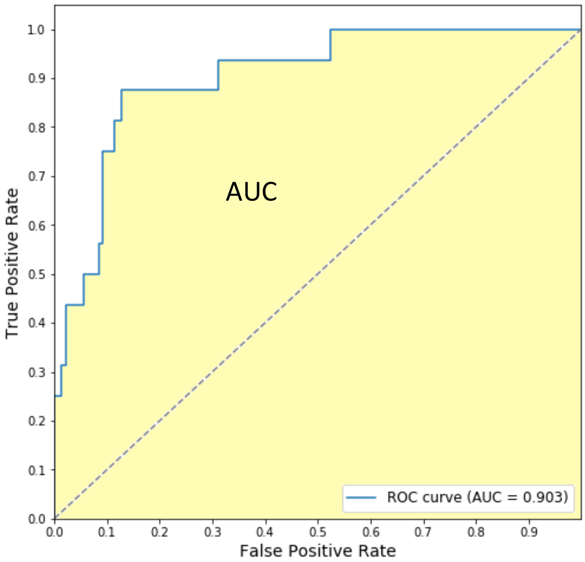
$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

F1-measure is a single metric which balances aspects of both precision and recall (the harmonic mean of precision and recall).

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Area under the ROC curve (AUC) is a metric used for the overall performance of a classifier (Fawcett, 2006). The Receiver Operating Characteristics (ROC) curve is a graph, as displayed in Figure 2.1, which shows the performance of a classification model at all possible thresholds for the true positive rate and the false positive rate. The two axes of the ROC curve present trade-offs between the true positives (correct classifications) and the false positives (errors) that a classifier makes for two classes.

Figure 2.1 : ROC curve and AUC



$$True\ Positive\ Rate = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

$$False\ Positive\ Rate = \frac{False\ Positives}{False\ Positives + True\ Negatives}$$

When classification for more than two classes is required, a separate ROC curve can be created for each class. Using multiple classes will be discussed next.

A QA system will need to classify questions into more than two categories. Table 2.3 shows an example where questions need to be classified as one of three categories and where the categories are mutually exclusive. In NLP, multinomial classification, also known as one-off classification, is used for this task (Jurafsky & Martin, 2019).

Table 2.3: Confusion matrix for multiple categories

		Actual Labels			
		Workshops	Tutorials	Quizzes	**Total**
Predicted Labels	Workshops	80	20	5	**105**
	Tutorials	116	270	55	**440**
	Quizzes	4	10	440	**455**
	Total	**200**	**300**	**500**	**1 000**

Precision	**Recall**	**F1**
76.2%	40.0%	52.5%
61.2%	90.0%	72.9%
96.9%	88.0%	92.2%
78.1%	**72.7%**	**72.5%**

Multinomial classification is done by building separate binary classifiers, for example the classifier for the “Workshops” category was trained on the 1 000 questions where 200 were labelled as positive and 800 as negative, the “Tutorials” category was trained where 300 were labelled as positive and 700 as negative, and the “Quizzes” category was trained where 500 were labelled as positive and 500 as negative. Given a test question, all three classifiers were used, and the label was selected from the classifier with the highest score. The example in Table 2.3 shows that the QA system incorrectly predicted 4 “Workshops” questions to be in the “Quizzes” category. A single metric can be used to determine how well the QA system performed by calculating the performance for each class, and then the average of this performance over all classes. In the example of using the F1-measure, the overall QA system performance would be 72.5%.

Using the average performance over all classes is also known as macro-averaging, where the performance of all classes is treated with equal importance. Another alternative would be to use micro-averaging, which can be considered if classes that occur more frequently are of higher importance. With micro-averaging, the outcomes of all categories are combined into a pooled confusion matrix as shown in Table 2.4. In this example, the micro-averaging F1-measure would be 84.3%.

Table 2.4: Separate confusion matrixes per category

		Class 1: Workshops		**Class 2: Tutorials**		**Class 3: Quizzes**		**Pooled**	
		Actual Labels		**Actual Labels**		**Actual Labels**		**Actual Labels**	
		True	False	True	False	True	False	True	False
Predicted Labels	True	80	25	270	171	440	14	790	210
	False	120	775	30	529	60	486	210	1 790

Human evaluation is where a test set is evaluated by humans in order to indicate whether a correct answer has been provided for a question. A general implementation of this measure is to use a five-point Likert scale rating and calculating the mean and standard deviation of ratings provided (Nguyen, 2018).

In the next section two QA systems used in learning environments will be discussed.

2.5 VTAs as a QA System

In this section two VTAs implemented as QA systems at well-known universities will be reviewed. The first is Jill Watson which has featured in several articles, but for which there is limited detail available as to how this QA system was built. Therefore, Percy was selected as a second VTA, as more implementation details have been disclosed in the case of this system, even if it has demonstrated poorer results than those claimed for Jill Watson.

Jill Watson, a VTA for Georgia Tech

Goel & Polepeddi (2016) identified the need for VTAs which can automatically answer student questions in the context of Massively Open Online Courses (MOOCs) where a course is run online via the World Wide Web and can have an unlimited number of students. Automation is essential to make MOOCs feasible.

The course run by the Georgia Institute of Technology (Georgia Tech) on Knowledge-Based Artificial Intelligence (KBAI) is an example of a MOOC where VTAs have been deployed. The first VTA version was created in January 2016 and was named Jill Watson, after the IBM Watson APIs used. The Jill Watson VTA is in essence a memory of question-answer pairs from previous course semesters posted on the digital course forum, and which have been organised into different question categories. For a new question, Jill will first classify it into a category using a predictive model and then retrieve an associated answer with a confidence score. If the score exceeds 97% then an answer is returned, otherwise the question is left for a human TA to answer. Goel and Polepeddi did not disclose any detail related to architecture, techniques, algorithms, tools or metric decisions for the creation of this VTA (Benedetto, Cremonesi & Parenti, 2019; Eicher, Polepeddi & Goel, 2018; Goel & Joyner, 2015; Goel & Polepeddi, 2016; Goel & Joyner, 2017; Goel et al., 2015; Joyner, 2018; Molnár & Szüts, 2018).

Students were not aware that Jill Watson is an AI agent but could select Jill as a TA to post questions to the online discussion forum of the course. Initially, all derived answers were first passed on to a human TA as a checkpoint before being posted on the forum. After two months of checking and improving the model, the human checkpoint could be removed and Jill was allowed to post answers autonomously.

In August 2016, the next VTA version was created by using proprietary software and open-source external Machine Learning (ML) libraries available in the public domain, instead of IBM Watson APIs. The reason for this change was to allow for the use of a larger set of questions and tasks for the new generation VTAs. Different names were assigned to the new VTAs, so that students could not distinguish between human and virtual TAs. This new VTA version was used to assist with a task where students introduce themselves on the forum during the first week of the course, and where a TA then replies to each student to welcome them on to the course. There were about 200-400 student introductions which needed to be replied to within one week and manual replies were time consuming, so this task was an appropriate new challenge for a VTA. A predictive model was used to take a student's introduction as input, and relevant concepts were then derived from this input using semantic processing and mapped to a suitable precompiled response.

Improvements were made by gradually expanding the range of question categories assigned to VTAs and more VTAs were used in following courses. Other improvements made were to extend the episodic memory of question-answer pairs, to use semantic information processing technology based on conceptual representations. For the task of welcoming students, the introduction provided by a student was mapped to relevant concepts which were then used as an index to retrieve the appropriate response.

Each new VTA performed better than its predecessor and, as the dataset of question-answer pairs grew, the quality of performance increased. The first VTA could only reply to 40% of student introductions, while the third version was able to reply to 60%. Students were only informed at the end of the course which TAs were not human and responses were very positive. During Course Instructor Opinion Survey (CIOS) one student responded by saying *"Just when I wanted to nominate Jill Watson as an outstanding TA in the CIOS survey!"*.

Research for the Georgia Tech VTAs was conducted over about two years and 12 000 question-answer pairs were used.

Percy, a VTA for Stanford University

A similar approach to the one taken by Georgia Tech was taken by Stanford University, where a VTA named Percy was created (Chopra, Gianforte & Sholar, 2016). The goal was to automatically answer repetitive questions for the Computer Science (CS) 221 Artificial Intelligence Principles and Techniques course, leaving unique questions to be answered by human TAs. Three high-level categories were created to classify policy-based, assignment and conceptual questions posted on the online course forums. Policy-based refers to course policy questions, assignment to specific course assignment questions, and conceptual to course content questions.

The selected architecture first classified any new question into one of three high-level categories. Once the category of the question was known, one sub method per category was used to find or generate the answer. The sub methods used were:

- Policy-based questions where regular expressions were used to further classify the new question into one of 15 subcategories. A set of precompiled answers was created and saved in a KB, and a second layer of regular expressions was then used to select the best answer by subcategory and by keywords within the question. If a question matched multiple subcategories, a relevant answer from each matching subcategory was then returned.
- Assignment questions where cosine similarity was used to find past questions most closely related to the new question. The answer paired to the most similar past question was then

used as the answer to the new question. TF-IDF vector representations for questions were used to calculate the cosine similarities. A threshold was set to help find the top cosine similarity for a new question. An answer of "*I don't know*" was returned when the threshold was not reached. The model failed to consistently find the best answer to a question using the top similarity score, but the best answer was often listed among the top few answers. Returning the top three answers instead of just a single answer was used as a way to address this problem, as the probability for the best answer being among the top three is higher than for it being the top one.

- Conceptual questions where intelligent IR on the course textbook was used. Two approaches were compared and the second approach was selected for use. The first approach compared every sentence of the textbook with a new question, but this ran very slowly and sometimes returned unrelated answers. The second approach divided the textbook into sections, first determining which sections of the textbook were most relevant, and only then comparing the question against every sentence in these sections. TF-IDF vector representations were used to calculate cosine similarities. The TF of the question was used when calculating the TF-IDF vector of the question, but the IDF of the entire textbook was used to determine the relevant sections. Once the top section was determined, the TF-IDF vector of the question was recalculated using the IDF of the top section to find the top three most relevant sentences in the section. The paragraph surrounding each of these sentences was then determined, as concepts related to conceptual questions usually require more than a single sentence to give context. These top three paragraphs were then returned as the answer to the question.

A linear SVM was used as a baseline classifier and then compared to Multinomial Naive Bayes (MNB) and RF classifiers. Token features such as unigram, bigram, and trigram counts were used. The token count vector representations were also compared with TF-IDF vector representations.

The RF classifier was further extended by using the Global Vector (GloVe) representations for words, developed by Stanford University, to determine if the use of these vectors could improve the model. The goal of using the GloVe word vectors was to exploit the learned similarities between words and phrases across large corpora in order to infer definitions of and relationships between words. The GloVe word vectors were pre-trained by Pennington, Socher & Manning (2014) on the combination of the 2014 Wikipedia dump and the Gigaword-5 dataset. In the case of each question, stop-words were first removed and GloVe word vectors were retrieved for remaining words. These vectors were then summed together to derive a single vector representation per question. The reason why the GloVe word vectors were only applied on the RF classifier is because the GloVe word vectors can be negative and the MNB predictor assumes a multinomial gaussian distribution.

Different datasets were used for each of the three high-level categories. For policy-based questions a set of precompiled answers was created, however, the paper published by Chopra, Gianforte & Sholar (2016) does not disclose how these answers were compiled. For assignment questions, a set of 1 500 question-answer pairs for the 2016 CS 221 course was used. For conceptual questions, the course textbook was used.

The evaluation metrics of precision, recall and F1-measure were used, along with an 80/20 split of training and testing data. The performance was the best for policy-based questions, moderate for assignment questions and poor for conceptual questions. Roughly 70% of the data fell into the category of assignment questions, 20% into that of policy-based questions and 10% into that of conceptual questions. A threshold was implemented where a response of "*I don't know*" was returned when the score of the predictive answer was below the set threshold (this also covers the scenario where a question was asked for the first time). The conclusion was made that conceptual questions were best left to human TAs.

VTAs are one type of QA system often used in learning environments. Another popular type of QA system found in websites is that of community-based question and answer services, which will be discussed in the next section.

2.6 Community-based QA Services

There are several community-based question and answer services in existence, where a question is answered by spreading it to a group of people with registered interest in the question topic. This is also referred to as Community Question Answering (CQA). Examples of websites hosting these CQA services are Stack Overflow, Quora, Yahoo! Answers and Google Answers. Users tend to use these websites instead of querying a web search engine when they struggle to express their question in a short query, have complex needs or when they perceive that humans would be more able to understand and answer their question than a machine would. However, answers do depend on the ability and willingness of other users in the community to address questions and this can lead to question starvation when questions are left unanswered.

Shtok et al. (2012) studied the Yahoo! Answers website and observed that 15% of questions do not receive answers and that 25% of questions are reoccurring. Their study investigated how to answer new questions, with the goal of avoiding question starvation by reusing past resolved questions using a predictive model specifically trained for this task. The Yahoo! Answers website was selected because it held the largest CQA repository of a billion posted answers at the time of the study, therefore offering the highest chance of reuse for past answers, and also offering easy open access through APIs and RSS feeds.

A closed-domain approach was taken where only three categories on the website were selected. Both content and intent similarity were identified as important. For example, the two questions "*How many pounds can you lose in 9 weeks?*" and "*Can I lose 9 pounds in a week?*" have high content similarity, but very different intent. Question formulation between different users with the same need may vary widely, for example one question could be concise and to the point where another might be lengthy and hypothetical.

The data used for their study consisted of 957 052 QA pairs for the period February to December 2010. Three categories from the Yahoo! Answers websites were used and their respective QA pair counts at the time: Beauty & Style (305 508), Health (449 890) and Pets (201 654). Questions consisted of a title (short description), body (detailed description) and an assigned category (chosen by the asker). Multiple answers could be provided by any user and consisted of detailed descriptions. The user who posted the question (the asker) could choose one of the answers as the best answer, or alternatively the community could vote for a best answer. The best answer was rated using the Likert scale by rewarding between one and five stars. A question was only marked as resolved once a best answer had been derived. For the 957 052 QA pairs, only those best answers chosen by the asker and those that were given a rating of at least three stars were used.

The IR paradigm was considered for the QA system, but CQA answers tend to be complex and not factoid. The NLP paradigm was therefore a better choice. The goal of the QA system was to find a single answer to a new question. To do this, an architecture consisting of a two-stage algorithm was used:

- First, those past resolved questions that were the most similar in content to the new question were identified. Only those questions where a best answer was selected and rewarded with at least three stars were considered. TF-IDF vector representations based on unigrams were used to calculate cosine similarities and to rank past questions. Only question titles were compared for similarity, and only past questions where the similarity score was above a set threshold were retained. These questions were then ranked by calculating a new similarity score using both "title-only" and "title and body" vectors. Finally, the top-ranked past question with its best answer was selected for the next stage. This approach could also be extended to use the top N, where the next stage would start with the top question and iterate to the next candidate if the top question did not pass the validation criteria.

- A classifier was applied to determine if the corresponding past answer met the need of the new question based on similar intent. The past answer was only used when the similarity confidence levels of both content and intent were high enough. As input text, the classifier took from:
    - New question title and body.
    - Past question title and body (as identified by the first stage of the algorithm).
    - Best answer for past question.

  Ninety-five additional features were then extracted for each of the three inputs, where these features could be categorised as:
    - Surface level statistics which tried to identify the focus, complexity and informativeness of the text. Most of these features were IDF related, for example: maximal IDF within all terms in the text, minimal IDF, average IDF, IDF standard deviation, etc.
    - Surface level similarity which measured how similar the input texts were in terms of lexical overlap. Features were based on cosine similarity between the TF-IDF weighted word unigram vectors of:
        - Titles of new and past questions.
        - Bodies of new and past questions.
        - Titles and bodies of new and past questions.
        - Title and body of past question and text of past answer.
        - Title and body of new question and text of past answer.
    - Latent topics where 200 Latent Dirichlet Allocation (LDiA) topics were learned from the corpus of past questions per category. The distribution of questions was inferred and topic quality features were generated.
    - Lexico-syntactic patterns where the Stanford dependency parser was used to determine the WH-question class and to generate the number of nouns, verbs and adjectives as question quality features. Features for negation and the negated predicated were also extracted.
    - Query clarity where a variant on the Kullback-Leibler divergence between the new question and the language model of past questions was used to calculate a clarity score.
    - Query feedback where a variant on the methods used for ad-hoc IR query feedback was used to generate measures for intra-question similarity, inter-question similarity and question-answer similarity.

    - Result list length where a feature that captured the number of candidates from past resolved questions as determined by the first stage of the algorithm.

Four classifiers were compared namely RF, Logistic Regression (LR), SVM and Naive Bayes (NB). The RF classifier consistently showed superior results. The hyperparameters selected were number of trees equal to 50 and number of features equal to 7. Little sensitivity to changes for these hyperparameters was observed.

Evaluation was done by using the metrics of F1-measure and area under ROC curve. A live system was also built to operate three robots that behaved as real users on the Yahoo! Answers website to provide predicted answers to live questions when the confidence level exceeded a set threshold (cosine similarity needed to be above 0.9).

Key observations from the study of Shtok et al. (2012) were:

- The best answer tends to contain terms which are repeated in the question as well as in the answer. Therefore, the dependency between question-answer pairs needs to be included in the model.
- Questions do not need to express the exact same intent in order to be satisfied by the same answer, for example a general informative answer can be used to satisfy differ questions connected by a central topic.
- It is important not to leave questions unanswered. By answering questions automatically, the “question starvation” problem can be avoided.
- Through the reuse of past answers “as is” and retaining human-generated content, opposed to auto-generation of texts, the community-based question and answer spirit was upheld.

2.7 Summary

QA is a well-researched field which can be described in terms of domains, paradigms, architectures, algorithms and the metrics needed to evaluate the reliability of answers generated by QA systems. Table 2.5 shows a summary of the QA systems reviewed in this chapter in terms of these aspects.

Table 2.5: Comparison of reviewed QA systems

		VTA Jill Watson	**VTA Percy**	**CQA system for Yahoo! Answers**
QA System Aspects	Domain	Closed-domain	Closed-domain	Closed-domain
	Paradigm	Hybrid	Hybrid	NLP
	Architecture	Not disclosed	Question classification, answer generation & answer provision	Top candidate selection from past questions, features extraction, classification, best answer selection
	Techniques, Algorithms & Tools	Not disclosed	Regexes, TF-IDF, SVM, Multinomial Naive Bayes, Random Forest, GLoVe word vectors	TF-IDF, Random Forest, Logistic Regression, SVM, Naive Bayes, LDA, Parser, KL-divergence
	Metric	Not disclosed	Precision, Recall & F1-measure	F1-measure & AUC
Data	# of QA pairs	12 000+	1 500	950 000+
	Period covered by QA pairs in months	24+	2	11
Research	Research period in months	24+	2	Not disclosed
	Year when research was published	2016	2016	2012

Challenges with QA systems are:

- Question classification.
- Formulation of correct queries.
- Ambiguity resolution.
- Semantic symmetry detection.
- Identification of temporal relationship in complex questions.
- Dealing with question unforeseen at the time of the QA system construction.

The following lessons learned from the literature reviewed have been selected to be applied in this study:

- A confined context in which to answer questions needs to be set. The problem becomes too difficult when combining many different categories and this results in poor performance.
- Quality of data is important and the cleaning of the data is vital.
- Increasing the amount of data helps to improve performance.
- Pattern matching using regular expression can be an effective way to help classify administrative (policy-based) questions.
- Finding the best answer to questions is difficult and returning the top three answers is a way to address this, as the probability for the best answer is higher in the top three than in the top one.
- Using word embedding, e.g. global word vector representations (GloVe), can be used with smaller datasets, but may not be suitable when the classification required is too specific.
- Setting a threshold to validate the confidence level of the selected answer to a question is needed. Returning a response of "*I don't know*" when the threshold has not been reached is better than returning a portion of an incorrect answer.

- Extracting additional features using various lexical, natural language and query performance prediction considerations can improve the predictive model.

In the next chapter the data relevant to this research will be discussed.

3. Data

The data for this research were sourced from the online teaching portal (also known as Vula) for the STA1000 course, taught at UCT at undergraduate level. Data for a 4-year period (2015 to 2018) were made available, during which the course was run 8 times and a total of about 6 400 students were registered. The selected data for this study are 509 question-answer pairs from the Administrative Queries site, where questions are posted by students and answered by the course administrator, the course convener or a lecturer. The course convener proposed selecting this particular subset of data because the goal of this study is to answer repetitive questions and because the administrative queries data were of a high standard. No ethical clearance was required because the data was obfuscated and generalised so that it could not be linked to any student.

During the discovery phase six data sources were investigated as potential data for this study. The data were extracted by the Learning Technologies Co-ordinator from the UCT online portal.

- **Chat Room** data where all participants (students, tutors and lecturers) can informally converse, as well as ask and answer course-related questions. This is the most unstructured source of data. The context of conversations is broad and can consist of anything considered pertinent to students. Not all posts are questions, and not all questions have answers. Question-answer pairs need to be constructed by linking a post from a tutor or lecturer to the relevant question asked by a student. This impacts the practicality of using this type of data and makes it a difficult problem to solve, due to the effort that would be required in cleaning the data, as well as the complex nature of communication in this forum.
- **Administrative Queries** where the context of the data is specific and where each post by a student is in the form of a question, which is then individually answered either by the course administrator, by the course convener or by a lecturer. This results in structured question-answer pairs where answers provided are of high quality. Using this type of data is more practical and makes it a reasonable problem to solve, due to less data cleaning required and because almost all questions have answers.
- **Questions for Your Lecturer** where formal course-related questions are managed. These questions are mostly content related and more complex to deal with, due to the variety of questions and the expert knowledge required to interpret and answer the questions correctly. Data is structured as question-answer pairs and categorised into topics to assist students in finding answers to past questions, before they choose to post a new question.
- **Questions for Your Tutor** and **Questions for Your Tutorial Group** where tutorial-related questions are managed, these being mostly content related. Posts are primarily questions but

can also consist of comments, and not all questions have answers. Data are unstructured and question-answer pairs need to be constructed by linking a post from a tutor to the relevant question asked by a student. This impacts the practicality of using this type of data and makes the auto-answering of content related questions a difficult problem to solve.

- Text from the **Course Information Pack**.
- Text from the **Course Textbook** (Underhill & Bradfield, 2014).

Table 3.1 shows a summary of these six data sources.

Table 3.1: Summary of potential data sources

Source	Observation Type	Context	Number of Observations*
Chat Room	Messages	Anything	1 812
Administrative Queries	Question-answer pairs	Administrative	509
Questions for Your Lecturer	Question-answer pairs	Content	1 242
Questions for Your Tutor	Messages	Content	1 625
2018 Course Information Pack	Words	Administrative	2 691
Course Textbook	Words	Content	133 521

* Number of data observations obtained over a 4-year period (2015 to 2018)

The final data used for this study consisted of 744 QA pairs, which are the result of:

- 509 administrative queries, where 12 of these queries were chosen for removal due to the following reasons:
    - 3 questions were related to protest action and therefore pertained to a very specific time period, meaning that these questions cannot be used for generalisation in the future. The 3 questions are also not enough to justify a separate question category and to train a predictive model on auto-answering question for this category type.
    - One question was the result of a student posting a question in the wrong forum.
    - Some questions were comments made on previous questions, so were not considered actual questions.
- 142 additional tutor questions. Due to the small number of administrative queries, additional questions were sourced by asking tutors to provide questions which are often asked by students. The approach taken is listed in Appendix A.1 which ensured that no bias was introduced when sourcing the data.
- 105 questions derived from the Course Information Pack, which is a valuable source for administrative queries. Table 3.2 lists 3 examples of how these questions and answers were derived (by reading the text and creating questions from the text).

Table 3.2: QA examples derived from the Course Information Pack

Text from the Course Information Pack	Question	Answer
There is also a strong emphasis on using Excel in this course, to perform calculations, to manipulate data and also to demonstrate principles of statistics. We hope that you will find that the skills you acquire in this course will empower you in your future decision making.	Why is Excel used in this course?	Excel is used in this course, to perform calculations, to manipulate data and also to demonstrate principles of statistics. We hope that you will find that the skills you acquire in this course will empower you in your future decision making.
The syllabus consists of 5 major modules, each divided into a number of work units. These will be laid out for you on Vula under Course Modules.	What is the syllabus of this course?	The syllabus consists of 5 major modules, each divided into a number of work units. These are laid out for you on Vula under Course Modules.
Within each work unit, you will be given some material to engage with (read notes, watch videos, complete quizzes, etc) in a step-wise fashion. This material will become available sequentially (a new section is revealed every Friday afternoon).	When is new course material made available?	New course material will become available every Friday afternoon. Within each work unit, you will be given some material to engage with (read notes, watch videos, complete quizzes, etc) in a step-wise fashion.

In the next section the data attributes will be discussed.

3.1 Attributes

Data extracted from the online portal have a total of 17 attributes of which only the question text and question date were used as predictors for the answer text. The other unused attributes consisted of identity values which could not be used to generalise on and were not significant enough to use in building a predictive model. These attributes are listed in Table 3.3. The "Answer Text" attribute was used as the label to predict a response for any given "Question Text".

Table 3.3: Original data attributes

Attribute	Type	Description	Significance
Site ID	GUID	Unique identifier for the course website	N/A
Site Title	String	Title or name of the course website, e.g. "STA1000S,2018"	N/A
Question ID	Integer	Unique identifier for a question	N/A
Question Owner ID	GUID	Unique identifier of student	N/A
Question Text	String	Full text of question posted	Predictor
Question Views	Integer	Number of views recorded on the website	N/A
Question Date	Date Time	Date and time when question was posted	Predictor
Question Published	Logical	Indicator if question was published on the website	N/A
Question Hidden	Logical	Indicator if question was hidden from the website	N/A
Answer ID	Integer	Unique identifier for an answer	N/A

Table 3.3: Original data attributes (continued)

Attribute	Type	Description	Significance
Answer Owner ID	GUID	Unique identifier of course administrator, course convener or lecturer	N/A
Answer Text	String	Full text of answer posted	Response
Answer Views	Integer	Number of views recorded on the website	N/A
Answer Date	Date Time	Date and time when answer was posted	N/A
Answer Approved	Logical	Indicator if answer was approved and posted on the website	N/A
Answer Private Reply	Logical	Indicator if answer was a private reply to the student	N/A
Answer Anonymous	Logical	Indicator if answer was marked as anonymous	N/A

The "Question Date" attribute can be used to derive a new "course week number" feature for when the question was asked. The course week number is a better predictor to use than the full date, as the course runs twice a year over a period of 12 weeks and the start date of each semester can be different across years. The course week number and the type of questions asked in that week are often linked. Table 3.4 provides examples of typical questions asked during the first two weeks of the course, relating to timetable clashes.

Table 3.4: Example questions with a mapping from question date to course week

Question Text	Question Date	Course Week*
Hi, I also have a clash for both workshop slots. Is it possible for another slot to become available. Perhaps on a Monday ?	2015-02-16	1
I have a lecture clash ,can I still send my timetable in so I can be placed in a suitable workshop.	2015-02-27	2
I have a timetable clash with every workshop other than the Wednesday 9 to 10 slot. But it is already full. What should I do?	2015-07-21	1
HI [*sic*] I have a clash with all the listed lecture times. Is it okay if I only do the workshops?	2015-07-26	1

* The course week number is derived from the question date, taking into account that the course runs multiple times a year.

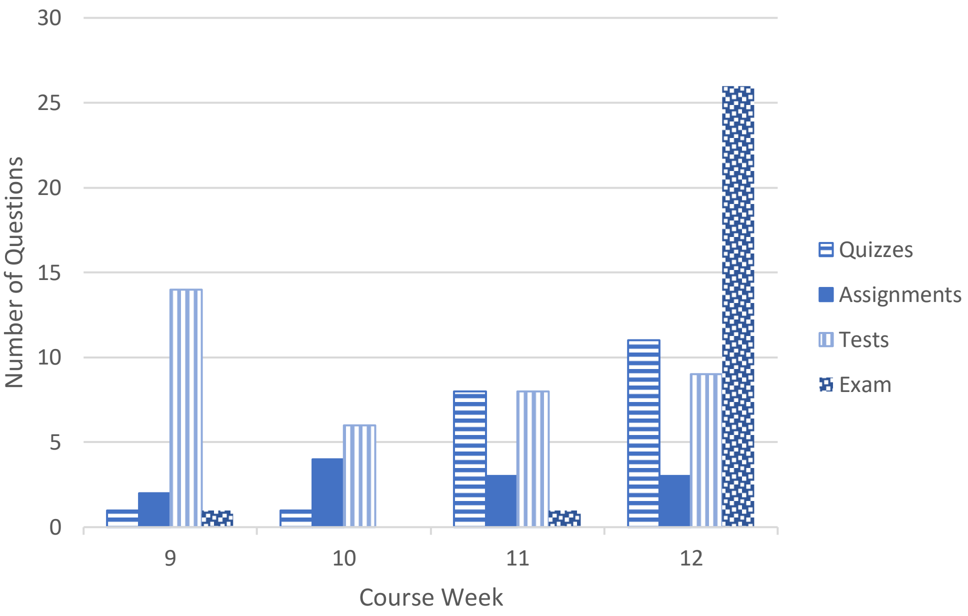
There are general milestones for some weeks as illustrated in Table 3.5. To validate the relevance of the course week number, the 509 administrative queries were analysed to see if the number of questions asked per week and their associated category aligned with these milestones. Most of these milestones only occur during the last 4 weeks of the course.

Table 3.5: Milestones linked to course weeks

Course Week	Milestone
5	First test
9	Second test
10 to 11	Excel test
12	Final exam

Figure 3.1 shows the number of questions for the last 4 weeks (e.g. weeks 9 to 12) by the top question categories. During course week 9, most questions fall into the "Tests" category, and in course week 12, most questions fall into the "Exam" category, as one would naturally expect. The assignment of question categories will be explained in more detail in Section 3.3.

Figure 3.1: Number of questions by course week and top question categories



The expectation would be for the predictive model to learn these associations between course weeks and the most likely question categories. This will be discussed in more detail in Chapter 6.

In the next section the data pre-processing steps needed to clean the data will be discussed.

3.2 Pre-processing

The extracted questions were in HTML format and required several text processing steps to clean the data. Table 3.6 lists all the text processing steps required.

The first step was to remove the HTML tags and formatting so that only the actual text of the question remained. The second step was to replace outdated course terminology. The course structure and terminology have changed over time, so it was important to standardise the terminology. For example, the terms "short workshop" (SWS) and "long workshop" (LWS) were used during a phase to try and run the course as a set of different types of workshops. "Short workshops" were equivalent to lectures and "long workshops" were equivalent to normal workshops. Other phrase replacements were also necessary and the full list of phrase replacements can be found in Appendix A.2. The third and fourth steps were to remove salutation phrases, closing phrases, punctuation and repeated spaces. The fifth step was to convert all text to lower case. In the sixth step stop-words were removed, as these do not add any meaning to the question asked. A custom list of 89 stop-words was used instead of a standard library, as it is important to have full control over which words should be considered stop-words (refer to Appendix A.3 for the custom stop-words list). A common issue with standard stop-word libraries is that they include negation words like “no” and “not”. In the seventh step, lemmatisation, the process where a word is converted to its meaningful base form, was applied. Lemmatisation differs from stemming, as it preserves the context of a word. The risk involved with stemming is that the meaning of a word can be lost when it is converted to its base form. For example, with stemming the word “better” would be converted to “bet”, while with lemmatisation it would be converted to “good”. The disadvantage of both stemming and lemmatisation is that in both cases some ambiguity of text is introduced, while the advantage for both is that text is generalised and this improves recall when a limited amount of text is available (Lane, Howard & Hapke, 2019). The improved recall allows the search application to associate more documents with the same query; this is similar to the aim of associating a new question with a past one. In the final step, text was converted into tokens and this process will be explained in more detail in Chapter 4.

Table 3.6: Text processing steps

Step	Question Text	Processing Note
0	<p>HI</p> <p>I have a clash with all the listed SWS times. Is it okay if I only do the LWS?</p>	Original text
1	HI I have a clash with all the listed SWS times. Is it okay if I only do the LWS?	Removed HTML tags and formatting
2	HI I have a clash with all the listed lecture times. Is it okay if I only do the workshops?	Replaced outdated terminology
3	I have a clash with all the listed lecture times. Is it okay if I only do the workshops?	Removed salutation and closing phrases
4	I have a clash with all the listed lecture times Is it okay if I only do the workshops	Removed punctuation and double spacing
5	i have a clash with all the listed lecture times is it okay if i only do the workshops	Converted to lower case
6	clash listed lecture times okay only workshops	Removed stop-words
7	clash list lecture time okay only workshop	Applied lemmatisation
8	{“clash”, “list”, “lecture”, “time”, “okay”, “only”, “workshop”}	Tokenized text

In the next section the data labelling process will be described, where answers were generalised and categories were assigned to each question.

3.3 Labelling

The answer for each question was reviewed and generalised. The original answers to questions were very specific and not well suited for generalisation. Table 3.7 provides some examples of how answers were generalised.

Table 3.7: Example QA pairs with generalisation

Question	Original Answer	Generalised Answer
HI [*sic*] I have a clash with all the listed SWS times. Is it okay if I only do the LWS?	Hi *Sam** There is different material covered in each of these sessions so ideally you should attend one SWS and one LWS per week. However, if you cannot attend a SWS session, the Monday 11am session is video recorded and you can watch that. But, yes, in your case, do make sure then that you can attend a LWS session where you can engage with lecturers and tutors and ask your questions! LS	There is different material covered in each of these sessions so ideally you should attend one lecture and one workshop per week. Do make sure then that you can attend a workshop session where you can engage with lecturers and tutors and ask your questions.
I have a timetable clash with other course lectures and cannot resolve it. What should I do?	Hi *Peter** You should ideally be able to attend one Short Workshop per week. If your timetable does not permit this you can make sure that you can at least attend a LWS and then view the recorded SWS of Monday 4th period. LS	You should ideally be able to attend one lecture per week that suits your timetable. If your timetable does not permit this, you can make sure that you at least attend a workshop.
I have a timetable clash with other course lectures and cannot resolve it. What should I do?	Hi *Peter** You should ideally be able to attend one Short Workshop per week. If your timetable does not permit this you can make sure that you can at least attend a LWS and then view the recorded SWS of Monday 4th period. LS	You should ideally be able to attend one lecture per week that suits your timetable. If your timetable does not permit this, you can make sure that you at least attend a workshop.
I have a timetable clash with every Long Workshop other than the Wednesday 9 to 10 slot. But it is already full. What should I do?	Hi *Ana** You should attend the Wed 9am LWS if it is the only one you can attend...it may be very full to start with but I expect will become less crowded as semester wears on as students who are able to do the questions on their own will not need to attend these sessions and may just come to check their answers are correct and then leave. LS	You should ideally be able to attend one lecture per week that suits your timetable. If your timetable does not permit this, you can make sure that you at least attend a workshop.

* Students names (i.e. Sam, Peter and Ana) are obfuscations of the actual student names related to these examples.

Specific rules were applied to provide generalised answers and these rules have been listed in Table 3.8.

Table 3.8: Generalisation rules

Rule	Examples related to the 1st original answer presented in Table 3.7
Remove salutation and closing phrases	The phrases "Hi Sam" and "LS" were removed. LS is the initials of the course convener who answered the question.
Replace outdated terminology	The abbreviations "SWS" and "LWS" were replaced with "lecture" and "workshop". The same phrase replacements as used for question text processing were used. The full list of phrase replacements can be found in Appendix A.2.
Remove text which was only relevant to the time when the question was asked, and which cannot be used as part of a general answer	The text "*However, if you cannot attend a SWS session, the Monday 11am session is video recorded and you can watch that*" and "*But, yes, in your case*" was removed.

Each question was assigned a category during the process of reviewing the original answers. A total of 16 categories were derived and these are listed in Table 3.9. Note the category imbalances as indicated by the frequency distribution, which range from 1% to 19%. The use of these categories in the process of automatically answering a question is explained in Chapter 4.

Table 3.9: Question categories with number of questions per category

Category Name	Category Description	Number of Questions	Frequency Distribution
Assignments	Questions related to assignments during the course.	86	12%
Content	These questions are content related and not administrative questions. Questions in this category have been asked in the incorrect forum and a standard answer will be provided to reflect this.	11	1%
DP Requirement	Questions related to the Duly Performed (DP) requirements of the course.	30	4%
Exam	Questions related to the final course exam.	43	6%
General	General questions related to accessing the course on the Vula website and making appointments with lecturers or the course convener.	77	10%
Hotseat Sessions	Questions related to the optional hotseat session in the lab where student can get help related to Excel or other course concepts.	24	3%
IntroStat Collection	Questions related to the course textbook and how a hardcopy can be acquired.	23	3%
Lectures	Lecture-related questions.	31	4%
Medical Certificate	Questions related to medical notes, the process of submitting them and the effect it may have on student performance.	31	4%
Quizzes	Weekly online quiz related questions.	88	12%
Tests	Test-related questions.	138	19%
Timetable clashes	Questions related to timetable clashes.	35	5%
Tutorials	Questions related to tutorial group meetings.	66	9%
Videos	Questions related to the online videos which students need to watch to guide them through the course material and weekly work units.	12	2%

Table 3.9: Question categories with number of questions per category (continued)

Category Name	Category Description	Number of Questions	Frequency Distribution
Winter/ Summer Term	Questions related to the additional winter and summer term short courses.	6	1%
Workshops	Questions related to weekly workshops.	43	6%

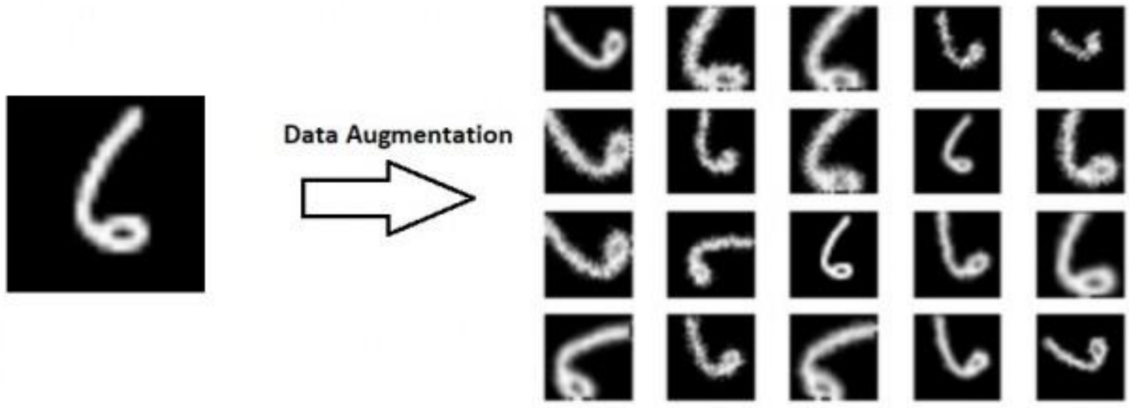
In the next section a data augmentation technique will be discussed, which can be used to increase the size of the available dataset.

3.4 Augmentation

Building a predictive model with small datasets does not generalise well, and often leads to overfitting. The predictive model may perform well on the training data (the known dataset), but with overfitting it will perform badly on new data. For these reasons, the 744 QA pairs were not sufficient and a systematic way to increase the dataset size needed to be implemented using the data augmentation technique.

Data augmentation is a technique frequently used with image data to help increase the number of data observations in order to avoid the problem of overfitting. With image augmentation, random transformations are applied to the original image, as illustrated in Figure 3.2. For this example, the handwritten number 6 is used as input and 20 new images are generated using random transformations based on rotating, zooming and shifting the image horizontally or vertically.

Figure 3.2: Example of image augmentation



Round-trip translation (RTT) is an augmentation method used for text, where the original text is translated into a different language (forward translation) and then back into the original language (back translation) (Marivate & Sefara, 2019). This method can result in synonym replacement, as well as dropping or adding words, while still preserving the meaning of the text. Table 3.10 shows an example where a question has been translated into Russian and then back into English. The same was done using Turkish and Dutch. The words that changed as a result of the back translation have been marked in gold.

