Predicting mergers and acquisitions using machine learning

by

Gordon Beckenstrater

Supervised by

Prof Patrick Marais

A Thesis submitted for the degree of
MSc by dissertation

in the

Department of Computer Science
University of Cape Town

August 2023

Declaration of Authorship

I, Gordon Beckenstrater, declare that this work presented is my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Signed by candidate

Date: 21/08/2023

Abstract

Mergers and acquisitions (M&As) play a crucial role in the expansion of companies. During a typical M&A deal, the target company is offered a significant premium over their current share price by the acquirer. This results in a material increase in the target company's share price on the announcement of acquisition. Therefore, accurately forecasting M&As, despite the challenge due to their rarity, presents a lucrative opportunity for investors. Traditional statistical forecasting techniques, reliant on fundamental and technical metrics along with a few macroeconomic indicators, often struggle to pick up underlying relationships between features and targets.

This study investigates the effectiveness of advanced machine learning techniques, which have found large success in stock price and fraud prediction, in predicting M&As. logistic regression, a popular statistical technique in M&A literature, serves as a baseline. The performance of algorithms such as random forest, LightGBM, long short-term memory networks (LSTM) and the TabTransformer are evaluated against the baseline. A secondary objective is the development of a robust ensemble model for potential use in an investment portfolio.

The algorithms were trained on a comprehensive historical dataset with diverse financial indicators. Given the considerable amount of missing values in the dataset, imputation was applied to allow all algorithms to function properly. Feature selection was conducted to remove redundant features, mitigating their impact on validation performance of the models. Data imbalance was addressed with data sampling techniques which proved substantial in improving validation performance.

The findings are that all the advanced algorithms surpassed the performance of logistic regression in M&A prediction, signalling a shift from traditional statistical methods to advanced machine learning techniques. LightGBM and the Ensemble model displayed the best performance in M&A prediction. These results also show that an investment portfolio, constructed based on the most confident predictions of the Ensemble model, forms the basis for a profitable investment strategy.

Acknowledgements

I would like to thank my family for their support over the period of time it took to complete this dissertation as well as Professor Patrick Marais for his guidance throughout the research process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mergers and acquisitions (M&A) provide an opportunity for companies to strengthen their market position and enhance collaboration. As there are many strategic factors that influence the decision to engage in a merger or acquisition, accurately forecasting these events presents a significant research challenge. The reason that this is such a difficult problem is mostly attributed to the highly imbalanced nature of the data, where M&A events make up only 0.5% of total observations [12]. Highly imbalanced data tends to skew model predictions to the majority class, thus compromising the capability to predict the minority class (M&A events). This thesis aims to leverage the power of advanced machine learning techniques to predict M&A events.

Historically, traditional approaches have primarily relied on analysis of corporate fundamentals, financial ratios and macroeconomic indicators in conjunction with basic statistical models such as logistic regression and discriminant analysis. These models are simplistic in nature and often focus on a limited set of features, due to a lack of computational power and large datasets. In recent years, a small number of studies have applied machine learning algorithms to the problem. Many of these studies still fall short of using state-of-the-art algorithms with large high-dimensional datasets, representing an opportunity for advancement in the field.

Modern machine learning algorithms are particularly suited to dealing with large high-dimensional datasets and learning complex underlying patterns in the data. These algorithms when integrated with data sampling techniques, can effectively deal with data imbalance. This opens the potential of improving accuracy and robustness in M&A predictions.

The array of algorithms utilised in this study are diverse, integrating an established algorithm in the domain of M&A, logistic regression, and sparsely explored algorithms in

M&A literature, such as random forest, LightGBM, long short-term memory networks (LSTMs) and TabTransformer. Logistic regression establishes a traditional baseline to contrast and analyse increasingly advanced machine learning algorithms. Random forest and LightGBM are robust against overfitting and are built to handle high-dimensional datasets. LSTMs capture temporal dependencies which are potentially relevant in the dynamics of M&A. Finally, the TabTransformer is an innovative model based on transformer architecture, known as one of the most sophisticated methodologies in machine learning. It is important to note that in this thesis, the focus is not on the development of novel machine learning algorithms, rather, it emphasises the application of established and robust machine learning techniques to improve predictive modelling of mergers and acquisitions.

This study embraces a 'big data' approach, using a high dimensional dataset with a significant number of samples. This differs from most traditional literature which is limited by smaller datasets both in terms of number of samples and number of features. The premise behind this approach is that having more data gives the algorithms a greater potential for accurately predicting M&As. This selection of algorithms and the use of an expansive dataset aims to fill the gap in the existing literature, which leans to more simplistic models and limited datasets for prediction.

1.1 Aims

The primary aim of this work is to conduct a detailed evaluation of optimised machine learning algorithms. Logistic regression serves as the benchmark due to its presence as the most frequently investigated model in the M&A space. The final evaluation on a holdout set requires rigorous validation on each individual algorithm to get the best possible model for each algorithm. This investigation will ultimately show the ability of these algorithms to anticipate M&A events in large imbalanced datasets.

A secondary objective is a short investigation into the feasibility of constructing a successful investment portfolio based on the predicted M&As. An aggregated ensemble of the single best validation model produced for each algorithm will be used to construct this ensemble model. This provides an estimate of potential returns for such a strategy and demonstrates a practical application of this work.

1.2 Contributions

The study contributes to investigating the latest, more sophisticated machine learning algorithms in forecasting M&As. The rigorous optimisation and evaluation of these algorithms on M&As advances our knowledge on their performance in severely imbalanced settings, especially in the context of financial tabular data.

The effectiveness of machine learning models is improved using a comprehensive dataset. This expansive dataset provides enhanced prediction over limited datasets by enabling the models to learn from a greater number of features and samples. This research helps contribute to understanding the role large complex data plays in M&A prediction.

The thesis assesses portfolio performance in a real-world scenario. The creation and subsequent evaluation of an ensemble of the best models from each algorithm provides another contribution. This contribution gives an estimate of potential returns and reinforces ensemble modelling as a robust technique when dealing with imbalanced financial datasets.

1.3 Thesis Structure

Chapter 2 lays the groundwork by providing foundational concepts on mergers and acquisitions, the stock market and machine learning. Chapter 3 presents a review of relevant literature concerning mergers and acquisitions, with particular emphasis on work that utilised machine learning for predictive purposes. Chapter 4 introduces the dataset and presents a high-level view of the experimental framework. In Chapter 5 the imputation and feature selection experiments are described. Chapter 6 focuses on hyperparameter tuning and details the process and parameter space that each algorithm is searched through. Chapter 7 discusses the data sampling experiments that investigate both undersampling and oversampling techniques that are necessary to address the data imbalance inherent in M&A datasets. Chapter 8 offers an in-depth analysis of the results and presents a possible investment strategy based on the Ensemble M&A model's predictions. The thesis concludes in Chapter 9 by providing a high-level conclusion, acknowledging the limitations and potential avenues for future work.

Chapter 2

Background

This chapter contains a summary of the background literature relevant to predicting mergers and acquisitions using machine learning techniques. First, an overview of the stock market, investment strategies and the efficient market hypothesis is presented. Then the mergers and acquisitions field is discussed, along with an introduction to important terminology and definitions. Classic techniques that have been used to try and predict M&As in the past, and general causes of an M&A, are then discussed. Subsequently, a brief description of machine learning is provided, along with various machine learning algorithms that can be applied to this task. Lastly, an overview of imputation techniques is given, in addition to a presentation of data sampling methods commonly used for addressing imbalanced datasets.

2.1 The Stock Market

The stock market is a market where buyers and sellers of shares (ownership claims) in companies can transact with each other and thus exchange ownership of shares of companies [13].

Investment Strategies

Investment strategies in finance are philosophies and processes (rules) that investors follow to select assets for a particular portfolio. Strategies can be broken down into one of two subgroups: active or passive [14]. Passive strategies simply mimic an index, whereas active strategies aim to outperform the index [15].

Other well-known strategies are value investing [16] (buying cheap companies), growth investing [17] (buying companies that are going to grow their earnings) and momentum investing [18] (buying shares that have been outperforming).

In the context of this thesis the investment strategy would be one of event prediction. This means anticipating the merger or acquisition and purchasing the target company's share prior to the M&A being announced. This is a subset of event-driven investing [19] as a portfolio strategy.

Market Efficiency

The efficient market hypothesis (EMH) states that asset prices reflect all available information. This implies that it is impossible to outperform the market on a risk adjusted basis. The hypothesis was expanded into three categories by Eugene Fama: weak-form, semi-strong form and strong-form [20].

The weak-form hypothesis states that the historical price information is discounted in the present price, meaning a stock's present price accurately reflects all historical price data on the stock. This rules out the ability to predict returns based on historical prices, rendering all forms of technical analysis and price momentum strategies ineffective. Semi-strong form efficiency implies that the market discounts all historical price-based information, as well as any publicly available (beyond historical) information. Strong-form efficiency implies that the market discounts not only historical prices and information in the public domain but also non-public private information.

The research study is a test of weak and semi-strong efficiency in that it tests to see if the machine learning algorithms can improve the odds of prediction above the base rate given via publicly available information.

2.2 Mergers and Acquisitions

Mergers and acquisitions (M&A) are the general way of describing the buying and selling of companies. More specifically it is the process of combining two companies into a single entity, where the goal of this combination is to create a more successful company. Strictly speaking the words 'merger' and 'acquisition' are different variations of buying and selling companies [21].

Mergers

A merger is an event where two or more companies of approximately the same size combine forming a new firm under a single corporate name [22]. Each company participating in the merger has an equal stake in the new firm.

Acquisitions

Also known as a takeover, an acquisition is when one company purchases another company outright. This requires the acquirer to successfully purchase a majority stake in the target company. In the M&A world, acquisitions are a far more frequent occurrence compared to mergers. Acquisitions are usually pursued as a growth strategy which allows the acquirer company to circumvent organically growing the business, as they are essentially buying established sales and profits [22]. Larger firms will often acquire smaller competitive firms in order to eliminate future competition.

Investment Opportunities

The ability to predict a merger or acquisition prior to it being announced will almost always allow an investor to earn an excess return on investment. M&A deals involving publicly owned companies are often covered by news outlets and draw investors' attention. This is due to the potentially significant upside created by the acquirer company paying a premium over the target company's share price. Thus, being able to correctly predict target firms is a lucrative prospect.

The reason this is such a complex challenge for investors is because the natural rate of M&A events is very low. An average of 0.5% of firms are actual targets at any point in time [12].

FIGURE 2.1: A visualisation of Nuance Corporation's (target company) share through announcement date [1].

Figure 2.1 above shows the typical reaction of a share's price immediately after the announcement of a M&A event. Nuance's (target company) share price jumps approximately 25% on the day Microsoft publicly announces its intention to acquire the company.

2.2.1 Machine Learning and M&As

Predicting potential mergers and acquisitions has been a topic of interest since the 1970's where multiple discriminant analysis was used by Simkowitz and Monroe to identify target companies before their announcement date [23]. Several studies prior to the introduction of machine learning indicated that target companies could be accurately predicted using publicly available financial characteristics. These have since been refuted by exposing the methodological flaws and leakage that was apparent in these studies [24]. Machine learning as a predictive tool has been relatively unexplored compared to classical techniques in the M&A context. In fact, only a few recent researchers have used modern machine learning techniques in their endeavour [12, 25–27]. The studies that do exist have mostly applied outdated machine learning algorithms. There are now newer, superior algorithms which can possibly perform better in this context. The research study plans to apply state-of-the-art machine learning algorithms as well as previous algorithms that have performed well on unbalanced datasets to find and exploit market inefficiency in the M&A database.

Mergers and Acquisitions by Sector

The frequency of mergers varies across different sectors of the business world. Plotting these frequencies is often the first step for investors trying to identify sectors that are more likely to experience consolidation. Figure 2.2 and Figure 2.3 were created using the Thomson Reuters global stock index from January 2000 to December 2018.

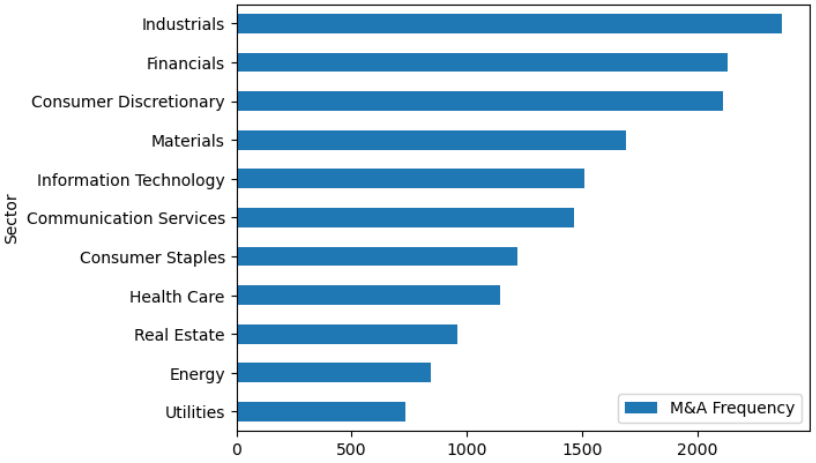


FIGURE 2.2: M&A Frequency

Plotting the frequency of merger and acquisitions as an adjusted percentage of their respective sector enables a more precise comparison between sectors, as larger sectors will often have a larger amount of M&As. This is most noticeable with the 'Industrials' sector in Figure 2.2, where it has the highest total M&A frequency and when adjusted for

the size of the industrial sector (Figure 2.3) it has the second lowest adjusted percentage frequency. The frequencies of M&As, both absolute and adjusted, are useful in designing models that predict M&As events by sector. Additionally, investors can manage risk by regulating exposure to more volatile sectors, thereby constructing a balanced and risk-adjusted portfolio.

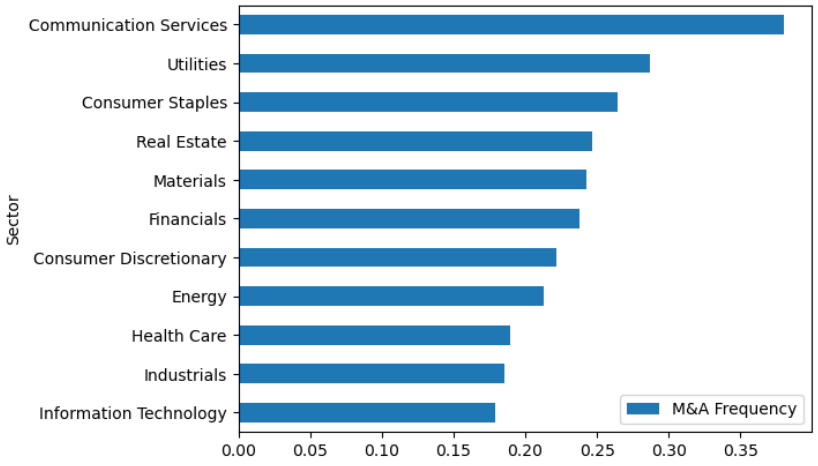


FIGURE 2.3: M&A Frequency Adjusted

Imbalanced Data

Imbalanced data refers to the type of dataset where the target class has an uneven distribution of observations, meaning one class label has a very high number of observations and the other has a very low number of observations.

Sequential Data

In a sequential dataset, samples are dependent on the other samples in the dataset. A time series dataset is a common example of this, with each point reflecting an observation at a certain point in time, such as a stock price or weather forecasting.

Learning of sequential data continues to be a fundamental task and a challenge in pattern recognition and machine learning. Applications involving sequential data may require prediction of new events, generation of new sequences, or decision making such as classification of sequences or sub-sequences.

This kind of data works well in conjunction with algorithms that can recall prior inputs such as LSTMs and gated recurrent units (GRUs). These algorithms have a 'memory' aspect that many other algorithms do not possess.

Structured Data

The database used in this study is of a structured/tabular format and this significantly impacts the choice of algorithms selected to undertake this task. Gradient-boosted trees

have emerged as the most qualified algorithms for these tasks, with the most recent deep learning techniques struggling with the nature of structured data [28].

2.3 Machine Learning Overview

Machine learning is a subset of artificial intelligence (AI), thus it is helpful to define AI moving forward. Artificial intelligence is the "science and engineering of making intelligent machines"[29]. Machine learning is an application of AI where systems, especially computer programs can learn and improve from experience without being explicitly programmed. Machine learning is based around algorithms that are designed to 'learn'. These algorithms are fed data and try to identify patterns from this data to make accurate predictions.

The two main approaches in machine learning are supervised and unsupervised learning. Supervised learning is the most widely used approach where the labels are used in the training process. Unsupervised learning algorithms are trained on samples with no labels and try to identify relationships and correlations between unlabelled feature vectors.

In both scenarios the data needs to be broken up into a training set and a test set. The training set is used to fit the model and tune hyperparameters: the model 'learns' relationships within the data. The test set consists of never seen before data where the trained model is 'tested'. Model fitting is the execution of an algorithm with a set of parameters. Hyperparameter tuning is the search for optimal parameters that minimise the loss function of the model.

A key definition in machine learning are feature vectors (inputs) which are elements of a d-dimensional space. A feature vector x in three dimensions is typically represented as $x = (x_1, x_2, x_3)$ where x_1, x_2 and x_3 are all individual features describing the feature vector x. Each feature is a measurable property that is assigned to a column in tabular datasets. All feature vectors (samples) in the training set must be of the same dimension, but there is no limit to this dimension. More specifically, if there were 100 features then each feature vector would have 100 dimensions.

The major difference in the two approaches is the use of target outputs, these are the outputs the model is trying to predict. In supervised learning feature vectors have corresponding target outputs to help identify what features vectors return certain outputs. In unsupervised there are only unlabelled samples which are plotted against each other to find underlying relationships between them [30]. The research study undertaken is a supervised learning problem, hence this approach warrants a more detailed discussion in the thesis.

2.3.1 Supervised Learning

Supervised learning is the most common approach and functions by learning from a set of labelled training samples, which come in pairs (a, b) where 'a' is the feature vector of d-dimensions and 'b' is the output value. The algorithm builds a model on the relationship between 'a' and 'b', this is the training phase. Then in the testing phase the trained model deals with unlabelled test samples 'a' and predicts the target value 'b' for each test sample. An example is the handwritten number recognition problem (multi-class classification) [31], the sample is an image of the number and the label space consists of the numbers 0-9. Therefore it will train from pairs in the form (image of number 5, 5) and then predict on unlabelled samples in the form (image of 5, prediction of number displayed).

Sample	Label
Image of 3	3
Image of 7	7
Image of 2	2

TABLE 2.1: Labeled training set

Sample	Label
Image of 3	?
Image of 1	?
Image of 5	?

TABLE 2.2: Unlabeled test set

The problem to be solved in supervised learning can either be classification (Tables 2.1, 2.2) or regression. The former being when the set of possible target outputs (discrete labels) is finite. Whereas the latter is when the possible outputs are infinite, namely, the set of real numbers. Thus, the only difference is the target variable space.

Regression

Regression's goal is to predict a continuous value (real number), a classic example of this is trying to predict a person's income based on attributes including years of study, field of study, job title and job location. The more training data relating the features to the target labels the better, this is because the more observations of a relationship often results in better predictive accuracy up to a certain point.