Table 3.10: Examples of text augmentation using back translation

Language	Question Text	Processing Note
English (original question)	I have a clash with all the listed lecture times. Is it okay if I only do the workshops?	Start with the processed question text from step 3 as displayed in Table 3.6. All HTML tags have been removed, outdated terminology has been replaced, salutations and closing phrases have been removed.
Russian	У меня есть конфликт со всеми перечисленными временами лекции. Это нормально, если я только делаю семинары?	Result when translating the question from English into Russian using the Google Translate API.
English (back translation)	I have a conflict with all the lecture times listed. Is it ok if I just do workshops?	Result when translating back from Russian into English.
Turkish	Listelenen ders süreleriyle bir çatışması var. Sadece atölye çalışmaları yaparsam sorun olur mu?	Another example where the original question has been translated from English into Turkish.
English (back translation)	There is a conflict with the course times listed. Is it okay if I just do workshops?	Result when translating back from Turkish into English.
Dutch	Ik heb een botsing met alle genoemde college tijden. Is het goed als ik alleen de workshops?	Yet another example where the original question has been translated from English into Dutch.
English (back translation)	I have a collision with all mentioned lecture times. Is it good if I only do the workshops?	Result when translating back from Dutch into English.

Only 80% of the QA pairs were used to train a predictive model and this resulted in 587 questions which could be augmented with RTT. The Google Translate API supports 106 languages. The 587 questions were turned into 6 457 questions by using the RTT method and selecting 10 languages. To select 10 languages which would produce the best augmentation results, the RTT method was applied to the question, "*I have a clash with all the listed lecture times. Is it okay if I only do the workshops?*", using all 106 languages. To determine the best 10 languages the following approach was taken:

- All the RTT results were reviewed and languages where translation mistakes occurred were disqualified. For example, the Samoan language was disqualified when the RTT result of "*You have a conflict of interest and time to speak. Is it ok if I only do school?*" was returned.

- The cosine similarity between the original question text and each of the RTT results was calculated. Languages were sorted in ascending order of cosine similarity, so that the RTT results which were the most dissimilar to the original question were displayed at the top. Cosine similarity will be discussed in more detail in Section 4.3.
- The number of people speaking each language was determined by doing a Google search. If the answer was less than 15 million people then the language was disqualified, as the generalisation was applied that the fewer people speak a language, the less accurate the RTT would be. The Greek language, for example, was disqualified as the Google search for "*how many people speak Greek?*" returned the answer "*13 million people*".
- The top 10 languages were then selected from the remaining results and these are displayed in Table 3.11.

Table 3.11: Languages with best RTT results

Language	RTT Result	Cosine Similarity	# of People Speaking the Language (Millions)
Turkish	There is a conflict during all the course hours listed. Would it be good to just do the workshops?	0.2857	88
Chinese (traditional)	I am in conflict with all the time listed. Is it ok if I only attend the workshop?	0.4286	1 200
Myanmar (Burmese)	There was a clash between the sessions I had listed. Is it ok if I just do a workshop?	0.4286	33
Dutch	I have a collision with all mentioned lecture times. Is it good if I only do the workshops?	0.5714	17
French	I have a conflict with all of the course times listed. Is it ok if I only do workshops?	0.5714	275
Portuguese	I have a conflict with all the class schedules listed. Is it okay if I only do the workshops?	0.5714	260
Swahili	I have a conflict with all of the lecture times listed. Is it ok if I only do seminars?	0.5714	150
Russian	I have a conflict with all the listed lecture times. Is it ok if I only do workshops?	0.7143	150
Spanish	I have a clash with all the reading times listed. Is it ok if I only do the workshops?	0.7143	437
Arabic	I have a collision with all times of the lectures listed. Is it okay if you only do the workshops?	0.8571	422

In the next section, data exploration which was needed in preparation for feature engineering will be discussed.

3.5 Exploration

Exploratory data analysis was applied to view administrative queries by course, year and month in order to see if the date when a question was asked aligned with when the course took place. This was needed as a preparation step to determine if the question date can be used to derive a new course week feature. One outlier was identified, and this observation was removed from the dataset that was used to train the predictive model. Table 3.12 provides a summary of this analysis where the anomaly is marked with an asterisk (*).

Table 3.12: Number of administrative queries by course, year and month

		Calendar Month Number										
Course	**Year**	**2**	**3**	**4**	**5**	**7**	**8**	**9**	**10**	**11**	**12**	**Sub Total**
STA1000F	2015	41	20	22	21	1						105
	2018	13	12	7	4							36
STA1000S	2015					32	24	13	6	3		78
	2016					51	16	17	10	12	1	107
	2017		1*				8	15	23	20		67
	2018					41	35	19	21			116
	Grand Total	**54**	**33**	**29**	**25**	**125**	**83**	**64**	**60**	**35**	**1**	**509**

* Questions identified as outliers will be excluded from the dataset used to train the predictive model

During the review of questions asked, some cases were identified where a student had posted a comment which linked to a previous question. These comments are not considered questions in their own right and therefore a decision was made to exclude them.

In the next section, feature engineering used to create new features will be discussed.

3.6 Feature Engineering

Feature engineering is the process by which domain knowledge is applied in order to create new features which can be used as input for machine learning algorithms.

The course week number described in Section 3.1 is one example of a new feature created. Based on the work from Shtok et al. (2012) the following extra features were created:

- Part-of-Speech (POS) tagging features. Words were categorised and counted as nouns, pronouns, verbs, adjectives and adverbs. This was done by using basic grammatical analysis of text as implemented by the TextBlob library (Loria et al., 2014). The result was 5 additional features.

- Question type features. The occurrence of the words “what”, “when”, “where”, “why”, “who”, “which” and “how” were counted, resulting in an additional 7 features.
- Negation feature. The occurrence of the words “not”, “no”, “neither”, “never”, “nobody”, “none”, “nor”, “nothing” and “nowhere” were counted within each question, resulting in a single new feature that represents an indication of negation.
- Keyword features. Keywords specific to question categories were identified by analysing the top five most frequently occurring words per question category, while ignoring all words that are stop-words or that form part of question type features or negation features. A unique list of these keywords was created, resulting in an additional 47 features. The full list of keywords can be found in Appendix A.4.

3.7 Summary

The data available for this study consist of 6 614 QA pairs, which are a combination of 4 sources:

- 497 QA pairs collected over a 4-year period from the Administrative Queries site for the STA1000 course taught at UCT. While reviewing the original 509 QA pairs, 12 were identified as irrelevant and were removed for the following reasons:
    - 1 was identified as an outlier during pre-processing (described in Section 3.5).
    - 8 “questions” were simply comments made on previous questions.
    - 3 questions related to student protest announcements and were specific to that time period only, and as a result count not be generalised. Initially it was considered to have a separate question category for “Protest” but using 2 questions for training and 1 question for testing did not produce good results.
- 142 additional questions created by course tutors.
- 105 questions derived from the Course Information Pack.
- 5 870 questions created by using the RTT augmentation method.

The first 3 sources resulted in a total of 744 questions and, by adding the 5 870 augmented questions to this amount, a grand total of 6 614 questions was created. As an investigation into the effectiveness of data augmentation forms part of this study, the 2 datasets (744 and 6 614) were compared when used to build predictive models. Table 3.13 shows a summary of general statistics for these 2 datasets.

Table 3.13: Comparison between the example questions and the full dataset

	744 Questions	6 614 Questions
Total # of Sentences	1 408	12 408
Total # of Words	17 414	156 794
Unique # of Words	14 937	134 127
Total # of Terms	7 608	65 796
Unique # of Terms	1 240	2 348
Avg. # of Sentences per Question	1.9	1.9
Avg. # of Words per Question	23.4	23.7
Avg. # of Unique Terms per Question	10.2	9.9
% of Questions with Sentence Count = 1	50.7 %	52.1 %
% of Questions with Sentence Count = 2	25.9 %	24.5 %
% of Questions with Sentence Count > 2	23.4 %	23.3 %

In the next chapter the methodology relevant to this study will be discussed.

4. Methodology

The aim of this study was to automatically answer any new question posted in Administrative Queries for the STA1000 course, using a QA system. This was done by predicting the category of a new question as one of 16 possible values, as described in the previous chapter (Section 3.3). Examples of such categories were "Assignments", "Workshops" and "Tutorials". The following step involved retrieving all past questions which fell into the predicted category and finding the past question most similar to the new question asked. Finally, the answer to the most similar past question could be used as the answer to the new question.

To predict the category of a question, a text categorisation method was needed. A wide range of methods are available of which the simplest is the NB classifier and a more complex method being the RF classifier. Section 4.7 will explore the methodology related to text categorisation.

To find the past question most similar to the new question asked, a similarity method was needed to compare the new question to past questions and calculate a similarity score. Section 4.3 will explore how the best-known method, cosine similarity, was used for this task.

Before one can predict the category of a question or calculate the similarity, one needs to find a way to represent the meaning of the question, so that it can be interpreted by a machine. A method is required to convert natural language text to numbers so that the question can be processed by a machine, while still preserving the meaning of the text. NLP is the field of research concerned with defining such methods. Therefore, this chapter will start with explaining the core methodologies of NLP (Section 4.1 to 4.6) before progressing to the algorithms and models needed to answer a question automatically.

In order to explain the underlying methodology, a small example is provided in Table 4.1 with 13 questions and their relevant categories. The first 8 questions are different variations of asking for the name of the venue where workshops take place. The category used for these 8 questions is "Workshops" and all questions were given the same answer of, "*John Day Lecture Theater 1*". The next 5 questions are different variations of asking how to change one's tutorial group. These 5 questions fall into the category "Tutorials", and were given the same answer of, "*Note that you will be allowed to change tutorial groups during the first TWO WEEKS of semester only. Please email the tutor co-ordinator as listed in the course outline to request a change.*".

Table 4.1: Example of 13 questions divided into 2 categories

ID	Question	Category
1	Where are workshops held?	Workshops
2	What is the venue for workshops?	
3	In which building and room do workshops take place?	
4	Where should I go to attend a workshop?	
5	What is the workshop room name?	
6	Can someone please assist? Where should I go for the workshop?	
7	Has the venue for workshops been communicated?	
8	I'm confused with workshops and tutorials. Where do workshops take place?	
9	Can I please change my tutorial slot?	Tutorials
10	How do I alter my tut group?	
11	I need to switch to a different tut please, what should I do?	
12	Is it possible to change to a different tutorial meeting?	
13	What's the process to swop to a new tutorial group?	

For this example, the corpus D is the collection of the 13 questions (each question can generally be referred to as a document), and the corpus has a total of 15 sentences. The vocabulary size within the corpus after punctuation was removed, is a total of 104 words, containing 100 unique words. Each question can be turned into a list of terms by applying text processing (refer to Table 3.6 in the previous chapter). Text processing is applied in order to convert the questions to lower case, to drop any stop-words and to lemmatise the words. In this case, this resulted in a total of 58 terms of which 31 were unique terms. These 31 unique terms were used as building blocks to represent the meaning of the 13 questions. Table 4.2 shows the notation used in this study, where the symbols in list A relate to NLP terminology, and list B is a general list of symbols used for equations.

Table 4.2: Notation used in this study

A) Symbols related to NLP

Symbol	Name	Definition	Example
D	Corpus	The set of documents.	{"Where are workshops held?", … , "What's the process to swop to a new tutorial group?"}
d	Document	A single document within the corpus. For this study a document is the same as a question.	"Can someone please assist? Where should I go for the workshop?"
s	Sentence	A single sentence within a document.	"Can someone please assist?"
w	Word	A single word within a document. Also called a unigram.	"Can"

Table 4.2: Notation used in this study (continued)

Symbol	Name	Definition	Example
t	Term	A single word (unigram) or a sequence of words (bigram, trigram, etc.). Also called a token.	“Can” OR “Can someone” OR “Can someone please”
V	Vocabulary	The set of all terms.	{“alter”, “assist”, … , “workshop”}
C	Class set	The set of all classes use for document categorisation. For this study it is the same as the question categories.	{“Workshops”, “Tutorials”}
c	Class	A single class within the set of classes.	“Workshops”

B) Symbols for equations

Symbol	Name	Definition	Example
N	Document count	Total number of documents.	13
M	Vocabulary size	Total number of unique terms (features).	31
K	Class count	Total number of classes.	2
Pr	Probability	The probability of a document being in a class.	$Pr($“Where are workshops held?” $\vert$ “Workshops”$)$ $= 1.0$
i	Document index	Index used to step through documents.	$1, \ldots, N$
j	Vocabulary index	Index used to step through terms.	$1, \ldots, M$
k	Class index	Index used to step through classes.	$1, \ldots, K$
$\vec{x}$	Feature vector	Vector of term values or any other features selected to train a predictive model.	$(x_1, \ldots, x_M)$
X	Feature matrix	Matrix of all feature values needed to train a predictive model. Rows represent documents and columns represent features.	$\begin{bmatrix} x_{11} & \cdots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{NM} \end{bmatrix}$
$\vec{y}$	Label vector	Vector of class labels that the predictive model will learn to predict.	$(y_1, \ldots, y_N)$
$\vec{w}$	Weight vector	Vector of weight values. Also called coefficients. Each value is a real number and associated with one of the input features. Weights are learned during model training.	$(w_1, \ldots, w_M)$
W	Weight matrix	Matrix of all weight values learned during model training. Rows represent documents and columns represent features.	$\begin{bmatrix} w_{11} & \cdots & w_{1M} \\ \vdots & \ddots & \vdots \\ w_{N1} & \cdots & w_{NM} \end{bmatrix}$
T	Transformation	A transformation mapping for a vector or matrix.	
b	Bias term	Also called the intercept. A real number added to the weighted inputs and learned during model training.	0.1
x_j	Feature value	Term value or any other feature value used as input for model training.	0.8549
^	Hat	The hat notation is used to estimate any value. For example, the predicted question category.	$\hat{c}$

4.1 Bag-of-Words

In the field of NLP, the corpus can be divided into documents, sentences or terms. A term can either be a single word (unigram) or a sequence of words (bigram, trigram, etc.). In the example of the 13 questions (listed in Table 4.1), a term was first associated with a single word. Tokenization is the process of splitting a document into separate terms. Each term therefore becomes a token, meaning that the words “term” and “token” could be used interchangeably in this example.

The 31 unique terms can be used to represent the meaning of the 13 questions, as displayed in Table 4.3. This matrix representation of the questions was created by using word counts. Note that only question 8 contains a word which occurs more than once, namely “workshop”, which was counted twice.

Table 4.3: Matrix representation of questions mapped to unique list of terms

		alter	assist	attend	building	change	communicate	confuse	different	go	group	hold	how	meeting	name	need	new	place	possible	process	room	slot	someone	switch	swop	take	tutorial	venue	what	where	which	workshop
Workshops	1											1																		1		1
	2																											1	1			1
	3				1													1			1					1					1	1
	4			1						1																				1		1
	5														1						1								1			1
	6		1							1													1							1		1
	7						1																					1				1
	8							1										1								1	1			1		2
Tutorials	9					1																1					1					
	10	1									1		1														1					
	11								1							1								1			1		1			
	12					1			1					1					1								1					
	13										1						1			1					1		1		1			

Note that zero values have been replaced with blank spaces

Each row in the matrix is a vector, also called a Bag-of-Words (BoW), which can be written out as follows:

```
BoW1  = {"where":1, "workshop":1, "hold":1}
BoW2  = {"what":1, "venue":1, "workshop":1}
BoW3  = {"which":1, "building":1, "room":1, "workshop":1, "take":1, "place":1}
BoW4  = {"where":1, "go":1, "attend":1, "workshop":1}
BoW5  = {"what":1, "workshop":1, "room":1, "name":1}
BoW6  = {"someone":1, "assist":1, "where":1, "go":1, "workshop":1}
BoW7  = {"venue":1, "workshop":1, "communicate":1}
BoW8  = {"confuse":1, "workshop":2, "tutorial":1, "where":1, "take":1, "place":1}
BoW9  = {"change":1, "tutorial":1, "slot":1}
BoW10 = {"how":1, "alter":1, "tutorial":1, "group":1}
BoW11 = {"need":1, "switch":1, "different":1, "tutorial":1, "what":1}
BoW12 = {"possible":1, "change":1, "different":1, "tutorial":1, "meeting":1}
BoW13 = {"what":1, "process":1, "swop":1, "new":1, "tutorial":1, "group":1}
```

A BoW vector is useful for summarising the essence of a question. The essence can still be interpreted even if the words are ordered lexically, for example "where workshop hold" vs. "hold where workshop". These vectors, as displayed in Table 4.3, provide a data structure which is easier for a machine to work with, and can be used to build a predictive model. The matrix equation for each element x_{ij} to determine the frequency of term t_j in document d_i is:

$$x_{ij} = count(t_j, d_i) \tag{4.1}$$

Where:

$$i = 1, 2, \dots, N$$

$$j = 1, 2, \dots, M$$

For the 13 questions example $N = 13$ and $M = 31$.

The vector equation for document d_i is:

$$\vec{x}_i = (x_{i1}, x_{i2}, \dots, x_{iM}) \tag{4.2}$$

In the next section another method will be described that can be used to convert natural language text to numbers.

4.2 Term Frequency times Inverse Document Frequency

Different methods can be used to calculate the Term Frequency (TF). The simplest method is that described for BoW and is called the raw term frequency. Another method is to normalise the frequency by dividing the term count by the total number of terms in the document (Lane, Howard & Hapke, 2019). For big documents, the frequency can be downweighed further by using the log of the frequency (Jurafsky & Martin, 2019).

Terms can be represented in binary by only indicating the presence or absence of a term, instead of using term counts. The essence of a question will still be retained using this approach. This method is also known as one-hot encoding (or one-hot word vectors). A disadvantage of this approach is that frequencies are lost, but term frequencies are more relevant for comparing long documents to one another. It is rare to find a repetition of terms in a single sentence (or question), so this disadvantage does not carry a lot of weight when the aim is to create a comparison between questions. On the other hand, the advantage is that the data will automatically be normalised. For the 13 questions example, word counts greater than 1 are an exception and can cause issues with feature scaling, therefore it would be advised to use one-hot encoding to replace the value of 2 in question 8 with the value of 1, ensuring that all values are normalised.

N-grams is an approach using a contiguous sequence of words, instead of individual words in a sentence. For example, a 1-gram (unigram) would be a single word like “where”, a 2-gram (bigram) would be a pair of words like “where workshop”, and a 3-gram (trigram) would be a triplet of words like “where workshop hold”. The disadvantage of using BoW is that the inherent meaning derived from the order of words in the sentence is lost. Instead, when using bigrams, the negation word “not” will remain attached to the word it is negating, for example “not available” vs. “is available”, therefore helping to retain the inherent meaning. Another advantage is that the meaning from fixed phrases (or concepts) like “public holiday” can be derived.

Another approach which is often used is Term Frequency times Inverse Document Frequency (TF-IDF). A term can be a word or an n-gram, and the TF-IDF value is calculated for each unique term in the vocabulary. Any of the methods described above can be used to calculate the term frequency. Inverse Document Frequency (IDF) means dividing the term frequency by the number of questions in which it occurs. The TF-IDF calculation for each term in a document can be done with the following steps:

- Start by calculating the normalised TF by dividing the term count by the total number of terms in the document:

$$tf_{t,d} = \frac{count(t,d)}{count(d)} \tag{4.3}$$

- Next, calculate the IDF by taking the natural logarithm of the fraction when dividing the total number of documents N in the corpus by the number of documents in which term t occurs:

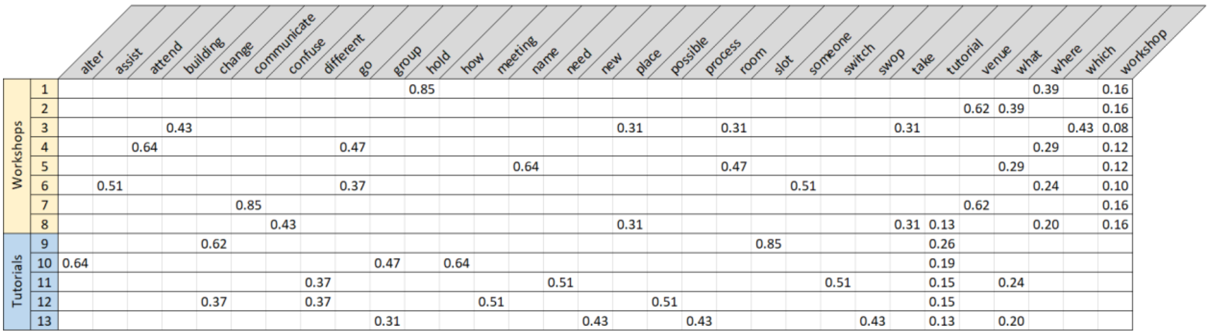
$$\begin{aligned} df_t &= count(d_t) \\ idf_t &= log_e\left(\frac{N}{df_t}\right) \end{aligned} \tag{4.4}$$

- Finally, calculate the TF-IDF weighting for term t in document d by multiplying TF with IDF:

$$tfidf_{t,d} = tf_{t,d} \times idf_t \tag{4.5}$$

Table 4.4 displays the matrix representation of the questions when using TF-IDF values for terms. Note that values are automatically normalised using this method and that the value for “workshops” for question 8 now has the same value of 0.16 as for most other questions in the category “Workshops”.

Table 4.4: Matrix representation of questions using TF-IDF



		alter	assist	attend	building	change	communicate	confuse	different	go	group	hold	how	meeting	name	need	new	place	possible	process	room	slot	someone	switch	swop	take	tutorial	venue	what	where	which	workshop
Workshops	1											0.85																		0.39		0.16
	2																											0.62	0.39			0.16
	3				0.43													0.31			0.31					0.31					0.43	0.08
	4			0.64						0.47																				0.29		0.12
	5														0.64						0.47								0.29			0.12
	6		0.51							0.37													0.51							0.24		0.10
	7						0.85																					0.62				0.16
	8							0.43										0.31								0.31	0.13			0.20		0.16
Tutorials	9					0.62																0.85					0.26					
	10	0.64									0.47		0.64														0.19					
	11								0.37							0.51								0.51			0.15		0.24			
	12					0.37			0.37					0.51					0.51								0.15					
	13										0.31						0.43			0.43					0.43		0.13		0.20			

Note that zero values have been replaced with blank spaces

The aim was to find values which represent the meaning of questions. For the first question, "*Where are workshops held?*", the word "hold" (the lemmatisation of "held") has the highest value, and the word "workshop" has the lowest value. This implies that "hold" is more important than "workshop" when representing the meaning of question 1. This, however, is not true. The word "hold" is more unique, as it only occurs in question 1, but logic and common sense tell us that the word "workshop" is more relevant in describing the context of this question.

One option is to adjust the TF-IDF vector values based on a list of keywords that best describe the known question categories. Table 4.5 lists example keywords per category for the 13 questions.

Table 4.5: List of keywords per category for the 13 questions example

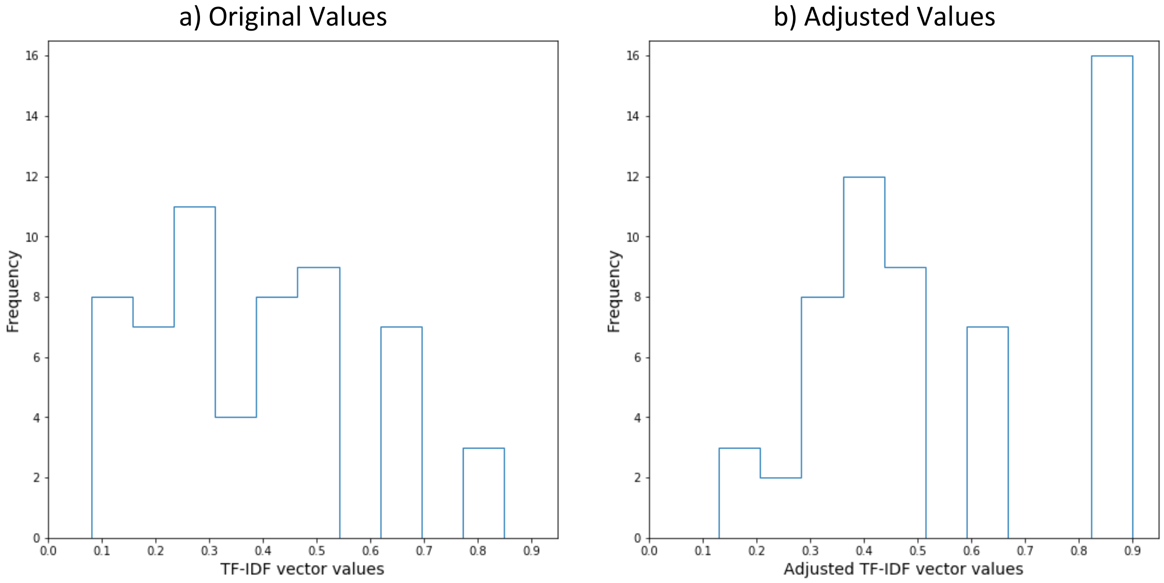
Category	Keywords	Mean TF-IDF vector value*
Workshops	venue	0.6239
	place	0.3120
	room	0.3900
	where	0.2799
	workshop	0.1335
Tutorials	alter	0.6412
	change	0.4991
	tutorial	0.1697
	group	0.3900

* The mean of all non-zero TF-IDF vector values

The normalised TF-IDF vector values range from 0 to 1, with a maximum value of 0.8549. Multiplying the keyword vector values for question 1 with a factor of 2 will result in a change of the value for the word "workshop", from 0.1618 to 0.3237, but this is not enough to raise its importance above the value of 0.8549 attributed to the word "hold". Multiplying the keyword vector values for question 2 with a factor of 2 will however change the value of the word "venue" from 0.6239 to 1.2478, which is outside of the normalised range. Therefore, factor multiplication is not the best option to use to adjust vector values. Instead, keyword vector values can be updated to a fixed value, which is more than the maximum value but still less than the value of 1, for example 0.9.

Figure 4.1 shows the frequency distribution of the original TF-IDF values, compared with the adjusted keyword values.

Figure 4.1: Frequency distribution of TF-IDF vector values



A disadvantage of using this approach is that the frequency distribution of vector values will change and this will impact the calculation of similarity scores between question vectors. Similarity will be discussed in the next section.

4.3 Similarity

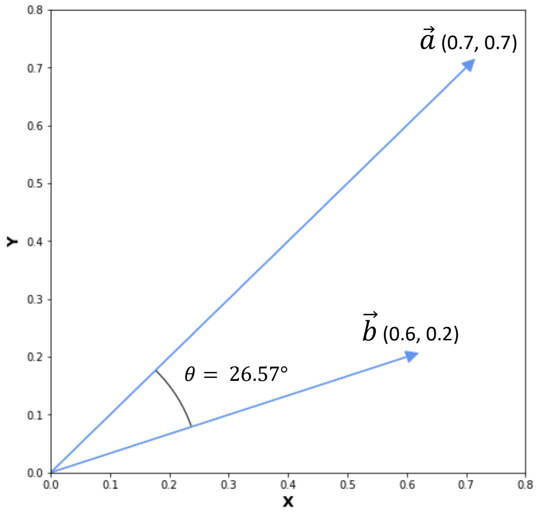
The best-known method to determine the similarity between documents (or sentences) is cosine similarity, where the cosine of the angle between documents is compared. The outcome of cosine similarity will be a value between -1 and 1. Documents which are exactly the same will have a cosine similarity of 1, and documents which are completely different will have a cosine similarity of -1. If the vector values used to represent documents are always greater than zero, as is the case with BoW and TF-IDF, then the cosine similarity cannot be negative and as a result will be bounded between 0 and 1. Word embeddings like Word2vec or GloVe can result in negative vector values, and are therefore examples of where the cosine similarity will be between -1 and 1.

The cosine similarity is calculated using the dot product and magnitude between two vectors as follows:

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{|\vec{a}||\vec{b}|} \tag{4.6}$$

Figure 4.2 shows two document vectors, $\vec{a}$ and $\vec{b}$, with an angle of 26.57° between them.

Figure 4.2: Cosine similarity between vectors in a 2D space



The cosine similarity between these vectors can be calculated as follows using equation 4.6:

$$\cos\theta = \frac{(0.7,\ 0.7)\cdot(0.6,0.2)}{|(0.7,0.7)||(0.6,0.2)|} = \frac{0.42+0.14}{\sqrt{0.49+0.49}\sqrt{0.36+0.04}} = \frac{0.56}{0.99\times 0.63} = 0.89$$

Which is of course the same as:

$$\cos 26.57° = 0.89$$

The angle between word vectors will not be known, but the cosine similarity can be calculated using the vector representations resulting from TF-IDF.

The power of cosine similarity is that a single value can be calculated for any vector, regardless of the dimensionality of the vector. These values capture the meaning of vectors and can be used to compare vectors (or questions in this example) to one another. Only the angle between vectors (the difference in direction) determines their similarity, not the magnitude of the vectors.

Table 4.6 shows the cosine similarity values for the 13 questions. The highest values have been marked in green, illustrating for example that question 1 is that which is most similar to question 8 (when using TF-IDF vector values as input for the cosine similarity calculation).

Table 4.6: Matrix representation of question similarities

		Workshops								Tutorials				
		1	2	3	4	5	6	7	8	9	10	11	12	13
Workshops	1		0.152	0.087	0.348	0.117	0.293	0.130	0.353					
	2	0.152		0.094	0.127	0.377	0.107	0.578	0.192			0.209		0.184
	3	0.087	0.094		0.072	0.297	0.061	0.080	0.451					
	4	0.348	0.127	0.072		0.097	0.500	0.108	0.295					
	5	0.117	0.377	0.297	0.097		0.082	0.108	0.148			0.161		0.142
	6	0.293	0.107	0.061	0.500	0.082		0.091	0.249					
	7	0.130	0.578	0.080	0.108	0.108	0.091		0.165					
	8	0.353	0.192	0.451	0.295	0.148	0.249	0.165		0.111	0.091	0.085	0.082	0.075
Tutorials	9								0.111		0.131	0.122	0.382	0.108
	10								0.091	0.131		0.100	0.096	0.287
	11		0.209			0.161			0.085	0.122	0.100		0.292	0.200
	12								0.082	0.382	0.096	0.292		0.079
	13		0.184			0.142			0.075	0.108	0.287	0.200	0.079	

Note that the diagonal and zero values have been replaced with blank spaces

In the next section semantic analysis will be described, a method used to convert text to numbers while still preserving the meaning of word combinations.

4.4 Semantic Analysis

Semantic analysis or semantic search is a further improvement for finding the meaning of word combinations within a document. Semantic analysis allows for finding topics in documents where topics are formed through the combination of certain words. TF-IDF word vectors can be used as a way to determine how important words are within a document, however, it must be noted that TF-IDF only supports key-word search and does not support semantic search.

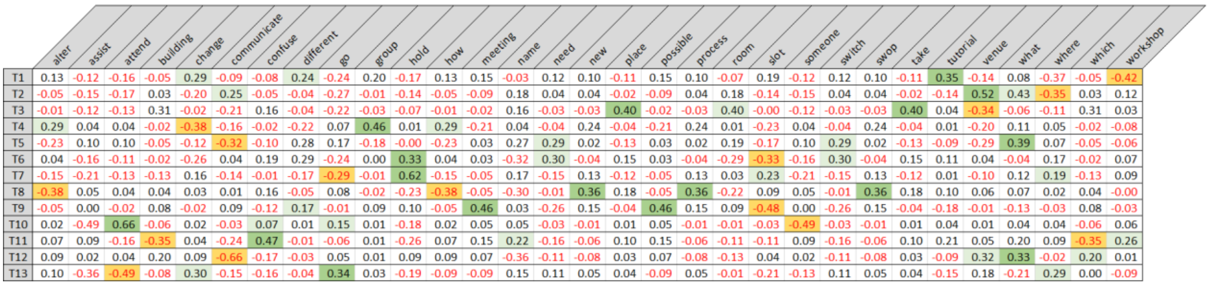
Latent Semantic Analysis (LSA) is a method used in NLP to reveal the meaning of word combinations within documents and to support semantic search. By using LSA, it is possible to discover topic vectors which can be used to describe a document. LSA uncovers the hidden semantics or meaning within documents. Either BoW or TF-IDF vectors can be used as the input for calculating the LSA topic vectors. TF-IDF vectors usually capture more meaning than BoW (Lane, Howard & Hapke, 2019), however, the two approaches were still compared in this study in order to confirm which produces the best results.

Using LSA topic vectors is an efficient way to represent the meaning of documents, as it is a dimension reduction method which uses the same algorithm as Principal Component Analysis (PCA), namely Singular Value Decomposition (SVD). With SVD, the input TF-IDF term-document matrix is broken down into three simpler matrices, which can be multiplied back together in order to produce the original matrix. This approach is also referred to as factorisation. The three simpler matrices that result from using SVD reveal properties of the original TF-IDF matrix, which can then be exploited. These matrices can be truncated by dropping some rows or columns before multiplying them back together, which results in dimension reduction. LSA helps to derive topics relating to different word combinations. The maximum number of topics which can be derived is equal to the minimum between the total number of unique words (dimensions) and the total number of documents. Therefore, only 13 topics can possibly be derived for the 13 questions example, which has a total of 31 dimensions. The first topic will represent the greatest amount of variance for words across the documents in which they appear. The second topic will represent the second greatest amount of variance, and so forth.

Table 4.7 shows the 13 topics with their LSA weights per word. The following observations can be made when looking at the top 3 topics:

- For topic one (T1) the word “tutorial” has the highest weight, followed by “change” and “different”. The word “workshop” has a negative weight, indicating that T1 is not about “workshop”. These observations indicate that T1 is a strong indicator for the “Tutorials” question category.
- For topic two (T2) the word “venue” has the highest weight, followed by “what” and “communicate”. The word “where” has a negative weight, indicating that T2 is not about “where”. These observations indicate that T2 is a strong indicator for the “Workshops” question category, as the words “venue” and “communicate” only occur in this category, but it also describes a bit about the “Tutorials” category, as the word “what” occurs in both categories (“Workshops” and “Tutorials”).
- For topic three (T3) the words “take” and “place” share the highest weight, followed by “room”. The word “venue” has a negative weight, indicating that T3 is not about “venue”. These observations indicate that T3 is a very specific indicator for questions in the “Workshops” category where the words “take”, “place” and “room” do occur, but the word “venue” does not.

Table 4.7: Matrix representation of LSA topic vectors



	alter	assist	attend	building	change	communicate	confuse	different	go	group	hold	how	meeting	name	need	new	place	possible	process	room	slot	someone	switch	swop	take	tutorial	venue	what	where	which	workshop
T1	0.13	-0.12	-0.16	-0.05	0.29	-0.09	-0.08	0.24	-0.24	0.20	-0.17	0.13	0.15	-0.03	0.12	0.10	-0.11	0.15	0.10	-0.07	0.19	-0.12	0.12	0.10	-0.11	0.35	-0.14	0.08	-0.37	-0.05	-0.42
T2	-0.05	-0.15	-0.17	0.03	-0.20	0.25	-0.05	-0.04	-0.27	-0.01	-0.14	-0.05	-0.09	0.18	0.04	0.04	-0.02	-0.09	0.04	0.18	-0.14	-0.15	0.04	0.04	-0.02	-0.14	0.52	0.43	-0.35	0.03	0.12
T3	-0.01	-0.12	-0.13	0.31	-0.02	-0.21	0.16	-0.04	-0.22	-0.03	-0.07	-0.01	-0.02	0.16	-0.03	-0.03	0.40	-0.02	-0.03	0.40	-0.00	-0.12	-0.03	-0.03	0.40	0.04	-0.34	-0.06	-0.11	0.31	0.03
T4	0.29	0.04	0.04	-0.02	-0.38	-0.16	-0.02	-0.22	0.07	0.46	0.01	0.29	-0.21	0.04	-0.04	0.24	-0.04	-0.21	0.24	0.01	-0.23	0.04	-0.04	0.24	-0.04	0.01	-0.20	0.11	0.05	-0.02	-0.08
T5	-0.23	0.10	0.10	-0.05	-0.12	-0.32	-0.10	0.28	0.17	-0.18	-0.00	-0.23	0.03	0.27	0.29	0.02	-0.13	0.03	0.02	0.19	-0.17	0.10	0.29	0.02	-0.13	-0.09	-0.29	0.39	0.07	-0.05	-0.06
T6	0.04	-0.16	-0.11	-0.02	-0.26	0.04	0.19	0.29	-0.24	0.00	0.33	0.04	0.03	-0.32	0.30	-0.04	0.15	0.03	-0.04	-0.29	-0.33	-0.16	0.30	-0.04	0.15	0.11	0.04	-0.04	0.17	-0.02	0.07
T7	-0.15	-0.21	-0.13	-0.13	0.16	-0.14	-0.01	-0.17	-0.29	-0.01	0.62	-0.15	-0.05	0.17	-0.15	0.13	-0.12	-0.05	0.13	0.03	0.23	-0.21	-0.15	0.13	-0.12	0.01	-0.10	0.12	0.19	-0.13	0.09
T8	-0.38	0.05	0.04	0.04	0.03	0.01	0.16	-0.05	0.08	-0.02	-0.23	-0.38	-0.05	-0.30	-0.01	0.36	0.18	-0.05	0.36	-0.22	0.09	0.05	-0.01	0.36	0.18	0.10	0.06	0.07	0.02	0.04	-0.00
T9	-0.05	0.00	-0.02	0.08	-0.02	0.09	-0.12	0.17	-0.01	0.09	0.10	-0.05	0.46	0.03	-0.26	0.15	-0.04	0.46	0.15	0.09	-0.48	0.00	-0.26	0.15	-0.04	-0.18	-0.01	-0.13	-0.03	0.08	-0.03
T10	0.02	-0.49	0.66	-0.06	0.02	-0.03	0.07	0.01	0.15	0.01	-0.18	0.02	0.05	0.05	-0.03	-0.01	0.01	0.05	-0.01	-0.01	-0.03	-0.49	-0.03	-0.01	0.01	0.04	0.01	0.04	0.04	-0.06	0.06
T11	0.07	0.09	-0.16	-0.35	0.04	-0.24	0.47	-0.01	-0.06	0.01	-0.26	0.07	0.15	0.22	-0.16	-0.06	0.10	0.15	-0.06	-0.11	-0.11	0.09	-0.16	-0.06	0.10	0.21	0.05	0.20	0.09	-0.35	0.26
T12	0.09	0.02	0.04	0.20	0.09	-0.66	-0.17	-0.03	0.05	0.01	0.09	0.09	0.07	-0.36	-0.11	-0.08	0.03	0.07	-0.08	-0.13	0.04	0.02	-0.11	-0.08	0.03	-0.09	0.32	0.33	-0.02	0.20	0.01
T13	0.10	-0.36	-0.49	-0.08	0.30	-0.15	-0.16	-0.04	0.34	0.03	-0.19	-0.09	-0.09	0.15	0.11	0.05	0.04	-0.09	0.05	-0.01	-0.21	-0.13	0.11	0.05	0.04	-0.15	0.18	-0.21	0.29	0.00	-0.09

Note values have been rounded to two decimal places for display purpose, but ranking is done on all decimal places

🟩 Vector weights with highest value per topic

▨ Vector weights with 2nd or 3rd highest value per topic

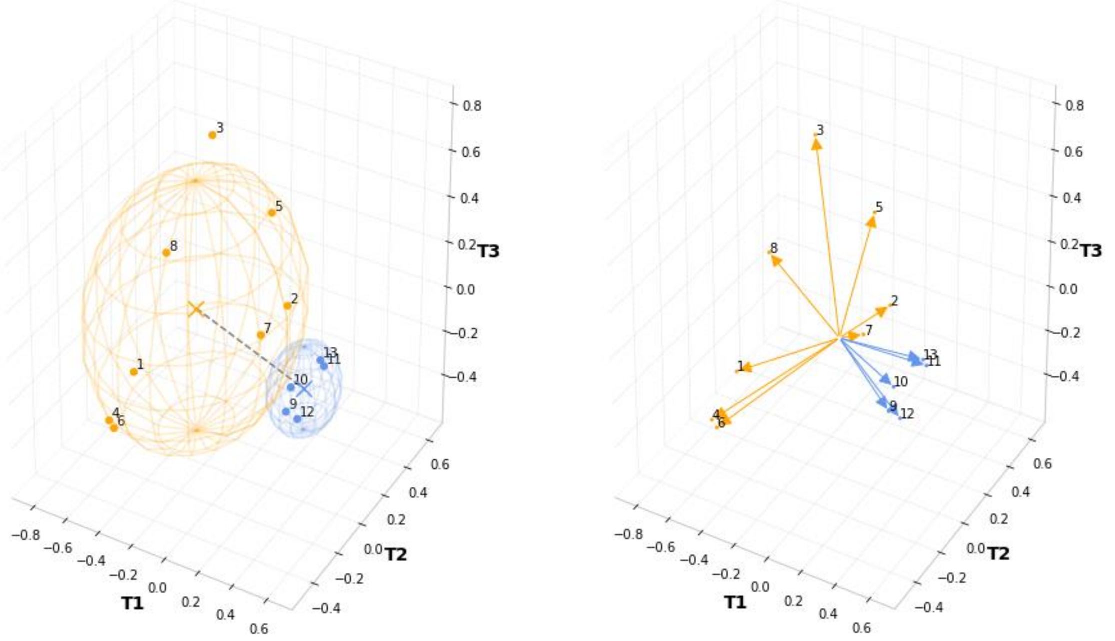
🟨 Vector weights with lowest value per topic (indicate what topic is not about)

These 3 topics are intuitive and do accurately describe the 2 categories which the 13 questions fall into. Figure 4.3 illustrates what these LSA vectors look like in a 3D space when the 31 dimensions for the 13 questions are reduced to 3 dimensions by using the top 3 LSA topics.