Simple linear regression is a basic type of regression technique where we predict the label y from the feature vector x according to the linear relationship between them. To predict a person's average sales each month based on the year we use the formula $Y = a + bX$ where Y is their estimated sales amount and X is the year. In this formula

a and b are the model coefficients that are learned from the data. The coeffecient a represents the y-intercept of the regression line, and b is the gradient of the regression line. The learning process aims to find values of a and b that minimise the cost function, which is a measure of the difference between the model's predicted sales and the actual sales [32].

Once the model is trained on data, the regression line that best fits the data is generated, thereby minimising error rate. This is far from a perfect solution. However, it provides a good idea of the trend of sales over the past couple years, and can help estimate sales in years to come.

Classification

Classification's aim is to predict a label which is selected from a predefined set of answers. There is two different types of classification, binary and multi-class classification [33]. In the case of binary classification there are only two possible labels. Multi-class classification applies to any problem with more than two possible labels.

A classic example of binary classification is the Titanic dataset [34] where the categories are survived or did not survive (binary). In this case, the attributes used to classify a person include passenger class, sex and age. Each of these attributes contributes greatly to the accuracy of the algorithm, for instance women and children and first class passengers were allowed priority on the lifeboats.

2.3.2 Unsupervised Learning

Unsupervised learning is the other end of the spectrum where the algorithm is trained on feature vectors with no labels. It searches for underlying structures between the samples and finds certain relationships and correlations between data points [33].

The problems of unsupervised learning fall into two different categories clustering and association. Clustering involves grouping data into 'clusters', meaning searching for inherent groupings in the data. Association is when a new representation of the data is created to allow other machine learning algorithms to have an easier time using the data or for humans to better understand the data.

Clustering

The objective of clustering is to find natural groupings in the data. The number of clusters that the algorithm searches for is a parameter that can be changed. There are many different types of clustering methods. A commonly used clustering scheme is the k-means clustering algorithm which involves developing k mutually exclusive clusters.

Here the number we choose for k will be an important factor. It starts by arbitrarily initialising k centroids (cluster centres). Then the algorithm performs iterative steps beginning with grouping each sample with the closest centroid. The centroids are then recomputed as the mean of grouped clusters. This process continues until the centroids have stabilised or the maximum number of iterations (defined prior) is reached [33].

2.4 Core Machine Learning Algorithms

Logistic Regression

This is the baseline classification model for predicting mergers and acquisitions as its results are well documented on M&A databases. Logistic regression [35] can output probabilities and classify unseen data using both continuous and discrete features. It is a classic baseline model in binary event classification problems.

A logistic regression classifier is a statistical model which makes use of the logit function to squeeze outputs of a linear equation within a range of 0 and 1 [36]. Thus, the outputs are represented as probabilities. A decision boundary is then created by setting the threshold value for these predicted probabilities, typically 0.5 for binary classification.

2.4.1 Tree-based Models

Decision Trees

The decision tree algorithm [37] (Figure 2.4), which forms the basis of tree-based learning, divides the dataset into progressively smaller subsets using a cost function. A decision tree consists of nodes and branches. The root node is the beginning of the decision tree and holds the whole dataset. Leaf nodes are the endpoints of decision trees and represent the class labels. Internal nodes are the nodes positioned between the root and leaf nodes. The branches that link these nodes are conjunctions of features which lead to class labels. At the root and each internal node, the algorithm uses feature values to split the data into subsets. A cost function like entropy or the Gini index [37] are used to find the best split possible at each node.

The trained decision tree takes a feature vector as input at the root node, this vector is passed down the tree recursively to the left or right to the next internal node. The algorithm repeats this process until it reaches a leaf node where the feature vector is given a predicted classification.

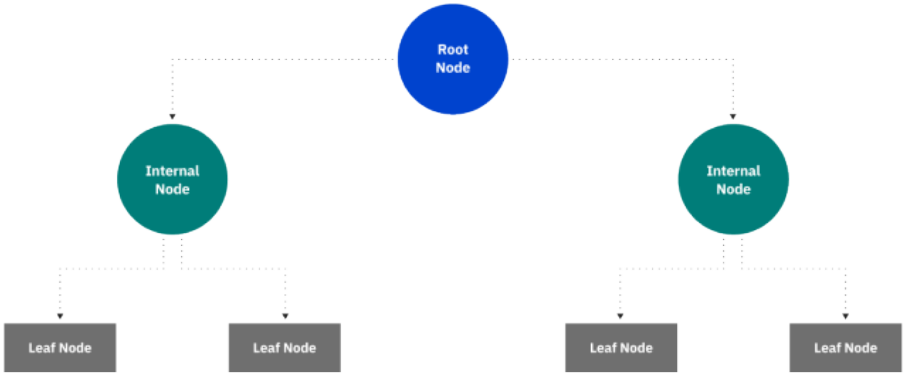


FIGURE 2.4: An illustration of a decision tree [2]

Random Forest

Random forest (Figure 2.5) is a supervised machine learning algorithm that builds a model using an ensemble of multiple decision trees, thus creating a 'forest'. The randomness of the model is introduced through the way the data is split. A standard decision tree would search all possible features for the most important feature when splitting a node. A random forest algorithm conducts this process differently. For each decision tree in a random forest the algorithm searches for the feature that best splits the data from a random subset of available features [38]. The decision trees are then aggregated to find the average result [38]. This randomness gives the model a wide diversification of results, which often helps generalisation of unseen data.

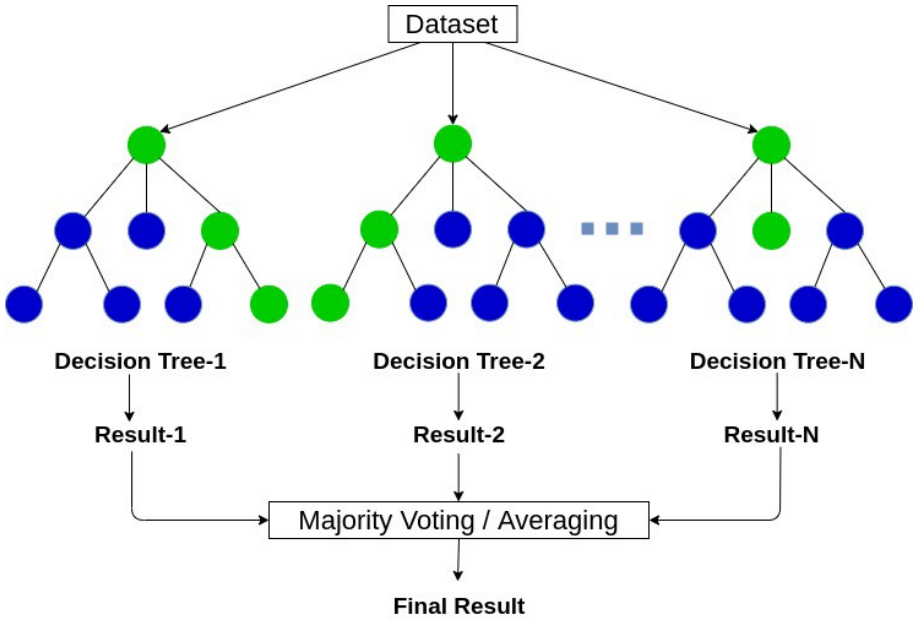


FIGURE 2.5: An illustration of a random forest [3]

Gradient Boosted Trees

Gradient boosting [39] can be broken down into three parts, choosing a loss function to be optimised, weak learners and an additive model which adds models sequentially. The choice of loss function will be dependent on the task, for example an imbalanced dataset would respond better to a metric like 'average precision', however a balanced dataset is better suited to 'log-loss'. A weak learner in this instance is a simple decision tree model that can predict slightly better than random. Gradient boosting uses weak learners to train on the residual errors of the strong learner. Gradient boosting focuses on using weak learners to improve areas where the existing learners have high residuals. These weak learners are added sequentially to the ensemble to minimise its overall error.

LightGBM

Light Gradient Boosting Machine (LightGBM) is a gradient boosting framework developed by Microsoft [40]. LightGBM is a computationally inexpensive, high-performance machine learning algorithm based on the decision tree algorithm. The 'Light' in the name of the algorithm refers to the fast computation speed of the algorithm and its ability to handle large-scale data. It uses histogram-based splitting, Gradient-Based One-Sided Sampling (GOSS) and Exclusive Feature Bundling (EFB) to create a highly efficient algorithm that produces exceptional results [40]. Histogram-based splitting is a technique which bins feature values by constructing a histogram to represent the distribution of the data. The binned feature values are then used to split the data on, which is much quicker than splitting on every possible value within a feature. Along with making the algorithm efficient, the binning process can often reduce overfitting by lessening the amount of noise in the feature values and managing outliers within the dataset.

GOSS is an innovative technique which reduces the number of samples used to build each decision tree whilst attempting to keep the distribution of the data the same. The samples with low residuals are randomly subsampled to reduce their number as they have a negligible impact on training, whereas samples with larger residuals are retained. Due to its focus on the most informative samples the algorithm can achieve similar or greater performance using fewer overall samples in training.

Exclusive Feature Bundling (EFB) is a feature reduction method that bundles up related sparse features into a single feature. This has a significant impact on large-scale sparse datasets, as the number of splits required by the algorithm is proportional to the dimensionality of the dataset[41]. Thus, EFB lowers the memory costs of training a LightGBM model.

2.4.2 Neural Networks

Artificial neural networks are powerful algorithms that are set up to operate similarly to the human brain [42]. Neural networks have many processing nodes that are interconnected. They are made up of layers of nodes which the data is fed to, usually in a feed-forward manner (one direction). Neural networks are applicable to a wide array of statistical learning tasks, possessing great capability at finding intricate patterns and correlations in the data provided.

Components of Neural Networks

Most artificial neural networks are made up of three components. The input layer, hidden layer(s) and an output layer.