Figure 4.3: The 13 questions represented in a 3D space using LSA topics

a) Question Points with Category Centroids

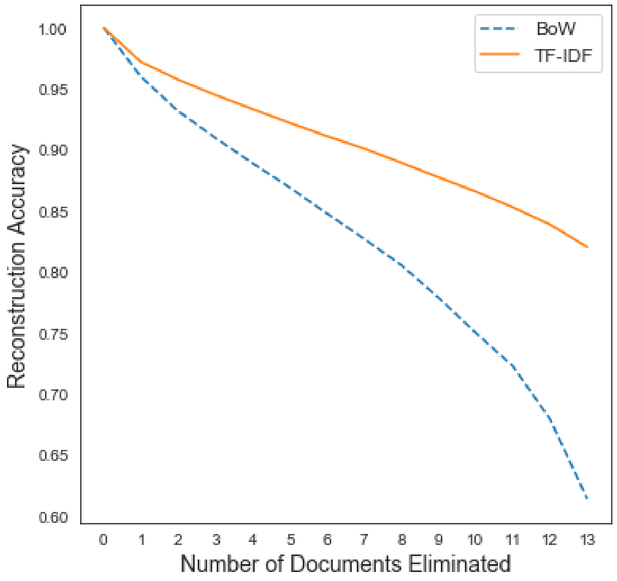
b) Question Vectors



The numbers next to the plotted points represent the question IDs. In Figure 4.3a, the centroid of each category has been calculated by taking the mean of the LSA vectors per category, and a sphere has been drawn around the centroid where the radius is the average distance of questions per category. In Figure 4.3b, the vectors are drawn from the zero point between the three topic axes. It is clear from both Figure 4.3a and 4.3b that the question vectors for the two categories are well separated, allowing for easy differentiation between the categories. This illustrates the power of this method.

In this case, only three topics were selected simply because it is the maximum number of dimensions that can be represented visually. Retaining all topics will ensure that the original TF-IDF matrix can be reconstructed with a 100% accuracy. Figure 4.4 illustrates a way in which to decide how many dimensions or documents can be eliminated, and what the impact will be on reconstruction accuracy. In the case of the example of retaining 3 topics, a total of 10 topics were eliminated, which resulted in a reconstruction accuracy of 87% when using TF-IDF.

Figure 4.4: LSA topic accuracy by number of documents eliminated



LSA produces a linear transformation, but depending on the data being analysed, a nonlinear transformation may be more suitable. LDiA is similar to LSA but uses a nonlinear algorithm to group words together, and can therefore yield better results in some situations. In this study, the acronym LDiA is used to differentiate Latent Dirichlet Allocation from Linear Discriminant Analysis (LDA). LDiA is based on the following assumptions:

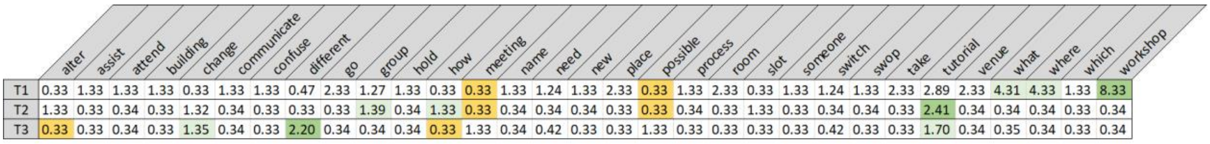
- Each document is a mixture of a predetermined number of topics.
- Each topic can be represented by a distribution of words (term frequencies).
- Both the probability of each topic within a document and the probability of a word associated with a topic fall within the Dirichlet probability distribution.

This makes LDiA more precise in the statistical allocation of words to topics than the linear algorithm used by LSA. The input for LDiA needs to be in the form of raw BoW vectors and not in the form of normalised TF-IDF vectors, due to the assumption that each topic is represented by term frequencies. LDiA is a stochastic algorithm relying on an initial random allocation of words to topics, therefore the weights of words per topic will differ with each training run, unless a fixed random seed is set.

Table 4.8 shows the top three topics with their LDiA weights per word. The following observations can be made:

- Topic one (T1) is a strong indicator for the “Workshops” question category, with high weights for words “workshop”, “where” and “what”.
- Topic two (T2) is a strong indicator for the “Tutorials” question category, with high weights for words “tutorial”, “group” and “how”.
- Topic three (T3) is a medium indicator for the “Tutorials” question category, with high weights for words “different”, “tutorial” and “change”.

Table 4.8: Matrix representation of LDiA topic vectors



	alter	assist	attend	building	change	communicate	confuse	different	go	group	hold	how	meeting	name	need	new	place	possible	process	room	slot	someone	switch	swop	take	tutorial	venue	what	where	which	workshop
T1	0.33	1.33	1.33	1.33	0.33	1.33	1.33	0.47	2.33	1.27	1.33	0.33	0.33	1.33	1.24	1.33	2.33	0.33	1.33	2.33	0.33	1.33	1.24	1.33	2.33	2.89	2.33	4.31	4.33	1.33	8.33
T2	1.33	0.33	0.34	0.33	1.32	0.34	0.33	0.33	0.33	1.39	0.34	1.33	0.33	0.34	0.34	0.34	0.33	0.33	0.34	0.33	1.33	0.33	0.34	0.34	0.33	2.41	0.34	0.34	0.34	0.33	0.34
T3	0.33	0.33	0.34	0.33	1.35	0.34	0.33	2.20	0.34	0.34	0.34	0.33	1.33	0.34	0.42	0.33	0.33	1.33	0.33	0.33	0.33	0.33	0.42	0.33	0.33	1.70	0.34	0.35	0.34	0.33	0.34

Note values have been rounded to two decimal places for display purpose, but ranking is done on all decimal places

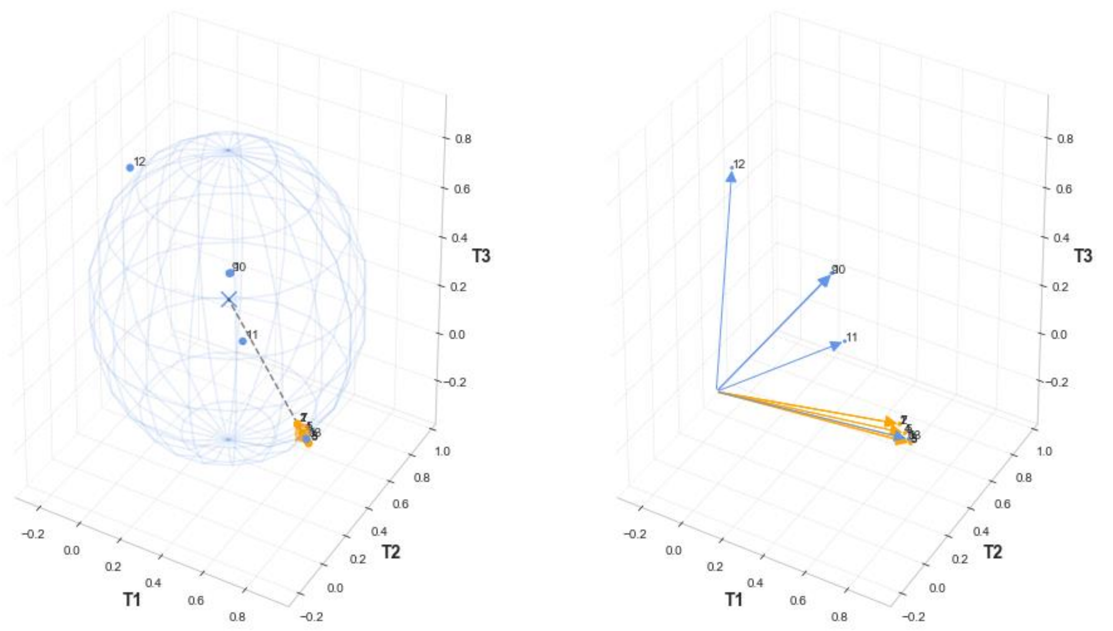
🟩 Vector weights with highest value per topic

🟩 Vector weights with 2nd or 3rd highest value per topic

🟨 Vector weights with lowest value per topic (indicate what topic is least about)

LDiA topic vectors are normally better suited to larger datasets and are not expected to perform well for a small dataset such as the 13 questions example. Usually a greater number of LDiA topics is needed compared to the number needed when using LSA, in order to achieve the same or an even better model performance. Therefore, a fair comparison cannot be done between the performance of LSA and that of LDiA in a case where the same number of topics were used for both methods. Figure 4.5 illustrates what the LDiA topic vectors look like in a 3D space.

Figure 4.5: The 13 questions represented in a 3D space using LDiA topics



a) Question Points with Category Centroids

b) Question Vectors

c) Vector Values per Topic and Question

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13
T1	0.832	0.832	0.903	0.865	0.865	0.887	0.831	0.898	0.085	0.068	0.586	0.057	0.885
T2	0.084	0.084	0.048	0.067	0.067	0.056	0.084	0.052	0.822	0.863	0.061	0.061	0.065
T3	0.084	0.084	0.048	0.067	0.067	0.056	0.084	0.050	0.093	0.069	0.353	0.883	0.051

Note that question 13 is an anomaly because the LDiA topics failed to describe it accurately, as it has been assigned to the "Tutorials" category but falls in the same LDiA vector space as those questions assigned to the "Workshops" category. An advantage of using LDiA is that vectors are more cleanly separated, as can be seen by the category centroid distance when comparing Figure 4.5a with Figure 4.3a.

In the next section a general method for deriving the sentiment of text will be described.

4.5 Sentiment Analysis

The methods described thus far help to extract the meaning of a text. Sentiment is a more general way in which to describe the overall feeling or emotion behind a text, for example if there is a sense of anger, urgency, disappointment or praise. On a very high-level, a sentiment of negative, neutral or positive can be derived, represented by a single real number between -1 and +1.

If a high number of questions are needing to be dealt with, then the sentiment of questions can be used to prioritise which questions need to be answered first. For example, questions with a negative sentiment can be given a higher priority.

There are two main approaches to sentiment analysis, namely the use of a rule-based algorithm or the use of a machine learning model learned from data. The first approach is based on heuristics where fixed rules are designed to associate a list of keywords with predefined sentiment values. The algorithm then adds up the scores for each keyword found in the document to produce an overall sentiment score. One can either create a custom rule-based algorithm by hand or use a general library like VADER (Valence Aware Dictionary for sEntiment Reasoning) which forms part of the Scikit-learn Python library and which was developed at Georgia Tech (Hutto & Gilbert, 2014).

For the second approach, labelled data is required to train a machine learning model, which will then derive the rules needed to perform the sentiment analysis. This approach requires a lot of labelled data and in fact Twitter feeds are often used as hash tags have already been added as labels for text. The machine learning algorithm mostly used for this approach is the NB classifier.

Table 4.9 shows the sentiment scores for the 13 questions example when using the VADER rule-based algorithm. The sentiment for most questions is neutral, as indicated by the value of 0.0. Question 8 has a negative sentiment due to the word “confused”. Questions 6, 9 and 11 have a positive sentiment due to the word “please”.

Table 4.9: Example of 13 questions with sentiment score

	ID	Question	Sentiment
Workshops	1	Where are workshops held?	0.00
	2	What is the venue for workshops?	0.00
	3	In which building and room do workshops take place?	0.00
	4	Where should I go to attend a workshop?	0.00
	5	What is the workshop room name?	0.00
	6	Can someone please assist? Where should I go for the workshop?	0.39
	7	Has the venue for workshops been communicated?	0.00
	8	I’m confused with workshops and tutorials. Where do workshops take place?	-0.32
Tutorials	9	Can I please change my tutorial slot?	0.32
	10	How do I alter my tut group?	0.00
	11	I need to switch to a different tut please, what should I do?	0.32
	12	Is it possible to change to a different tutorial meeting?	0.00
	13	What’s the process to swop to a new tutorial group?	0.00

Table 4.10 shows the sentiment scores for some extra questions and comments in order to illustrate how punctuation, smiley or sad faces added to text influence the final score. Note that for question 20 the word "love" has a score of 0.6369 and the word "hate" a score of -0.5719, resulting in a combined score of 0.2003.

Table 4.10: Extra questions or comments with sentiment score

ID	Question or Comment	Sentiment
14	I would like to compliment my tutor who helped me to finally understand the basics of stats. John you the best.	0.87
15	I would like to compliment my tutor who helped me to finally understand the basics of stats. John you the best!	0.88
16	I would like to compliment my tutor who helped me to finally understand the basics of stats. John you the best! :-) :-)	0.93
17	I'm not happy with my tutor, where can I complain?	-0.67
18	I just don't understand stats	0.00
19	I just don't understand stats :-(	-0.36
20	I have a love hate relationship with this course!	0.20

In the next section, an advanced method that can be used to convert natural language text to numbers using language modelling and feature learning techniques will be described.

4.6 Word Embeddings

Word embeddings are the result of a neural network model where a neighbourhood of words is used to derive a word's meaning within a sentence inside a document. The neighbourhood of words can be controlled to ensure that nearby sentences do not influence a word's meaning.

When using semantic analysis, word meaning is derived from word combinations within a document and this results in being able to identify synonyms, antonyms and words which belong together. Even better results can be achieved by controlling the neighbourhood of words.

Word embeddings are dense vectors (no zeros) represented by real numbers which enable queries and logical reasoning. These vectors represent word semantics or meaning, including literal and implied meaning. These vectors also capture the connotation of words with regards to concepts, things, places, people, etc.

Using word embeddings makes it possible to answer analogy questions, for example "*'King' is to 'man' as what is to 'woman'?*". The answer to this question is "queen", which can be written in vector math as follows:

$$\vec{v}_{queen} = \vec{v}_{king} - \vec{v}_{man} + \vec{v}_{woman} \tag{4.7}$$

Word2vec (Mikolov, Yih & Zweig, 2013c; Mikolov et al., 2013a; Mikolov et al., 2013b) is a pretrained word-embedding model released by Google, which used unsupervised learning to derive dense vector representations for more than 100 billion words, and can perform the vector math described. In 2013, Word2vec was trained on unlabelled Google news articles in which 20% of the words in the news articles had been randomly masked and the model had to learn to predict these words. The words near to the target word that was to be predicted became the input features by which the neural network was trained. It's not the predictions themselves which make Word2vec powerful, instead the power lies in the word embeddings (the weights of a hidden layer in the neural network) which the model needs to learn to enable the predictions. These word embeddings are typically between 50 and 300 dimensions when using pretrained models, but the dimensionality can be set to any number based on the problem needing to be solved, training data and the processing capability available. Generally, word embeddings capture many more word meanings than the amount that either LSA or LDiA topic vectors are capable of.

There are two approaches that can be used to train the Word2vec embeddings, namely Continuous Bag-of-Words (CBOW) where the model predicts the context of words from the current word, and continuous skip-gram where the model predicts the target word from the context of words. According to Mikolov, the skip-gram approach works well with small data sets and rare terms, due to the neural network structure which generates more training examples. However, CBOW produces higher accuracy for more frequent words, and is faster to train.

Word2vec was the first acclaimed pretrained word-embedding model, but others followed thereafter. Table 4.11 provides a summary of pretrained word-embedding models available on the open domain at the time of writing this dissertation.

Table 4.11: Summary of pretrained word-embedding models

Model Name	Date	Institution	Trained on Data From	Model Type
Word2vec	2013	Google	Google News	Context-free Unidirectional
GloVe	2014-2015	Stanford	Wikipedia	Context-free Unidirectional
fastText	2015-2018	Facebook	Wikipedia and Crawl	
ConceptNet	2017	Luminoso	Astronomy in WordNet, Wiktionary, OpenCyc and DBPedia	
ELMo	2018	Allen Institute for AI	Wikipedia and Crawl	Contextual shallow Bidirectional
WT103	2018	FastAI	Wikipedia	
ULMFiT	2018	FastAI	Wikipedia	Contextual shallow Bidirectional
BERT	2019	Google	Wikipedia and BookCorpus	Contextual deeply Bidirectional

Global Vectors for Word Representation (GloVe) was released by Stanford NLP researchers (Pennington, Socher & Manning, 2014) who developed an approach based on SVD instead of using a neural network. The advantage of using SVD is that SVD is more efficient in ensuring convergence to a global optimum. The underlying data used (taken from Wikipedia) is also more relevant to this study than the Google News used by Word2vec. Therefore, the pre-trained GloVe word vectors were selected for this study.

Figure 4.6 shows the results when comparing the word vectors for the 13 questions example using GloVe. At first, a GloVe pre-trained model of 50 dimensions was used and then compared to 100, 200 and 300 dimensions. From the comparisons it is clear that the vectors for the two question categories are more distinctly separated, as a result of more dimensions being used. Some findings are:

- With 50 dimensions, question 7, which falls into the "Workshops" category, seems closer to the tutorial questions. But when compared to 100 dimensions, question 7 moves closer to the other workshop questions.
- With 300 dimensions the spread of questions is more distinct, as can be observed by looking at the angle and label of the vectors. For example, questions 1 and 4 can be distinguished more easily when compared to 200 dimensions.
- Question 9 has a total vector magnitude which is almost thrice the value of any other question, due to the fact that it is the only question that does not have a negative value for any of the PCA topics. Table 4.12 shows the three PCA topic vectors learned by using the GloVe 200 pre-trained word vectors.

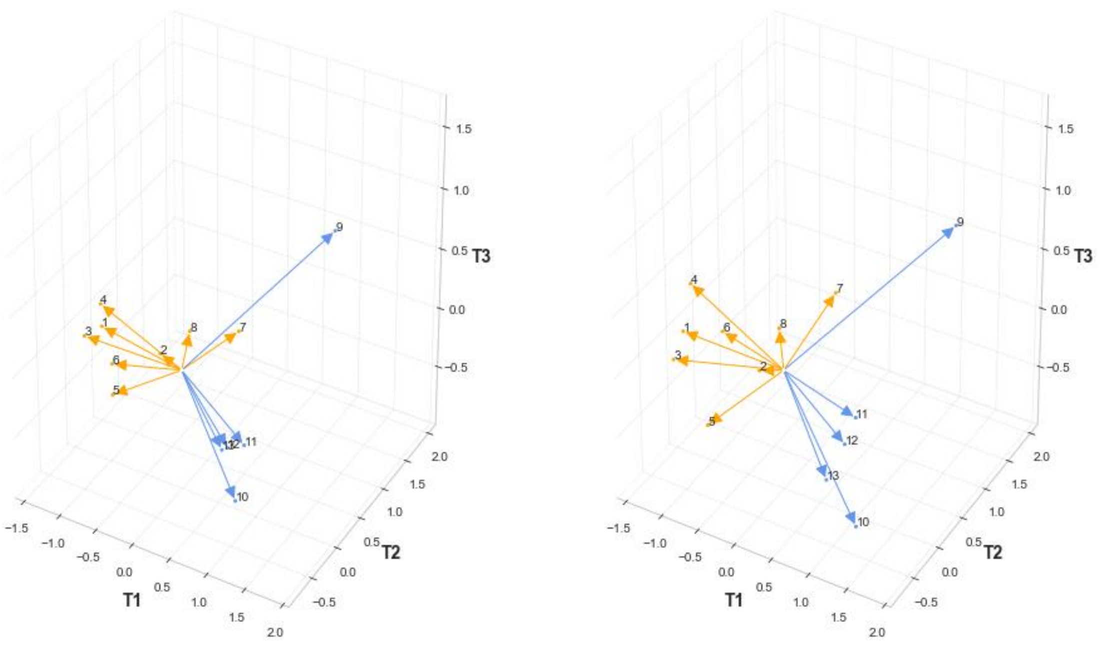
Table 4.12: PCA topics to represent the 13 questions in a 3D space with GloVe 200 vectors

		T1	T2	T3	Total
Workshops	1	-1.396	-0.356	0.495	-1.257
	2	-1.076	0.712	-0.579	-0.943
	3	-0.707	-0.771	0.032	-1.446
	4	-1.286	-0.516	0.635	-1.167
	5	-0.472	-0.636	-0.409	-1.518
	6	-0.465	-0.697	0.385	-0.777
	7	-1.081	1.989	-0.404	0.504
	8	-0.400	0.214	0.290	0.104
Tutorials	9	1.907	0.794	1.642	4.343
	10	1.367	-0.048	-0.848	0.471
	11	1.485	-0.152	-0.202	1.130
	12	1.091	-0.061	-0.420	0.610
	13	1.034	-0.470	-0.618	-0.054

Figure 4.6: The 13 questions represented in a 3D space using GloVe pre-trained word vectors

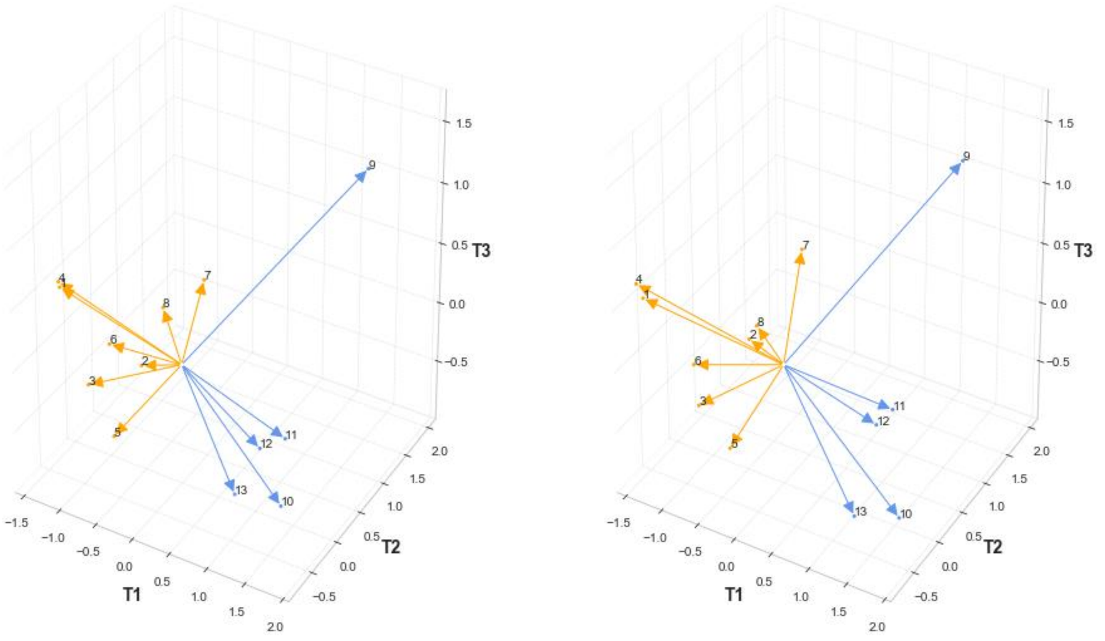
a) 50 dimensions reduced to 3

b) 100 dimensions reduced to 3



c) 200 dimensions reduced to 3

d) 300 dimensions reduced to 3



The sections up to this point have focused on converting text to numbers. In the next section methods used to predict the category of a question using the numeric features will be described.

4.7 Text Categorisation

To predict the category of a question, a text categorisation method is needed. The goal of text categorisation is to classify text into two or more categories. As mentioned before, part of the aim of this study is to classify administrative queries into a total of 16 categories.

Different classification models (also known as classifiers) can be applied to perform text categorisation and the following classifiers are commonly used in the reviewed literature: NB, LR, SVMs, Stochastic Gradient Descent (SGD), RF and neural networks. Due to the small dataset available, only the traditional supervised learning techniques are discussed, while neural networks and deep learning techniques will be out of scope for this study.

The NB classifier is normally used as the baseline model against which to compare other models because it is the simplest model. All the classifiers will aim to predict the probability of document d being in class c, as defined in equation 4.8. The collection of all classes is represented by C and the full document corpus by D.

$$\begin{gathered} Pr(c|d) \\ c \in C \\ d \in D \end{gathered} \tag{4.8}$$

In this section the methodology for the NB and LR classifiers will be described in detail, while the methodology for SVMs, SGD and RF will be summarised briefly.

4.7.1 Naive Bayes

NB is a classification algorithm based on Bayes' theorem of probability which makes an assumption of independence among predictors, in other words, that all predictors are unrelated. Predictors can be interdependent but the algorithm will still consider them independently. This assumption simplifies computation, hence the name "naive". MNB is an extension of the algorithm used for multiple classes.

In the following sections, Bayes' theorem will first be described, followed by a look at the logic behind the NB classifier, and the components needed to train a machine learning classifier.

Bayes' Theorem

Bayes' theorem was named after Reverend Thomas Bayes (Bayes & Price, 1763) who formalised the algorithm given in equation 4.9 that describes the probability of an event A based on prior knowledge of a related event B.

$$Pr(A|B) = \frac{Pr(B|A)Pr(A)}{Pr(B)} \tag{4.9}$$

For any two given events A and B, there exist two conditional probabilities which can be considered (Underhill & Bradfield, 2014):

$$Pr(A|B) = \frac{Pr(A \cap B)}{Pr(B)} \tag{4.10}$$

$$Pr(B|A) = \frac{Pr(A \cap B)}{Pr(A)} \tag{4.11}$$

From equation 4.11 the following can be derived by multiplying both sides by $Pr(A)$:

$$Pr(A \cap B) = Pr(B|A)\, Pr(A) \tag{4.12}$$

Bayes' theorem connects the two conditional probabilities of A and B by starting with equation 4.10 and substituting equation 4.12 as follows:

$$\begin{aligned} Pr(A|B) &= \frac{Pr(A \cap B)}{Pr(B)} \\ &= \frac{Pr(B|A)\, Pr(A)}{Pr(B)} \end{aligned} \tag{4.13}$$

The conditional probability of A given B can therefore be broken down into three further probabilities:

- The likelihood of B occurring given that A is true: $Pr(B|A)$
- The prior probability of observing A: $Pr(A)$
- The marginal probability of observing B: $Pr(B)$

The Naive Bayes Classifier

The idea of applying Bayesian inference to text categorisation was introduced by Mosteller & Wallace (1964) and from here the Bayesian classifier as shown in equation 4.14 was developed (Jurafsky & Martin, 2019). This classifier returns the estimated correct class $\hat{c}$ for a document d out of all possible classes $c \in C$, where $\hat{c}$ has the maximum posterior probability for d.

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}}\, Pr(c|d) \tag{4.14}$$

The intuition of Bayesian classification is using Bayes' theorem to transform equation 4.14 into the three probabilities mentioned for equation 4.13. Through substitution, the following equation can be derived:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \frac{Pr(d|c)\, Pr(c)}{Pr(d)} \tag{4.15}$$

Equation 4.15 can be simplified by dropping the denominator $Pr(d)$ because it will be the same for all possible classes and only one method is needed to select the most probable class. Therefore, it is possible to select the class that maximises the product of the likelihood of document d being of class c and the prior probability of class c.

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}}\, Pr(d|c)\, Pr(c) \tag{4.16}$$

In Section 4.1 to 4.2 it was demonstrated how a document d can be represented in terms of the words w it consists of. By substituting document d with its word features w, the following can be derived:

$$\begin{aligned} \hat{c} &= \underset{c \in C}{\operatorname{argmax}}\, Pr(w_1, w_2, \dots, w_M|c)\, Pr(c) \\ &= \underset{c \in C}{\operatorname{argmax}}\, Pr(w_1|c) Pr(w_2|c) \dots Pr(w_M|c)\, Pr(c) \\ &= \underset{c \in C}{\operatorname{argmax}}\, Pr(c) \prod_{j \in M} Pr(w_{\mathrm{j}}|c) \end{aligned} \tag{4.17}$$

This equation 4.17 for the Bayesian classifier makes two assumptions. The first is the BoW assumption where it is assumed that the position of word w is of no importance in document d. In other words, it assumes that the word “workshop” has the same effect on classification regardless of where it occurs in the document. The second assumption is commonly known as the naive Bayes assumption of conditional independence among the individual word probabilities of class c. This naive assumption allows for the probabilities $Pr(w_{\mathrm{j}}|c)$ to be multiplied together.

A final improvement that is commonly used for language modelling in order to increase computation speed and to avoid underflow, is to perform all calculations in log space (Lavrenko, 2010). Therefore, equation 4.17 is generally expressed as:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \log Pr(c) + \sum_{j \in M} \log Pr(w_{\mathrm{j}}|c) \tag{4.18}$$

Training the Naive Bayes Classifier

To train the NB classifier, the probabilities for $Pr(c)$ and $Pr(w_{\mathrm{j}}|c)$ need to be learned. The first step is to calculate the prior probabilities of each class by using the frequencies in the data. This can be done using equation 4.19 where N is the total number of documents and N_c the number of documents of class c.

$$\widehat{Pr}(c) = \frac{N_c}{N} \tag{4.19}$$

The second step is to calculate the likelihood of word w_{j} occurring given class c. This can be done using equation 4.20 by calculating the fraction of times the word w_{j} occurs among all words in all documents of class c. The $count(w_{\mathrm{j}}, c)$ is calculated by concatenating all documents of class c together and counting word w_{j} in this set. On the other hand, the $\sum_{w \in V} count(w, c)$ counts the occurrence of all words w in the complete vocabulary V (not just the words in one class c).

$$\widehat{Pr}(w_{\mathrm{j}}|c) = \frac{count(w_{\mathrm{j}}, c)}{\sum_{w \in V} count(w, c)} \tag{4.20}$$

One issue with maximum likelihood training, is that if an estimate is made for a word and class combination which did not occur in the training data then the estimated probability will be zero, regardless of any evidence of this word occurring in other classes. This is due to the naive assumptions where all probabilities are multiplied together. A solution for this issue is to simply add 1 to both the numerator and the denominator. This is also known as Laplace smoothing.

$$\begin{aligned} \widehat{Pr}(w_{\mathrm{j}}|c) &= \frac{count(w_{\mathrm{j}}, c) + 1}{\sum_{w \in V}(count(w, c) + 1)} \\ &= \frac{count(w_{\mathrm{j}}, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + V} \end{aligned} \tag{4.21}$$

However, with Laplace smoothing the estimated probability will be 1 for a word which did not occur in the training set for any of the classes. The solution for dealing with such unknown words is to identify them before testing is done and to remove them from the test document so that these words are excluded from probability calculations.

Example using Bag-of-Word Counts

One document from the 13 questions example will be used in this section to illustrate how the NB classifier can estimate the category of a document. The selected document d for this illustration is, "*I'm confused with workshops and tutorials. Where do workshops take place?*". This is a good selection to use as some words can be found in both the category of "Workshops" and in that of "Tutorials" (refer to Section 4.1 for a reminder of word distributions for the question between the two categories). The individual prior class probabilities for the two categories can be calculated using equation 4.19 as follows:

$$\widehat{Pr}(workshops) = \frac{8}{13} \qquad \widehat{Pr}(tutorials) = \frac{5}{13}$$

The likelihood probabilities can be calculated from the BoW counts as listed in Table 4.3 and for document d it is as follows: {"confuse":1, "workshop":2, "tutorial":1, "where":1, "take":1, "place":1}. Using these counts and equation 4.21, the likelihood probability calculations are:

$$\widehat{Pr}(confuse|workshops) = \frac{1+1}{35+31} \qquad \widehat{Pr}(confuse|tutorials) = \frac{0+1}{23+31}$$

$$\widehat{Pr}(workshop|workshops) = \frac{9+1}{35+31} \qquad \widehat{Pr}(workshop|tutorials) = \frac{0+1}{23+31}$$

$$\widehat{Pr}(tutorial|workshops) = \frac{1+1}{35+31} \qquad \widehat{Pr}(tutorial|tutorials) = \frac{5+1}{23+31}$$

$$\widehat{Pr}(where|workshops) = \frac{4+1}{35+31} \qquad \widehat{Pr}(where|tutorials) = \frac{0+1}{23+31}$$

$$\widehat{Pr}(take|workshops) = \frac{2+1}{35+31} \qquad \widehat{Pr}(take|tutorials) = \frac{0+1}{23+31}$$

$$\widehat{Pr}(place|workshops) = \frac{2+1}{35+31} \qquad \widehat{Pr}(place|tutorials) = \frac{0+1}{23+31}$$

The prior probability and individual likelihood probabilities per word and class can now be multiplied as follows:

$$\begin{aligned}\widehat{Pr}(workshops)\widehat{Pr}(d|workshops) &= \frac{8}{13} \times \frac{2 \times 10 \times 2 \times 5 \times 3 \times 3}{66^6} \\ &= 1.34 \times 10^{-8}\end{aligned}$$

$$\begin{aligned}\widehat{Pr}(tutorials)\widehat{Pr}(d|tutorials) &= \frac{5}{13} \times \frac{1 \times 1 \times 6 \times 1 \times 1 \times 1}{54^6} \\ &= 9.31 \times 10^{-11}\end{aligned}$$

Finally, the maximum posterior probability selected from above is 1.34×10^{-8} which results in the estimated class for document d to be "Workshops".

Application

The NB classifier can be implemented in multiple ways and the algorithms used vary between application libraries. The equation for MNB as implemented by the Scikit-learn library is:

$$\widehat{Pr}(x_j|c) = \frac{N_{cj} + \alpha}{N_c + \alpha M}$$
$$\widehat{Pr}(d|c) = \widehat{Pr}(x_1|c) \cdot \widehat{Pr}(x_2|c) \cdot \ldots \cdot \widehat{Pr}(x_M|c) \tag{4.22}$$

Where:

$$d = (x_1, x_2, \ldots, x_T)$$

$$N_{cj} = \sum\nolimits_{x \in V} x_j$$

$$N_c = \sum\nolimits_{j=1}^{M} N_{cj}$$

With N_{cj} being the number of times feature x_j is in a document of class c in the vocabulary V, and N_c is the total count of all features of class c.

The equation 4.22 is an estimation of the probability for feature x_j being of class c. Document d is made up of feature values x which can be word counts, TF-IDF or topic values, and M is the size of the vocabulary V.

MNB is suitable for classification with discrete features such as text classification. It is easy to implement and very fast to train, as little optimisation is required. It has further advantages in that it is appropriate to use when the number of features is high (Hastie, Tibshirani & Friedman, 2009) which happens when working with text, and it works exceptionally well on very small datasets or short documents (Jurafsky & Martin, 2019).

The disadvantages of using MNB are that correlated features may cause over estimation for them as they are assumed to be independent. The frequency-based probability estimate will be zero if no occurrences of a class label have been sampled, as all the probabilities are multiplied together. Furthermore, although NB is known to be a decent classifier, it is also known to be a bad estimator, so the probability outputs are not to be taken too seriously (Zhang, 2004).

The hyperparameters which can be tuned for equation 4.22 are:

- α (real number between 0 and 1) as an additive smoothing parameter. Value of 0 for no smoothing, 1 for Laplace smoothing or any value in between for Lidstone smoothing.

- Fit prior (Boolean value) to indicate whether the classifier is to learn class prior probabilities or not. If false, a uniform prior will be used.

Another popular classification model is LR, which will be introduced in the following section.

4.7.2 Logistic Regression

LR is based on linear regression, but has the advantage of being more appropriate for qualitative responses when more than two classes are involved. Instead of modelling responses directly, LR models the probability that the response belongs to a specific category.

The NB classifier described in the previous section is a generative classifier, while LR is a discriminative classifier. Different frameworks are needed to build a generative machine learning model compared to a discriminative model. A generative model, like NB, makes use of the likelihood term $Pr(d|c)$ (refer to equation 4.16) which expresses how to generate the features of a document d given that the class c is known. On the other hand, a discriminative model will attempt to directly compute $Pr(d|c)$, by learning weights for features and assigning higher weights to those features which directly improve the classifier's ability to discriminate between classes.

In the following sections linear regression will be described first, followed by a description of LR and the components needed to train a discriminative machine learning classifier. Multinomial Logistic Regression (MLR), the extension of LR for multiple categories, will also be introduced.

Linear Regression

The 13 questions example with LDiA topic features as displayed in Figure 4.5 can be used to explain how linear regression works and the challenges related to multi-class categorisation. The dummy variable approach can be used to encode a response variable as follows, in order to enable questions to be categorised as either "Workshops" or "Tutorials":

$$y = \begin{cases} 1 & \text{if workshops} \\ 0 & \text{if tutorials} \end{cases}$$

A linear regression can be fitted to this binary response and used to predict the category "Workshops" if $\hat{y} > 0.5$ and "Tutorials" otherwise. For this binary case it can be shown that $\hat{w}x$ obtained using linear regression is in fact an estimate of $Pr(workshops|x)$ (James et al., 2017). The linear regression function to calculate this probability is shown in equation 4.23 where b is the bias term (also known as the intercept) and w is the weight of feature x. The weight represents how important the feature is in predicting the category. A positive weight indicates that the feature is associated with the "Workshops" category, while a negative weight indicates that it is not associated.

$$Pr(workshops|x) = p(y = 1) = b + wx \tag{4.23}$$

The first problem with linear regression is that some of the estimates may be outside of the range of 0 and 1 (see Figure 4.7a where values to the left are less than 0), making them hard to interpret as probabilities. The second problem is that the dummy variable approach cannot be extended easily to accommodate more than two categories, therefore in this case a more appropriate method like LR is needed. This will be introduced in the following section.

Logistic Regression

LR uses the logistic function to model $p(y = 1)$ so that the output values are guaranteed to be between 0 and 1 for all values of x. The equation for the logistic function is:

$$p(y = 1) = \frac{e^{b+wx}}{1 + e^{b+wx}} \tag{4.24}$$

From the logistic function one can derive the sigmoid function as shown in equation 4.25.

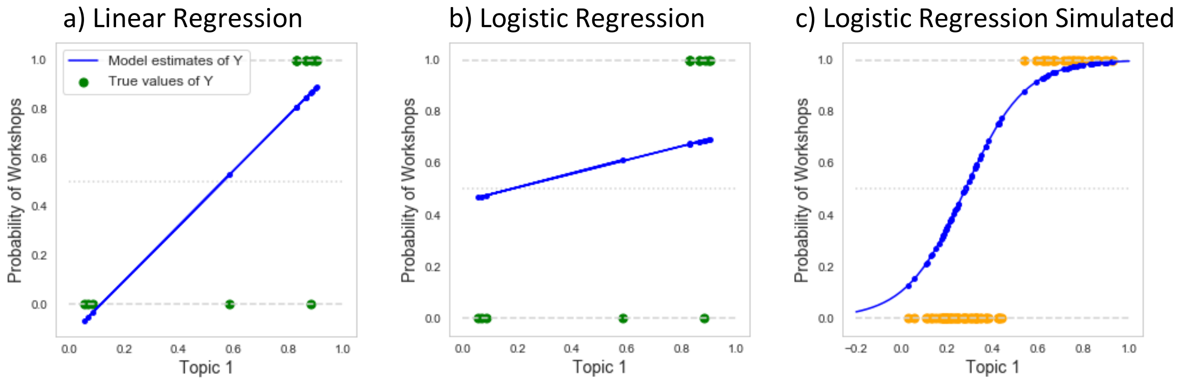
$$p(y = 1) = \frac{1}{1 + e^{-(b+wx)}} \tag{4.25}$$

To allow calculating the probability for both categories ("Workshops" and "Tutorials"), one needs to ensure that the sum of the two probabilities is equal to 1. Equation 4.26 shows how to derive the formula to model $p(y = 0)$.

$$\begin{aligned} Pr(workshops|x) + Pr(tutorials|x) &= 1 \\ p(y = 1) + p(y = 0) &= 1 \\ p(y = 0) &= 1 - p(y = 1) \\ &= 1 - \frac{1}{1 + e^{-(b+wx)}} \\ &= \frac{e^{-(b+wx)}}{1 + e^{-(b+wx)}} \end{aligned} \tag{4.26}$$

Figure 4.7 shows a comparison between linear regression and LR where, the 13 questions example has been used as input to train the models. The linear regression model on the left calculates values which are not always bounded between 0 and 1. The LR model in the middle uses the sigmoid function to ensure that calculated values are always between 0 and 1. The LR model appears as a straight line due to the small dataset of only 13 training samples. The model on the right is another LR model based on a larger simulated dataset and displays the typical “S”-shaped curve associated with the sigmoid function.

Figure 4.7: Comparing linear and logistic regression using a single LDiA topic feature



$$p(y = 1) = -0.13 + 1.128x$$

$$p(y = 1) = \frac{1}{1 + e^{0.191 - 1.115x}}$$

$$p(y = 1) = \frac{1}{1 + e^{2.142 - 7.653x}}$$

With equations 4.25 and 4.26 it is possible to use the probabilities to determine whether the estimated category of an observation is “Workshops” or not. For the test observation x' the probability $Pr(y = 1|x')$ can be calculated. If the value is more than 0.5 then the decision is yes, and if not the decision is no. The value 0.5 is called the decision boundary which the model uses to predict the category “Workshops”.

In Figure 4.7b the equation for the LR model is shown, where the learned values for the bias term b and the weight w for the first LDiA topic feature x are 0.191 and -1.115 respectively. The next section will describe the training process for the LR classifier to learn these values.

Training the Logistic Regression Classifier

Supervised learning can be used to learn the bias term b and weight w for equations 4.25 and 4.26 presented in the previous section, as the correct label y for each training observation x is known. The trained model is then able to calculate an estimate value $\hat{y}$ for the true value of y by using these learned values.

Two components are required to learn values for b and w. The first is a measure to evaluate the difference between the estimated value $\hat{y}$ and the true value of y. Instead of measuring the similarity, the distance between $\hat{y}$ and y are calculated. The cross-entropy loss function is commonly used for NLP to calculate this distance (Eisenstein, 2019). The second component required is an optimisation algorithm for iteratively updating the weights during training, with the goal of minimising the loss function. The standard optimisation algorithm for this is gradient descent.

Cross-entropy Loss Function

The goal of the trained LR classifier is to learn weights that maximise the probability of predicting the correct labels. The conditional maximum likelihood estimation is used for this task, where the parameters b and w are selected that maximize the log probability of the true y labels in the training data given the observations x. The resulting loss function is the negative log likelihood loss function, also known as the cross-entropy loss function (Jurafsky & Martin, 2019).

The probability of predicting the correct label $Pr(y|x)$, is a Bernoulli distribution for the true value of y, since there are only two discrete outcomes (1 or 0). Therefore, equation 4.27 simplifies to $\hat{y}$, if $y = 1$ and to $1 - \hat{y}$ if $y = 0$.

$$Pr(y|x) = \hat{y}^{y}(1 - \hat{y})^{1-y} \tag{4.27}$$

The cross-entropy loss function L_{CE} can be derived from equation 4.27 by flipping the sign (changing it from maximise to minimise) and taking the log of both sides, as any value which maximises a probability will also maximise the log of the probability:

$$\begin{aligned} L_{CE}(\hat{y}, y) &= -\log[\hat{y}^{y}(1 - \hat{y})^{1-y}] \\ &= -[y \log \hat{y} + (1 - y) \log(1 - \hat{y})] \end{aligned} \tag{4.28}$$

Finally, by replacing $\hat{y}$ with its definition $\hat{y} = \sigma(b + wx)$, the following can be derived:

$$L_{CE}(b, w) = -\big[y \log \sigma(b + wx) + (1 - y) \log\big(1 - \sigma(b + wx)\big)\big] \tag{4.29}$$

The negative log is a convenient loss metric as it ranges from 0 (negative log of 1) to infinity (negative log of 0), where 0 implies no loss and infinity implies infinite loss. This loss function guarantees that the probability of the correct answer is maximised, while the probability of the incorrect answer is minimised. This is because the sum of the two is equal to 1, and any increase in the one probability comes at the expense of the other, hence the loss function name of cross-entropy.

Gradient Descent

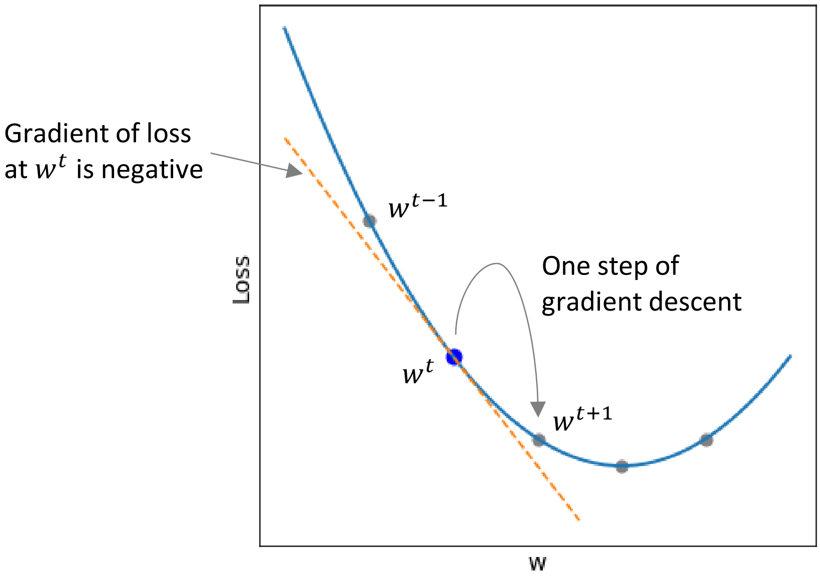
Gradient descent is the optimisation algorithm used for iteratively updating the weights during training, with the aim of minimising the loss between $\hat{y}$ and y. The loss function L_{CE} is parameterised by the weights, which in the case of LR is referred to as $\theta = b, w$. Equation 4.30 shows the gradient descent optimisation algorithm.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} L_{CE}(y_i, x_i; \theta) \tag{4.30}$$

The intuition of gradient descent can be explained using the analogy of hiking in a canyon with the goal of descending to the river at the bottom as quickly as possible, but there is heavy fog which makes visibility extremely low. One strategy to use when faced with this problem is to make a $360°$ assessment to find out where the ground is sloping the steepest, and selecting the opposite direction.

The loss function for LR is conveniently convex as shown in Figure 4.8. A convex function has just one minimum, so starting a descent from any point, one is guaranteed to find the minimum.

Figure 4.8: Gradient descent



Gradient descent starts at a randomly initialised weight w^t, assuming the loss function has the shape in Figure 4.8. Then the algorithm needs to decide in which direction to move the weight with the goal of reaching the minimum loss, either towards w^{t-1} (decreasing the weight value) or w^{t+1} (increasing the weight value). The algorithm will use the gradient of the loss function at point w^t to make this decision. Because the gradient is negative, the weight will be moved in the opposite direction, in other words increasing the weight value.

The amount by which to increase or decrease the weight value is determined by the gradient, weighted by a learning rate α as shown in equation 4.31.

$$w^{t+1} = w^t - \alpha \frac{\partial}{\partial w^t} L_{CE}(y, x; \theta) \tag{4.31}$$

If α is too small, it may take a very long time for the gradient descent algorithm to reach the minimum loss. On the other hand, if α is too big, the optimal loss may be missed and the algorithm will fail to converge. Hyperparameter tuning can be used to select the most appropriate value for α.

The iterative updating of weights can either be done for a single observation at a time or by using a batch of training observations at a time. The former is called SGD and the latter is commonly referred to as mini-batch training. An issue with using a single observation at a time is that it can result in irregular movements which can be avoided by computing the average gradient for a batch of training observations. Selecting the batch size can also be done using hyperparameter tuning.

Learning weights that match the training data too perfectly may not generalise well for any new data and can result in an overfitted model. A regularisation term $R(\theta)$ can be added to the objective function to help avoid overfitting and will penalise large weights. Equation 4.31 can be adjusted by adding the regularisation term, as well as changing the minimum loss to a maximise log probability, and by removing the $\frac{1}{N}$ term which doesn't affect the argmax. The new objective function is:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \log Pr(y_i|x_i) - \lambda R(\theta) \tag{4.32}$$

Two common ways by which the regularisation term $R(\theta)$ can be computed are L2 or L1 regularisation, also known as Ridge and Lasso Regression. L2 regularisation is a quadratic function of the weight values and gets its name from the use of the L2 norm on θ. This L2 norm, $\|\theta\|_2$, is also known as the Euclidean distance of vector θ from the zero point. If θ consists out of M weights, then $R(\theta)$ can be calculated as follows:

$$R(\theta) = \|\theta\|_2^2 = \sum_{j=1}^{M} \theta_j^2 \tag{4.33}$$

Through substitution, the L2 regularised objective function becomes:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left[\sum_{i=1}^{N} \log Pr(y_i|x_i) \right] - \lambda \sum_{j=1}^{M} \theta_j^2 \tag{4.34}$$

L1 regularisation is a linear function of the weight values and get its name from the use of the L1 norm on θ. This L1 norm, $\|\theta\|_1$, is also known as the Manhattan distance of vector θ from the zero point. If θ consists out of M weights, then $R(\theta)$ can be calculated as follows:

$$R(\theta) = \|\theta\|_1 = \sum_{j=1}^{M} |\theta_j| \tag{4.35}$$

Through substitution, the L1 regularised objective function becomes:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left[\sum_{i=1}^{N} \log Pr(y_i | x_i) \right] - \lambda \sum_{j=1}^{M} |\theta_j| \tag{4.36}$$

L2 regularisation is easier to optimise because the derivative of θ^2 is just 2θ. L1 regularisation is however more complex, as the derivative of $|\theta|$ is non-continuous at zero. The advantage of L1 regularisation is that it will cause some weights to become zero, leading to sparser weight vectors and the outcome of the dependence on fewer features. L2 regularisation will lead to smaller weights but will still depend on all features.

Only a single feature and two categories have been considered up to this point. The algorithms can easily be extended to cater for multiple features and more than two categories. MLR, also known as Softmax Regression, is used for modelling responses with more than two classes. The softmax function, a generalisation of the sigmoid function, is used to calculate the probability $Pr(y = c|x)$ of y being in each potential class $c \in C$. It takes a vector $z = [z_1, z_2, \dots, z_k]$ of k values and maps them to a probability distribution, where each output value is in the range 0 to 1, and where the sum of the values is equal to 1. The following equation shows the defined softmax function:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}; 1 \leq i \leq k \tag{4.37}$$

By using this softmax function, any input vector $z = [z_1, z_2, \dots, z_k]$ can be transformed into a vector of probabilities as follows:

$$\text{softmax}(z) = \left[\frac{e^{z_1}}{\sum_{j=1}^{k} e^{z_j}}, \frac{e^{z_2}}{\sum_{j=1}^{k} e^{z_j}}, \cdots, \frac{e^{z_k}}{\sum_{j=1}^{k} e^{z_j}} \right] \tag{4.38}$$

Finally, the probabilities for the MLR is given by equation 4.39.

$$Pr(y = c|x) = \frac{e^{b_c + w_c \cdot x}}{\sum_{j=1}^{k} e^{b_j + w_j \cdot x}} \tag{4.39}$$

The loss function for MLR is slightly different from the loss function for binary LR, due to the softmax function. The loss function is the sum of the log of the K output classes:

$$\begin{aligned} L_{CE}(\hat{y}, y) &= -\sum_{k=1}^{K} 1\{y = k\} \log Pr(y = k|x) \\ &= -\sum_{k=1}^{K} 1\{y = k\} \log \frac{e^{b_k + w_k \cdot x}}{\sum_{j=1}^{K} e^{b_j + w_j \cdot x}} \end{aligned} \tag{4.40}$$

Equation 4.40 makes uses of the binary function $1\{y = k\}$ which evaluates to 1 if the condition $y = k$ is true and to 0 otherwise.

In the next section an example will be considered to help explain how the above steps fit together.

Example using LDiA Topic Features

The 13 questions example with 3 LDiA topic features as displayed in Figure 4.5 will be used in this section to illustrate how the LR classifier can be trained and used to estimate the category of a document. The gradient descent algorithm, in combination with the cross-entropy loss function, is used to learn values for the bias and weight terms.

Assume the initial bias and weights in θ^0 are all set to 0 and the learning rate α is 0.1:

$$b = w_1 = w_2 = w_3 = 0$$
$$\alpha = 0.1$$

Starting with the first training record the following values can be observed:

$x_1 = 0.832$ (1st LDiA topic value)
$x_2 = 0.084$ (2nd LDiA topic value)
$x_3 = 0.084$ (3rd LDiA topic value)
$y = 1$ (encoded value for category “Workshops”)

The first update step for θ requires the calculation of the gradient, multiplied by the learning rate:

$$\theta^{t+1} = \theta^t - \alpha \nabla_\theta L_{CE}\left(f\left(x^{(i)}; \theta\right), y^{(i)}\right)$$

The gradient vector has four dimensions due to the four parameters b, w_1, w_2 and w_3. The computation of the gradient is as follows:

$$\nabla_{w,b} = \begin{bmatrix} \dfrac{\partial L_{CE}(b,w)}{\partial b} \\ \dfrac{\partial L_{CE}(b,w)}{\partial w_1} \\ \dfrac{\partial L_{CE}(b,w)}{\partial w_2} \\ \dfrac{\partial L_{CE}(b,w)}{\partial w_3} \end{bmatrix} = \begin{bmatrix} \sigma(b + w \cdot x) - y \\ (\sigma(b + w \cdot x) - y)x_1 \\ (\sigma(b + w \cdot x) - y)x_2 \\ (\sigma(b + w \cdot x) - y)x_3 \end{bmatrix} = \begin{bmatrix} \sigma(0) - 1 \\ (\sigma(0) - 1)x_1 \\ (\sigma(0) - 1)x_2 \\ (\sigma(0) - 1)x_3 \end{bmatrix} = \begin{bmatrix} -0.5 \\ -0.5x_1 \\ -0.5x_2 \\ -0.5x_3 \end{bmatrix} = \begin{bmatrix} -0.5 \\ -0.416 \\ -0.042 \\ -0.042 \end{bmatrix}$$

Now that the gradient is known, the new parameter vector θ^1 can be calculated by moving θ^0 in the opposite direction from the gradient:

$$\theta^1 = \begin{bmatrix} b \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} - \alpha \begin{bmatrix} -0.5 \\ -0.416 \\ -0.042 \\ -0.042 \end{bmatrix} = \begin{bmatrix} 0.05 \\ 0.042 \\ 0.004 \\ 0.004 \end{bmatrix}$$

After the first training iteration, the bias and weights have shifted from all zeros to $b = 0.05$, $w_1 = 0.042$, $w_2 = 0.004$ and $w_3 = 0.004$ respectively. The parameter vector θ will be updated until the gradient reaches a value of zero or becomes positive. For this example, the vector updates will stop at iteration eight, after which the process will be repeated for the next training example until all 13 examples have been used.

In this simplified example the sigmoid function with no regularisation was used. The final model for the LR classifier is:

$$p(y = 1) = \frac{1}{1 + e^{-3.246 + 6.81x_1 - 2.88x_2 - 7.176x_3}}$$

$$p(y = 0) = \frac{e^{-3.246 + 6.81x_1 - 2.88x_2 - 7.176x_3}}{1 + e^{-3.246 + 6.81x_1 - 2.88x_2 - 7.176x_3}}$$

Application

The LR classifier can be implemented in multiple ways and the algorithms used vary between application libraries. The equation for MLR as implemented by the Scikit-learn library is:

$$\hat{P}(d|c) = \frac{e^{b_c + w_c \cdot x}}{\sum_{k=1}^{K} e^{b_k + w_k \cdot x}} \tag{4.41}$$

The optimisation algorithm is:

$$\min_{w,b} \left\{ \frac{1 - p}{2} w^T w + p\|w\|_1 + R \sum_{i=1}^{N} \log(\exp\left(-y_i\left(X_i^T w + b\right)\right) + 1) \right\} \tag{4.42}$$

With y being a one versus all transformation of the class label, and where p and R are hyperparameters.

The advantages of LR are that the model is easy to interpret and the Lasso regularisation performs feature selection causing insignificant features to be discarded. When compared to NB, it is more robust when dealing with correlated features and will also assign a more accurate probability.

Disadvantages are that it relies on all features and their dimensionality to be predefined. It cannot find a solution in higher dimensions beyond the number of features provided. Overfitting can also occur easily due to the logit model used.

The hyperparameters which can be tuned for equation 4.42 are:

- p (real number between 0 and 1) for controlling the strength of L1 vs. L2 regularisation. A value of 0 is used for L2 regularisation, 1 for L1 regularisation or any value in between for Elastic-Net regularisation.
- R as a regularisation parameter. No regularisation amounts to setting R to a very high value.
- Which solver to use in order to minimise the cost function. Options are “liblinear”, “newton-cg”, “lbfgs”, “sag” or “saga”.
- A tolerance for stopping criteria for solvers. Training iterations will stop as soon as the gradient is within the set tolerance. Default value is 0.0001.
- If weights should be associated with classes. The “balanced” mode uses the training class labels to automatically adjust weights inversely proportional to class frequencies in the training data as $\frac{N}{K \times N_c}$, where N is the number of samples, K the number of classes and N_c the number of samples within class c.
- Dual or primal formulation (Boolean value). Dual formulation is only implemented for L2 regularisation using the “liblinear” solver. The Scikit-learn documentation recommends setting dual=False when the number of samples is more than the number of features.

In the next section three other classification models will be introduced briefly.

4.7.3 Other Classification Models

The NB and LR classifiers have been described in detail in the previous sections. There are, however, many other classifiers which can be considered for text categorisation, for example SVMs, SGD and RF, which are often mentioned in the literature. In this section each of these will be described briefly.

Support Vector Classifier

Support Vector Classifier (SVC) is an algorithm which forms part of SVMs, and extracts the best possible hyperplane (or set of hyperplanes) for separating classes into a higher dimension feature space. It is based on feature expansion, using kernels which use the inner products between all pairs of training observations, as well as finding the maximal margin classifier (Stitson et al., 1996).

The algorithm for SVC as implemented by the Scikit-learn library is used to solve the following:

$$\min_{\alpha}\left\{\frac{1}{2}\alpha^T Q\alpha - e^T\alpha\right\} \tag{4.43}$$
$$\text{subject to } y^T\alpha = 0, 0 \leq \alpha_j \leq R, j = 1, \ldots, M$$

The decision function is:

$$\operatorname{sgn}\left(\sum_{j=1}^{M}\alpha_j y_j K\left(x_j, x_{j'}\right) + b\right) \tag{4.44}$$

Where:

- α is a vector for slack values for the support vectors.
- y is a one-versus-one transformation of the class labels.
- e is a vector of all 1s.
- R as a regularisation term and the upper bound for α values, which aims at maximising the decision function's margin. The penalty is a squared L2 penalty with $R > 0$. The parameter is for the total amount of slack allowed for observations outside of the margin for the decision boundary between classes.
- Q is an M by M positive semidefinite matrix, $Q_{jj'} \equiv y_j y_{j'} K\left(x_i, x_{j'}\right)$.
- K is the kernel function with $K\left(x_j, x_{j'}\right) = \tau\left(x_j\right)^T \tau\left(x_{j'}\right)$. Options for the kernel function are linear, polynomial, radial basis, sigmoid, precomputed or any other callable function.
- τ is a function which maps training vectors to a higher dimensional space.
- b is the intercept.

The hyperparameters that can be tuned are the regularisation term R, the selection of a kernel function and a tolerance for stopping criteria. Training iterations will stop as soon as the gradient is within the set tolerance (default value is 0.001). When training an SVM with the radial basis kernel function, two parameters must be considered: *C* and *gamma*. *Gamma* defines how far the influence of a single training example reaches, with low values indicating "far" and high values indicating "close". The class weight *C* can be set for unbalanced classes.

The advantages of using SVM and SVC are that kernel functions allow for finding an efficient solution in higher dimensions beyond the number of features present and they work well for a high number of dimensions. The model remains effective even if the number of dimensions is greater than the number of samples. It is memory efficient as it uses a subset of training points in the decision function (only the support vectors are used). Lastly, it is versatile because of its ability to use different kernel functions; even custom kernel functions can be used.

The disadvantages of using SVM and SVC are that the estimation of probabilities is not part of the algorithm and therefore more effort to estimate is required. It also does not perform feature selection as with Lasso for LR, which results in all features being used, and this makes model interpretation difficult.

Stochastic Gradient Descent

SGD is itself not a classifier, but it is a method for optimising a loss function and can be applied to linear classifiers (or regressors). The SGD classifier is an estimator that implements regularised linear models with SGD learning. The true gradient of the loss is estimated by using a single training sample at a time, and the model is updated along the way with a decreasing learning rate.

The algorithm for SGD as implemented by the Scikit-learn library is minimising the regularised training error given by:

$$E(b,w) = \frac{1}{N}\sum_{i=1}^{N} L\big(y_i, f(x_i)\big) + \rho R(w) \tag{4.45}$$

The linear scoring function to learn is:

$$f(x) = b + w^T x \tag{4.46}$$

The update rule for learning the weights w is:

$$w \leftarrow w - \alpha\left(\rho\frac{\partial R(w)}{\partial w} + \frac{\partial L(y_i, b + w^T x_i)}{\partial w}\right) \tag{4.47}$$

Where:

- E is the error function.
- L is the loss function, with options of Hinge, Perceptron, Modified Huber, Log, Least-Squares, Huber or Epsilon-Insensitive.
- $(x_1, y_1), \dots, (x_N, y_N)$ are the training samples, with $x_i \in \boldsymbol{R}^m$ and $y_i \in \{-1, 1\}$.
- R is the regularisation term that penalises model complexity. Options are L1 norm, L2 norm or Elastic Net.
- b is the intercept, with $b \in \boldsymbol{R}$.
- w are the weights, with $w \in \boldsymbol{R}^m$.
- ρ is a constant that multiplies the regularisation term, with $\rho > 0$.

- α is the learning rate which controls the step-size in the parameter space. The default value is $\alpha^{(t)} = \frac{1}{\rho(t_0+t)}$, with t being the time step and the determining of t_0 being based on a heuristic proposed by Léon Bottou (Bottou, 1998).
- To only make predictions, the sign of $f(x)$ needs to be considered.

The hyperparameters that can be tuned are the loss function L, the regularisation term R, the constant ρ to increase the regularisation term, the learning rate α and lastly a tolerance for stopping criteria. Training iterations will stop as soon as the gradient is within the set tolerance (default value is 0.001).

The advantages of using SGD are that it is a simple and efficient algorithm for discriminative learning of linear classifiers, and it is easy to implement. It can easily scale to problems with more than 10^5 training samples and more than 10^5 features, given that the data is sparse.

The disadvantages of using SGD are that it is dependent on the number of iterations required to train the model, as well as on the number of hyperparameters to be tuned, and it is sensitive to feature scaling.

Random Forests

RF is an ensemble algorithm where several decision trees are built using bootstrapped training examples and a majority vote for the final classification results is taken. While building the decision trees, each time that a split in a tree is considered, a random sample of m features is selected out of the full set of M features (where $m \approx \sqrt{M}$) as split candidates. This allows for trees to be decorrelated, making the final result less variable and more reliable (Breiman, 2001).

The main hyperparameters that can be tuned are the number of trees in the forest and the size of the random subsets of features to consider when splitting a node. In addition to this, the maximum depth of the trees or the minimum number of samples required to be at a leaf node can be set.

The advantages of using RF are improved accuracy compared to using a single tree or bagged trees, due to the removal of bias when using multiple trees and the presence of a different subset of features for each tree. The importance of features can be measured by using the average decrease in the Gini index that occurs as a result of the splits over a given feature. More important features will have a bigger decrease. Lastly, RF works well on any type of feature or any combination of features, both categorical and numerical, as well as when there are values missing.

The disadvantages of using RF are that model interpretation is lost due to the use of multiple trees which are not as easy to interpret as a single tree. The model is also complex and requires more computation resources and more time to train.

4.8 Summary

In this chapter, the methodology used to convert natural language text to numbers so that it can be processed by a machine, while still preserving the meaning of text, was explained. The building blocks of NLP were introduced, starting with BoW, followed by the TF-IDF technique, semantic analysis and word embeddings.

Text categorisation methods, which are needed as the first step in automatically answering questions, were explained, followed by a look at the second step, which involves identifying the most similar past question. Here, the cosine similarity method was discussed.

A toy dataset of 13 questions was used to explain the methodology. In this case the vocabulary consisted of 31 words learned from these questions. In practice, however, the vocabulary size can easily grow to a number as large as 10 000 words. Using BoW or one-hot word vectors results in sparse vectors, as each question will only contain a fraction of the total number of words in the vocabulary, but the dimension of the vectors will be equal to the vocabulary size. This high dimensionality leads to the “curse of dimensionality” which makes it difficult to find the similarity between vectors, as they get exponentially farther away from one another as the dimensionality increases. Table 4.3 showed the one-hot encoding of the 13 questions example and illustrated the sparse matrix effect.

In the next chapter the methodology introduced in Chapter 4 will be tested on the two datasets (the base data and the extra RTT data) for answering administrative queries for the STA1000 course, to determine how feasible it would be to build a QA system with these datasets.

5. Results and Discussion

This chapter consolidates the approach used for training and testing models, the final model results and a discussion of the findings of this study. The results are structured based on three phases that were applied to the QA system. Firstly, the success of predicting the category of a question was tested. Secondly, the success of finding the most similar past question was tested. Thirdly, the success of the extracted answer was tested. Additionally, an evaluation was done to determine the effectiveness of using sentiment analysis to prioritise questions, as well as to evaluate the success of using the extra RTT data.

The questions posed by this study and discussed in this chapter are the following: which features preserved the meaning of questions the best? Which classification model performed the best in predicting the question category? Was it a good idea to apply augmentation to the base dataset to increase the number of training samples? How can the best past answer be selected for a new question? Can sentiment analysis be used to help prioritise the answering of questions? Was it possible to use the available data to create a QA system which can automatically answer administrative queries with reasonable results? Can a simulator be built to demonstrate and test the QA system? What lessons were learned?

5.1 Training and Testing Approach

In order to create separate training and testing datasets, each of the 16 question categories were considered individually, and an 80/20 stratified split of randomly selected observations were applied within each category. The index values for these training and testing datasets were saved to ensure the same observations could be used consistently during the process of training and testing models. This approach ensured that all question categories were represented in both the training and testing datasets, thereby avoiding the risk that the categories with few observations (e.g. “Winter/ Summer Term” with 6 observations and “Content” with 11 observations) may otherwise be excluded in the testing dataset, if random sampling was applied on the dataset as a whole. By using this approach, the category imbalances, as described during data labelling (refer to Section 3.3), were also dealt with. Another advantage of using this approach is that the performance of all models could be compared in a fair manner for the 16 question categories. To tune the hyperparameters for the different models during the training phase, the Scikit-learn library for grid search with ten-fold cross validation on the training dataset was used (Pedregosa et al., 2011).

Table 5.1 shows a summary with the sizes of the available data, split between the training and testing datasets, as well as a split between the base data and the extra RTT data. Note that only the training dataset was increased with the extra RTT data, as a decision was made to do testing only on the base data to allow for consistent comparison of all results, and to determine if the use of RTT produced better models or not.

Table 5.1: Summary of training and testing dataset sizes

	All data	Training data	Testing data
Base data (744 Questions)	744	587	157
Base + extra RTT data (6 614 Questions)	6 614	6 457	157

A systematic approach was taken for testing, where 5 model types and 30 feature types were combined. The five model types used were: MNB, MLR, SVM, SGD and RF. The 30 feature types were a combination of standard word terms, semantic analysis topic vectors, engineered features and GloVe word embeddings.

The next section will describe these feature types and the resulting models in more detail.

5.2 Results

The results of predicting the category of a question is presented first, followed by the results for finding the most similar past question. The F1-measure with macro-averaging (described in Section 2.4) was used to evaluate the success of predicting the category of a question. Human evaluation was used to evaluate the success of selecting the most similar past question. A binary rating for human evaluation was used, instead of using the five-point Likert scale, because a clear distinction between correct or incorrect answers were required.

5.2.1 Prediction of Question Categories

A total of 146 unique models were compared as the result of combining 5 model types and 30 feature types. The 4 feature types which had negative values were not used for the MNB model because it assumes a multinomial distribution. The top 10 models for predicting question categories are listed in Table 5.2 and the full list of all models can be found in Appendix A.6, with a summary of the 30 feature types in Table A.3.

Table 5.2: Top 10 models for predicting the question category

				F1-measure		
Model	**Feature Type**	**# Samples**	**# Features**	**Train**	**Test**	**Rank**
MLR	TF-IDF unigrams + bigrams	587	5 688	0.978	0.848	1
MLR	LSA TF-IDF unigrams (probabilities)	587	55	0.844	0.844	2
MLR	BoW counts	587	1 101	0.934	0.841	3
SGD	TF-IDF unigrams (SGD with Least-Squares loss)	587	1 101	0.951	0.835	4
MLR	LSA TF-IDF unigrams (raw values)	587	55	0.886	0.831	5
SGD	TF-IDF unigrams + bigrams (Modified Huber loss)	587	5 688	0.992	0.825	6
SGD	BoW counts (Modified Huber loss)	587	1 101	0.942	0.825	7
MLR	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.783	0.825	8
SGD	TF-IDF unigrams + bigrams + trigrams (Hinge loss)	587	10 972	0.998	0.821	9
MNB	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.723	0.820	10

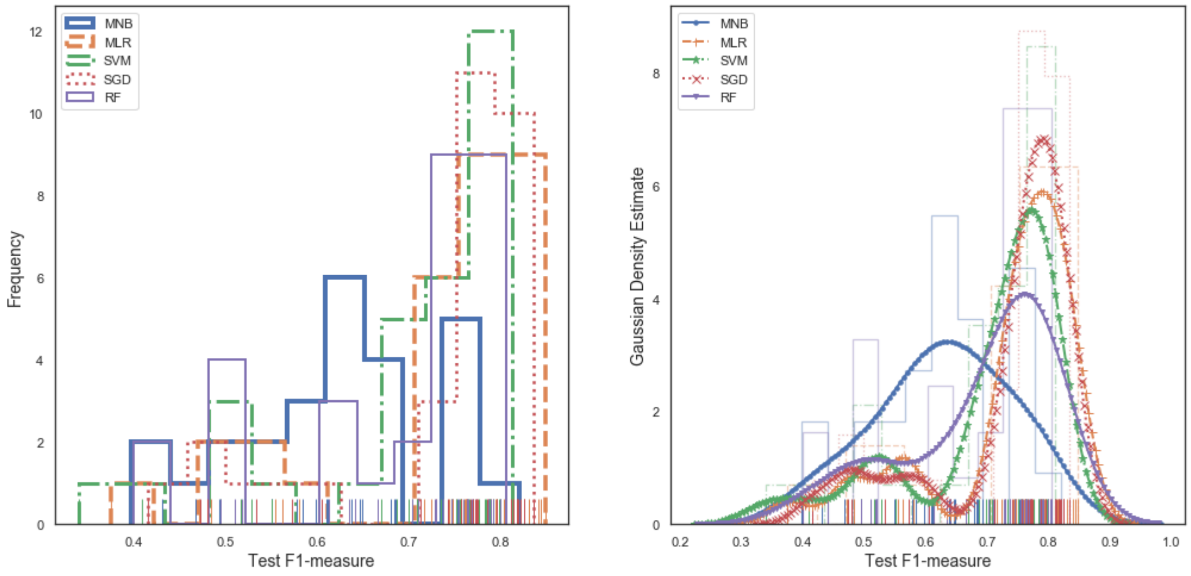
The MLR model with TF-IDF unigram and bigram features (built on the smaller base dataset) performed the best, with a test F1-measure of 84.8%. The MLR model type occurred five times in the top 10 and also ranked second and third. Only three of the five model types occurred in the top 10 list; second and third to MLR, the SGD model type occurred four times, and the MNB model type occurred once. Table 5.3 lists all 5 model types with the best performing feature types and how they ranked among the 146 models.

Table 5.3: Summary of five model types with best performing feature types

				F1-measure		
Model	**Feature Type**	**# Samples**	**# Features**	**Train**	**Test**	**Rank**
MLR	TF-IDF unigrams + bigrams	587	5 688	0.978	0.848	1
SGD	TF-IDF unigrams (SGD with Least-Squares loss)	587	1 101	0.951	0.835	4
MNB	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.723	0.820	10
SVM	LSA BoW unigrams	587	91	0.966	0.812	16
RF	TF-IDF unigrams + bigrams	587	5 688	1.000	0.806	21

When looking at the top 20% of the 146 models, the model types that occurred the most in order of best performance are: MLR (11 times), SGD (9 times), RF (5 times), SVM (4 times) and MNB (once). Figure 5.1 shows a comparison of the five model types using the test F1-measures with frequencies and Gaussian density estimates. The histograms are useful to view the discrete data points for highest test F1-measures per model type, while the Gaussian density functions are useful to see a continuous estimate for the test F1-measures per model type.

Figure 5.1: Model types with histograms and Gaussian density functions for the test F1-measures



In Figure 5.1, the highest Gaussian density estimate is reached by the SGD model type, and in Table 5.4 the SGD model type also reached most of the highest descriptive statistic values (as marked in green). From these observations it can be concluded that the SGD model type is the most versatile classifier. The inverse can also be observed; that the MNB model type demonstrates the worst performance (the lowest descriptive statistic values have been marked in light orange).

Table 5.4: Model types with descriptive statistics for test F1-measures

	Test F1-measure Statistics								
Model	**Count All**	**Count in Top 20%**	**Min**	**Mean**	**Max**	**Std**	**P25**	**P50**	**P75**
MNB	26	1	0.398	0.628	0.820	0.110	0.558	0.640	0.691
MLR	30	11	0.375	0.730	0.848	0.124	0.729	0.762	0.810
SVM	30	4	0.340	0.701	0.812	0.126	0.689	0.753	0.787
SGD	30	9	0.416	0.732	0.835	0.118	0.733	0.773	0.813
RF	30	5	0.400	0.691	0.806	0.120	0.637	0.737	0.781

An analysis of the 30 feature types revealed that the TF-IDF unigram and bigram feature type performed the best and was also the most versatile, followed by BoW counts and LSA TF-IDF unigrams with extra RTT data (probabilities). The worst performing feature types were LDiA BoW unigrams and TF-IDF bigrams.

The standard LSA feature values (referred to as “raw values” in this study) can be negative and therefore cannot be used as input for the MNB classifier. For this reason, an additional feature type was derived where the standard LSA feature values were converted into probabilities.

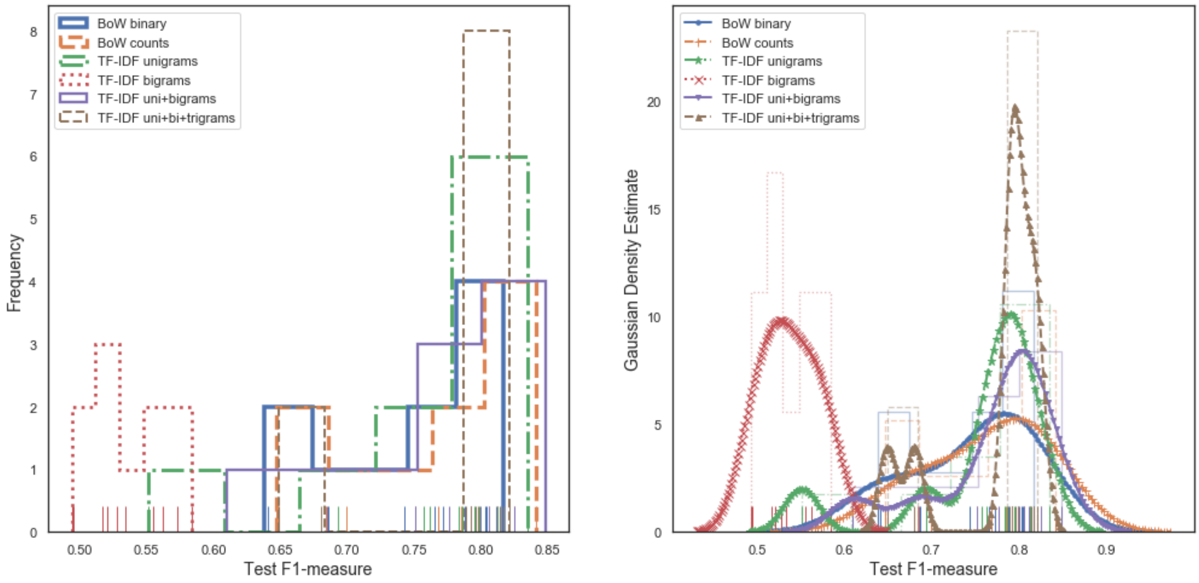
The top 10 feature types with their best performing models are listed in Table 5.5 (the full list of 30 feature types can be found in Appendix A.6).

Table 5.5: Top 10 feature types with best performing models

			Best	F1-measure		
Feature Type	**# Samples**	**# Features**	**Model**	**Train**	**Test**	**Rank**
TF-IDF unigrams + bigrams	587	5 688	MLR	0.978	0.848	1
LSA TF-IDF unigrams (probabilities)	587	55	MLR	0.844	0.844	2
BoW counts	587	1 101	MLR	0.934	0.841	3
TF-IDF unigrams	587	1 101	SGD	0.951	0.835	4
LSA TF-IDF unigrams (raw values)	587	55	MLR	0.886	0.831	5
LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	MLR	0.783	0.825	8
TF-IDF unigrams + bigrams + trigrams	587	10 972	SGD	0.998	0.821	9
TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	SGD	0.977	0.817	12
BoW binary values	587	1 101	SGD	0.947	0.817	13
Engineered features	587	61	SGD	0.911	0.813	15

The TF-IDF bigram feature type performed badly because the derived terms were too specific and did not generalise well across all questions. On the other hand, combining the TF-IDF unigram and bigram terms worked well because it utilizes the individual word occurrences (unigrams) as well as word combinations (bigrams) that allow for identifying negation. A comparison of BoW and TF-IDF feature types are shown in Figure 5.2.