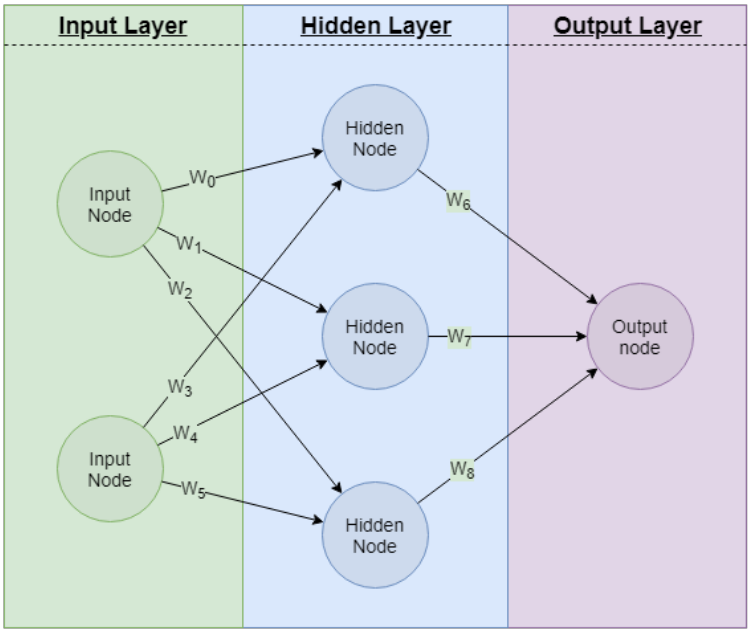


FIGURE 2.6: Neural network architecture [4]

Neurons

An artificial neuron, or node, is a simple unit of a neural network that receives information, processes simple calculations and passes the computed value further. The information a neuron receives can either be from the raw data into the input layer or from neurons in a previous layer. The neurons within a neural network are organised in layers. The input neurons receive the raw data, hidden neurons process this information and the output neurons provide the result.

Weights

The weights are scalar values that are adjusted during training according to underlying patterns that are found within the input data. Neurons are connected by synapse which have weights assigned to them. These weights determine the strength of connection and thus the overall influence of the previous neuron.

Activation Functions

A neural networks activation function (Figure 2.7) determines the threshold at which a neuron is activated. It essentially defines the output of the neurons using the weighted sum of inputs for a particular layer in the neural network.

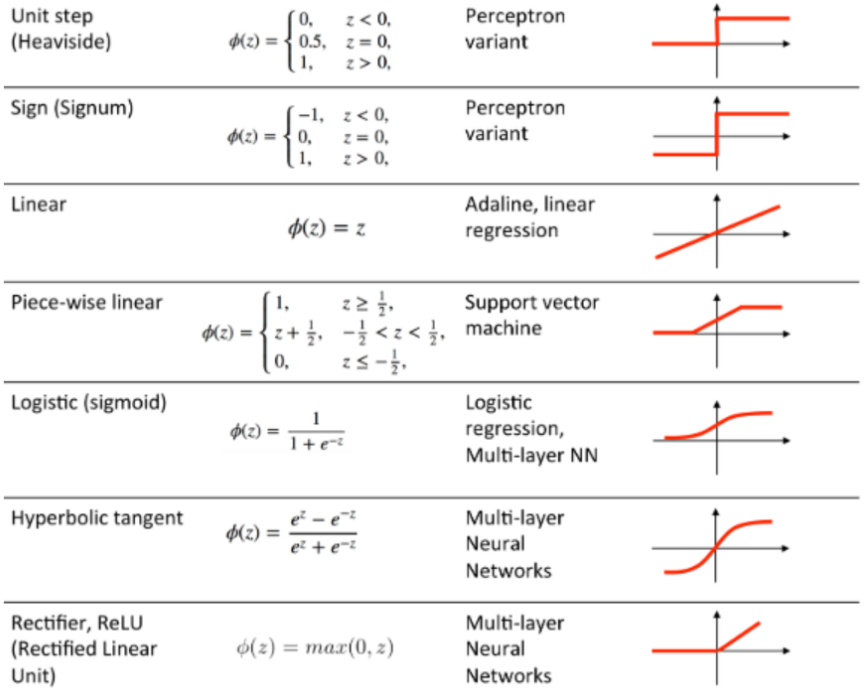


FIGURE 2.7: Activation functions [5]

Often non-linear activation functions are chosen, as these functions can model more complex problems. Neural networks are considered 'universal function approximators', meaning that they can approximate any function that maps real numbers to a given output. The popular non-linear *sigmoid* function outputs values that are squeezed between $(0, 1)$ in an S-shaped distribution. The *tanh* function uses the same S-shaped distribution but allows for positive and negative outputs with a wider range between $(-1, 1)$. *ReLU* is another popular activation function due to to its ease of use and efficiency, it cuts off all negative values at 0. Thus, the range of ReLU is between $(0, \infty)$.

Input Layer

The input layer is the start of the neural network, where the data is fed through. The dimensions of a neural network's input layer must have the same number of dimensions as the input data. The dimension is the number of features in each sample of the input data.

Hidden Layer

The layers between the input and output of the neural network are the hidden layers, this is where the major computation of the algorithm is done. The hidden layer applies non-linear transformations by applying weights to inputs and using an activation function to produce a desired output. The more hidden layers in a network the 'deeper' the network is.

Output Layer

The output layer is the final layer of neural network, where the number of neurons is set to the desired number of outputs for the prediction. It receives numerical data that is combined and summed before being subject to an activation function. The output is then in the format required for the task. If the task was binary classification, then the number of neurons in the output layer is set to two.

Types of Neural Networks

Feed-Forward Neural Networks

Feed-forward neural networks only process information in one direction, there are no cycles within the network. Feed-forward networks are usually applied to non-sequential data, due to it having no ability to look 'backwards' in time.

Perceptron

The most shallow feed-forward neural network is a perceptron, which is also known as a single-layered neural network. There is no hidden layer within a perceptron, they consist of an input layer, weighted connectors and an output layer [43].

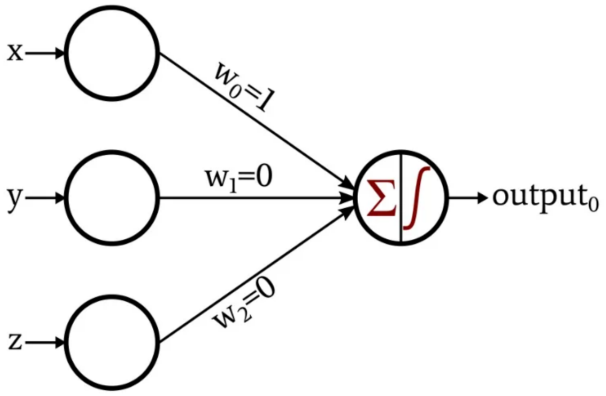


FIGURE 2.8: Perceptron, single-layer neural network [6]

Multi-layered Perceptron

The multi-layered perceptron (MLP) [44] is a feed-forward neural network that has hidden computational layer(s) as seen in Figure 2.6. The term 'multi-layer' refers to the hidden layer(s) that are located between the input and output layers. The MLP neural network is trained using the backpropagation algorithm [45] which computes the gradient of the loss function with respect to the weights. The difference between the predicted outputs and the true outputs is used, along with the computed weight gradients, to update the weights with the aim of minimising the loss function. It is known as backpropagation due to the manner in which the error is propagated backwards from the output layer to the input layer.

Recurrent Neural Networks

Recurrent neural networks (RNNs) [46] are a type of neural network that was created to specifically process sequential data. RNNs (Figure 2.9) have a 'memory' that allows the algorithm to retain important information from prior inputs in the hidden state of the network. This hidden state is updated after each time step. The hidden state gives RNNs the ability to capture temporal dependencies in the data using a feedback loop, where the algorithm uses previous outputs (stored information) as inputs.

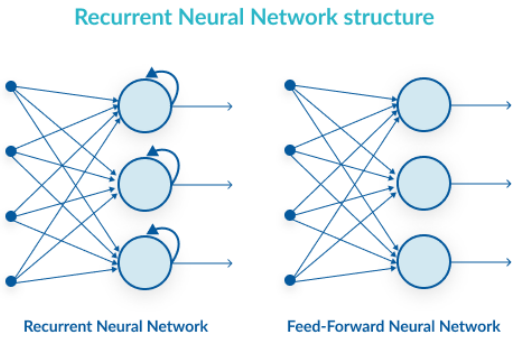


FIGURE 2.9: Recurrent Neural Network [7]

Long Short-Term Memory Networks

The shortcoming of most RNNs is their struggle to capture long-term dependencies in sequential data due to the vanishing gradient problem. The vanishing gradient problem occurs when computed gradients get smaller and smaller as they propagate backwards through the network, thus making it difficult to learn long-term relationships in the dataset.

Long short-term memory (LSTM) networks [47] were created to solve this issue, by including a memory cell which can systematically recall or forget information over time. The LSTMs memory cell is made up of three *gates*: input, forget and output which control the memory cell. The input gate governs how new data enters the memory cell, the forget gate decides what data is kept or forgotten, and the output gate determines the data used to generate the output.

TabTransformer

A transformer is a state-of-the-art neural network architecture, used mainly in the domain of natural language processing (NLP) tasks. Transformers are thus often used for sequence-to-sequence machine learning tasks, where an encoder is fed an input sequence and maps it to a higher dimensional space. This higher dimensional vector is then received by the decoder which generates an output sequence. An example application is the generation of new text from input given by a user. This could, for instance, be used by chatbots that help people use services online. Another example of the power of transformers is Google Translate, which in 2017 switched from a stacked LSTM model to a full transformer-based model to translate languages on the web.