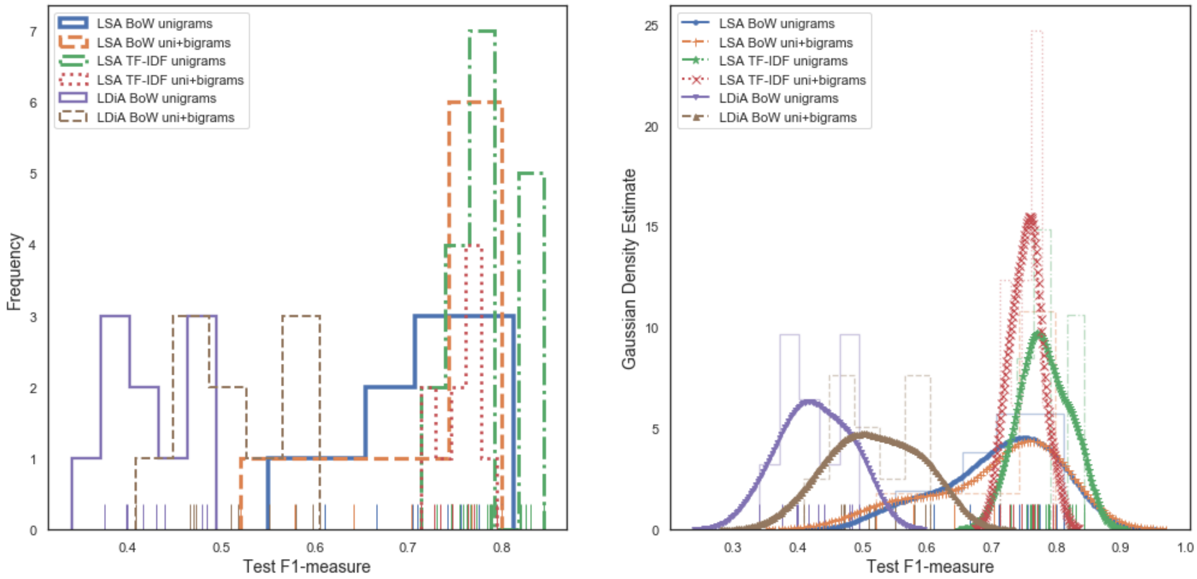
Figure 5.2 BoW and TF-IDF feature types with histograms and Gaussian density functions



The BoW feature types performed well with test F1-measures consistently above 0.638, reaching the ranked position of third out of the 146 models with BoW counts. The TF-IDF feature types had a wider range of values with the lowest being 0.494 for TF-IDF bigrams, however, the top rank (0.848) was achieved with the TF-IDF unigram and bigram features.

LDiA works well for solving topic modelling problems for large datasets and tends to outperform LSA when it comes to identifying topics that are easy to interpret by humans. Compared to LSA, LDiA generally associates more of the available topics to a document. In addition to this, all LDiA topic values will be positive while LSA values can be negative, thereby indicating unrelatedness. LSA, however, is better at reducing dimensionality and therefore outperformed the LDiA features when used to build a question category classifier. Details of how the number of LSA and LDiA topics were selected can be found in Appendix A.5. Figure 5.3 shows a comparison of the LSA and LDiA feature types.

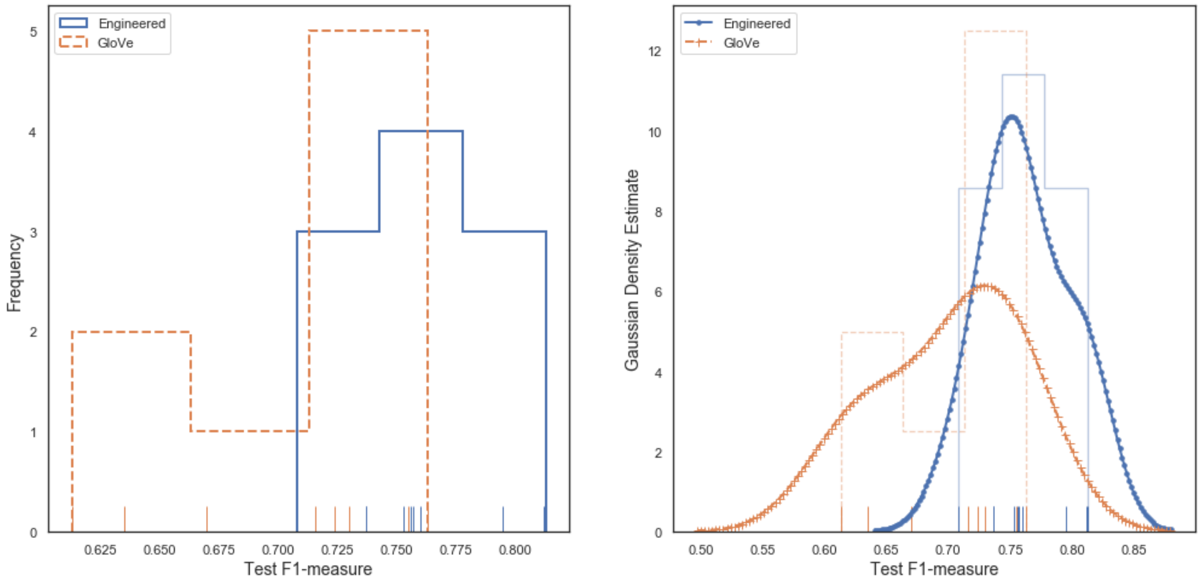
Figure 5.3: LSA and LDiA feature types with histograms and Gaussian density functions



The maximum test F1-measure for all LDiA features was 0.605, which is lower than what the minimum test F1-measure for all BoW features was. The best model that could be built with LDiA features only ranked 113th out of all 146 models, while the best LSA model ranked second. The last 13 models ranked were all based on LDiA features. Further analysis is needed to understand why the LDiA features performed so badly.

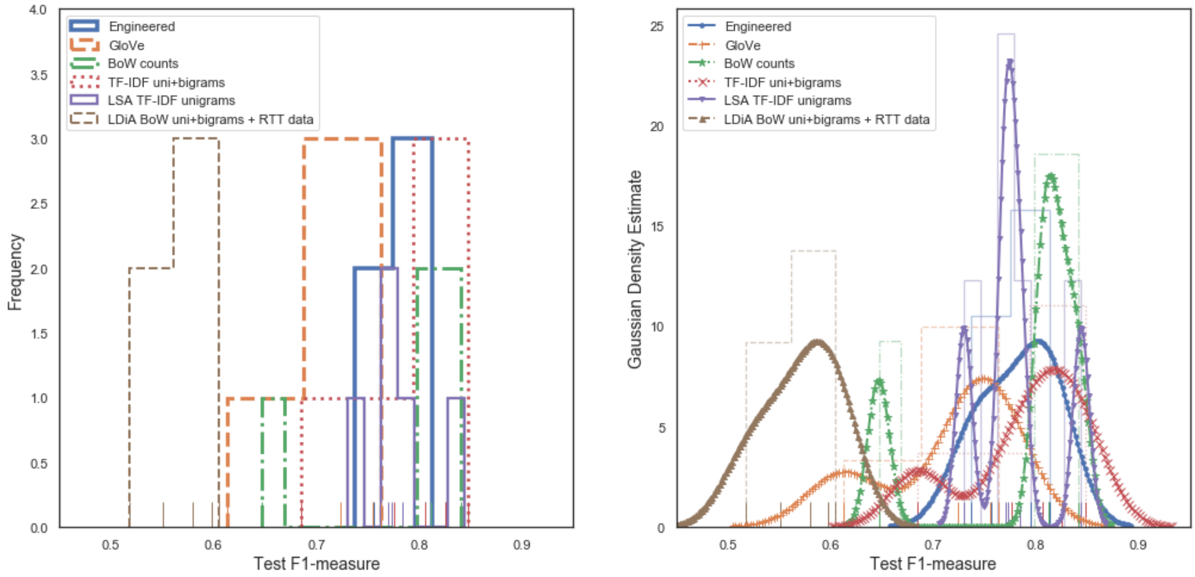
Figure 5.4 shows a comparison of the engineered and GloVe feature types. The engineered features were described in Section 3.6 and the GloVe features in Section 4.6. The engineered features performed well and the best model built using these features ranked in the 15th position. The GloVe features underperformed based on word-embedding expectation, the best model ranking only in the 60th position.

Figure 5.4 : Engineered and GloVe feature types with histograms and Gaussian density functions



The engineered feature type consisted out of 61 individual features and performed better than the GloVe feature type that consisted out of 300 individual features. Figure 5.5 shows a comparison done for the best performing feature types per category of engineered, GloVe, BoW, TF-IDF, LSA and LDiA features.

Figure 5.5: Summary of feature types with histograms and Gaussian density functions



The feature type summary in Figure 5.5 reveals how badly the LDiA features performed, as well as the difference between the engineered and GloVe feature types when compared with the BoW, TF-IDF and LSA features. Table 5.6 shows the descriptive statistics for the feature type summary where the feature types have been ordered by their maximum test F1-measure values.

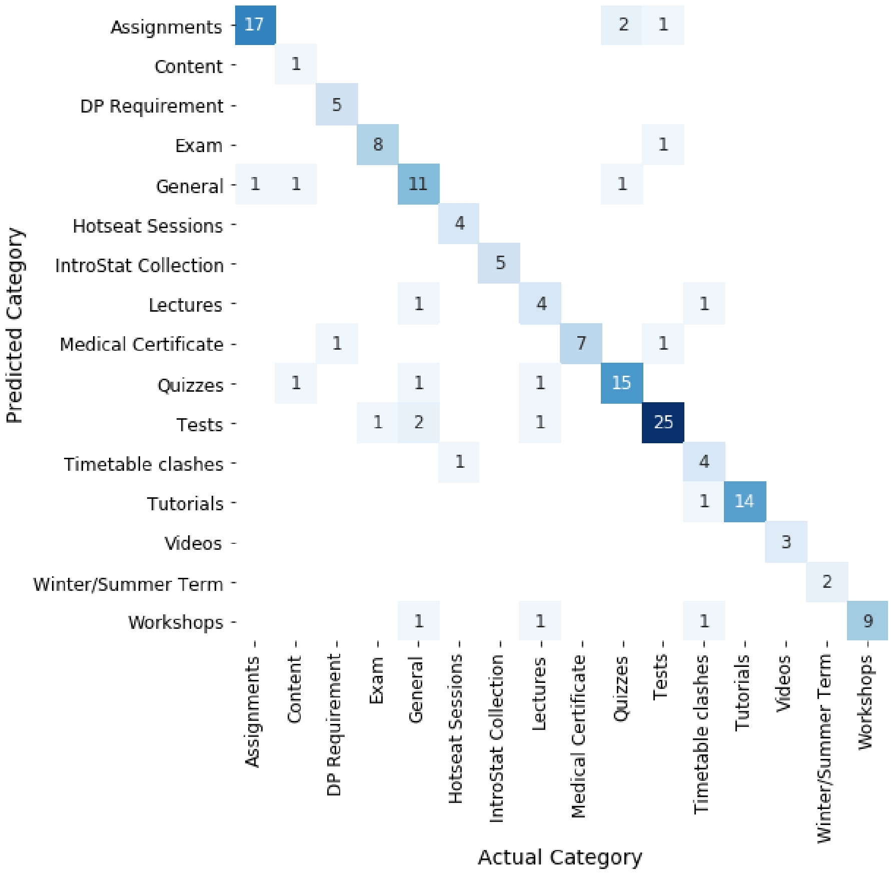
Table 5.6: Feature type summary with descriptive statistics for test F1-measures

	Test F1-measure Statistics							
Feature Type	**Count**	**Min**	**Mean**	**Max**	**Std**	**P25**	**P50**	**P75**
TF-IDF unigrams + bigrams	5	0.685	0.788	0.848	0.063	0.776	0.806	0.825
LSA TF-IDF unigrams (probabilities)	5	0.730	0.780	0.844	0.041	0.770	0.773	0.784
BoW counts	5	0.647	0.786	0.841	0.079	0.806	0.812	0.825
Engineered	5	0.737	0.783	0.813	0.034	0.756	0.795	0.812
GloVe	4	0.613	0.714	0.763	0.069	0.696	0.740	0.757
LDiA BoW uni+bigrams + RTT data	5	0.518	0.570	0.605	0.036	0.551	0.580	0.598

The engineered features had the best performance if the minimum test F1-measure needed to be the highest, or if the standard deviation of the test F1-measure needed to be at a minimum. The performance difference, based on the maximum test F1-measure, between the engineered features and the top ranked model with TF-IDF features was only 3.5%. The BoW features had the best percentile values but also had the highest standard deviation of the test F1-measure values.

Figure 5.6 shows the question category prediction results of the test data in the form of a confusion matrix, using the best performing model (MLR with TF-IDF unigram and bigram features). Each column represents the instances in an actual class, while each row represents the instances in a predicted class. The diagonal represents the true positives, in other words the number of class instances correctly predicted to be in that class. All values outside of the diagonal are errors (the false negatives).

Figure 5.6: Question category prediction results of the test data using the best MLR model



Predicted Category

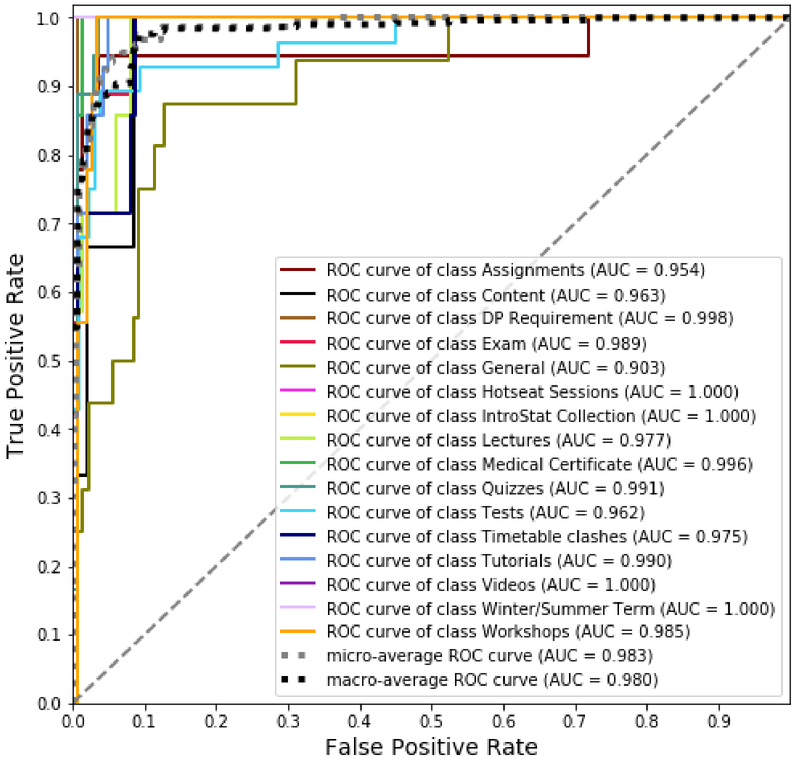
Actual Category

	Assignments	Content	DP Requirement	Exam	General	Hotseat Sessions	IntroStat Collection	Lectures	Medical Certificate	Quizzes	Tests	Timetable clashes	Tutorials	Videos	Winter/Summer Term	Workshops
Assignments	17									2	1					
Content		1														
DP Requirement			5													
Exam				8							1					
General	1	1			11					1						
Hotseat Sessions						4										
IntroStat Collection							5									
Lectures					1			4				1				
Medical Certificate			1						7		1					
Quizzes		1			1			1		15						
Tests				1	2			1			25					
Timetable clashes						1						4				
Tutorials												1	14			
Videos														3		
Winter/Summer Term															2	
Workshops					1			1				1				9

There are only 6 of the 16 categories for which no errors occurred, namely “IntroStat Collection”, “Medical Certificate”, “Tutorials”, “Videos”, “Winter/Summer Term” and “Workshops”. All these categories, with the exception of “Tutorials”, had less than the average number of test observations, while “Videos” and “Winter/Summer Term” had the least test observations of all 16 categories. The category with the worst performance was “Content” where only one of the three questions was categorised correctly. This was expected as content questions are the most difficult category to classify due to the vast number of ways these questions can be asked. More than the average number of test observations will be needed for this category and additional features may need to be engineered, for example testing to see if any of the keywords related to the other categories occurred for these content related questions.

The “Tests” category, which had the greatest number of test observations, namely 28, had an average performance with 3 errors occurring. The “General” category had the lowest AUC value as can be seen in Figure 5.7 where the ROC curves with AUC values per question category are shown using the best MLR model. Separate one-versus-rest LR classifiers were built for each category to allow for calculating AUC values for multi-label classification (as described in Section 2.4).

Figure 5.7: ROC curves with AUC values per question category using the best MLR model



These AUC values per question category are also shown in Table 5.7 with the error rates and F1-measure values. The question categories have been ordered by the F1-measure values (descending order). The error rate is equal to the number of false negatives divided by the count of testing observations per category.

Table 5.7: Error rate, AUC and F1 values per question category using the best MLR model

Question Category	Count	True Positives	False Negatives	Error Rate	AUC	Precision	Recall	F1
IntroStat Collection	5	5	0	0.000	1.000	1.000	1.000	1.000
Videos	3	3	0	0.000	1.000	1.000	1.000	1.000
Winter/Summer Term	2	2	0	0.000	1.000	1.000	1.000	1.000
Tutorials	14	14	0	0.000	0.990	0.933	1.000	0.966
DP Requirement	6	5	1	0.167	0.998	1.000	0.833	0.909
Assignments	18	17	1	0.056	0.954	0.850	0.944	0.895
Hotseat Sessions	5	4	1	0.200	1.000	1.000	0.800	0.889
Exam	9	8	1	0.111	0.989	0.889	0.889	0.889
Tests	28	25	3	0.107	0.962	0.862	0.893	0.877
Medical Certificate	7	7	0	0.000	0.996	0.778	1.000	0.875
Workshops	9	9	0	0.000	0.985	0.750	1.000	0.857
Quizzes	18	15	3	0.167	0.991	0.833	0.833	0.833
General	16	11	5	0.313	0.903	0.786	0.688	0.733
Timetable clashes	7	4	3	0.429	0.975	0.800	0.571	0.667

Table 5.7: Error rate, AUC and F1 values per question category using the best MLR model (continued)

Question Category	Count	True Positives	False Negatives	Error Rate	AUC	Precision	Recall	F1
Lectures	7	4	3	0.429	0.977	0.667	0.571	0.615
Content	3	1	2	0.667	0.963	1.000	0.333	0.500
			Average Values	**0.165**	**0.980**	**0.835**	**0.884**	**0.844**

There were four categories with AUC values of 1, namely “IntroStat Collection”, “Videos”, “Winter/Summer Term” and “Hotseat Sessions”. The “Hotseat Sessions” category had one error, but with the one-versus-rest method this error reduced to zero. Also, there were four questions for which the “Hotseat Sessions” category was predicted correctly, which resulted in a precision value of 1 for this category. The “DP Requirement” category is another example of where the precision value was 1 and where the AUC value was higher than the F1-measure value.

Based on the error rate and the F1-measure, the “Content” category displayed the worst performance, which is in line with previous observations, even though the AUC values suggest that it was the “General” category which showed the worst performance. The AUC values also ranked the “Tutorials” category in the eighth position, while the rank for this category, based on the F1-measure, was fourth. Another difference is where the AUC values ranked the “Hotseat Sessions” category in the fourth position, while the rank for this category, based on the F1-measure, was seventh.

This concludes that the F1-measure is a better performance measure to use in this case where multiple categories need to be evaluated, compared with the AUC values. In the next section the success of selecting the most similar past question will be discussed.

5.2.2 Selection of Most Similar Past Questions

Once the question category is known, all past questions which fell into that category can be compared against the new question asked, in order to determine the most similar past question. The best-known method is cosine similarity, but other distance methods like Euclidean or Manhattan can also be considered. A comparison between these methods will be discussed in this section.

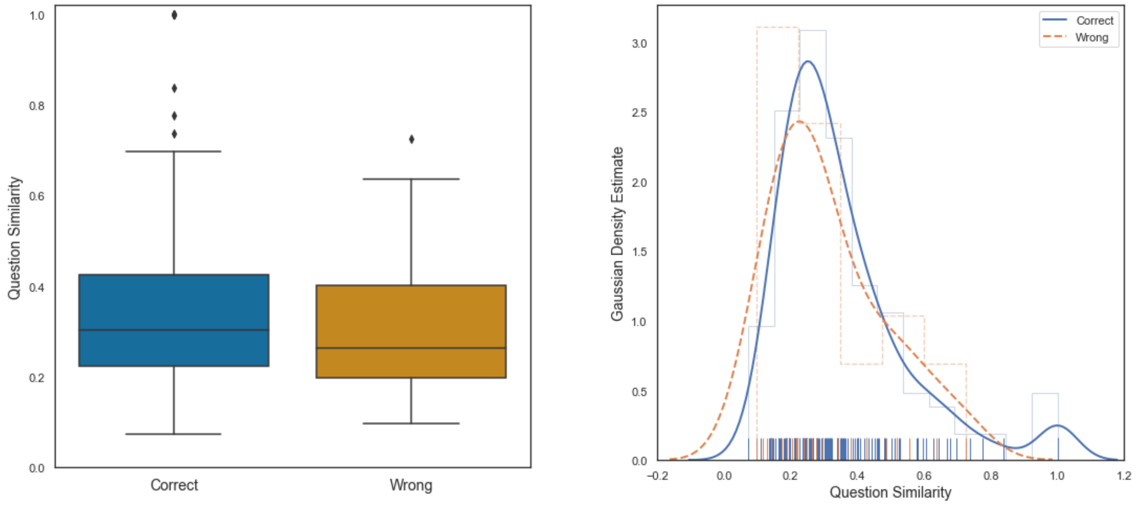
In Section 4.8 the “curse of dimensionality” was mentioned. This makes it difficult to find the similarity between vectors because they get exponentially farther away from one another as the dimensionality increases. The MLR classifier based on TF-IDF unigram and bigram features has a total of 5 688 dimensions, which leads to this curse of dimensionality. Therefore, the 2nd ranked MLR classifier based on LSA features, built from TF-IDF unigram features and which resulted in a total of 55 dimensions, was compared to see if this could improve the precision of similarity. The process for selecting the number of LSA topics is described in Appendix A.5.

In Figure 5.8 the distributions and densities of the similarity values for the 157 test questions are compared for the TF-IDF unigram and bigram features, and for the LSA features (built from TF-IDF unigram features). The process followed was to first predict the question category of a test question, then retrieve all training questions which fell into the predicted category and calculate the cosine similarity between the test question and each of the training questions. The highest cosine similarity value for each test question was collected, as well as the outcome of the top predicted question category, which could either be "correct" or "wrong", based on the available question category label. This derived "correct" or "wrong" label for the predicted question category could then be used to categorise the similarity value of the selected training question.

Figure 5.8: Comparison of the distributions and densities of similarity values

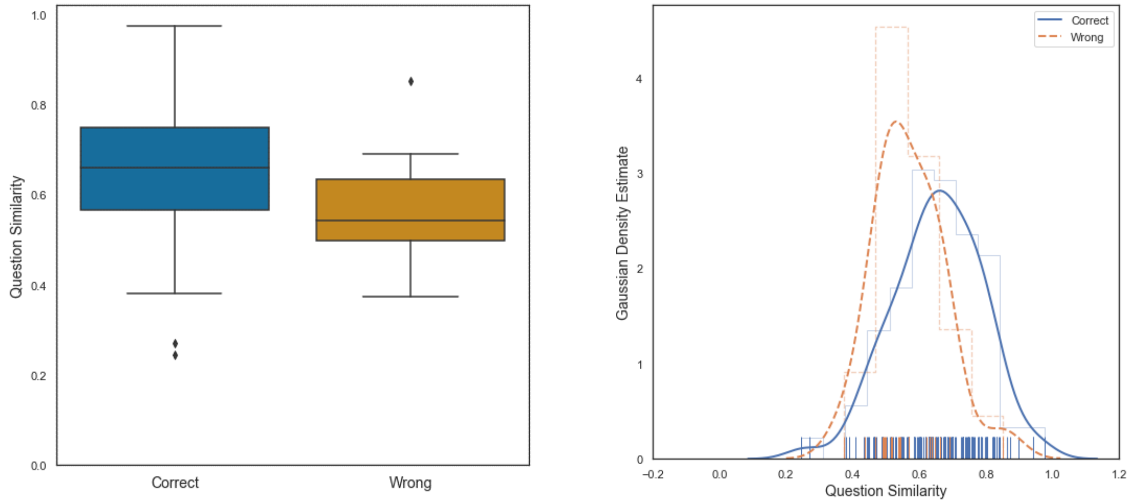
a) Distributions based on TF-IDF features

b) Gaussian density based on TF-IDF features



c) Distributions based on LSA features

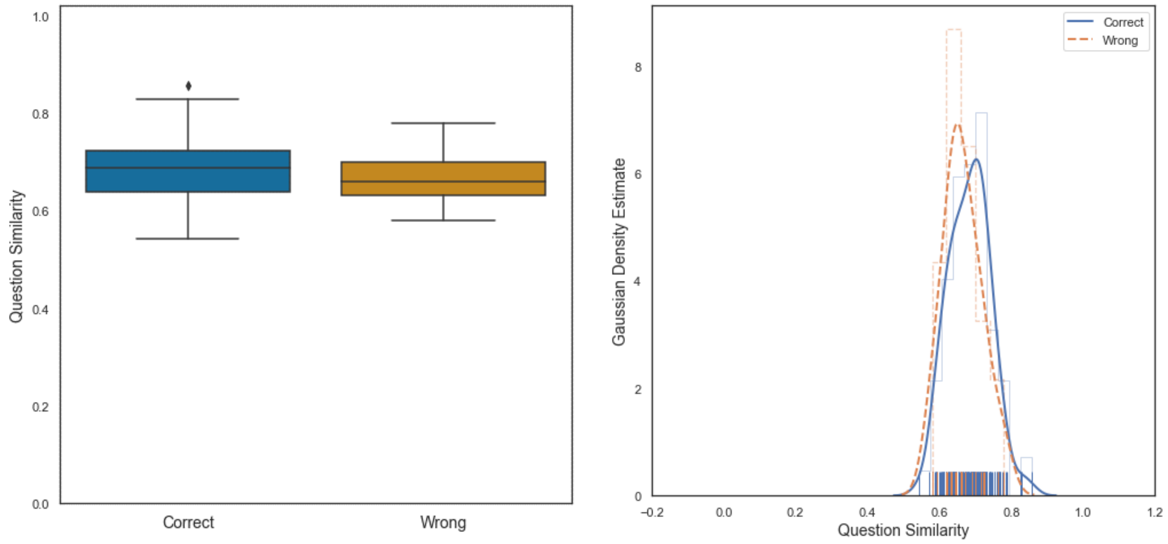
d) Gaussian density based on LSA features



The challenge of distinguishing between similarity values falling into the “correct” or “wrong” categories when using TF-IDF features is evident in Figure 5.8a and 5.8b, as the distribution and density values for both categories are almost identical. The advantage of using the LSA features for calculating similarity values can be seen in Figure 5.8c and 5.8d, where the distribution and density values become more distinct.

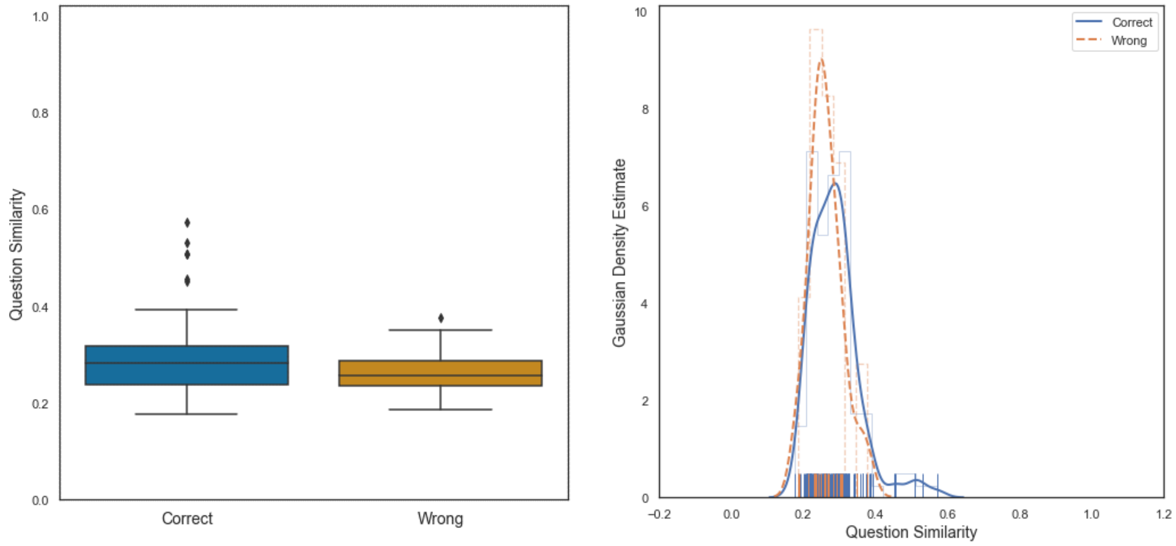
Instead of using cosine similarity, one could consider using the Euclidean or Manhattan distance methods. Vectors with the shortest distance between them are considered most similar. The distributions and densities of using the Euclidean and Manhattan distance methods for the LSA features are shown in Figure 5.9.

Figure 5.9: Comparison of similarity values based on Euclidean and Manhattan distances



a) Distributions based on Euclidean distance

b) Density based on Euclidean distance



c) Distributions based on Manhattan distance

d) Density based on Manhattan distance

It is clear form this comparison that the Euclidean and Manhattan distance methods do not achieve the same distinction between question similarities labels as “correct” or “wrong”, as was possible when using the cosine similarity method.

In the next section, the accuracy of extracting answers as a result of predicting the question category using the MLR classifier based on TF-IDF unigram and bigram features and selecting the most similar past question using cosine similarity based on the 55 LSA features (built from the TF-IDF unigram features) will be discussed.

5.2.3 Evaluation of Extracted Answers

It would not be possible to answer all questions and therefore it is important to set thresholds to help determine when the confidence level is high enough before attempting to return an answer. Thresholds for both the question category probabilities and the past question similarities need to be determined. The descriptive statistics as shown in Table 5.8 can be used and the 10th percentile for the “correct” category has been selected to determine these thresholds. For the question category probabilities, the selected threshold is for values to exceed 0.14 and for past question similarities the selected threshold is for values to exceed 0.48.

Table 5.8: Descriptive statistics for category probabilities and past question similarities

		Descriptive Statistics								
Measure Type	**Category**	**Count**	**Min**	**Mean**	**Max**	**Std**	**P10**	**P25**	**P50**	**P75**
Category Probability (TF-IDF features)	Correct	134	0.093	0.226	0.435	0.076	0.140	0.172	0.213	0.270
	Wrong	23	0.099	0.167	0.298	0.056	0.112	0.122	0.147	0.200
Question Similarity (LSA features)	Correct	134	0.245	0.654	0.975	0.133	0.477	0.567	0.660	0.749
	Wrong	23	0.373	0.570	0.852	0.104	0.473	0.498	0.542	0.634

To evaluate the QA system and the correctness of the extracted answer, 1 question from each of the 16 question categories was randomly selected from the test data. These questions are listed in Table 5.9 along with their predicted category, the best past question similarity score and the extracted answer (the training dataset represents past questions). Two of the 16 questions had question category probabilities below the threshold of 14% and 1 question had its best past question similarity below the threshold of 48%. Because the thresholds were not reached for these three questions, a standard answer of “*The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.*” was extracted. Values linked to thresholds not reached are marked in blue. Predicted categories or extracted answers that are correct are marked in green, while incorrect values are marked in light orange.

Table 5.9: Random selection of one test question per category with extracted answer

No	Question Category	Test Question Text	Predicted Category	Category Probability	Past Question Similarity	Past Question Text	Extracted Answer
1	Assignments	Hi there I have a bit of an issue with assignment 4. It seems that the Excel file does not contain the values given as the right answers for the two questions concerning the five number summary, in part A. In fact, I searched the file for all the values given, and they are not in the file at all. I would appreciate any clarification you could provide on the matter. Thank you in advance.	Assignments	12.8%	N/A	N/A	The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.
2	Content	Please can you confirm that we are only being examined up to work unit 7. The announcement said up to the end of chapter 9 but some of the material such as pages 215-220 are not in work unit 7 but are after this unit. Please can you clarify this as these pages are work unit 8.	General	12.2%	N/A	N/A	The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.
3	DP Requirement	What is the DP requirement? The Outline says just to complete the quizzes with 90% for each quiz? Do we have to get a minimum mark for our class record?	DP Requirement	26.0%	78.0%	Do I have to get 90% for quiz 1.0 for DP as well? The course outline quiz?	You need to obtain a minimum of 90% for each online quiz. These quizzes do not contribute towards your Class Record. However, there are bonus marks awarded for completing your quizzes timeously.
4	Exam	Hi, is it possible to release atleast [*sic*] 5+ past papers for exams? Or more questions for revision. I am requesting this as the quizzes are 10-20 times easier than the test as they are very basic, the LWS questions are a bit challenging but again they are a few and don't necessarily play the tricks that tests do same goes for SWS. At the same time all of the resources test you on the week's content and thus you also know what method to use. Test just connect two or more topics at once and you never get that kind of practice. Thank you	Tests	19.6%	64.4%	Please could we have past test papers?	They are under "Week4" in "Weekly Material"
5	General	Where is the Quiet Room?	General	9.3%	N/A	N/A	The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.
6	Hotseat Sessions	The times listed for the Excel hotseat in the info pack were different to those stated in the short workshop. What is the correct time?	Hotseat Sessions	15.4%	64.3%	What time are the hotseats?	Monday (Extra Help group only) Tuesday (Meridian) Thursday (3rd period) Friday (4th or 5th period)

Table 5.9: Random selection of one test question per category with extracted answer (continued)

No	Question Category	Test Question Text	Predicted Category	Category Probability	Past Question Similarity	Past Question Text	Extracted Answer
7	IntroStat Collection	Where can I collect the course text book?	IntroStat Collection	34.2%	88.0%	When can I collect the course text book?	Attend your first lecture during the first week of semester and then collect your copy of Introstat afterwards.
8	Lectures	Am I right in saying that the STA1000S lectures shown on PeopleSoft count as short workshops? And hence only one of those need to be attended? :)	Lectures	15.3%	54.9%	Must we do the lecture questions before we go to the lecture, or must we only do them during the lecture?	Yes, if you have not prepared for the lectures by covering the relevant sections of the work beforehand, then they will be of little benefit to you.
9	Medical Certificate	Good Afternoon! Who do you talk to if you are unable to go to a tutorial and miss an assignment hand in because of medical reasons? Thank you in advance for your assistance.	Medical Certificate	31.6%	78.6%	Can I submit a medical note if a missed an assignment hand in?	Note that we do not accept medical notes for assignments as we accommodate for occasional absences by only taking your top 10 out of 12 assignments into account for Class Record purposes.
10	Quizzes	How do we get feedback from quizzes?	Quizzes	40.4%	85.6%	Quiz 2.2 does not allow feedback? Help	Please email a screenshot of the problem you are experiencing to the course convener.
11	Tests	Hello, Will test two be negatively marked? Thanks	Tests	24.3%	94.6%	Is there negative marking?	There is no blanket rule with regard to negative marking but you will be informed at each test as to how the test will be marked.
12	Timetable clashes	Hi, I also have a clash for both LWS slots. Is it possible for another slot to become available. Perhaps on a Monday ?	Timetable clashes	19.4%	61.5%	Hello. I have clashes on both wednesday [*sic*] and thursday [*sic*] LWS, can we have atleast [*sic*] another on the other days of the week. Thank you	Please check recent announcements on timetable clashes. If you still have a clash, then email a copy of your timetable and detail of your clashes to the course convener so that we can attempt to make a suitable alternate arrangement.
13	Tutorials	Are tutorials compulsory?	Tutorials	26.9%	30.6%	I accidentally swapped tutorials after being in one the whole semester, is there any way I can get changed back from tutorial 57 to tutorial 30?	The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.
14	Videos	Please can you upload the videos for week 4. I always try and get through as much stats as i [*sic*] can on the weekend so having the material including the videos on a friday [*sic*] would be such a great help.	Videos	20.1%	63.1%	Hi, My Video 6 is not playing at all. Please could this be fixed?	Please try downloading the video and watching it offline, or opening it up in an alternative browser
15	Winter/ Summer Term	Hello, could someone please give me the contact details of the lecturer in charge of teaching winter school?	Winter/ Summer Term	14.3%	63.8%	Is STA1000 offered for Winter/Summer School?	Yes it is. Winter term runs from mid June for a period of 4 weeks. Summer school runs for 4 weeks from end November to December. Winter/summer school is online only.
16	Workshops	If we failed test 1, will we need to attend a workshop this week?	Workshops	24.2%	78.8%	I failed test 1 but passed test 2 - am I still obliged to attend the weekly workshops?	No, you are not, well done on the improvement! Although, if you reckon that attending the workshops helped you improve, then it might be wise to attend the last few as well.

The following sections will look at specific examples, starting with where the threshold for the question category probabilities was not reached. Different examples where incorrect answers were extracted will be discussed: first where the predicted questions category was correct and then where the predicted questions category was incorrect. New questions posted on the QA simulator will also be discussed in two sections, firstly where answers were incorrect and secondly where answers were correct.

Examples where the threshold for the question category probabilities were not reached

The probability of the predicted question category for the first test question in Table 5.9 was 12.8% and below the threshold (14%), but the predicted category was correct. The most similar past question of "*I think the excel file given for assignment 4 is incorrect. It doesn't relate to the questions. Can the correct one please be uploaded*" had a similarity of 77.7% and was indeed relevant to the test question. The answer for the most similar past question "*You may be looking at the Workshop Excel file and not the Assignment Excel file?*" was also a valid response for the test question. This points out how a threshold can potentially cause the inability to answer a question where it would have been possible without the threshold.

For the second test question, the probability of the predicted question category was also below the threshold, but in this case the predicted category was incorrect. The most similar past question had a similarity of 85%, much higher when compared to the first test question where the predicted category was correct, but for this second test question the most similar past question and its answer were not relevant. The reason for the high similarity value of 85% was due to a weakness of cosine similarity where short documents are favoured. The test question had a total of 16 tokens, while the most similar past question only had a total of 5 tokens. When comparing this to the first test question, it was noticed that the ratio for tokens between the test question and the most similar past question was 46% while the ratio for this second test question was only 31%. Therefore, an additional measure could be considered in combination with the threshold tests in order to check for the ratio between the new question and the most similar past question. Jeon, Croft & Lee (2005b) have shown in their research that language-modelling-based similarity measures outperform cosine similarity and address the issue of favouring short documents. These language-modelling-based similarity measures and using a measure for the ratio between the new question and the most similar past question are suggestions for future study.

Examples of incorrect answers where the predicted question category was correct

A total of five test questions was found where the predicted question category was correct and the best past question similarity score was high, but where the extracted answer was incorrect.

The seventh test question "*Where can I collect the course text book?*" has an extracted answer of "*Attend your first lecture during the first week of semester and then collect your copy of Introstat afterwards*" which is based on the past question "*When can I collect the course text book?*". A more similar past question, by human evaluation, would be "*Where can I collect the handbook?*". The reason for this selection is that the core of the test question is captured by the terms "where", "collect" and "text book". By human evaluation one knowns that "text book" and "handbook" are equal, which results in the same derived terms for both questions. The context for the test question and the past question selected by human evaluation are the same, but the term "when" which occurs in the past question selected by machine evaluation, results in a different context. Table 5.10 shows a comparison between these three questions, where matching tokens are marked in gold.

Table 5.10: Evaluation of a test question for the "IntroStat Collection" category

Type	Question Text	Question Tokens	Answer
Test question	Where can I collect the course text book?	{"where", "collect", "course", "text", "book"}	You can collect the IntroStat handbook any time from Statistical Sciences reception, PD Hahn building level 5
Most similar past question	When can I collect the course text book?	{"when", "collect", "course", "text", "book"}	Attend your first lecture during the first week of semester and then collect your copy of Introstat afterwards
Human evaluation	Where can I collect the handbook?	{"where", "collect", "textbook"}	You can collect the IntroStat handbook any time from Statistical Sciences reception, PD Hahn building level 5

The term "handbook" in the past question selected by human evaluation was replaced by "textbook", based on a data pre-processing rule, but this single token "textbook" cannot be matched to the two tokens "text" and "book" found for the test question. An extra data pre-processing rule would be needed to remove the space and convert all "text book" terms to "textbook" in order to resolve this issue. The current list of phrase replacements can be found in Appendix A.2. The cosine similarity between the test question and the most similar past question by machine evaluation was 88%, while the cosine similarity between the test question and the past question selected by human evaluation was 78%.