The TabTransformer (Figure 2.10) is a transformer-based deep learning machine learning algorithm that seeks to improve on the Multi-layer Perceptron (MLP) [8]. This is done mainly by transforming categorical embeddings into robust contextual embeddings in the structured dataset. The TabTransformer embeds the categorical features in the tabular dataset using self-attention based transformers to create contextual embeddings. The standard way of feeding categorical features into a machine learning algorithm is through one-hot encoding. The problem with this technique is that it there is no 'context' between the categorical feature vectors. Transformers were designed for natural language processing where context between words is of utmost importance. Therefore, the contextual embeddings are created through transformers that capture intricate relationships between categorical features and variables, often resulting in improved performance compared to the MLP. Overall the research on transformers and tabular data is still in its infancy, however research such as [8, 48] demonstrates that transformers

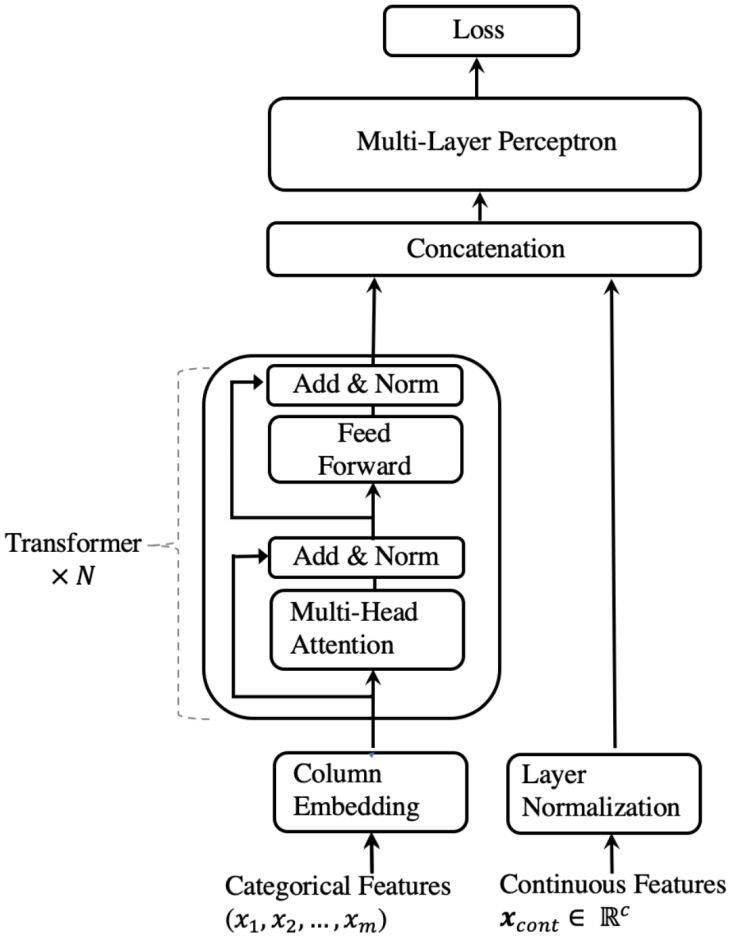


FIGURE 2.10: TabTransformer Architecture [8]

can match the performance of tree-based models and outperform other deep learning methods including MLPs, Sparse MLPs and TabNet on tabular data.

2.5 Data Imputation

Data imputation is the process whereby missing data values (N/As) are replaced with substitute values to preserve most of the information within the dataset. Missing data causes a variety of problems including introducing bias into the data, being difficult to analyse and the majority of machine learning algorithms are unable to work with N/A values.

Simple Imputation

Listwise Deletion (Complete Case Analysis)

The most reliable way of dealing with missing data is listwise deletion, this involves deleting any data points that contain a N/A value in any of its features [49]. This ensures that data is not replaced incorrectly and has a low computational cost. Listwise deletion is not a feasible option when a large fraction of samples contain a N/A value, which is often the case with large datasets with a high dimensionality. If there are 300 features per sample and a datapoint is only missing a single feature then deleting the datapoint would be wasting valuable information.

Mean and Median Imputation

The mean/median is calculated for each feature F in the dataframe using non-missing values of the same feature. These are then used to impute the missing values of F. Median values are more robust to outliers, and thus is preferred when the data is skewed. Computing mean/median has a low computational cost. This method can however lead to biased estimates of variances and covariances. It can also introduce leakage due to calculating mean/median over the whole dataset.

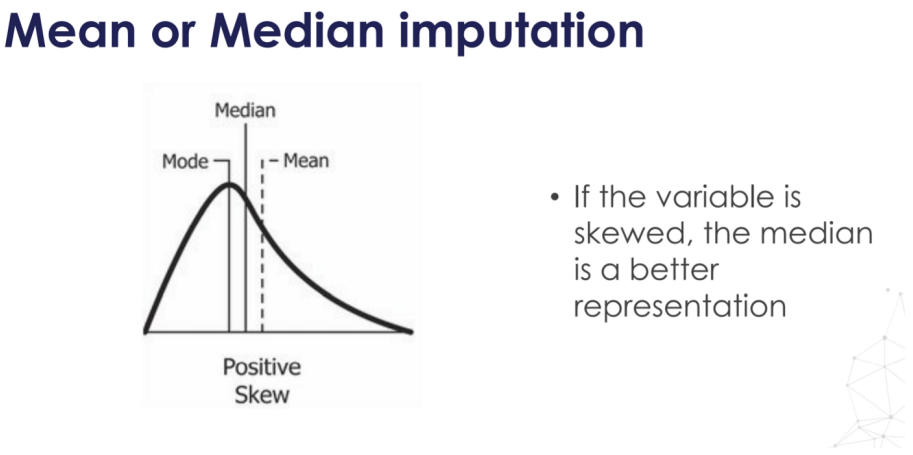


FIGURE 2.11: Mean or Median [9]

A rolling mean/median can be used to remove any sort of leakage when imputing. The rolling imputation utilises a window parameter, which looks back a certain number of recent observations and computes the mean/median of these observations. This way the imputation technique never looks forward in the database to complete its imputation.

Next Observation Carried Backward

Next observation carried backward (NOCB) fills missing values by finding the first non-missing value after the N/A and backward filling the N/A with this value. It is not as

popular as its counterpart last observation carried forward (LOCF) due to its tendency to introduce leakage into the dataset.

Last Observation Carried Forward

Last observation carried forward (LOCF) is similar to NOCB, however it imputes in the opposite direction. LOCF fills missing values with the last observed non-missing value within the feature F. LOCF is mostly applied to sequential research where feature vectors are consistently observed at pre-specified intervals [50]. The drawback for this technique is it cannot fill missing values if they are at the beginning of the database (there are no previous values to carry forward).

Conditional imputation

Conditional imputation uses independent variables to identify groups in the dataframe and imputes missing values within these groups. An example of this in financial data is grouping stocks by their tickers. Thus, if a stock has a N/A value, then previous timestamps of the same stock will be consulted to determine the substitute for the missing values. Conditional imputation can be applied to mean/median, NOCB and LOCF imputation.

Multiple Imputation

Multiple imputation (MI) estimates missing feature values using all other features available. This technique was proposed by David Rubin [51] to overcome the noise created using single imputation methods. Multiple imputation creates multiple imputed datasets using an estimator of choice, these imputed datasets are then averaged to give the final imputed values. The goal of MI is to not to predict the missing values exactly, but rather an exercise to try and achieve valid statistical inference [52]. The MICE and Miss Forest statistical packages both perform multiple imputation and have been shown to be reliable with large tabular datasets [53, 54]. Multiple imputation does have a much higher computational cost compared to simple imputation and is important to note moving forward.

Multiple imputation is broken down into 3 sequential steps:

- Imputation of m generated datasets using an estimator of choice.
- Analysis of the m generated datasets is conducted, each imputed dataset will have slightly different imputed values.
- Aggregating of the m generated datasets' imputed values to have a consolidated result with less variance than single imputers.

MICE

The most popular MI technique is multivariate imputation by chained equation (MICE). It calculates the mean of every column that contains N/A values and uses it as a starting point [53, 54]. It then runs chained equations to impute each missing N/A value sequentially, this is most often done starting with the features with the least N/As to the features with the most N/As in the original dataset. MICE functions like any machine learning algorithm in that it has feature variables and a target variable. The target variable is the column whose N/As are trying to be substituted and the feature variables are all other features in the feature matrix. Iteratively, MICE repeats this process m times, changing the starting variables each time to create a robust estimation. MICE is able to impute missing values of mixed data types which is a significant advantage over most simple imputers.

MissForest

MissForest is a non-parametric multiple imputation technique which works on mixed data types [55], it can thus work with both categorical and continuous data. It is a random forest regressor which predicts missing features using all other features available. MissForest can do imputations in parallel which lowers its computational cost. However, it tends to have a high computational cost when dealing with large datasets.

KNNImputer

The k-Nearest Neighbors imputer (KNNImputer) is a multivariate imputer which predicts missing feature values using the nearest k-samples by Euclidean distance. The missing value is substituted with the mean of the k values found in the dataset.

2.6 Data Sampling

Imbalanced datasets are often helped by transforming the dataset to achieve a less skewed distribution. There are a variety of techniques to 'rebalance' training data which in turn helps algorithms train on more samples from the minority class. This often reduces the bias in datasets that often causes algorithms to highly favour the majority class when fitting.

Random Oversampling

Random oversampling is the most basic way of balancing a skewed dataset. The technique randomly duplicates samples from the minority class to make the training dataset more balanced. The scale of that balance is a choice made during implementation.

SMOTE

Synthetic Minority Oversampling Technique (SMOTE) is a way of increasing the volume of the minority class by creating synthetic examples of the minority class. SMOTE differs from simple random oversampling in that it creates new unique minority class samples, rather than cloning the original minority class samples to balance the class distribution. SMOTE creates this new information using the original minority samples feature space and thus the values produced are considered plausible for a minority class sample. The SMOTE technique as defined by the original paper [56] is implemented as follows: A feature vector and its nearest neighbour are selected from the minority class and subtracted from one another. This difference is multiplied by a randomly generated float in the range of 0 to 1, and then added to the first feature vector. Thus, every feature vector created is at a point between two of the original samples in the minority class feature space.

ADASYN

Adaptive Synthetic Sampling Approach (ADASYN) is a more complex oversampler which creates synthetic samples based on the local distribution of the minority class feature vectors. ADASYN generates more synthetic feature vectors in the less populated regions within the local feature space and produces less samples in the more populated areas [57]. The two functions that are provided by ADASYN include: reducing bias within the database by creating a more balanced dataset and shifting the main oversampling focus to the area around the decision boundary (least populated area).

Random Undersampling

The simplest undersampler randomly removes samples from the majority class to balance the dataset.

TOMEK LINKS

Tomek Links is an undersampling technique that strives to remove noisy borderline majority class feature vectors. Tomek's algorithm [58] searches for pairs of feature vectors that are of opposite classes and are closest in terms of Euclidean distance. Once the pair is found the majority class feature vector is deleted from the pair, this often rids the dataset of ambiguity around the decision boundary.

Edited Nearest Neighbours

The Edited Nearest Neighbours (ENN) technique [59] is similar to Tomek Links in that it aims to remove majority samples near the decision border. The technique sets k to 3 by default for the kNN algorithm. ENN runs through kNN on every sample from the majority class, if the sample is correctly classified as a majority class it remains, otherwise it is deleted from the dataset. ENN uses the same procedure on each sample from the minority class, where the neighbours within $k = 3$ that are of the majority class are deleted.

Condensed Nearest Neighbour

The Condensed Nearest Neighbours (CNN) [60] is a method that only keeps majority samples that the kNN algorithm struggles to identify. The resampling starts with a subset of all the minority samples. Then for every minority sample in the dataset the kNN algorithm is used to find the closest majority sample (different label) to add to this subset.

One-Sided Selection

One-Sided Selection (OSS) is a combination of CNN and Tomek Links. The Tomek Links technique is applied first to remove noisy borderline majority samples. Then CNN is used to target and remove samples that are further from the decision border [61].

Neighbourhood Cleaning Rule

The Neighbourhood Cleaning Rule (NCR) is another combination method which uses CNN and ENN. It is based on similar principles to OSS, where one technique is used to remove noisy samples by the decision border (ENN). Then CNN is applied to reduce the number of samples closer in the interior of the majority class [62].

2.7 Summary

This chapter provides a broad overview of the various fields that are involved in the prediction of mergers and acquisitions. Having surveyed the background, these concepts will be fundamental in the following chapters. The next chapter is more specifically interested in related literature on predicting M&As, comparing their research and identifying fundamentals and flaws that should be explored in the thesis.

Chapter 3

Literature Review

This chapter evaluates related literature around imbalanced classification, and more specifically the prediction of potential acquisition targets using statistical and machine learning models. An evaluation is conducted on the performance of statistical and machine learning models when predicting M&As. There are six foundational hypotheses which represent the kinds of companies that most often become mergers or acquisitions. These hypotheses are used to select relevant financial characteristics for the creation of acquisition prediction models. Many past studies on the prediction of M&As choose a set of features due to their popularity in the financial world and then cull that down using empirical analysis.

Six hypotheses are considered fundamental in M&A literature when predicting takeover targets. The hypotheses are also considered when choosing financial features to create a model. The inefficient management hypothesis suggests that acquirers will bid for poorly run companies. The bid is made under the premise that the market value of the firm will improve with a different management structure [24]. The inefficient market hypothesis has been proxied by financial features such as profitability, dividend yield and excess stock return [63].

Growth-resource mismatch hypothesis states that companies with a large mismatch between growth and available resources are likely targets. These are either companies with a high level of growth opportunities and no financial slack [64], or companies with fewer growth opportunities and available resources. The features important to the growth-resource mismatch hypothesis include average sales growth rate, liquidity and leverage [24]. The growth portion of the hypothesis is measured by the average sales growth rate whilst the last two features represent the resources available to the firm.

The industry disturbance hypothesis is an economic theory that suggests that acquisitions can cluster by industry. Thus the likelihood of a firm being acquired is increased if there are recent acquisitions made within the same industry [65, 66]. The features extracted for this include GICSSector and GICSIndustryGroup. These variables can then be used to create more variables such as the percentage of industry population acquired in recent periods.

The company size hypothesis claims that a company is more likely to be acquired the smaller the size of the company. This is due to the transaction cost of a takeover [24], this can be especially prominent in the event of a hostile takeover. Smaller companies are easier to overwhelm compared to larger companies that are more equipped to defend themselves against potential takeover bids [65]. The features used to describe company size are the net book value of the company and the market cap of the company.

Market-to-book hypothesis refers to companies whose market value is considered low relative to their book values. This is what many analysts will look for when stock picking as the market-to-book hypothesis is based on buying companies that are 'cheap' considering the financial ratios they produce [24]. The features related to the market-to-book hypothesis are market value, book value and the ratio between them.

In the Price-to-Earnings (P/E) ratio hypothesis, companies with high P/E's will buy out companies with low P/E's to create an almost instant capital gain [24]. This capital gain is a result of the combination of the P/E of the acquirer company and the target company growing the acquiring firm's P/E. Thus, the P/E ratio is used in the M&A dataset.

3.1 Traditional Statistical Architecture

Hasbrouck (1985) [67] conducts an empirical analysis of the difference in financial characteristics between target and non-target companies using logistic regression. The share samples are controlled by matching the non-target firms with target firms by industry and size. This supposedly created a more level view of the companies' financial characteristics which can then be compared more 'fairly'. Hasbrouck found that Q-ratio (market value/assets' replacement cost), current financial liquidity and financial leverage are important in differentiating target and non-target firms. This was one of the earliest papers on predicting M&As and thus the data sample is small with a total of 258 firms used in the analysis.

Palepu (1986) [24] examined the fundamental hypotheses in M&A literature using robust modelling techniques to avoid bias being introduced into the data sample. To further

improve robustness, Palepu did not incorporate matched sampling, a technique that ensures for every target company there is a similar non-target company in terms of size and industry. Matched sampling was a popular technique at the time, but its application created a dataset that did not correctly represent the population. This led to biases that would overstate the predictive ability of the models. Logistic regression models were used as they can handle more noisy data and have the added benefit of choosing a probability cutoff. The cutoff is the point which separates samples into positives or negatives. The standard cutoff is 0.5, where any sample with a predicted probability of greater than 0.5 would be classified as positive and below 0.5 would be classified as negative. The cutoff can be adjusted to a higher value to only classify the most confident predictions as target firms or to a lower value to allow more target predictions. Palepu stresses the importance of the optimal cutoff. Too high a cutoff will misclassify more targets as non-targets and result in fewer predicted targets. On the contrary, a low cutoff will predict many non-targets as targets and have a larger number of predicted targets. Palepu's model was tested on 1117 companies, of which 30 were targets and the remaining 1087 were non-targets. It achieved an accuracy of 80% (24/30) on the 30 target firms, but also predicted 55% (601/1087) of the non-target companies as targets. The study concludes that the model does not predict the takeovers accurately.

3.2 Machine Learning Architecture Analysis

There are only a handful of machine learning algorithms which have been researched and applied to the merger and acquisition context. It is important to note most research around predicting takeovers predated modern machine learning and focused on empirical and traditional statistical models. The research articles are systematically evaluated within the taxonomy, with individual strengths and weaknesses identified, serving as a basis for scoring. These features include the ability to deal with imbalanced data, feature selection analysis, computational cost, prediction window and evaluation quality. Dealing with imbalanced data refers to the performance of the models used in the paper, feature selection analysis: the reasoning behind the features selected for the dataset, evaluation quality: the insight given into the results of the paper, the use of graphs and tables to help explain the performance. The prediction window is the amount of time that we are trying to predict into the future, and computational cost is the compute time and power necessary for these algorithms to achieve the results displayed in the paper.

Table 3.1 is a high-level literature review of the machine learning applications in the field of predicting mergers and acquisitions. The table compares research papers based

on some key aspects, the higher the number recieved the better the aspect has been implemented. It assesses whether each study addresses the issue of imbalanced data, a common problem in predicting M&A events. The degree of feature selection analysis, indicating the extent the research has engaged in identifying the most relevant features for predicting mergers and acquisitions. Computational cost is considered, offering insights into the resources necessary to perform the study. The prediction window column refers to the temporal scope in which the models try to make predictions. Lastly, the evaluation quality metric provides a rating for the robustness of the validation methods used to assess the ability of the models to predict M&As.

Literature Comparison					
Author	Deals with Imbalanced Data	Feature Selection Analysis	Computational Cost	Prediction Window	Evaluation Quality
Ragothaman et al. (2003)	1	1	Low	Annually	2
Tsagkanos et al. (2007)	1	2	Low	Annually	1
Wei et al. (2009)	2	2	Low	N/A	2
Xiang et al. (2012)	2	3	High	Annually	3
D' Angelo (2012)	1	2	Low	N/A	2
Francisco (2017)	2	3	High	Annually	3
Luo (2017)	3	3	High	Monthly	3
Arroyo et al. (2019)	3	3	High	Annually	3
Geha (2021)	2	2	Moderate	Annually	2
Aramyan (2021)	2	2	Low	Annually	3

TABLE 3.1: Literature comparison of machine learning M&A papers

Ragothaman et al. (2003) [68] embarked on a study to forecast corporate acquisition events utilising uncertain reasoning through rule induction. The research details the creation and evaluation of ACQTARGET, a prototype expert system designed to assess potential acquisition targets. The study used a dataset made up of eight financial variables informed by previous academic investigations into M&A. A training set of 130 firms, equally comprised of 65 acquired companies and 65 non-acquired companies. The

holdout set was also balanced but only consisting of 32 acquired and 32 non-acquired. The performance of ACQTARGET was benchmarked against traditional statistical models, namely discriminant analysis and logit models. The findings of Ragothaman et al. demonstrate that the ACQTARGET expert system is on par with established statistical models in classifying firms as acquisition targets or non-targets. This result is particularly noteworthy as it suggests that expert systems with induced rules can serve as valuable tools for financial analysis in the domain of corporate acquisitions. A limitation of the study is the balanced nature of the training and holdout set, where M&A events are far more rare than this dataset would suggest.