The most similar past question for the tenth test question "*How do we get feedback from quizzes?*" was "*Quiz 2.2 does not allow feedback? Help*" with a similarity score of 85.6%. An incorrect answer of "*Please email a screenshot of the problem you are experiencing to the course convener*" was extracted.

The past question with the correct answer for the test question was "*Can we get feedback on the quizzes because i* [*sic*] *don't understand why i'm* [*sic*] *getting some of them wrong?*" and only ranked in the fifth position based on similarity ranks. Table 5.11 shows the matching tokens between these three questions, where it becomes apparent that all three questions had the same matching tokens, but that the number of non-matching tokens decreased the similarity score.

Table 5.11: Evaluation of a test question for the "Quizzes" category

Type	Question Text	Question Tokens	Answer
Test question	How do we get feedback from quizzes?	{"how", "feedback", "quiz"}	Scroll down on the "Test & Quizzes" page and go to the bottom. The quizzes you have completed are listed there with the scores you achieved. Click on the quiz to get feedback.
Most similar past question	Quiz 2.2 does not allow feedback? Help	{"quiz", "not", "allow", "feedback", "help"}	Please email a screenshot of the problem you are experiencing to the course convener
Human evaluation	Can we get feedback on the quizzes because i [*sic*] don't understand why i'm [*sic*] getting some of them wrong?	{"feedback", "quiz", "because", "not", "understand", "why", "wrong"}	Scroll down on the "Test & Quizzes" page and go to the bottom. The quizzes you have completed are listed there with the scores you achieved. Click on the quiz to get feedback.

The core difference between the test question and the most similar past question by machine evaluation is that the former has a focus on "get feedback", while the latter has a focus on "allow feedback". The term "get" was dropped as part of the data pre-processing step where custom stop-words were removed. The model can potentially be improved by excluding the term "get" from the custom stop-words list, as this term will increase the similarity with the past question selected by human evaluation. However, thorough testing will be needed to ensure this change does not have any negative impact on other questions where this term occurs. The current list of custom stop-words can be found in Appendix A.3.

For the 16th test question, "*If we failed test 1, will we need to attend a workshop this week?*", the most similar past question was "*I failed test 1 but passed test 2 - am I still obliged to attend the weekly workshops?*" with a similarity score of 78.9%. An incorrect answer of "*No, you are not, well done on the improvement! Although, if you reckon that attending the workshops helped you improve, then it might be wise to attend the last few as well.*" was extracted. The best answer by human evaluation was "*No, but should you fail Test1* [*sic*], *then it will become compulsory for you to attend one workshop a week (in addition to tutorials)*" which was linked to the fourth ranked past question of "*Are workshops compulsory?*". This answer is not perfect for the test question due to the fact that it starts with "*No, but*", which may cause confusion and points to an issue with reusing past question answers where the context is different. Table 5.12 shows the matching tokens between these three questions, where it becomes apparent that the past question selected by human evaluation does not contain enough tokens matching the test question and therefore could not be selected by machine evaluation.

Table 5.12: Evaluation of a test question for the "Workshops" category

Type	Question Text	Question Tokens	Answer
Test question	If we failed test 1, will we need to attend a workshop this week?	{"fail", "test", "will", "need", "attend", "workshop", "week"}	If you failed Test 1 then you will be required to attend both workshops and tutorials (as a DP requirement).
Most similar past question	I failed test 1 but passed test 2 - am I still obliged to attend the weekly workshops?	{"fail", "test", "pass", "test", "still", "oblige", "attend", "weekly", "workshop"}	No, you are not, well done on the improvement! Although, if you reckon that attending the workshops helped you improve, then it might be wise to attend the last few as well.
Human evaluation	Are workshops compulsory?	{"workshop", "compulsory"}	No, but should you fail Test1 [*sic*], then it will be become compulsory for you to attend one workshop a week (in addition to tutorials).

An issue that can be observed from the way tokens have been derived for these questions is that the test numbers have been dropped from tokens but are in fact required because the terms "test 1" and "test 2" are different and test numbers are important factors to take into account in determining if workshops are compulsory for a student, as would be the case if test 1 was failed. An investigation to determine why some numbers were dropped exposed an error during data pre-processing. Correction of this error could potentially improve all model test results and will be left for future study.

Example of an incorrect answer where the predicted question category was incorrect

There was only one test question where the probability of the predicted question category was above the threshold, but where both the predicted category and the extracted answer were incorrect. For the "Exam" category the test question was "*Hi, is it possible to release atleast* [*sic*] *5+ past papers for exams? Or more questions for revision. I am requesting this as the quizzes are 10-20 times easier than the test as they are very basic…*" but the top predicted questions category was "Tests" with a probability of 19.6%. The correct category of "Exam" ranked third with a predicted probability of 8.0%. The extracted answer was of course incorrect, but the most similar past question of "*Please could we have past test papers?*" was close with a similarity score of 64.4%. If the predicted question category was "Exam" then the most similar past question would have been "*Would it be possible for the 2014 past papers to be put online please?*" with a similarity score of 59.9% and a correct answer of "*Have a look under Week 12 Material*" would have been extracted. Table 5.13 shows a comparison between these three questions.

Table 5.13: Evaluation of a test question for the "Exam" category

Type	Question Text	Question Tokens	Answer
Test question	Hi, is it possible to release atleast [*sic*] 5+ past papers for exams? Or more questions for revision. I am requesting this as the quizzes are 10-20 times easier than the test as they are very basic, the LWS questions are a bit challenging but again they are a few and don't necessarily play the tricks that tests do same goes for SWS. At the same time all of the resources test you on the week's content and thus you also know what method to use. Test just connect two or more topics at once and you never get that kind of practice.	{"possible", "release", "atleast", "past", "paper", "exam", "much", "question", "revision", "request", "quiz", "10", "20times", "easy", "test", "very", "basic", "workshop", "bite", "challenge", "again", "few", "not", "necessarily", "play", "trick", "same", "go", "lecture", "time", "resource", "week", "content", "thus", "know", "what", "method", "use", "connect", "two", "topic", "once", "never", "kind", "practice"}	We will not be releasing any more exam prep material. For preparation, please refer to the exam revision questions under the weekly material tab. There are also videos which you can watch.
Most similar past question	Please could we have past test papers?	{"past", "test", "paper"}	They are under "Week4" in "Weekly Material"
Human evaluation	Hi There [*sic*] Would it be possible for the 2014 past papers to be put online please? I understand that the course structure has changed a lot between 2013 and 2014, and having the 2014 paper might be more beneficial for us. Please could someone advise me where I could find them other wise.	{"possible", "2014", "past", "paper", "put", "online", "understand", "course", "structure", "change", "lot", "between", "2013", "much", "beneficial", "someone", "advise", "where", "find", "other", "wise"}	Have a look under Week 12 Material

This test example also illustrated how the number of non-matching tokens decreased the similarity score as the past question selected by human evaluation had more matching tokens than the most similar past question by machine evaluation.

Examples of new questions posted on the QA simulator and answered incorrectly

A simulator was built to illustrate how a QA system can be used to facilitate the auto-answering of student questions and allow lecturers to test the QA system (refer to Appendix A.7 for more detail). The first version received negative feedback, as the system failed to answer basic questions like "*Where are workshops held?*" and "*Are lectures compulsory?*". An investigation into the reason for answering these questions incorrectly revealed that there were no training data for these questions, as students had been asking such questions through other platforms and not through the Administrative Queries site from where all training data were sourced at the time. To resolve this issue additional questions and answers were derived from the Course Information Pack as described in Chapter 3.

The question "*What do I need to know for test 1?*" was correctly categorised as a "Tests" question (with a probability of 23.4%), but the provided answer "*Please refer to the Course Information Pack for test dates*" was incorrect. Table 5.14 shows a comparison between the test question, the most similar past question by machine evaluation and the best past question selected by human evaluation. The similarity score between the new question and the most similar past question by machine evaluation was 57.3%, while the similarity score with the question selected by human evaluation was only 33.5%.

Table 5.14: Evaluation of a new question for the "Tests" category

Type	Question Text	Question Tokens	Answer
New question	What do I need to know for test 1?	{"what", "need", "know", "test", "1"}	Please refer to the Course Information Pack for test dates
Most similar past question	What are the test dates?	{"what", "test", "date"}	
Human evaluation	What is the content for test 1?	{"what", "content", "test"}	Modules 1, 2 & 3

The same issue as previously mentioned, can be observed for the question selected by human evaluation where numbers are dropped from tokens. Interestingly enough, the number in the new question was preserved as a token. All three questions have the same matching tokens, but the shorter question was selected by machine evaluation due to the fact that cosine similarity favours shorter questions. If the issue with dropped token numbers could be resolved then the past question "*What is the content for test 1?*" would have the highest similarity score and the correct answer "*Modules 1, 2 & 3*" would be extracted.

The question "*How long is the course?*" was correctly categorised as a "General" question (with a probability of 23%) but the provided answer "*Notices will be posted on the STA1000S Vula site*" was incorrect. Table 5.15 shows a comparison between the test question and the most similar past question. There was no past question that was related to this new question and therefore it was impossible to extract a correct answer. Unfortunately, the similarity score between the new question and the most similar past question was high (93.2%) because both questions were short. This points out an issue that cannot easily be solved as there were also other examples where both the new question and the most similar past question were short but where a correct answer was extracted.

Table 5.15: Evaluation of a new question for the “General” category

Type	Question Text	Question Tokens	Answer
New question	How long is the course?	{“how”, “long”, “course”}	Notices will be posted on the STA1000S Vula site
Most similar past question	How is [*sic*] course notices communicated?	{“how”, “course”, “notice”, “communicate”}	

The question “*What is a classification?*” was incorrectly categorised as a “Tests” question with a probability of 11%. The correct question category “Content” ranked last among all 16 categories with a probability of 3.1%. Due to the threshold that was not met, the general answer of “*The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.*” was extracted for this question. The only token in this new question that could be matched to any past questions was “what” and the “Tests” category contains the most occurrences of this token.

Example of a combination question

The question “*I was ill and missed the test, who do I need to submit my sick certificate to and when will we receive confirmation of the next test dates?*” is an example of a combination question for categories “Medical Certificate” and “Tests”. Table 5.16 shows a split of this question into three steps where the first is the “Medical Certificate” question and the second is the “Tests” question. The question categories for the individual questions were correctly predicted and the similarity scores were 77.1% for the first question and 54.3% for the second question.

Table 5.16: Evaluation of a new question for a combination of two categories

Step	Type	Question Text	Question Tokens	Answer
1	New question	I was ill and missed the test, who do I need to submit my sick certificate to?	{“ill”, “miss”, “test”, “who”, “need”, “submit”, “sick”, “certificate”}	We do not accept an email of a medical certificate (doctor's note). You will need to bring a copy as well as the original to stats department.
	Most similar past question	Can I submit a medical note if a missed a tutorial test?	{“submit”, “medical”, “note”, “miss”, “tutorial”, “test”}	
2	New question	When will we receive confirmation of the next test dates?	{“when”, “will”, “receive”, “confirmation”, “next”, “test”, “date”}	Please check for announcement posted on test times, venues and sign up procedures.
	Most similar past question	Test date and time confirmation When will receive confirmation of the test dates and times? The course outline contains only provisional test dates and times.	{“test”, “date”, “time”, “confirmation”, “when”, “will”, “receive”, “confirmation”, “test”, “date”, “time”, “course”, “outline”, “contain”, “only”, “provisional”, “test”, “date”, “time”}	

Table 5.16: Evaluation of a new question for a combination of two categories (continued)

Step	Type	Question Text	Question Tokens	Answer
3	New combination question	I was ill and missed the test, who do I need to submit my sick certificate to and when will we receive confirmation of the next test dates?	{“ill”, “miss”, “test”, “who”, “need”, “submit”, “sick”, “certificate”, “when”, “will”, “receive”, “confirmation”, “next”, “test”, “date”}	We do not accept an email of a medical certificate (doctor's note). You will need to bring a copy as well as the original to stats department.
	Most similar past question	I missed my tutorial this week due to illness - who do I email my doctor's note to?	{'miss', 'tutorial', 'week', 'due', 'illness', 'who', 'email', 'doctor', 'note'}	

An observation for the second question is that there was a total of six unique matching tokens with the most similar past question but, due to the length difference between the questions, the similarity score was average (54.3%). The combination question (step 3) was categorised as a “Medical Certificate” question with a probability of 16.6%; this probability was lower than that of the first question (24.5%). The combination question also found a different most similar past question to the first question, with a similarity score of 65.5%. This was also lower than that of the first question (77.1%) and there were only two matching tokens between the combination question and the most similar past question. This example points out that the model can only answer one question at a time, but it is not clear why the combination question has a different most similar past question to that of the first question (step 1) which has more matching tokens. This is probably due to the cosine similarity being calculated using the LSA features, but more analysis is needed.

Examples of new questions posted on the QA simulator and answered correctly

The focus of the previous sections was on questions answered incorrectly. This section will look at a few examples of questions posted on the QA simulator where the extracted answers were correct.

Table 5.17 lists the new questions for six of the question categories with their most similar past question, the matching tokens and the extracted answer.

Table 5.17: Examples of new questions answered correctly

Category	Type	Question Text	Question Tokens	Answer
Assignments	New question	Should the answers for assignment questions be submitted as fractions or decimal values?	{“answer”, “assignment”, “question”, “submit”, “fraction”, “decimal”, “values”}	As per the instructions on the assignment, round to 3 decimal places. If you fail to adhere to this, then Vula will mark the answer as incorrect.
	Most similar past question	For the assignment, can we submit our answers as fractions or do they have to be in decimals?	{“assignment”, “submit”, “answer”, “fraction”, “decimal”}	

Table 5.17: Examples of new questions answered correctly (continued)

Category	Type	Question Text	Question Tokens	Answer
DP Requirement	New question	How many tutorial sessions can I miss and still get DP?	{"how", "many", "tutorial", "sessions", "miss", "still", "dp"}	If there is a good reason (such as illness) for missing your tutorial, then you need to ask your tutor to excuse you from the tutorial and provide proof. Otherwise, you need to attend all tutorials and at least one workshop per week.
	Most similar past question	If we failed test one - so tutorials are for DP and compulsory; how many tutorials are we allowed to miss to still obtain DP?	{"fail", "test", "one", "tutorial", "dp", "compulsory", "how", "many", "tutorial", "allow", "miss", "still", "obtain", "dp"}	
Exam	New question	How much weight does the final exam carry?	{"how", "much", "weight", "final", "exam", "carry"}	The Final Mark is calculated as: (0.3 * CLASS RECORD%) + (0.7 * EXAM MARK%) Students who have medical exemption for a test will have their exam mark weighted 75%
	Most similar past question	How much is the exam weighted?	{"how", "much", "exam", "weight"}	
Hotseat Sessions	New question	What is a hot seat session?	{"what", "hotseat", "session"}	Tutors are on hand to assist you with any queries, particularly those related to Excel operations or the statistical concepts being portrayed in the videoed Excel exercises.
	Most similar past question	What are the hotseat sessions for?	{"what", "hotseat", "session"}	
IntroStat Collection	New question	Where do I get a copy of the book	{"where", "copy", "book"}	You can collect the IntroStat handbook any time from Statistical Sciences reception, PD Hahn building level 5
	Most similar past question	Where do I collect my copy of Introstat?	{"where", "collect", "copy", "textbook"}	
Tutorials	New question	I can't find my tutors email address. Please help!	{"not", "find", "tutor", "email", "address", "help"}	Tutor email addresses are listed in the course information pack and under "Course Outline" on Vula
	Most similar past question	Good Morning Where could one find the emails for the tutors - Ruth Amoore in particular?	{"where", "one", "find", "email", "tutor", "ruth", "amoore", "particular"}	

The question "*Should the answers for assignment questions be submitted as fractions or decimal values?*" was matched with the most similar past question of "*For the assignment, can we submit our answers as fractions or do they have to be in decimals?*" with a 66.5% similarity. All five tokens in the most similar past question were found in the new question. However, the new question had two additional tokens that did not occur in the past question.

The question "*What is a hot seat session?*" has three tokens where the term "hot seat" was replaced with "hotseat" during data pre-processing. All three tokens matched with the most similar past question, which also had only three tokens. The probability of the question category was 49.8% and the similarity score of the most similar past question was 58.6%. One would expect a higher similarity score based on the length of both questions and on the fact that all three tokens in both questions matched, but similarity was calculated using the LSA features, which do not have a direct mapping to the tokens.

The question "*Where do I get a copy of the book*" was answered correctly by chance, even though the similarity score between the question and the most similar past question was high (84.9%). Any other short question that contains the words "where" and "copy" and does not match any other question category tokens, would also be given the same answer. For example, "*Where can I go make a copy of my ID?*" will have a similarity score of 82.5% with the most similar past question of "*I haven't picked up a copy of introstat yet. Where can I go to get one?*".

The next section will evaluate sentiment analysis as a means of prioritising questions for the scenario where the most important question needs to be answered first.

5.2.4 Sentiment of Questions

Deriving the sentiment of the question can help to prioritise which question to answer first when dealing with multiple questions asked at the same time. VADER, the general word sentiment library developed by Georgia Tech (Hutto & Gilbert, 2014), was used to classify all questions into one of three categories, namely negative, neutral or positive. The VADER library assigns a sentiment score between -1 and +1 for an input text. If the score was below -0.1 the negative category was assigned, if the score exceeded 0.1 the positive category was assigned, in all other cases the neutral category was assigned.

Five random questions from each sentiment category were selected and are listed in Table 5.18 where questions have been ordered by the sentiment score, suggesting the order in which questions should be answered if all 15 questions were received at the same time. It is debateable how well this ordering works, especially when considering questions in the neutral or positive categories. For example, is the question with ID 6 really more important than the question with ID 15? There is merit in prioritising questions falling into the negative category, but questions in the neutral or positive categories might rather be considered to be answered in the order in which the questions were received.

Table 5.18: Random sample of 15 questions with sentiment scores and categories

		Sentiment	
ID	**Question**	**Score**	**Category**
1	Hello I mistakenly forgot to submit my assignment on time. I understand that this was a critical and self-inflicted error. Am I right in saying only the top 10 out of 12 assignments contribute towards my marks? Or is my DP jeopardized? If so is there anyway [*sic*] to rectify my situation?	-0.713	Negative
2	What are the consequences of me failing to submit an assingment [*sic*]?	-0.511	Negative
3	Will I get zero for an assignment if I missed the submission due date due to medical reasons?	-0.296	Negative
4	I am sick, but I need to attend compulsory workshops. What do I do?	-0.285	Negative
5	I missed a tutorial last week and didn't attend a make-up, but I have a medical note. Who can i [sic] give it to?	-0.153	Negative

Table 5.18: Random sample of 15 questions with sentiment scores and categories (continued)

ID	Question	Sentiment Score	Sentiment Category
6	Good day, was just wondering why for assignment 5 question four it is only to one decimal place, when nowhere in the assignment did it state that we had to round to 1 decimal place. Since i [*sic*] wrote the answer 1.200 i [*sic*] got it wrong however i [*sic*] don't understand why this is the case.	-0.052	Neutral
7	Where is the link for assignment 7? I don't see it anywhere.	0.000	Neutral
8	I have dislocated my right shoulder during a rugby game, I have a medical certificate to prove this. I am right handed and now I cant [*sic*] write anything. Will I be able to write the test without needing to write any calculations?	0.000	Neutral
9	There was mention of downloading the videos in one of the messages but I cant [*sic*] seem to find how to do that. What should I do, or where should I go on Vula to download the videos?	0.000	Neutral
10	When will we be able to sign up for test slots for the test on the 2nd of October?	0.000	Neutral
11	Hi, my email did not get a response. Is it possible to submit an assignment after the deadline? I completed it but was distracted by the test on friday [*sic*] and forgot to submit it. if i [*sic*] am able to, will there be any reductions on my mark? Thank you.	0.131	Positive
12	About my previous question concerning the quizzes, Sorry i [*sic*] found the quiz that i [*sic*] did not do. My mistake	0.187	Positive
13	1 - Will the max of exam or exam+Cr [*sic*] apply for deferred exams too? 2 - if i [*sic*] have already deferred my full block of exans [*sic*] but would like to write in November rather will i [*sic*] be allowed to?	0.394	Positive
14	Hi, is it possible to be emailed memos for all the tutorials that cover work in the class test? (Tutorials 1 - 5) Thank you!	0.420	Positive
15	On Sunday 22 October the announcement for Test 2 sign up was posted and stated there is a separate Vula site, STA 1000 2017 Exam, that will be used for testing. The announcement further stated that we have access to this web page as of 19:30 the very same evening. However I seem unable to find this site on my own Vula. I was wondering if I might have not been added to the participants on this site? If so how do I gain access to the site, so I may choose a test slot and write Test 2?	0.615	Positive

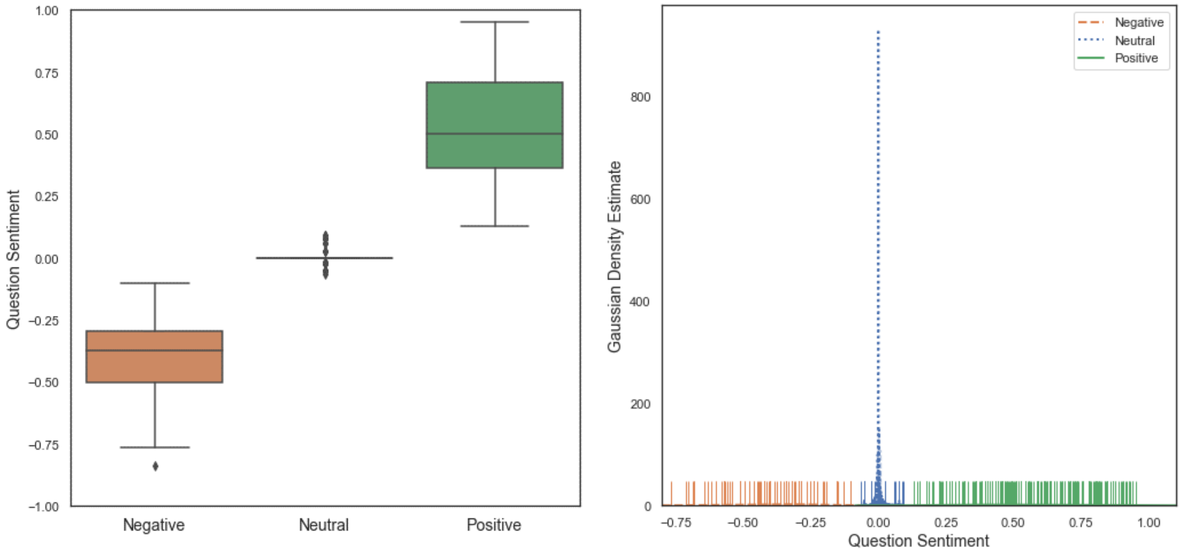
Table 5.19 shows the general statistics for the sentiment scores for all 744 questions. Almost half of the questions (48.3%) fell into the neutral category, 14.9% into the negative category and 36.8% into the positive category.

Table 5.19: Sentiment categories with descriptive statistics for all 744 questions

	Descriptive Statistics								
Category	Count	% of Total	Min	Mean	Max	Std	P25	P50	P75
Negative	111	14.9%	-0.836	-0.389	-0.103	0.157	-0.502	-0.374	-0.296
Neutral	359	48.3%	-0.064	0.001	0.094	0.013	0.000	0.000	0.000
Positive	274	36.8%	0.130	0.536	0.953	0.207	0.361	0.499	0.710

Some interesting observations are that the 25th, 50th and 75th percentile values for the neutral category are all zero and how low the standard deviation is for the neutral category when compared to the other categories. Figure 5.10 shows a boxplot for all the category sentiment scores in combination with the Gaussian density estimates per category. The boxplot reveals how densely the sentiment scores for the neutral category occurred, compared with the negative and positive categories. The graphs for the Gaussian density functions first include all categories where the density of the neutral category completely overpowers the other categories. Looking at the Gaussian density functions for only the negative and positive categories (Figure 5.10c) however show how well these categories are separated and that there is merit in using these sentiment classifications.

Figure 5.10: Boxplot and Gaussian density functions per sentiment category

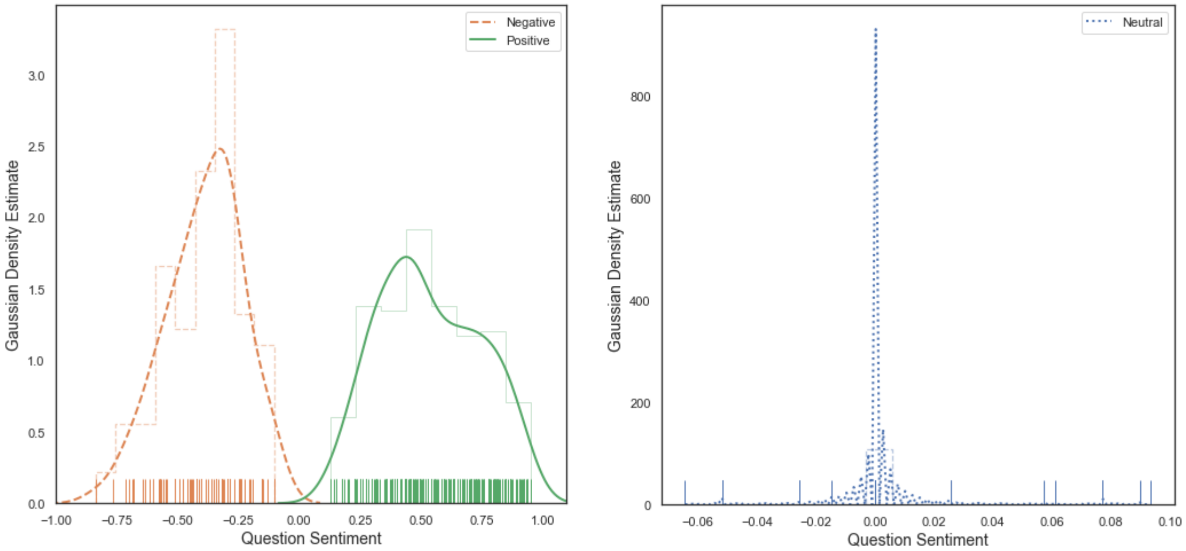


a) Boxplot

b) Gaussian density for all categories

c) Gaussian density for two categories

d) Gaussian density for neutral category

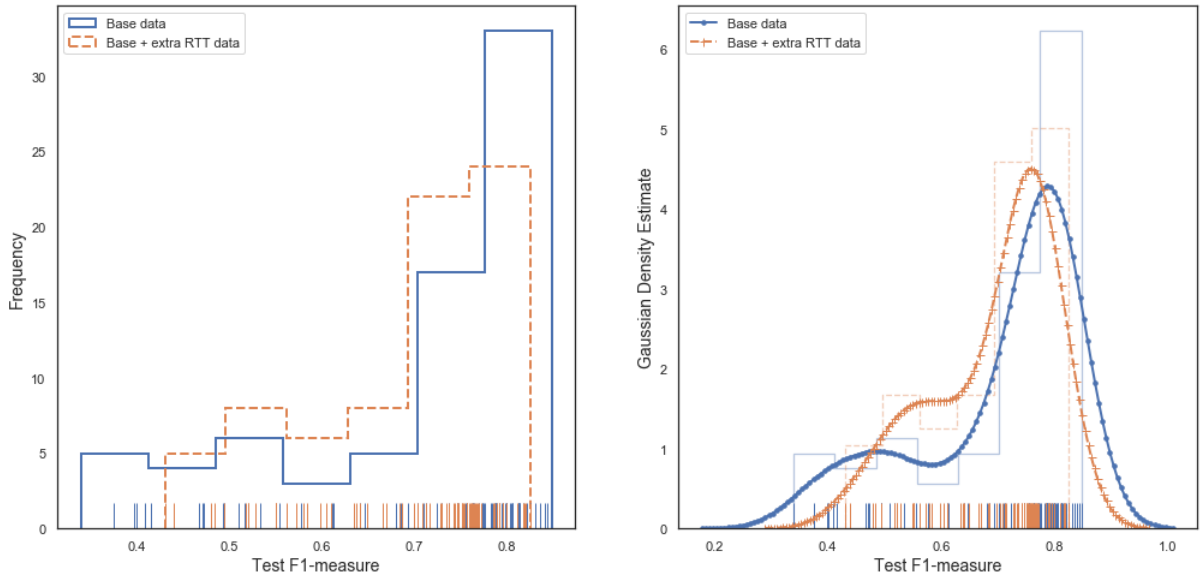


The next section will evaluate the success of using the extra RTT data in an attempt to increase the dataset size and to avoid model overfitting, which can easily occur with small datasets.

5.2.5 Base Data vs. Extra RTT Data

Most models performed best using the smaller base dataset, except for MNB which performed the best with LSA TF-IDF unigram features, built on the extended RTT dataset (refer to Table 5.3). In Figure 5.11 a comparison is shown for the performance of all 146 models grouped by the data type used (base data or the extra RTT data).

Figure 5.11: Base data vs. extra RTT data with histograms and Gaussian density functions



Some interesting observations observed when using the extra RTT data were that the minimum test F1-measure increased by 9.1% (from 34% to 43.1%) while the maximum test F1-measure decreased by only 2.3% (from 84.4% to 82.5%). The descriptive statistics for test F1-measures, grouped by the data type, are shown in Table 5.20.

Table 5.20: Data type summary with descriptive statistics for test F1-measures

	Test F1-measure Statistics								
Data Type	**Count All**	**Count in Top 20%**	**Min**	**Mean**	**Max**	**Std**	**P25**	**P50**	**P75**
Base data	73	23	0.340	0.703	0.848	0.140	0.641	0.770	0.803
Base + extra RTT data	73	7	0.431	0.693	0.825	0.106	0.610	0.742	0.766

The standard deviation of the test F1-measures is less for the extra RTT data compared to the base data, but all percentile values were higher for the base data. Also, the base data had a total of 23 models in the top 20% of all 146 models, while there were only 7 models in the top 20% for the extra

RTT data. From these observations one can conclude that there are no substantial benefits of using the extra RTT data.

5.3 Summary

In this chapter the approach for training and testing models was described and the results of this study were discussed. A total of 146 models for predicting the question category were compared, of which the top performer was an MLR model with TF-IDF unigram and bigram features, built on the smaller base dataset and having a test F1-measure of 84.8%.

The first challenge for a QA system is to correctly predict the question category, which the MLR model managed to do successfully. The second challenge is to extract the correct answer and for this task the cosine similarity was calculated using LSA topics built from TF-IDF unigram features. Using the LSA topics for cosine similarity resolved the “curse of dimensionality”, as mentioned in Section 4.8 and Section 5.2.2. The success of answering questions in each of the 16 question categories was evaluated, firstly through a random selection of one test question per category and the application of human evaluation to the extracted answer (refer to Table 5.9) and secondly by evaluating new questions posted on the QA simulator and applying human evaluation to the extracted answers, in combination with using the user feedback of the ratings provided for a correct or incorrect answer. Correct answers were only possible if a similar past question for the new question existed. Based on the evaluation done for new questions posted on the simulator, 55% of extracted answers where classified as correct.

Using thresholds for both the question category probabilities and the past question similarities assisted in determining if a question could be answered with a reasonable confidence level. Where thresholds could not be reached, a standard answer of “*The confidence level for the predicted category is too low to attempt answering this question automatically. The course convener will be notified of your question.*” was extracted.

In evaluating the success of predicting multiple question categories, the F1-measure was more informative than AUC. The best prediction performance for question categories was achieved for the “IntroStat Collection”, “Videos” and “Winter/Summer Term” categories, while the “Content” category had the worst performance.

It was determined that it is possible to prioritise questions by deriving the sentiment of the question text, where questions with the lowest sentiment should be answered first. It was however concluded that questions with a neutral or positive sentiment should rather be answered in the order in which the questions were received.

No significant improvements were found when using the extended dataset created by using RTT as an augmentation method for text. An evaluation was however only done for the main task of predicting the question category, and the evaluation to determine if RTT holds any benefits for finding the most similar past question has been left for future study.

The outcome of the results is that the main question raised by this study − to determine if it would be possible to use the available data to create a QA system which can automatically answer administrative queries with reasonable results − can be answered positively.

In the next chapter the conclusion for this study will be presented.

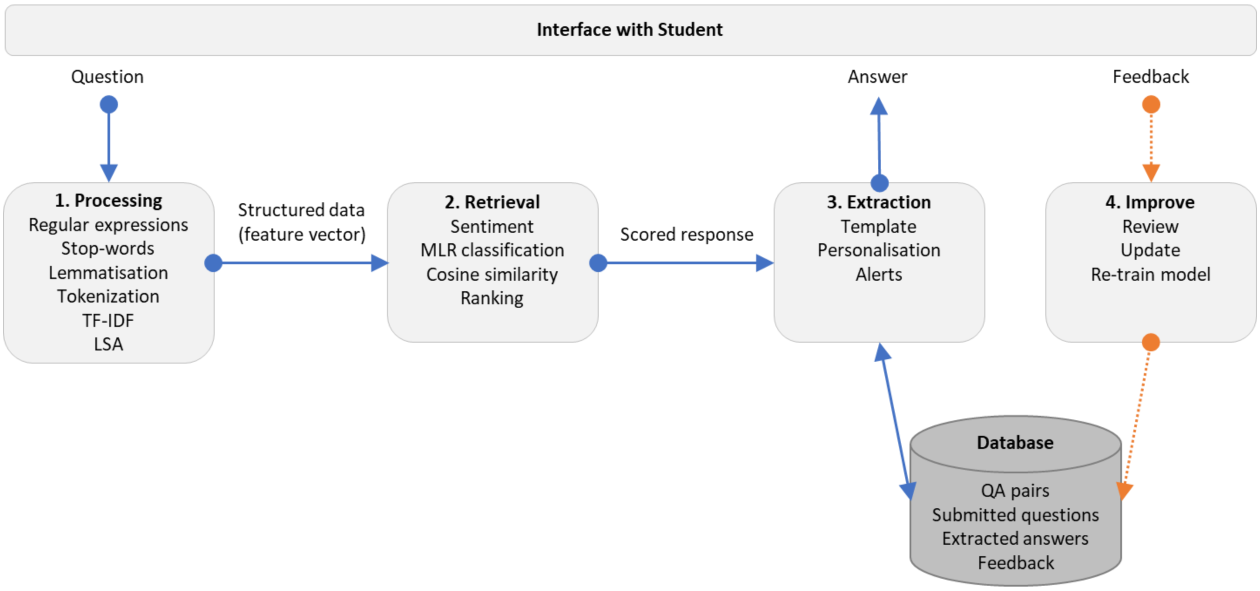
6. Conclusion

This chapter concludes this study, in which the architecture and design needed for a QA system to automatically answer administrative queries for the STA1000 course has been discussed. The architecture describes how the reviewed literature was used in constructing the QA system, and the design describes how all the related components were brought together to create a practical solution. In this final chapter, the findings and limitations of this study are presented and recommendations for future study are made.

6.1 Architecture and Design

This section provides a summary of how the architecture and aspects required for constructing a QA system (as described in Chapter 2) were applied to create the implemented solution to automatically answer administrative queries for the STA1000 course. Figure 6.1 shows the high-level design of the core architecture, featuring the macro modules of question processing, document retrieval and answer extraction. An extra module has been added to help improve the system over time, by allowing users to provide feedback on the answers extracted.

Figure 6.1: Conceptual design for implemented QA system



In the following sections the additional aspects that were needed to implement this design will be discussed, namely: the domain, paradigm, architecture, algorithms and metrics needed to evaluate the reliability of answers generated by the QA system.

Domain and Paradigm

The type of questions requiring answering by the QA system need to be restricted to administrative queries for the STA1000 course taught at UCT. Therefore, a closed-domain was selected instead of an open-domain and this should prevent the system from attempting to provide answers to questions that fall outside of the 16 question categories, as described in Chapter 3 and listed in Table 3.9. The thresholds that were set for question category probabilities and past question similarities (described in Section 5.2.3) are a means of enforcing this closed-domain restriction.

The paradigm or approach that was used for the implemented system is NLP. The other options available were IR, KB or a hybrid approach. A decision was made not to use the IR paradigm, as it looks to find short text segments within a collection of documents or on the World Wide Web, and the requirement for this study was to return full answers and not only fragments. The KB paradigm requires a structured database and was therefore not selected due to the additional effort that would have been required to design and create such a database. The hybrid approach is a more complex one and, in this case, fell outside of the scope for this dissertation due to the level of complexity, but may be a consideration for future study as it tends to provide better results.

Architecture

A standard QA system architecture consisting of three macro modules (question processing, document retrieval and answer extraction) was used. For the question processing module, the option to derive a question type in terms of factoid, confirmation, causal or complex was decided against, as the scope of questions had already been restricted to a closed-domain and these question types are more relevant to an open-domain QA system. A basic document retrieval module consisting of retrieving all past questions for a predicted category and finding the most similar past question to the new question was selected. A basic answer extraction module was chosen, one that would return the full answer to the most similar past question asked. An alternative would have been to split the text of past answers into smaller passages and the answer extraction module would then have selected the most appropriate passage (or combination of passages) as an answer to the new question. This alternative was not used due to the level of complexity involved.

An extra module was added to allow feedback on the answer extracted. Here, the user could rate the accuracy of the predicated answer as either correct or incorrect, as well as provide any comments on the predicted answer, or even propose a more suitable answer. This extra module proved useful during testing, where students, lecturers and the course convener were asked to test different versions of the model. It also allowed for model improvements. A database was used to store the submitted question, extracted answer, predicted question sentiment, predicted question category, most similar past question and optional feedback from the user in terms of rated accuracy of the answer, plus any comments or proposals for a better suited answer. This database can be queried at any time to view any of the stored data, and could be used to improve the system over time.

Techniques, Algorithms and Tools

The techniques, algorithms and tools that were selected for the implemented solution are a combination of regular expressions, stop-words, lemmatisation, tokenization, TF-IDF features, LSA topics, MLR classifier, cosine similarity and sentiment analysis.

The regular expressions were used for data pre-processing as described in Section 3.2. The full list of phrase replacements is listed in Appendix A.2. A custom set of stop-words was removed and this set is listed in Appendix A.3. Lemmatisation also forms part of data pre-processing, and the NLTK WordNet library was used for this purpose (Bird, Klein & Loper, 2009).

Tokenization was used to turn the submitted question into a combination of unigrams and bigrams, after first having applied pre-processing to it. These unigrams and bigrams were then converted into 5 688 TF-IDF features as described in Section 4.2, which provided the input for the MLR classifier to predict the question category outlined in Section 5.2.1.

Cosine similarity was used to find the most similar past questions based on LSA topics. The compared questions were first converted to TF-IDF unigrams and then reduced to 55 LSA topics as described in Section 4.4. Using LSA features instead of the TF-IDF features resolved the “curse of dimensionality” as mentioned in Section 4.8 and Section 5.2.2.

Sentiment analysis was used to identify questions with a negative sentiment, in order to assist in prioritising multiple questions (Section 5.2.4). The raw question text was used as input to the VADER library and thresholds were applied to classify the sentiment score as either negative, neutral or positive.

Evaluation Metrics

The F1-measure with macro-averaging was used to evaluate 146 models in predicting question categories (Section 5.2.1), and human evaluation was used to evaluate the success of finding the most similar past question (Section 5.2.2) and extracting the most relevant answer (Section 5.2.3).

The next section will discuss the findings and limitations of this study.

6.2 Findings and Limitations

No substantial benefits were found when using the extra RTT data, instead the best performing classifier proved to be MLR with 5 688 TF-IDF unigram and bigram features (built on the smaller 744 questions dataset), with a test F1-measure of 84.8%. However, for the purpose of finding the most similar past question, the best performing features were 55 LSA topics built from the TF-IDF unigram features.

Predicting the question category is the most important task of a QA system. A total of 146 models were compared for the prediction of the question category and in this case MLR performed the best. The SGD model type produced the most versatile classifiers, while the MNB model type demonstrated the worst performance. The feature type with the best performance turned out to be TF-IDF unigrams and bigrams, while LDiA features displayed the worst performance. The BoW count features had the highest 25th, 50th and 75th percentile values out of all feature types, but also had the highest standard deviation for the test F1-measure.