Tsagkanos et al. (2007) [69] used machine learning as a new approach to predicting the M&As on the Greek (Athens) Stock Exchange. The purpose of their study was to see if a profitable predictive accuracy can be attained using novel machine learning approaches. Greece is a small open economy, whilst most studies done prior to this were on large open economies such as the United States of America, Canada and the United Kingdom. Thus, their research provides focus on the prediction of takeovers in small open economies. This is the first study using decision trees to make predictions on M&As as all previous studies had used multiple discriminant analysis or logistic regression. The dataset included Greek stock market data in the period 1995-2002 containing 51 target companies and 290 non-target companies. They achieved a predictive accuracy of 42.86% on targets in the holdout set. The high precision of 42.86% is due to the holdout set having a 1:10 ratio of targets to non-targets. The real population of the Greek dataset only had 2% of the firms as targets. The paper concluded that predicting takeovers is challenging and did not achieve any meaningful results.

Wei et al. (2009) [70] explored the potential of patent data as a strategic resource in forecasting M&A activities using a Naive Bayes model. Their work, presented in the context of e-business systems design, suggests that the quantity and nature of patent filings provide valuable insights into a firm's strategic direction, potentially identifying companies as targets or acquirers before an M&A event occurs. In their innovative approach, Wei et al. address the gaps in traditional M&A literature by incorporating technological indicators from patent analyses and considering compatibility of both bidders and potential targets in their predictive modelling. The researchers defined a comprehensive set of features for their analysis, including the number of patents granted to a company, the number and impact of recent patents, and the company's technological breadth. In their study the application of ensemble learning algorithms to rebalance datasets significantly mitigated data skewness, achieving a true positive rate of 46.43% in identifying acquisitions among 2,394 companies, of which 61 were actual acquisition targets. A limitation of the study is the temporal relevance of the model, as models

based on data in a specific period may not perform as well in future periods due to the evolution of market dynamics.

Xiang et al. (2012) [71] presented a novel approach using textual information compared to past research in the field which only used numerical features. The work is the first in the M&A prediction space to explore topical modelling over news articles, and using premier sources such as TechCrunch. Xiang et al. focused on extracting features from CrunchBase profiles, including basic, financial, and managerial aspects, and combined these with topic features obtained from TechCrunch articles using Latent Dirichlet Allocation (LDA). This approach allowed them to create a more robust prediction model by leveraging the richness of unstructured textual data alongside factual company information. The study's experimental setup was notable for its scale, analysing a substantial corpus of TechCrunch news articles and CrunchBase profiles. The results demonstrated a high true positive rate (TP) between 60% to 79.8% with a relatively low false positive rate (FP) between 0% to 8.3%, showing the effectiveness of their methodology in accurately identifying potential M&A targets. The inclusion of diverse data sources like TechCrunch represents a significant advancement in M&A prediction.

Geha (2021) [25] conducted a study to predict corporate takeover events with a variety of machine learning algorithms. The stock market data was collected as monthly data of 14370 firms where 12% were targets, the dataset spans from 2010-2019 using two different databases to create the dataset. The two databases include the Securities Data Company (SDC) Platinum database, providing key information on takeover targets (the target variable) and the Compustat Merged (CRSP) database for all the independent variables (features) needed to create the prediction models. Geha's study was detailed, using many different machine learning algorithms and performance metrics to find the best predictive ability on mergers and acquisitions. The algorithms tested were logistic regression, lasso, ridge regression, CART and random forest. The performance metrics to measure the algorithm's performance included accuracy, sensitivity/recall, specificity, false positive rate (FPR), false negative rate (FNR) and area under the curve (AUC). The random forest performed the best out of sample on the aggregate of metrics mentioned above, notably sensitivity of 35.7% and AUC of 0.66. The research shows the variables that each algorithm favoured when training models and are important to note. Geha's study has an inherent flaw of not representing the real rate of takeovers within the holdout set. The holdout has approximately 12% of firms as targets, the actual population percentage is closer to 1-2%.

Luo (2017) [12] proposes an ensemble of models to create the SMAP (Systematic Merger and Acquisition Prediction) model. This ensemble includes logistic regression, random forest, AdaBoost, support vector machine and an artificial neural network. Due to the

highly skewed data, bagging and paired sampling were utilised with the goal to reduce bias. The ensemble proves to be more robust and produces a higher predictive power than any of the machine learning algorithms independently. The model is trained on monthly data extracted from the Russell 3000 index over 30 years. The target firms represent 0.5% of the universe, the SMAP model achieves approximately 10% precision when predicting target firms. This is the most optimal study to date using a holdout set that correctly represents the M&A population. The SMAP model generates a probability of being taken out for each company in the holdout set. When the results are grouped into deciles (10 groups) the precision increases with each predicted decile. To put this in perspective the first decile (Q1) captures 2.5% of possible takeovers whereas the last decile (Q10) captures 25% of takeovers.

Francisco (2017) [72] utilising CrunchBase data, created models including random forest, logistic regression and a support vector machine. The random forest produced the best results, achieving a True Positive Rate (TPR) of 94.1% and a False Positive Rate (FPR) of 7.8%, the highest accuracy reported with this dataset. The model stands out for its ability to classify start-ups with high precision (92.2%) and an AUC of 93.2%, surpassing previous studies' results. The study's innovative approach involved transforming features into binary variables, which, while increasing computational time, enhanced the model's effectiveness. A limitation of the study is within the dataset's composition, where 16.8% of companies were involved in M&A events, this is significantly higher than the natural occurrence of around 0.5% [12]. Thus, the model may not perform as well in typical market conditions where the frequency of M&As is far lower.

Arroyo et al. (2019) [73] explored the application of machine learning techniques to enhance decision-making in venture capital investments, particularly for early-stage companies. Recognizing the high uncertainty in this sector, their study sought to provide a more comprehensive risk assessment tool than those currently available. They focused on expanding beyond the traditional binary classification of predicting whether a company would be acquired or go public. Instead, their model also predicted other outcomes like receiving further funding rounds or closure, providing a broader perspective on potential investment returns. They incorporated a variety of algorithms including decision trees, random forest, gradient boosted tree and support vector machine. Among the various algorithms tested, the Gradient Boosted Tree model stood out, particularly for its ability to predict acquisitions (AC). This algorithm achieved a precision of 0.55, making it the most effective method in the study for identifying companies likely to be acquired. This is a significant result as the proportion of M&A events in the dataset was only 2.7%, closely aligning with the naturally occurring rate in the global market.

Aramyan (2021) [27] set out to build a mergers and acquisition predictive model that could provide an excess return on investment. The logistic regression model was chosen due to its robustness and proven performance in M&A literature, features were selected based on past empirical studies. The dataset is made up of large US companies from April 2010 - June 2021 and is pulled from the Refinitiv workspace. The holdout set contains 84 target companies and 1704 non-target companies, thus the holdout set is about 5% of the universe. The out of sample model achieved the following results: AUC ROC: 0.6, Accuracy: 0.58, Precision: 0.06, Recall: 0.57, False Positive Rate: 0.42 and F1 Score: 0.11.

3.3 Summary

This chapter examined related work that aimed at forecasting mergers and acquisitions. A combination of traditional and machine learning architecture was studied to find features that may be important in the prediction of M&A events. After reviewing the machine learning architectures in the context of mergers and acquisitions, it is evident that there is a significant research gap in leveraging more powerful machine learning algorithms such as gradient-boosted trees and neural networks. The works of Luo et al. [12] and Geha [25] were particularly insightful into the methodologies required to conduct a comprehensive study on the prediction of mergers and acquisitions.

Chapter 4

Experimental Design and Datasets

This chapter contains a summary of the data used to conduct the experiments, a high-level view of the experimental framework and the computing environment in which the experiments were performed.

4.1 Datasets

The data used in the experiments is financial stock market data pulled from the Refinitiv Eikon database. Data was extracted at weekly periodic intervals, as this was seen as a good compromise of efficiency and amount of data obtained. The daily data yields five times the weekly data but introduces a large amount of noise that occurs due to the large volatility of the market from day to day. The stock market data is pulled over a twenty-year period for this study, thus daily data has the added disadvantage of creating too large a dataset for the compute power available as well as hitting data limits on the Refinitiv Eikon license. Monthly data although more robust to the volatility of the market, allows far less data to be obtained compared to weekly data.

The data extracted is from the beginning of January 2000 to the end of July 2019, this period was chosen in order to avoid the COVID-19 pandemic which started at the end of December 2019. The data was pulled weekly on Friday after the close of the New York Stock Exchange (NYSE) and the NASDAQ. The US stock exchanges are the largest in the world and represents a large portion of the dataset, thus the choice to pull data on their closing time makes the most sense for our database.

To create the database, the top 10000 companies in terms of market cap (size) were pulled from the global database every week. Any companies that were involved in mergers or acquisitions were classified as a 1 in the week before the announcement. Any company that was not a merger or acquisition was labelled as a 0. This created our highly imbalanced dataset where 0.4% of the data were M&As. The features used in the dataset are listed in the appendix (Figure A.5). The database is quantile transformed (grouped by date) to a normal distribution to normalise skewed distributions and lessen the impact of outliers [74]. Once quantile transformed the data is min-max scaled (grouped by date) between 0 and 1 as many machine learning algorithms perform better when input features fall within a relatively small range.