During evaluation of the extracted answers an issue with data pre-processing was discovered where some numbers were not converted to tokens. This issue only occurred in the case of some questions and an investigation is suggested to find and correct the cause of this issue. For questions in the "Tests" category, the test number mentioned in a question plays an important role in determining the best answer for the question, so fixing this issue would improve the performance in answering test-related questions.

The current version of the QA system is not suited to be used in practice because of the high likelihood of answering a question incorrectly. This is due to the small dataset on which the QA system was trained. If a new question does not match a similar past question, then the probability for not being able to answer the question, or extracting an incorrect answer, becomes very high.

The biggest limitation in the design of the QA system is that past answers are extracted "as is" and cannot be adjusted for a different context based on a comparison between the context of the new question and the most similar past question asked. For example, in the case of the new question "*If we failed test 1, will we need to attend a workshop this week?*", which had the extracted answer of "*No, but should you fail Test1* [*sic*]*, then it will become compulsory for you to attend one workshop a week (in addition to tutorials)*", the system has no means of knowing that the first part of the extracted answer "*No, but*" should be removed, as it is only relevant to the past question "*I failed test 1 but passed test 2 - am I still obliged to attend the weekly workshops?*". All past answers were reviewed, rewritten and standardised prior to model building in an attempt to make these past answers more general and reusable. The original answers were too specific and did not generalise well, but the evaluation of the extracted answers revealed that further generalisation can be applied to all past answers. A suggestion to lecturers who are currently answering administrative queries on the STA1000 Vula website, would be to keep potential reuse of answers in mind and to write answers in a more standard and precise way.

The current design also does not make provision for synthesising answers from more than one past question, meaning that it cannot answer combined questions and that only one question can be answered at a time.

6.3 Recommendations for Future Study

A number of ways in which the QA system could be improved have been observed. The first would be to correct the issue with numbers being dropped from some questions during data pre-processing. Also, a data pre-processing step to correct spelling should be considered, as spelling errors in new questions would result in the inability to match misspelled words with correctly spelled words in past questions. The TextBlob correct function was investigated for this purpose, but it made some errors such as changing the word "where" to "there" when a sentence began with this word.

The QA system cannot currently answer combination questions and more work is needed to make provision for this. Instead of a basic answer extraction module where the full answer for the most similar past question is used "as is", an alternative could be to consider splitting the text of past answers into smaller passages. In this way, the answer extraction module could then select the most appropriate passage (or combination of passages) as an answer to the new question. In addition to this, the use of information extraction could be considered to build a KB for specific facts, for example, who the current course convener is, the name of the venue where lectures take place, etc. Using a KB would also allow for updating answers over time, for example, if a new course convener was appointed.

The extra RTT data was only tested for the task of predicting the question category. An investigation to test if the extra RTT data would provide any benefits for the task of selecting the most similar past question could be considered.

The engineered features were also only tested for the task of predicting the question category, and for this purpose they performed well. Combining the engineered features with the LSA features may work well for cosine similarity, as the total number of features would not increase significantly and so the “curse of dimensionality” could still be avoided, but advantage could still be taken of the added domain knowledge.

Questions which are similar but semantically very different from one another are difficult to identify. To solve this problem, it would be useful to determine the similarity between answers, with the assumption that if two answers are similar then their corresponding questions should also be similar.

Instead of using cosine similarity, which favours short documents and often leads to the selection of the wrong past question, more advanced language-modelling-based similarity measures should be considered. Either pre-trained word embeddings, like GloVe, could be used for a language model, or a custom language model could be trained. A generic UCT language model could be valuable and it would be preferable to build such a language model using the entire digital content of Vula across all faculties where students converse, ask questions or post comments. Building a QA system using this UCT language model is foreseen to offer better results, as terms like “Vula”, “DP”, “SciLab” and “jdlt1” would be endowed with more meaning.

If more data can be sourced, then it would make sense to use a deep learning model. Some of the advantages would be that deep learning models can learn better semantic similarity between questions and that deep learning models could also be used for answer generation (especially transformers). This will greatly expand on the problem of retrieving the correct answer or even generating a new answer.

This study only considered answering questions asked in English because the data available were limited to English questions and answers. It would be valuable if a QA system could be able to answer questions asked in any of the 11 official languages. To support a QA system in other languages the question language first needs to be identified (Duvenhage, 2019), after which the Google Translate API can be used to translate the question to English. Alternatively, additional input data can be sourced in local languages.

6.4 Closing Remarks

This study has shown that it is indeed possible to use the available data to create a QA system that can automatically answer administrative queries, with reasonable results, for the STA1000 course. There are a number of ways in which the QA system can be improved, and these have been listed in Section 6.3, “Recommendations for Future Study”.

Building a QA system using a small dataset proved to be a challenging task, as it can become a complex undertaking with all the processing steps required. Restricting the domain of questions and applying multiple data pre-processing steps proved valuable.

The initial feedback from the course convener on the performance of the QA system was not positive, but the feedback on the final version was well received, the comment being, "*Wow, the simulator is pretty impressive! I asked a number of questions and was very happy with all the answers!*".

Appendices

This appendix chapter lists additional information supporting the main body of this dissertation.

A.1 Method for Sourcing Additional Tutor Questions

Due to the small number of observations, an additional 142 questions were sourced by asking tutors to provide questions often asked by students. The following approach was taken to ensure that no bias was introduced with this additional data:

- A target of 10 new questions per tutor was set, which would result in a total of 130 new questions for the 13 tutors of the current academic year.
- The frequency distribution of the 509 administrative queries, across different categories such as exam related questions, were determined (refer to Table 3.9 in Section 3.3 on labelling for the list of all categories).
- The tutors were randomly assigned to create new questions for 3 to 5 categories, while ensuring that the frequency distribution of the new questions matched the frequency distribution of the original 509 questions.
- One tutor opted to provide a total of 20 questions instead of 10.
- Only tutors provided the questions. These questions were manually answered by the researcher and reviewed by the course convener.
- The frequency distribution of the additional 142 questions was compared against the frequency distribution of the original 509 questions to ensure that they were the same.

Note that for the engineered feature “Course week number” values had to be assigned manually for these extra tutor questions. Week numbers were assigned by looking at the most prominent week for related questions in the base dataset of 509 questions.

A.2 Phrase Replacements

During data pre-processing phrase replacements were necessary to clean the data. Two types of phrase replacements were needed, namely contractions and general replacements.

List of phrase replacements related to contraction:

"aren't" → 'are not'
"can't" → 'can not'
"cannot" → 'can not'

"don't" → 'do not'
"didn't" → 'did not'
"doesn't" → 'does not'
"hasn't" → 'has not'
"haven't" → 'have not'
"hadn't" → 'had not'
"isn't" → 'is not'
"won't" → 'will not'
"couldn't" → 'could not'
"wouldn't" → 'would not'
"shouln't" → 'should not'
"could've" → 'could have'
"would've" → 'would have'
"should've" → 'should have'
"couldn've" → 'could not have'
"wouldn've" → 'would not have'
"wasn't" → 'was not'
"weren't" → 'were not'
"we'd" → 'we had'
"you've" → 'you have'
"i'm" → 'i am'
"i've" → 'i have'
"it's" → 'it is'
"there's" → 'there is'

List of general phrase replacements:

'lws' → 'workshop'
'long workshops' → 'workshop'
'long workshop,' → 'workshop'
r'long workshop\.' → 'workshop'
'long workshop' → 'workshop'
'short/long workshop' → 'workshop'
'sws' → 'lecture'
'short workshops' → 'lecture'
'short workshop,' → 'lecture'
r'short workshop\.' → 'lecture'
'short workshop' → 'lecture'
'classes' → 'lecture'
'class' → 'lecture'
'hot seats' → 'hotseat'
'hot seat' → 'hotseat'
'hotseats' → 'hotseat'
'hot desk' → 'hotseat'
'help sessions' → 'hotseat session'
'help session' → 'hotseat session'
'microsoft excel' → 'excel'
r'tuts\.' → 'tutorial'
'tuts,' → 'tutorial'
'tuts' → 'tutorial'
'tut,' → 'tutorial'

r'tut\.' → 'tutorial'
'tut' → 'tutorial'
r'dr\.scott' → ''
r'dr\. scott' → ''
r'prof\.scott' → ''
r'prof\. scott' → ''
'hi there' → ''
'hi leanne' → ''
'hi kim' → ''
'hi celene' → ''
'hi stefan' → ''
'hi' → ''
'howzit' → ''
'hello sir/madam' → ''
'hello sir' → ''
'hello madam' → ''
'hello' → ''
r'dear mrs\. jansen-fielies' → ''
'dear leanne' → ''
'dear kim' → ''
'dear celene' → ''
'dear stefan' → ''
'to whom this may concern' → ''
'to whom it may concern' → ''
'goodevening' → ''
'good evening' → ''
'evening' → ''
'goodday' → ''
'good day' → ''
'goodafternoon' → ''
'good afternoon' → ''
'afternoon' → ''
'goodmorning' → ''
'good morning' → ''
'morning' → ''
'greetings' → ''
'youve' → 'you'
'vula' → 'webpage'
'introstats' → 'textbook'
'introstat' → 'textbook'
'handbook' → 'textbook'
'scilab' → 'lab'
'comlab' → 'lab'
'jdlt1' → 'venue'
'sta1000f' → 'course'
'sta1000s' → 'course'
'stats1000' → 'course'
'stats1000s' → 'course'
'sta1100' → 'course'
'sta1100s' → 'course'
'sta1000' → 'course'

'doc' → 'doctor'
'dr' → 'doctor'
'examination' → 'exam'
'examined' → 'exam'
'examine' → 'exam'
'iam' → 'i am'
'percent' → 'percentage'
'pracs' → 'practical'
'prac' → 'practical'
'quizz' → 'quiz'
'quizes' → 'quiz'
'mednotes' → 'medical note'
'mednote' → 'medical note'
'first' → 'one'
'mondays' → 'monday'
'tuesdays' → 'tuesday'
'wednesdays' → 'wednesday'
'thursdays' → 'thursday'
'fridays' → 'friday'
'saturdays' → 'saturday'
'sundays' → 'sunday'
'12 am' → '12h00'
'12am' → '12h00'
'12 pm' → '12h00'
'12pm' → '12h00'
'1 pm' → '13h00'
'1pm' → '13h00'

A.3 Custom Stop-words

The following custom list of 89 stop-words was used instead of a standard library as it was important to have full control over which words should be seen as stop-words:

'a', 'about', 'after', 'all', 'almost', 'also', 'am', 'among', 'an', 'and', 'are', 'as', 'at', 'be', 'been', 'but', 'by', 'can', 'could', 'dear', 'did', 'do', 'doe', 'does', 'for', 'from', 'get', 'got', 'had', 'has', 'have', 'he', 'her', 'hers', 'him', 'his', 'however', 'i', 'if', 'in', 'im', 'm', 'into', 'is', 'it', 'its', 'just', 'let', 'may', 'me', 'might', 'most', 'my', 'of', 'off', 'on', 'or', 'our', 'please', 'said', 'say', 'says', 'she', 'should', 'since', 'so', 'some', 'than', 'that', 'the', 'their', 'them', 'then', 'there', 'these', 'they', 'this', 'to', 'too', 'us', 'want', 'wants', 'was', 'we', 'were', 'while', 'with', 'would', 'yet', 'you', 'your'

A.4 Keyword Features

The following list of 47 features specific to question categories was identified by analysing the top 5 most frequent occurring words per question category, while ignoring all words which had already been included in question type features, negation features or stop-words features:

'access', 'answer', 'assignment', 'attend', 'clash', 'collect', 'compulsory', 'copy', 'course', 'dp', 'email', 'exam', 'feedback', 'find', 'hand', 'hotseat', 'lab', 'lecture', 'lecturer', 'mark', 'medical', 'note', 'question', 'quiz', 'requirement', 'school', 'session', 'sign', 'slot', 'student', 'submit', 'summer', 'term', 'test', 'textbook', 'time', 'tutor', 'tutorial', 'venue', 'video', 'watch', 'way', 'webpage', 'winter', 'work', 'workshop', 'write'

Note that these words were identified after the eight data pre-processing steps were performed as listed in Table 3.6.

A.5 Selecting the Number of Topics for LSA and LDiA

To find the number of topics for which the LSA and LDiA models performed best, a comparison was made between the topic coherence scores and the test F1-measure values per number of topics. For LSA, which normally requires fewer topics than LDiA, topic numbers between 1 and 100 with a step size of 2 were evaluated. For LDiA, the number of topics evaluated was between 1 and 200, with a step size of 5. The Gensim library was used to build the LSA and LDiA models, as well as to calculate the topic coherence scores (Röder, Both & Hinneburg, 2015).

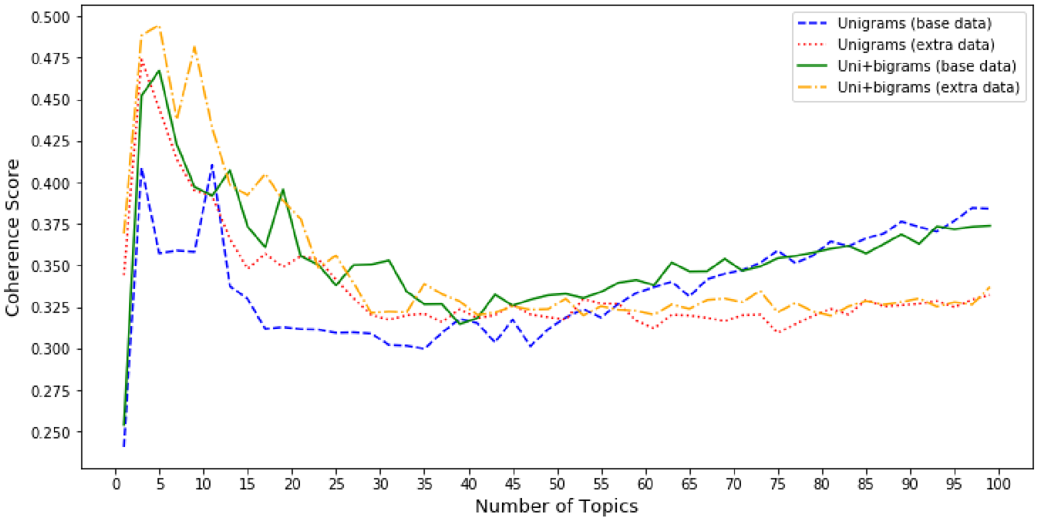
The process followed for calculating the test F1-measure values was to build an LSA or LDiA model with N number of topics, where the input is the test questions that have been converted to either BoW or TF-IDF features. The resulting topic values are then used as input features for an MLR model where the question category is predicted and the test F1-measure are calculated. The MLR model was used to generate the test F1-measure values as this model proved to perform the best on all other feature types.

The standard LSA topic values (referred to as “raw values” in this study) can be negative, and therefore cannot be used as input for the MNB classifier as it assumes a multinomial distribution. For this reason, the standard LSA topic values were also converted to probabilities in order to create a new feature type, which could be tested across all five model types that were evaluated in this study.

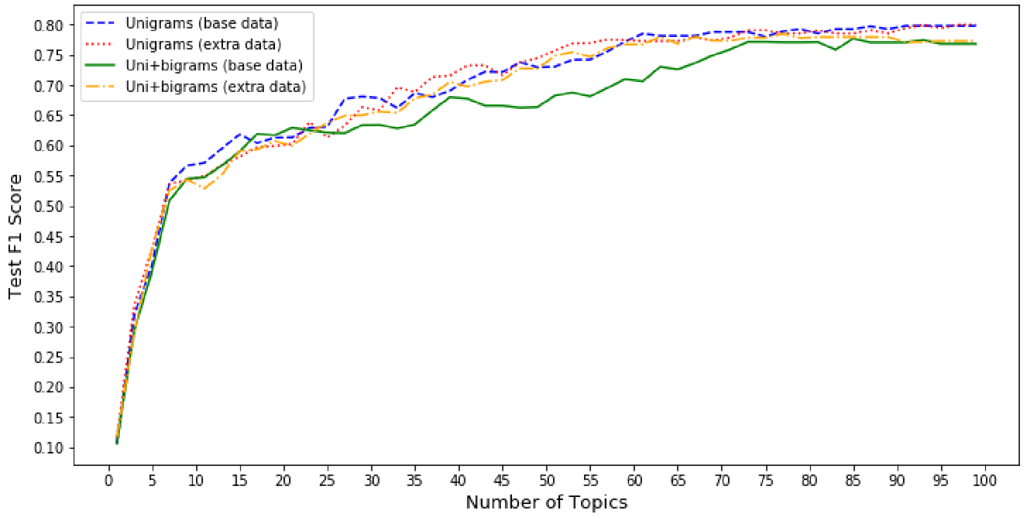
In Figure A.1 the evaluation results are presented for LSA using BoW features as input, with a comparison between the topic coherence scores, the test F1-measure values with raw LSA topic values as input, and the test F1-measure values with LSA topic values converted into probabilities. Four types of BoW features are also compared, namely unigrams and the combination of unigrams with bigrams, as well as using the base dataset and the extended RTT dataset (referred to as “extra data”).

Figure A.1: Evaluation of number of Topics for LSA using BoW

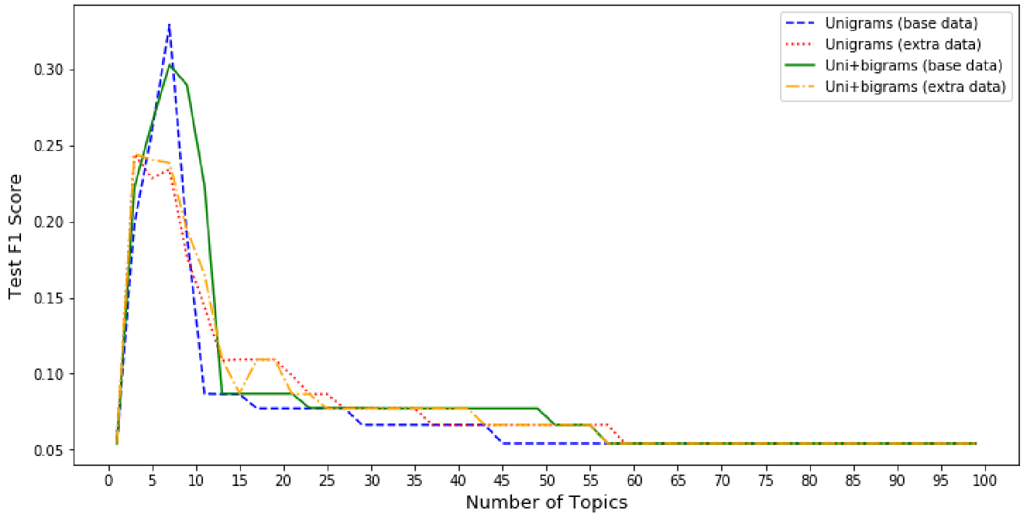
a) Topic Coherence per number of topics



b) Test F1-measure (raw values) per number of topics



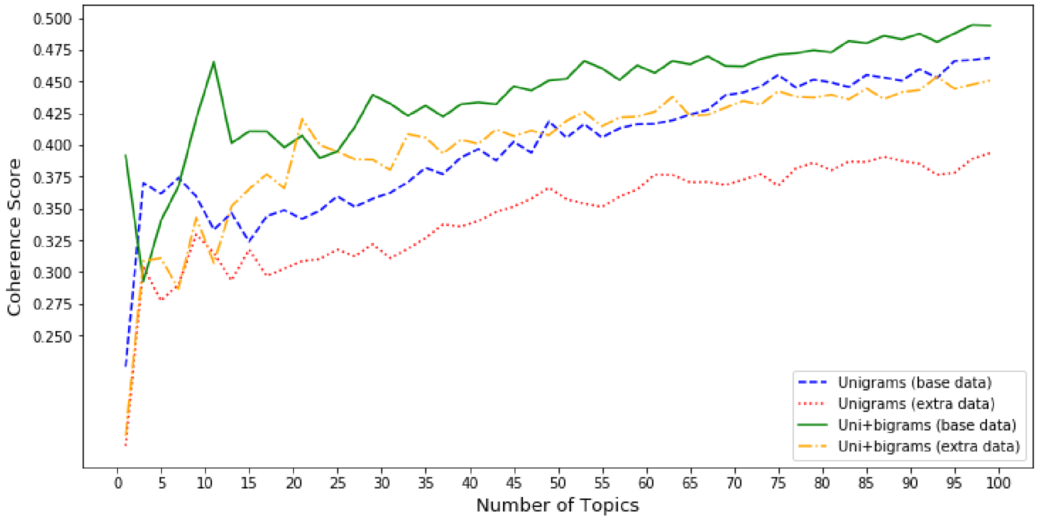
c) Test F1-measure (probabilities) per number of topics



In Figure A.2 the evaluation results are presented for LSA using TF-IDF features as input, with a comparison between the topic coherence scores, the test F1-measure values with raw LSA topic values as input, and the test F1-measure values with LSA topic values converted into probabilities.

Figure A.2: Evaluation of number of Topics for LSA using TF-IDF

a) Topic Coherence per number of topics



b) Test F1-measure (raw values) per number of topics

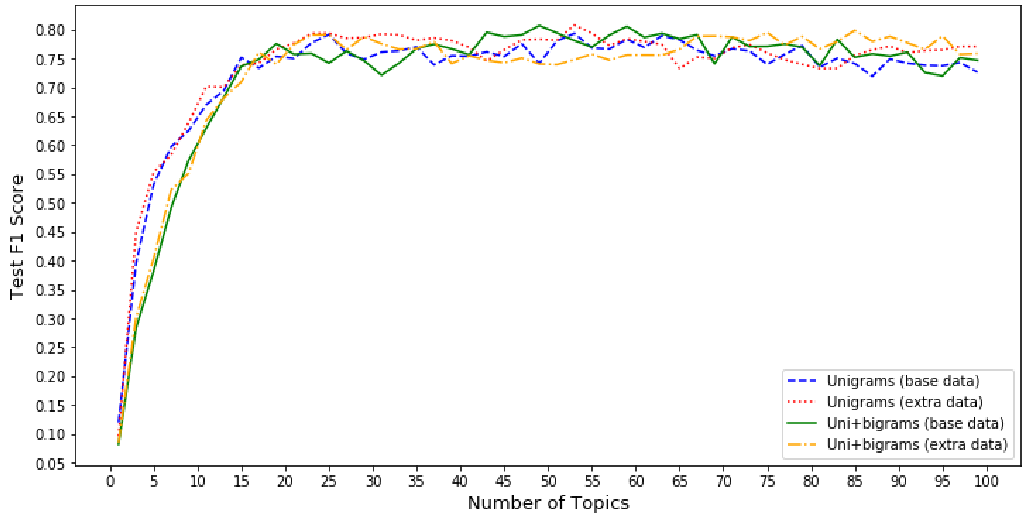
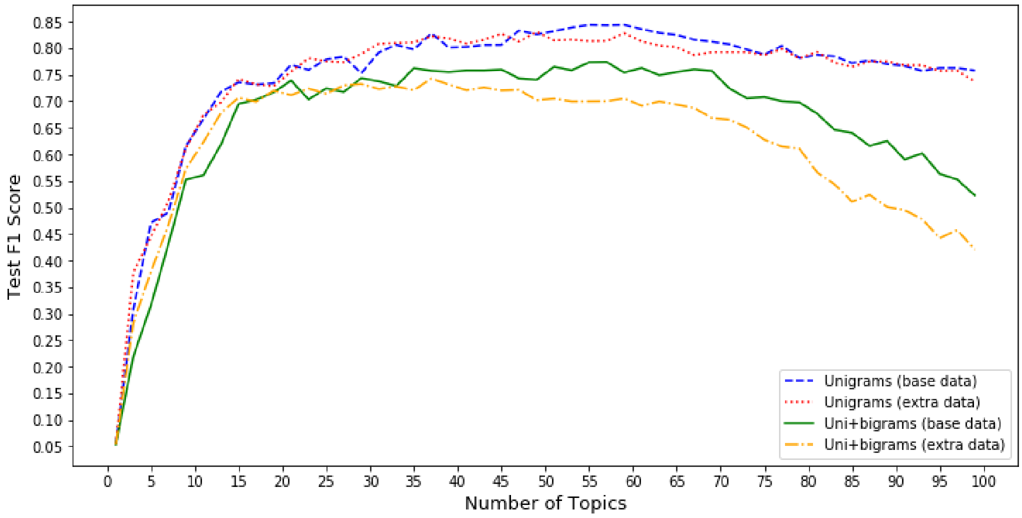


Figure A.2: Evaluation of number of Topics for LSA using TF-IDF (continued)

c) Test F1-measure (probabilities) per number of topics



In Figure A.3 the evaluation results are presented for LDiA using BoW features as input, with a comparison between the topic coherence scores and the test F1-measure values with raw LDiA topic values as input. Note that only BoW features and not TF-IDF features can be used for an LDiA model.

Figure A.3: Evaluation of number of Topics for LDiA using BoW

a) Topic Coherence per number of topics

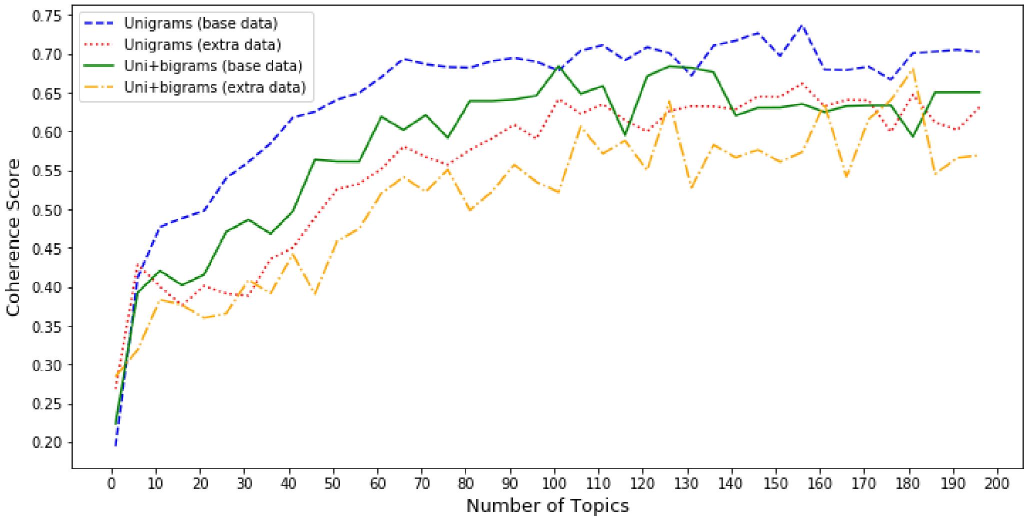
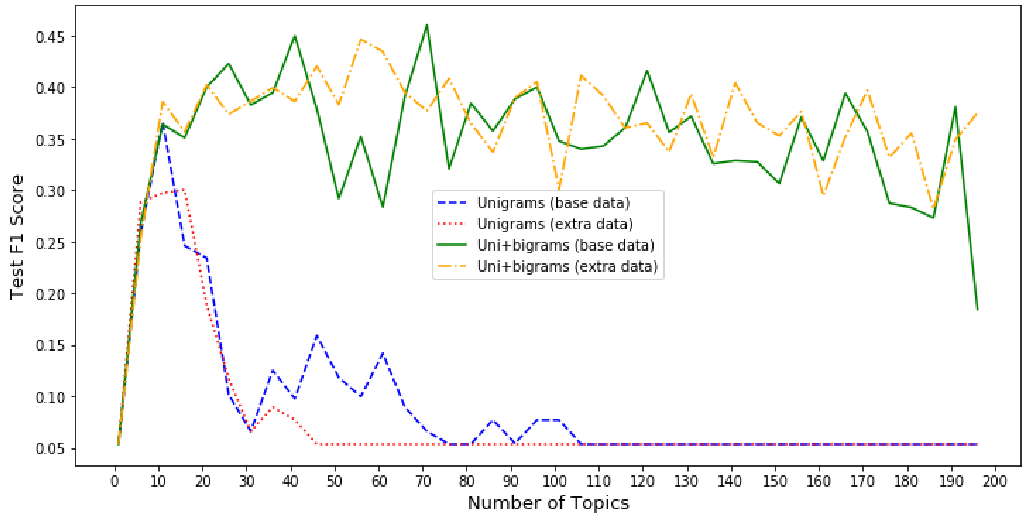


Figure A.3: Evaluation of number of Topics for LDiA using BoW (continued)

b) Test F1-measure (raw values) per number of topics



Various tests were done to compare the performance of the MLR classifier based on the best coherence values. For example, for the BoW unigram and bigram features, the best topic coherence score was 0.467 for 5 LSA topics (Figure A.1a), which resulted in a test F1-measure of only 0.391 for the MLR classifier built on these 5 LSA features (Figure A.1b). However, if one decided to ignore the topic coherence and only consider the top F1-measure, then the outcome would be to select 85 LSA features, which would result in a much better test F1-measure of 0.777 (Figure A.1b). For this reason, a decision was made to use the number of topics that performed the best for the test F1-measure regardless of what the topic coherence score turned out to be.

A summary of the suggested number of topics for LSA and LDiA as a result of using the best topic coherence scores or the top test F1-measures is shown in Table A.1.

Table A.1: Summary of suggested number of topics for LSA and LDiA

			Suggested Number of Topics			
			for LSA based on		for LDiA based on	
Input Feature			**Topic Coherence**	**Test F1-measure**	**Topic Coherence**	**Test F1-measure**
BoW	Unigrams	Base data	11	91	156	11
		Extra data	3	97	156	16
	Uni+bigrams	Base data	5	85	101	71
		Extra data	5	63	181	56
TF-IDF	Unigrams	Base data	99	55	N/A	N/A
		Extra data	99	49	N/A	N/A
	Uni+bigrams	Base data	97	49	N/A	N/A
		Extra data	93	85	N/A	N/A

The final selected number of topics for LSA and LDiA were those based on the test F1-measure.

A.6 Model Results for Predicting the Question Category

This section contains the training and testing detail for all models compared to predict the question category. More than 350 different model versions were compared when counting the initial default models with the subsequent tuned model versions. Table A.2 lists the final results where only the top performing model was selected per model type and feature type. The result was 146 unique models for the combination of five model types and 30 feature types. The MNB model type could only be combined with 26 feature types, because the algorithm used is not able to deal with negative numbers, which can occur in the feature types values for raw LSA or GloVe (both base and extra RTT data versions of these feature types).

Table A.2: List of 146 question category prediction models ranked by performance

				F1-measure		
Model	**Feature Type**	**# Samples**	**# Features**	**Train**	**Test**	**Rank**
MLR	TF-IDF unigrams + bigrams	587	5 688	0.978	0.848	1
MLR	LSA TF-IDF unigrams (probabilities)	587	55	0.844	0.844	2
MLR	BoW counts	587	1 101	0.934	0.841	3
SGD	TF-IDF unigrams (SGD with Least-Squares loss)	587	1 101	0.951	0.835	4
MLR	LSA TF-IDF unigrams (raw values)	587	55	0.886	0.831	5
SGD	TF-IDF unigrams + bigrams (Modified Huber loss)	587	5 688	0.992	0.825	6
SGD	BoW counts (Modified Huber loss)	587	1 101	0.942	0.825	7
MLR	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.783	0.825	8
SGD	TF-IDF unigrams + bigrams + trigrams (Hinge loss)	587	10 972	0.998	0.821	9
MNB	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.723	0.820	10
SGD	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.754	0.819	11
SGD	TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	0.977	0.817	12
SGD	BoW binary values (Modified Huber loss)	587	1 101	0.947	0.817	13
MLR	TF-IDF unigrams + bigrams + trigrams	587	10 972	1.000	0.814	14
SGD	Engineered features (Modified Huber loss)	587	61	0.911	0.813	15
SVM	LSA BoW unigrams	587	91	0.966	0.812	16
MLR	Engineered features	587	61	0.905	0.812	17
SVM	BoW counts	587	1 101	0.981	0.812	18
MLR	TF-IDF unigrams	587	1 101	0.946	0.812	19
SGD	TF-IDF unigrams + bigrams + extra RTT data (Hinge)	6 457	22 148	0.972	0.811	20
RF	TF-IDF unigrams + bigrams	587	5 688	1.000	0.806	21
RF	BoW counts	587	1 101	1.000	0.806	22
RF	BoW binary values	587	1 101	1.000	0.804	23
MLR	BoW binary values	587	1 101	0.936	0.803	24
RF	TF-IDF unigrams + bigrams + extra RTT data	6 457	22 148	0.998	0.800	25
MLR	LSA BoW unigrams	587	91	0.922	0.800	26
SVM	LSA BoW unigrams + bigrams	587	85	0.956	0.799	27
RF	TF-IDF unigrams	587	1 101	1.000	0.799	28
SVM	TF-IDF unigrams + bigrams + trigrams	587	10 972	1.000	0.798	29
MLR	TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	0.997	0.796	30
SVM	Engineered features	587	61	0.938	0.795	31
SGD	TF-IDF unigrams + extra RTT data (Log loss)	6 457	2 263	0.935	0.795	32
MLR	LSA TF-IDF unigrams + bigrams	587	49	0.824	0.794	33
RF	TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	0.998	0.792	34
SVM	TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	0.996	0.792	35
MLR	TF-IDF unigrams + bigrams + extra RTT data	6 457	22 148	0.997	0.792	36

Table A.2: List of 146 question category prediction models ranked by performance (continued)

				F1-measure		
Model	**Feature Type**	**# Samples**	**# Features**	**Train**	**Test**	**Rank**
RF	TF-IDF unigrams + bigrams + trigrams	587	10 972	1.000	0.789	37
SGD	LSA TF-IDF unigrams + extra RTT data (raw values)	6 457	49	0.753	0.789	38
SVM	LSA TF-IDF unigrams (raw values)	587	55	0.950	0.788	39
MLR	LSA BoW unigrams + bigrams	587	85	0.898	0.788	40
SVM	BoW binary values	587	1 101	0.973	0.788	41
SGD	LSA TF-IDF unigrams (raw values)	587	55	0.882	0.787	42
RF	TF-IDF unigrams + extra RTT data	6 457	2 263	0.998	0.786	43
SGD	BoW counts + extra RTT data	6 457	2 263	0.973	0.785	44
SVM	LSA TF-IDF unigrams (probabilities)	587	55	1.000	0.784	45
SVM	TF-IDF unigrams	587	1 101	0.974	0.783	46
SVM	TF-IDF unigrams + bigrams	587	5 688	1.000	0.776	47
SGD	LSA BoW unigrams	587	91	0.953	0.775	48
SGD	LSA BoW unigrams + bigrams	587	85	0.943	0.773	49
MNB	LSA TF-IDF unigrams (probabilities)	587	55	0.828	0.773	50
SGD	BoW binary values + extra RTT data	6 457	2 263	0.970	0.772	51
MNB	LSA TF-IDF unigrams + bigrams + extra RTT data	6 457	85	0.764	0.770	52
SGD	LSA TF-IDF unigrams (probabilities)	587	55	0.811	0.770	53
RF	BoW counts + extra RTT data	6 457	2 263	0.998	0.768	54
SGD	LSA BoW unigrams + bigrams + extra RTT data	6 457	63	0.794	0.767	55
SVM	LSA TF-IDF unigrams + extra RTT data (raw values)	6 457	49	0.817	0.766	56
RF	BoW binary values + extra RTT data	6 457	2 263	0.998	0.765	57
SVM	LSA TF-IDF unigrams + bigrams	587	49	0.877	0.764	58
RF	LSA TF-IDF unigrams + bigrams	587	49	1.000	0.763	59
SGD	GloVe word embeddings	587	300	0.937	0.763	60
SGD	LSA TF-IDF unigrams + bigrams + extra RTT data	6 457	85	0.792	0.763	61
MLR	LSA TF-IDF unigrams + extra RTT data (raw values)	6 457	49	0.774	0.762	62
MLR	TF-IDF unigrams + extra RTT data	6 457	2 263	0.954	0.762	63
MLR	Engineered features + extra RTT data	6 457	65	0.819	0.760	64
RF	LSA TF-IDF unigrams + extra RTT data (raw values)	6 457	49	0.998	0.759	65
SGD	Engineered features + extra RTT data	6 457	65	0.782	0.757	66
SVM	TF-IDF unigrams + extra RTT data	6 457	2 263	0.998	0.757	67
MLR	LSA BoW unigrams + extra RTT data	6 457	97	0.867	0.756	68
MNB	Engineered features	587	61	0.863	0.756	69
SVM	GloVe word embeddings	587	300	0.991	0.755	70
RF	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.998	0.754	71
MNB	Engineered features + extra RTT data	6 457	65	0.764	0.753	72
MLR	BoW counts + extra RTT data	6 457	2 263	0.980	0.751	73
SVM	TF-IDF unigrams + bigrams + extra RTT data	6 457	22 148	0.996	0.751	74
MNB	LSA TF-IDF unigrams + bigrams	587	49	0.805	0.747	75
MLR	LSA TF-IDF unigrams + bigrams + extra RTT data	6 457	85	0.904	0.747	76
RF	LSA TF-IDF unigrams (raw values)	587	55	1.000	0.746	77
MLR	LSA BoW unigrams + bigrams + extra RTT data	6 457	63	0.815	0.746	78
SVM	LSA BoW unigrams + bigrams + extra RTT data	6 457	63	0.895	0.746	79
MLR	BoW binary values + extra RTT data	6 457	2 263	0.978	0.743	80
SGD	LSA BoW unigrams + extra RTT data	6 457	97	0.842	0.742	81
RF	Engineered features	587	61	1.000	0.737	82
RF	Engineered features + extra RTT data	6 457	65	0.995	0.737	83
SVM	LSA TF-IDF unigrams + bigrams + extra RTT data	6 457	85	0.990	0.734	84
RF	LSA TF-IDF unigrams (probabilities)	587	55	1.000	0.730	85
SGD	GloVe word embeddings + extra RTT data	6 457	300	0.857	0.730	86
RF	LSA TF-IDF unigrams + bigrams + extra RTT data	6 457	85	0.998	0.728	87
MLR	GloVe word embeddings	587	300	0.995	0.724	88
MLR	GloVe word embeddings + extra RTT data	6 457	300	0.921	0.716	89
SVM	LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	0.989	0.713	90

Table A.2: List of 146 question category prediction models ranked by performance (continued)

				F1-measure		
Model	**Feature Type**	**# Samples**	**# Features**	**Train**	**Test**	**Rank**
SGD	LSA TF-IDF unigrams + bigrams	587	49	0.827	0.713	91
RF	LSA BoW unigrams	587	91	1.000	0.710	92
SVM	Engineered features + extra RTT data	6 457	65	0.847	0.708	93
SVM	LSA BoW unigrams + extra RTT data	6 457	97	0.935	0.704	94
RF	LSA BoW unigrams + bigrams	587	85	1.000	0.703	95
SVM	BoW counts + extra RTT data	6 457	2 263	0.994	0.699	96
MNB	TF-IDF unigrams	587	1 101	0.945	0.693	97
SVM	BoW binary values + extra RTT data	6 457	2 263	0.997	0.686	98
MNB	TF-IDF unigrams + bigrams	587	5 688	0.990	0.685	99
MNB	BoW counts + extra RTT data	6 457	2 263	0.935	0.684	100
MNB	TF-IDF unigrams + bigrams + trigrams	587	10 972	0.997	0.680	101
SVM	GloVe word embeddings + extra RTT data	6 457	300	0.998	0.670	102
RF	LSA BoW unigrams + extra RTT data	6 457	97	0.998	0.666	103
MNB	TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	0.996	0.649	104
MNB	BoW counts	587	1 101	0.940	0.647	105
MNB	BoW binary values	587	1 101	0.945	0.641	106
RF	LSA BoW unigrams + bigrams + extra RTT data	6 457	63	0.998	0.641	107
MNB	BoW binary values + extra RTT data	6 457	2 263	0.939	0.638	108
RF	GloVe word embeddings + extra RTT data	6 457	300	0.998	0.635	109
RF	GloVe word embeddings	587	300	1.000	0.613	110
MNB	LSA BoW unigrams	587	91	0.654	0.611	111
MNB	TF-IDF unigrams + bigrams + extra RTT data	6 457	22 148	0.995	0.610	112
SGD	LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	0.554	0.605	113
MNB	LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	0.509	0.598	114
MNB	TF-IDF bigrams + extra RTT data	6 457	19 885	0.995	0.584	115
MLR	LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	0.565	0.580	116
MNB	LSA BoW unigrams + bigrams	587	85	0.629	0.578	117
SGD	TF-IDF bigrams + extra RTT data	6 457	19 885	0.986	0.573	118
MLR	TF-IDF bigrams + extra RTT data	6 457	19 885	0.997	0.562	119
MLR	TF-IDF bigrams	587	4 587	1.000	0.555	120
SVM	LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	0.536	0.551	121
MNB	TF-IDF unigrams + extra RTT data	6 457	2 263	0.956	0.551	122
MNB	LSA BoW unigrams + extra RTT data	6 457	97	0.580	0.550	123
SGD	TF-IDF bigrams	587	4 587	1.000	0.534	124
SVM	TF-IDF bigrams + extra RTT data	6 457	19 885	0.995	0.528	125
MNB	LSA BoW unigrams + bigrams + extra RTT data	6 457	63	0.561	0.521	126
RF	TF-IDF bigrams + extra RTT data	6 457	19 885	0.997	0.521	127
RF	LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	0.999	0.518	128
SVM	TF-IDF bigrams	587	4 587	1.000	0.517	129
RF	LDiA BoW unigrams + bigrams	587	71	1.000	0.510	130
RF	TF-IDF bigrams	587	4 587	1.000	0.495	131
SVM	LDiA BoW unigrams + extra RTT data	6 457	16	0.470	0.495	132
MNB	TF-IDF bigrams	587	4 587	0.998	0.494	133
MLR	LDiA BoW unigrams + extra RTT data	6 457	16	0.506	0.484	134
SGD	LDiA BoW unigrams + extra RTT data	6 457	16	0.497	0.480	135
SGD	LDiA BoW unigrams + bigrams	587	71	0.519	0.473	136
MLR	LDiA BoW unigrams + bigrams	587	71	0.553	0.471	137
MNB	LDiA BoW unigrams + bigrams	587	71	0.542	0.467	138
RF	LDiA BoW unigrams + extra RTT data	6 457	16	0.998	0.440	139
MNB	LDiA BoW unigrams + extra RTT data	6 457	16	0.457	0.431	140
SGD	LDiA BoW unigrams	587	11	0.377	0.416	141
SVM	LDiA BoW unigrams + bigrams	587	71	0.444	0.409	142
RF	LDiA BoW unigrams	587	11	1.000	0.400	143
MNB	LDiA BoW unigrams	587	11	0.418	0.398	144

Table A.2: List of 146 question category prediction models ranked by performance (continued)

				F1-measure		
Model	**Feature Type**	**# Samples**	**# Features**	**Train**	**Test**	**Rank**
MLR	LDiA BoW unigrams	587	11	0.354	0.375	145
SVM	LDiA BoW unigrams	587	11	0.365	0.340	146

Table A.3 lists the 30 feature types alphabetically, along with the results of the best performing model and where it ranked among the 146 models listed in Table A.2.

Table A.3: Summary of 30 feature types with best performing model

			Best	F1-measure		
Feature Type	**# Samples**	**# Features**	**Model**	**Train**	**Test**	**Rank**
BoW binary values	587	1 101	SGD	0.947	0.817	13
BoW binary values + extra RTT data	6 457	2 263	SGD	0.970	0.772	51
BoW counts	587	1 101	MLR	0.934	0.841	3
BoW counts + extra RTT data	6 457	2 263	SGD	0.973	0.785	44
Engineered features	587	61	SGD	0.911	0.813	15
Engineered features + extra RTT data	6 457	65	MLR	0.819	0.760	64
GloVe word embeddings	587	300	SGD	0.937	0.763	60
GloVe word embeddings + extra RTT data	6 457	300	SGD	0.857	0.730	86
LDiA BoW unigrams	587	11	SGD	0.377	0.416	141
LDiA BoW unigrams + extra RTT data	6 457	16	SVM	0.470	0.495	132
LDiA BoW unigrams + bigrams	587	71	RF	1.000	0.510	130
LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	SGD	0.554	0.605	113
LSA BoW unigrams	587	91	SVM	0.966	0.812	16
LSA BoW unigrams + extra RTT data	6 457	97	MLR	0.867	0.756	68
LSA BoW unigrams + bigrams	587	85	SVM	0.956	0.799	27
LSA BoW unigrams + bigrams + extra RTT data	6 457	63	SGD	0.794	0.767	55
LSA TF-IDF unigrams (probabilities)	587	55	MLR	0.844	0.844	2
LSA TF-IDF unigrams (raw values)	587	55	MLR	0.886	0.831	5
LSA TF-IDF unigrams + extra RTT data (probabilities)	6 457	49	MLR	0.783	0.825	8
LSA TF-IDF unigrams + extra RTT data (raw values)	6 457	49	SGD	0.753	0.789	38
LSA TF-IDF unigrams + bigrams	587	49	MLR	0.824	0.794	33
LSA TF-IDF unigrams + bigrams + extra RTT data	6 457	85	SGD	0.792	0.763	61
TF-IDF bigrams	587	4 587	MLR	1.000	0.555	120
TF-IDF bigrams + extra RTT data	6 457	19 885	MNB	0.995	0.584	115
TF-IDF unigrams	587	1 101	SGD	0.951	0.835	4
TF-IDF unigrams + extra RTT data	6 457	2 263	SGD	0.935	0.795	32
TF-IDF unigrams + bigrams	587	5 688	MLR	0.978	0.848	1
TF-IDF unigrams + bigrams + extra RTT data	6 457	22 148	SGD	0.972	0.811	20
TF-IDF unigrams + bigrams + trigrams	587	10 972	SGD	0.998	0.821	9
TF-IDF unigrams + bigrams + trigrams + extra RTT data	6 457	53 279	SGD	0.977	0.817	12

Table A.4 lists the test F1-measure statistics for all 30 feature types.

Table A.4: Summary of 30 feature types with descriptive statistic for test F1-measure

	Descriptive Statistic for Test F1-measure								
Feature Type	**Count All**	**Count in Top 20%**	**Min**	**Mean**	**Max**	**P25**	**P50**	**P75**	**Std**
BoW binary	5	3	0.641	0.771	0.817	0.788	0.803	0.804	0.073
BoW binary + extra RTT data	5	0	0.638	0.721	0.772	0.686	0.743	0.765	0.057
BoW counts	5	4	0.647	0.786	0.841	0.806	0.812	0.825	0.079
BoW counts + extra RTT data	5	0	0.684	0.737	0.785	0.699	0.751	0.768	0.044
Engineered features	5	2	0.737	0.783	0.813	0.756	0.795	0.812	0.034
Engineered features + extra RTT data	5	0	0.708	0.743	0.760	0.737	0.753	0.757	0.021
GloVe	4	0	0.613	0.714	0.763	0.696	0.740	0.757	0.069
GloVe + extra RTT data	4	0	0.635	0.688	0.730	0.661	0.693	0.719	0.044
LDiA BoW unigrams	5	0	0.340	0.386	0.416	0.375	0.398	0.400	0.029
LDiA BoW unigrams + extra RTT data	5	0	0.431	0.466	0.495	0.440	0.480	0.484	0.029
LDiA BoW unigrams + bigrams	5	0	0.409	0.466	0.510	0.467	0.471	0.473	0.036
LDiA BoW unigrams + bigrams + extra RTT data	5	0	0.518	0.570	0.605	0.551	0.580	0.598	0.036
LSA BoW unigrams	5	2	0.611	0.742	0.812	0.710	0.775	0.800	0.083
LSA BoW unigrams + extra RTT data	5	0	0.550	0.684	0.756	0.666	0.704	0.742	0.083
LSA BoW unigrams + bigrams	5	1	0.578	0.728	0.799	0.703	0.773	0.788	0.092
LSA BoW unigrams + bigrams + extra RTT data	5	0	0.521	0.684	0.767	0.641	0.746	0.746	0.104
LSA TF-IDF unigrams (probabilities)	5	1	0.730	0.780	0.844	0.770	0.773	0.784	0.041
LSA TF-IDF unigrams (raw values)	4	1	0.746	0.788	0.831	0.777	0.788	0.799	0.035
LSA TF-IDF unigrams + extra RTT data (probabilities)	5	3	0.713	0.786	0.825	0.754	0.819	0.820	0.050
LSA TF-IDF unigrams + extra RTT data (raw values)	4	0	0.759	0.769	0.789	0.761	0.764	0.772	0.014
LSA TF-IDF unigrams + bigrams	4	0	0.713	0.759	0.794	0.751	0.764	0.772	0.034
LSA TF-IDF unigrams + bigrams + extra RTT data	4	0	0.728	0.743	0.763	0.733	0.741	0.751	0.016
TF-IDF bigrams	5	0	0.494	0.519	0.555	0.495	0.517	0.534	0.026
TF-IDF bigrams + extra RTT data	5	0	0.521	0.554	0.584	0.528	0.562	0.573	0.028
TF-IDF unigrams	5	3	0.693	0.784	0.835	0.783	0.799	0.812	0.055
TF-IDF unigrams + extra RTT data	5	0	0.551	0.730	0.795	0.757	0.762	0.786	0.101
TF-IDF unigrams + bigrams	5	3	0.685	0.788	0.848	0.776	0.806	0.825	0.063
TF-IDF unigrams + bigrams + extra RTT data	5	2	0.610	0.753	0.811	0.751	0.792	0.800	0.083
TF-IDF unigrams + bigrams + trigrams	5	3	0.680	0.780	0.821	0.789	0.798	0.814	0.058
TF-IDF unigrams + bigrams + trigrams + extra RTT data	5	2	0.649	0.769	0.817	0.792	0.792	0.796	0.068

Table A.5 lists the 6 feature categories, ranked in order of the best performing model. Only two of the five model types, MLR and SGD, occurred as the best model type to use per feature category.

Table A.5: Summary of 6 feature categories with best performing model

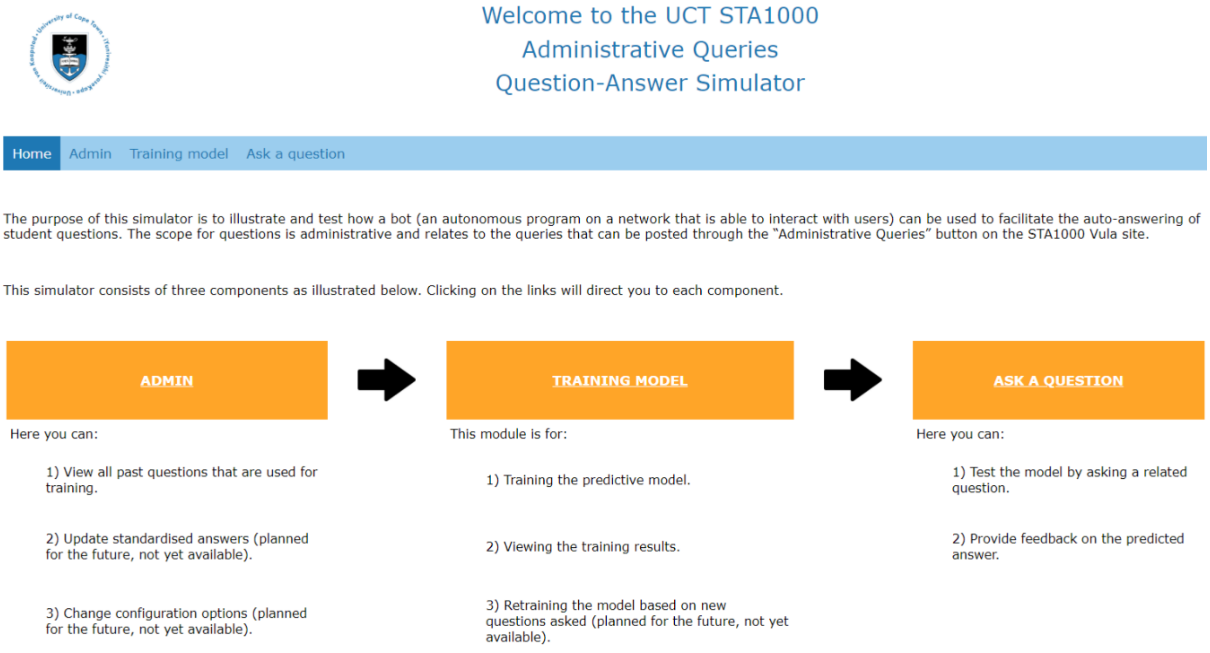
Category	Best Feature Type per Category	# Samples	# Features	Best Model	F1-measure		Rank
					Train	Test	
TF-IDF	TF-IDF unigrams + bigrams	587	5 688	MLR	0.978	0.848	1
LSA	LSA TF-IDF unigrams (probabilities)	587	55	MLR	0.844	0.844	2
BoW	BoW counts	587	1 101	MLR	0.934	0.841	3
Engineered	Engineered features	587	61	SGD	0.911	0.813	15
GloVe	GloVe	587	300	SGD	0.937	0.763	60
LDiA	LDiA BoW unigrams + bigrams + extra RTT data	6 457	56	SGD	0.554	0.605	113

A.7 The QA Simulator

A simulator was built to illustrate and test how a bot (an autonomous program on a network that is able to interact with users) can be used to facilitate the auto-answering of student questions. The PythonAnywhere online-integrated-development environment and web-hosting service was used to create the simulator, and can be accessed using this link: https://sta1000bot.pythonanywhere.com.

This section provides a layout of the different components included in this simulator, the first of which is the home page of the QA simulator as shown in Figure A.4.

Figure A.4: Home page of QA simulator



The purpose of this simulator is to illustrate and test how a bot (an autonomous program on a network that is able to interact with users) can be used to facilitate the auto-answering of student questions. The scope for questions is administrative and relates to the queries that can be posted through the "Administrative Queries" button on the STA1000 Vula site.

This simulator consists of three components as illustrated below. Clicking on the links will direct you to each component.

Here you can:

1) View all past questions that are used for training.

2) Update standardised answers (planned for the future, not yet available).

3) Change configuration options (planned for the future, not yet available).

This module is for:

1) Training the predictive model.

2) Viewing the training results.

3) Retraining the model based on new questions asked (planned for the future, not yet available).

Here you can:

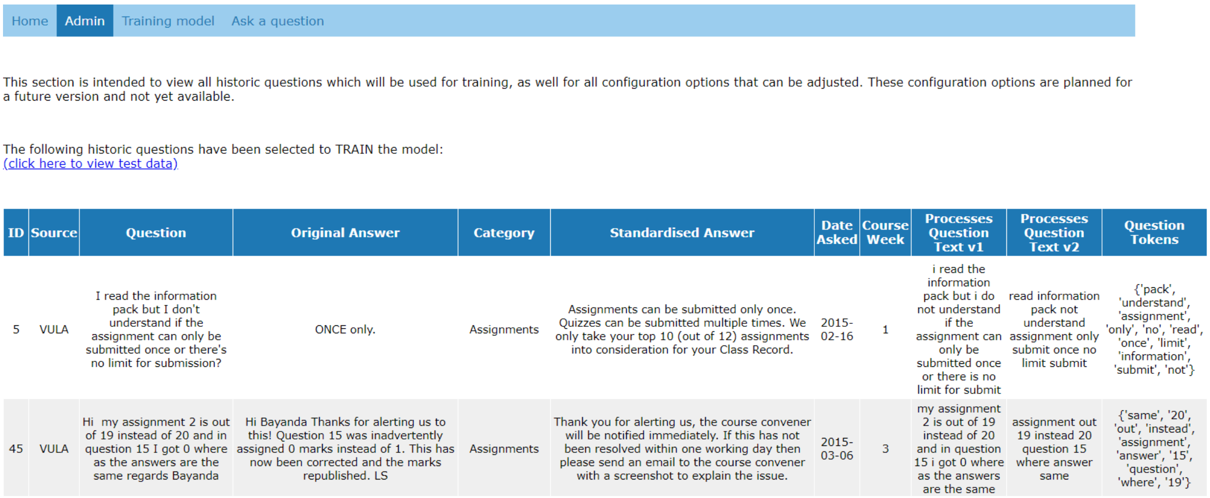
1) Test the model by asking a related question.

2) Provide feedback on the predicted answer.

From the home page a user can either navigate to the "Admin" page, the "Training Model" page or the "Ask a Question" page. One does not need to navigate between the pages in a specific order, but the pages have been laid out in the logical sequence which was used to create the simulator.

The “Admin” page provides a view of the training and testing data used to build the QA system as shown in Figure A.5.

Figure A.5: Admin page of QA simulator



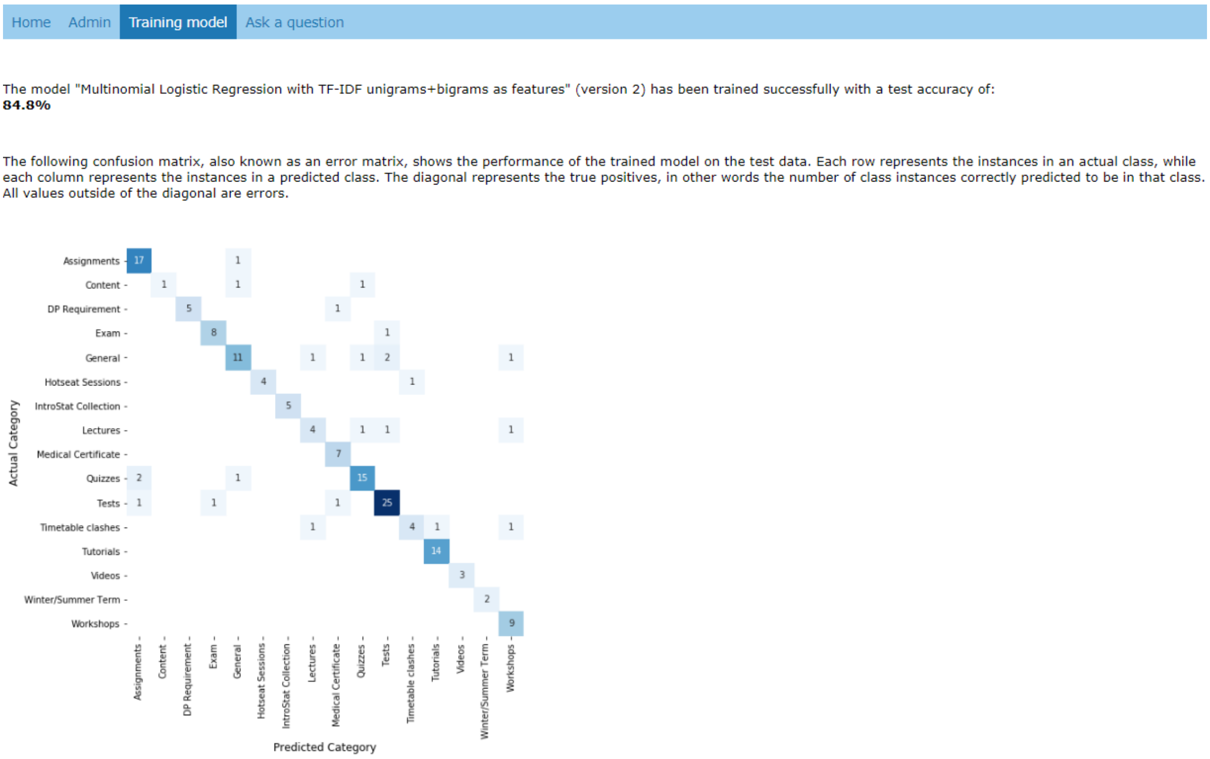
This section is intended to view all historic questions which will be used for training, as well for all configuration options that can be adjusted. These configuration options are planned for a future version and not yet available.

The following historic questions have been selected to TRAIN the model:
(click here to view test data)

ID	Source	Question	Original Answer	Category	Standardised Answer	Date Asked	Course Week	Processes Question Text v1	Processes Question Text v2	Question Tokens
5	VULA	I read the information pack but I don't understand if the assignment can only be submitted once or there's no limit for submission?	ONCE only.	Assignments	Assignments can be submitted only once. Quizzes can be submitted multiple times. We only take your top 10 (out of 12) assignments into consideration for your Class Record.	2015-02-16	1	i read the information pack but i do not understand if the assignment can only be submitted once or there is no limit for submit	read information pack not understand assignment only submit once no limit submit	{'pack', 'understand', 'assignment', 'only', 'no', 'read', 'once', 'limit', 'information', 'submit', 'not'}
45	VULA	Hi my assignment 2 is out of 19 instead of 20 and in question 15 I got 0 where as the answers are the same regards Bayanda	Hi Bayanda Thanks for alerting us to this! Question 15 was inadvertently assigned 0 marks instead of 1. This has now been corrected and the marks republished. LS	Assignments	Thank you for alerting us, the course convener will be notified immediately. If this has not been resolved within one working day then please send an email to the course convener with a screenshot to explain the issue.	2015-03-06	3	my assignment 2 is out of 19 instead of 20 and in question 15 i got 0 where as the answers are the same	assignment out 19 instead 20 question 15 where answer same	{'same', '20', 'out', 'instead', 'assignment', 'answer', '15', 'question', 'where', '19'}

The “Training Model” page, as shown in Figure A.6, provides information on the trained model used for predicting the question category of a new question. This model supports the main task of the QA system.

Figure A.6: Training model page of QA simulator



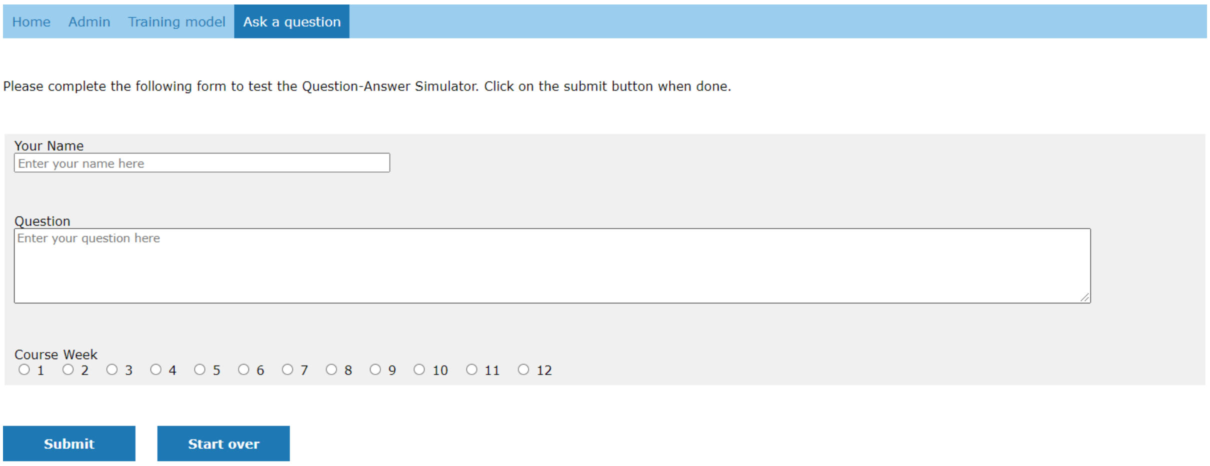
The model "Multinomial Logistic Regression with TF-IDF unigrams+bigrams as features" (version 2) has been trained successfully with a test accuracy of:
84.8%

The following confusion matrix, also known as an error matrix, shows the performance of the trained model on the test data. Each row represents the instances in an actual class, while each column represents the instances in a predicted class. The diagonal represents the true positives, in other words the number of class instances correctly predicted to be in that class. All values outside of the diagonal are errors.

	Assignments	Content	DP Requirement	Exam	General	Hotseat Sessions	IntroStat Collection	Lectures	Medical Certificate	Quizzes	Tests	Timetable clashes	Tutorials	Videos	Winter/Summer Term	Workshops
Assignments	17				1											
Content		1			1					1						
DP Requirement			5						1							
Exam				8							1					
General					11			1		1	2					1
Hotseat Sessions						4						1				
IntroStat Collection							5									
Lectures								4		1	1					1
Medical Certificate									7							
Quizzes	2				1					15						
Tests	1			1					1		25					
Timetable clashes								1				4	1			1
Tutorials													14			
Videos														3		
Winter/Summer Term															2	
Workshops																9

The “Ask a Question” page provides the means of testing the QA system as shown in Figure A.7.

Figure A.7: Ask a question page of QA simulator



Please complete the following form to test the Question-Answer Simulator. Click on the submit button when done.

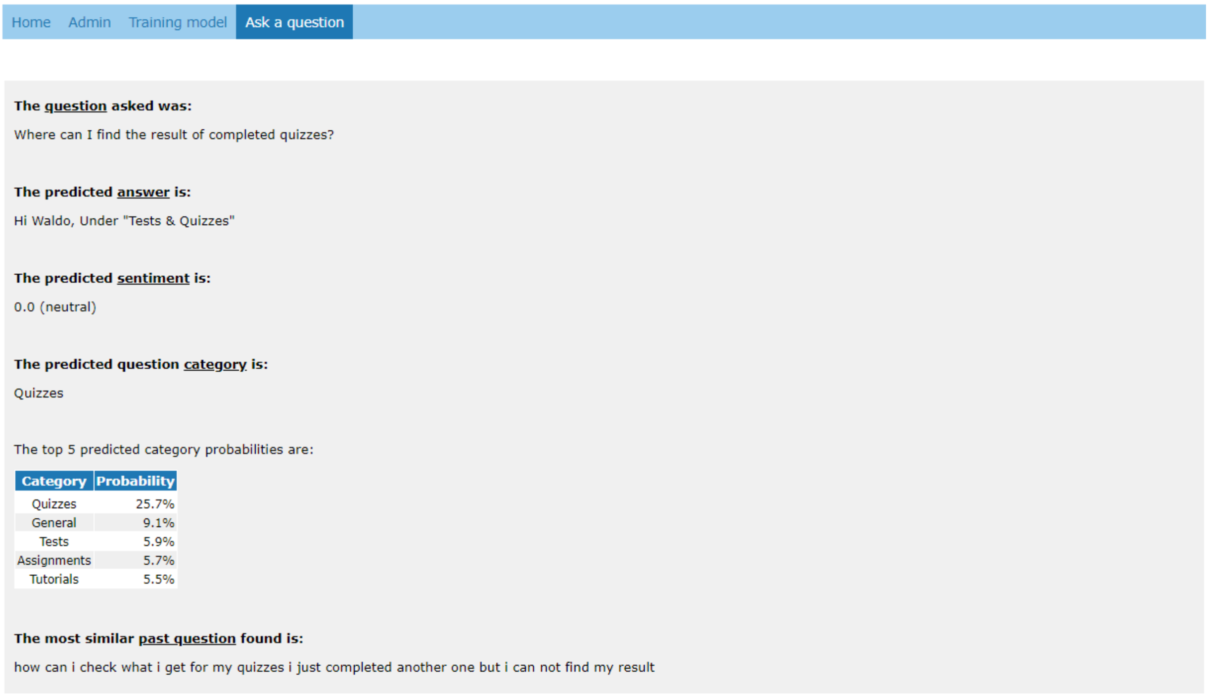
Your Name
Enter your name here

Question
Enter your question here

Course Week
○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 ○ 8 ○ 9 ○ 10 ○ 11 ○ 12

After a question has been submitted the answer page is displayed as shown in Figure A.8.

Figure A.8: Example of an answer returned by the QA simulator



The question asked was:

Where can I find the result of completed quizzes?

The predicted answer is:

Hi Waldo, Under "Tests & Quizzes"

The predicted sentiment is:

0.0 (neutral)

The predicted question category is:

Quizzes

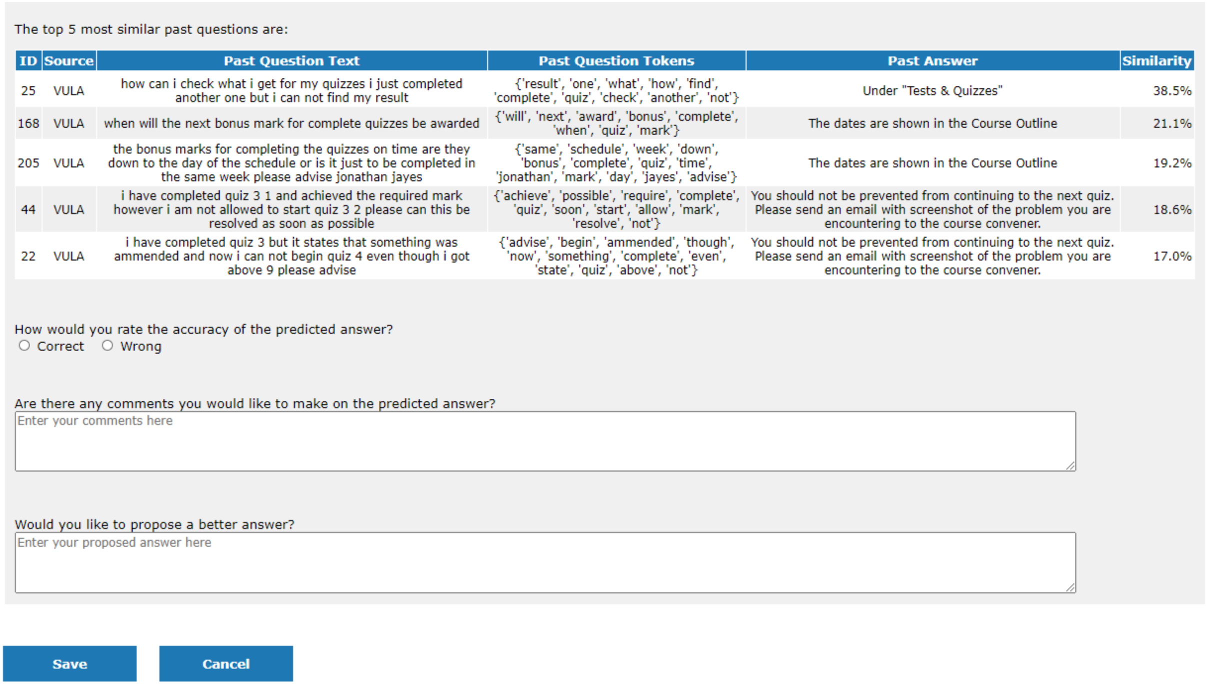
The top 5 predicted category probabilities are:

Category	Probability
Quizzes	25.7%
General	9.1%
Tests	5.9%
Assignments	5.7%
Tutorials	5.5%

The most similar past question found is:

how can i check what i get for my quizzes i just completed another one but i can not find my result

Figure A.8: Example of an answer returned by the QA simulator (continued)



The top 5 most similar past questions are:

ID	Source	Past Question Text	Past Question Tokens	Past Answer	Similarity
25	VULA	how can i check what i get for my quizzes i just completed another one but i can not find my result	{'result', 'one', 'what', 'how', 'find', 'complete', 'quiz', 'check', 'another', 'not'}	Under "Tests & Quizzes"	38.5%
168	VULA	when will the next bonus mark for complete quizzes be awarded	{'will', 'next', 'award', 'bonus', 'complete', 'when', 'quiz', 'mark'}	The dates are shown in the Course Outline	21.1%
205	VULA	the bonus marks for completing the quizzes on time are they down to the day of the schedule or is it just to be completed in the same week please advise jonathan jayes	{'same', 'schedule', 'week', 'down', 'bonus', 'complete', 'quiz', 'time', 'jonathan', 'mark', 'day', 'jayes', 'advise'}	The dates are shown in the Course Outline	19.2%
44	VULA	i have completed quiz 3 1 and achieved the required mark however i am not allowed to start quiz 3 2 please can this be resolved as soon as possible	{'achieve', 'possible', 'require', 'complete', 'quiz', 'soon', 'start', 'allow', 'mark', 'resolve', 'not'}	You should not be prevented from continuing to the next quiz. Please send an email with screenshot of the problem you are encountering to the course convener.	18.6%
22	VULA	i have completed quiz 3 but it states that something was ammended and now i can not begin quiz 4 even though i got above 9 please advise	{'advise', 'begin', 'ammended', 'though', 'now', 'something', 'complete', 'even', 'state', 'quiz', 'above', 'not'}	You should not be prevented from continuing to the next quiz. Please send an email with screenshot of the problem you are encountering to the course convener.	17.0%

How would you rate the accuracy of the predicted answer?
○ Correct ○ Wrong

Are there any comments you would like to make on the predicted answer?

Would you like to propose a better answer?

A user has the option to provide feedback on the answers extracted and this can then be used to improve the system over time, as all responses are saved in a MySQL database hosted on the PythonAnywhere server.

Bibliography

Allam, A.M.N. & Haggag, M.H. 2012. The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS).* 2(3).

Bayes, T. & Price, R. 1763. An essay toward solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFRS. *Philosophical Transactions.* 1683-1775.

Benedetto, L., Cremonesi, P. & Parenti, M. 2019. A virtual teaching assistant for personalized learning. *arXiv preprint arXiv:1902.09289.*

Berwick, R.C. & Chomsky, N. 2016. *Why only us: Language and evolution.* MIT press.

Bird, S., Klein, E. & Loper, E. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* O'Reilly Media, Inc.

Bottou, L. 1998. Online Algorithms and Stochastic Approximations. *Cambridge University Press.* Available: http://leon.bottou.org/papers/bottou-98x.

Breiman, L. 2001. Random forests. *Machine learning.* 45(1): 5-32.

Chopra, S., Gianforte, R. & Sholar, J. 2016. Meet Percy: CS 221 Teaching Assistant Chatbot. *ACM Transactions on Graphics.* 1(1).

Duvenhage, B. 2019. Short text language identification for under resourced languages. *arXiv preprint arXiv:1911.07555.*

Eicher, B., Polepeddi, L. & Goel, A.K. Eds. 2018. Jill Watson Doesn't Care if You're Pregnant: Grounding AI Ethics in Empirical Studies. *AIES*. 88-94.

Eisenstein, J. 2019. *Introduction to Natural Language Processing.* MIT Press.

Fawcett, T. 2006. An introduction to ROC analysis. *Pattern recognition letters.* 27(8): 861-874.

Fellbaum, C. 2010. WordNet. In *Theory and applications of ontology: computer applications.* Springer. 231-243.

Ferrucci, D.A. 2012. Introduction to “This is Watson”. *IBM Journal of Research and Development.* 56(3.4): 1: 1-1: 15.

Goel, A.K. & Joyner, D.A. 2015. *An experiment in teaching cognitive systems online*. Georgia Institute of Technology. Available: http://hdl.handle.net/1853/53704.

Goel, A.K. & Polepeddi, L. 2016. *Jill Watson: A virtual teaching assistant for online education*. Georgia Institute of Technology. Available: http://hdl.handle.net/1853/59104.

Goel, A.K. & Joyner, D.A. 2017. Using AI to teach AI: lessons from an online AI class. *AI Magazine.* 38(2): 48-59.

Goel, A.K., Creeden, B., Kumble, M., Salunke, S., Shetty, A. & Wiltgen, B. Eds. 2015. Using watson for enhancing human-computer co-creativity. *2015 AAAI Fall Symposium Series*.

Hao, X., Chang, X. & Liu, K. Eds. 2007. A Rule-based Chinese question Answering System for reading Comprehension Tests. *Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007)*. IEEE. 325-329.

Hastie, T., Tibshirani, R. & Friedman, J. 2009. *The elements of statistical learning: data mining, inference, and prediction.* Springer Science & Business Media.

Hutto, C.J. & Gilbert, E. Eds. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth international AAAI conference on weblogs and social media*.

James, G., Witten, D., Hastie, T. & Tibshirani, R. 2017. *An Introduction to Statistical Learning: with Applications in R, 7th Edition.* New York: Springer.

Jeon, J., Croft, W.B. & Lee, J.H. Eds. 2005a. Finding similar questions in large question and answer archives. *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM. 84-90.

Jeon, J., Croft, W.B. & Lee, J.H. Eds. 2005b. Finding semantically similar questions based on their answers. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. 617-618.

Joyner, D. Ed. 2018. Squeezing the limeade: policies and workflows for scalable online degrees. *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. ACM. 53.

Jurafsky, D. & Martin, J.H. 2019. *Speech and Language Processing (3rd ed. draft).* Prentice Hall PTR, Upper Saddle River, NJ, USA. https://web.stanford.edu/~jurafsky/slp3/.

Lane, H., Howard, C. & Hapke, H.M. 2019. *Natural language processing in action: Understanding, analyzing, and generating text with python.* Manning Publications Company.

Lavrenko, V. Ed. 2010. Introduction to Probabilistic Models for Information Retrieval. *proceedings of the 33rd International ACM SIGIR conference on Research and Development in Information Retrieval edited by Hsin-His Chen, Efthismis N. Efthimiadis, Jacques Savoy, Fabio Crestani Lugano and Stephane Marehand-Maillet, Association for Computing Machinery, New York*. Citeseer.

Loria, S., Keen, P., Honnibal, M., Yankovsky, R., Karesh, D. & Dempsey, E. 2014. Textblob: simplified text processing. *Secondary TextBlob: Simplified Text Processing.* 3.

Marivate, V. & Sefara, T. 2019. Improving short text classification through global augmentation methods. *arXiv preprint arXiv:1907.03752.*

Mikolov, T., Yih, W.-t. & Zweig, G. Eds. 2013c. Linguistic regularities in continuous space word representations. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 746-751.

Mikolov, T., Chen, K., Corrado, G. & Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. & Dean, J. Eds. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*. 3111-3119.

Mollá, D. & Vicedo, J.L. 2007. Question answering in restricted domains: An overview. *Computational Linguistics.* 33(1): 41-61.

Molnár, G. & Szüts, Z. Eds. 2018. The Role of Chatbots in Formal Education. *2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY)*. IEEE. 000197-000202.

Montgomery, C.A. Ed. 1972. Is natural language an unnatural query language? *Proceedings of the ACM annual conference-Volume 2*. 1075-1078.

Mosteller, F. & Wallace, D.L. 1964. *Inference and Disputed Authorship: The Federalist.* First edition. Reading, MA: Addison-Wesley.

Nguyen, D. Ed. 2018. Comparing automatic and human evaluation of local explanations for text classification. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 1069-1078.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P. et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research.* 12: 2825-2830.

Pennington, J., Socher, R. & Manning, C. Eds. 2014. Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532-1543.

Phillips, A.V. 1960. *A question-answering routine*. Massachusetts Institute of Technology.

Pudaruth, S., Boodhoo, K. & Goolbudun, L. 2016. An Intelligent Question Answering System for ICT. *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT).* 2895-2899.

Reddy, A.C.O. & Madhavi, K. 2017. A Survey on Types of Question Answering System. *IOSR-JCE.* 19(6): 19-23.

Riloff, E. & Thelen, M. Eds. 2000. A rule-based question answering system for reading comprehension tests. *Proceedings of the 2000 ANLP/NAACL Workshop on Reading comprehension tests as evaluation for computer-based language understanding sytems-Volume 6*. Association for Computational Linguistics. 13-19.

Röder, M., Both, A. & Hinneburg, A. Eds. 2015. Exploring the space of topic coherence measures. *Proceedings of the eighth ACM international conference on Web search and data mining*. 399-408.

Shtok, A., Dror, G., Maarek, Y. & Szpektor, I. Eds. 2012. Learning from the past: answering new questions with past answers. *Proceedings of the 21st international conference on World Wide Web*. ACM. 759-768.

Simmons, R.F. 1964. *Answering English questions by computer: a survey.* Santa Monica, California: System Development Corporation.

Soares, M.A.C. & Parreiras, F.S. 2018. A literature review on question answering techniques, paradigms and systems. *Journal of King Saud University-Computer and Information Sciences.*

Stitson, M., Weston, J., Gammerman, A., Vovk, V. & Vapnik, V. 1996. Theory of support vector machines. *University of London.* 117(827): 188-191.

Underhill, L. & Bradfield, D. 2014. *INTROSTAT (Statistics Textbook).* University of Cape Town: Department of Statistical Sciences.

Yao, X. 2014. *Feature-driven question answering with natural language alignment.* Ph.D. thesis. Johns Hopkins University.

Zhang, H. 2004. *The optimality of naive Bayes*. University of New Brunswick.