4.2 Experimental Design

Classifiers

The classifiers utilised when conducting the study are displayed in table 4.1. The shorthand is also given which will be used to refer to these algorithms going forward. The logistic regression, random forest and LGBM implementations are imported from *Scikit-learn* machine learning library. The LSTM and TabTransformer are implemented through *Keras*, a high-level API of *Tensorflow 2*.

LR	Logistic Regression
RF	Random Forest
LGBM	Light Gradient-boosting Machine
LSTM	Long Short-Term memory
TT	TabTransformer

Table 4.1: Classifier Definitions

Machine Learning Framework Implementations

Python The de facto standard programming languages for data science problems are Python and R. Python has been chosen due to familiarity with the language and it holds powerful tools for data science. These tools include data science libraries such Keras, Scikit-Learn, Matplotlib, PyTorch and many more.

Scikit-Learn Scikit-learn (Sklearn) is a machine learning software library that allows easy access and use of otherwise complex machine learning algorithms. The algorithms in Sklearn that are utilised for this study include the random forest classifier, logistic regression and LGBM classifier.

Keras Keras is a high-level framework that is built on top of Tensorflow 2, it provides a python interface of neural networks. The algorithms that we use from Keras include a perceptron, a multi-layered perceptron and a LSTM.

Jupyter Notebook Jupyter Notebook is an integrated development environment (IDE) that runs via a web browser. The environment has been designed around the field of data science and research applications. It has powerful data visualisation and text editing tools that are not offered with other IDEs.

Imbalanced-Learn Imbalanced-learn (Imblearn) is an open-source MIT-licensed library which allows access to tools specifically created for imbalanced datasets.

Metrics

The most popular metric in machine learning is the classification accuracy measurement:

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}}$$

The classification accuracy metric is avoided when measuring the effectiveness of a model on imbalanced data, as the accuracy will be skewed by the large difference in class frequency. A high accuracy can be achieved by just predicting the majority class. This phenomenon is known as the accuracy paradox, high accuracy does not necessarily mean strong predictive ability.

There are, however, a variety of metrics that give us a good idea of the true performance of a model in relation to imbalanced datasets. The confusion matrix is a visual summary of prediction results for a classification problem, it helps identify where your algorithms strengths and weaknesses are in terms of prediction. To construct a confusion matrix two parameters are necessary, true labels and predicted labels.

In the binary case this table will also exactly represent true positives, false positives, false negatives and true negatives. The table below shows the confusion matrix for a binary classification problem with classes A and B.

	MA_{true}	$non - MA_{true}$
MA_{pred}	TP	FP
$non - MA_{pred}$	FN	TN

- True Positive (TP) - Where class A is predicted and it is class A.
- False Positive (FP) - Where class A is predicted but it is class B.
- False Negative (FN) - Where class B is predicted but it is class A.

- True Negative (TN) - Where class B is predicted and it is class B.

Using TP, FP, FN and TN we can derive many metrics that are more applicable than classification accuracy when dealing with class-imbalanced datasets, as their results aren't skewed by class frequency. These include precision, recall, F1 score, area under the precision-recall curve (PRAUC) and area under the receiving operating characteristic curve (AUC ROC). The precision recall curve illustrates the trade-off between precision and recall at all classification thresholds. Similarly a ROC curve plots recall against the false positive rate across the same thresholds. The Area Under the Curve (AUC) of both these plots is a single-number summary often used to quantify a classifiers performance in an imbalanced context.

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

FIGURE 4.1: Precision measures the proportion of true positives out of all predicted positives.

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

FIGURE 4.2: Recall (or Sensitivity) measures the proportion of true positives out of all actual positives.

$$\text{False Positive Rate (FPR)} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

FIGURE 4.3: False Positive Rate measures the proportion of false positives out of all actual negatives.

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

FIGURE 4.4: F1 Score is the harmonic mean of precision and recall, balancing both metrics.

$$\text{F0.5 Score} = (1 + 0.5^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{(0.5^2 \cdot \text{Precision}) + \text{Recall}}$$

FIGURE 4.5: F05 Score is a variant of F Score, giving more two times weight to precision than to recall

High-level Architecture Diagram

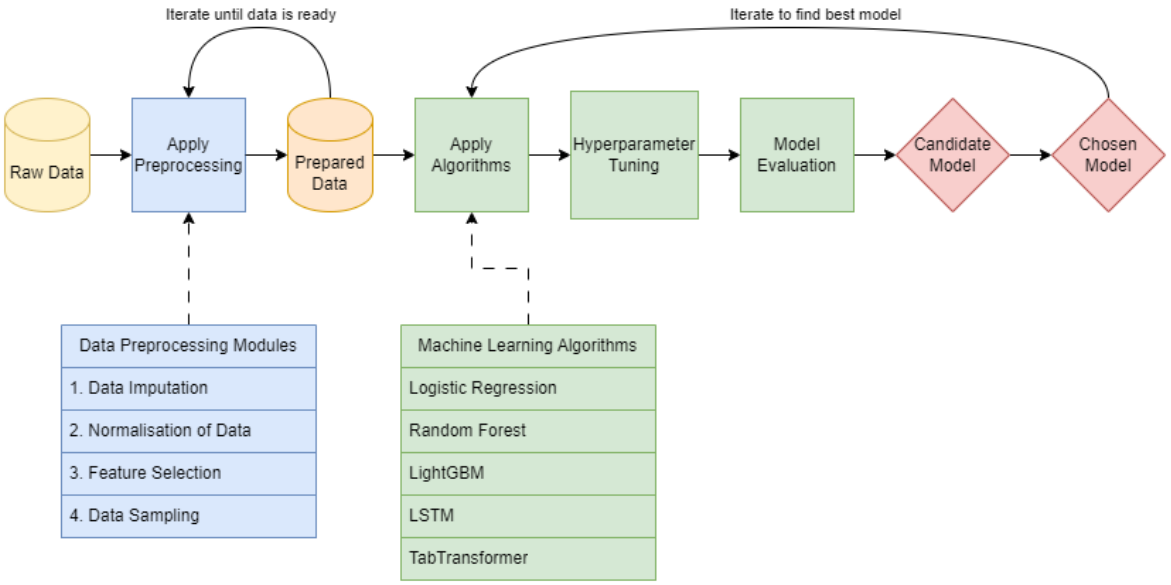


FIGURE 4.6: High-level architecture diagram

This study is setup as a binary classification problem, where target firms are labelled as 1 and non-targets firms are labelled as 0. The framework developed will use machine learning techniques to evaluate the models ability to predict mergers and acquisitions.

The workflow of the project consists of three main components: data collection, data preprocessing and implementing machine learning algorithms. Data collection will be done via the Eikon API. It is indispensable to collect a large amount of data to compensate for the highly imbalanced nature of the data. This also helps with the dimensionality problem that is inherent in having a large number of features (Figure A.5).

The preprocessing step will include imputation, scaling of the raw data, feature selection and lastly data sampling to put the dataset in an optimal state for algorithms to extract performance from the dataset. Thus, heavy experimentation is required for each part of the preprocessing that has been identified as a key aspect.

The Scikit-Learn machine learning package encompasses the majority of algorithms we will examine. The random forest classifier, logistic regression and LightGBM will be imported from Scikit-Learn. The LSTM will be imported from the Keras API, a software library which provides a Python interface for artificial neural networks. An algorithm will be selected from the focus subset to build a model to use on the preprocessed data. The Eikon data will then be split into two parts, training set and test set. The training set contains a training subset and a validation set. To find optimal hyperparameters purged time series five-fold cross validation will be run on the training subset and validation

set. The best parameters found will subsequently be used on the whole training subset to create an optimal model. This model predicts on the test set and is evaluated using several metrics.

Cross Validation

The standard K-fold cross validation seems to fail when applied to financial datasets [75], as it can contribute to overfitting. The reason K-fold CV fails in a financial context is because it assumes the data is independently and identically distributed (IID). Several features in the dataset could demonstrate autocorrelation, a condition whereby a value is correlated with its own historical values. Additionally, due to the temporal proximity of the observations, there is a risk of data leakage at the database level. Data leakage gives models an unfair advantage and thus does not provide a realistic assessment of the models performance. To mitigate this problem when using cross validation a small gap of data will be left out before the validation set of each fold and before beginning the next fold another gap will be left out. Thus, time series purged cross validation will be used as seen in Figure 4.7.

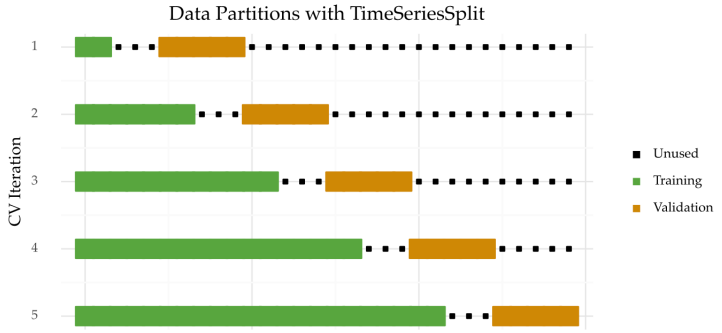


FIGURE 4.7: Time-series 5-fold CV [10]

GridSearch is the most simple and widely used hyperparameter tuning method, however more sophisticated methods are now available. One such method is Hyperopt a Python library for Bayesian optimization. Both Hyperopt and GridSearch are libraries for machine learning hyperparameter optimization, however they operate in slightly different ways. A straightforward and popular technique for tuning hyperparameters is GridSearch. A grid of parameter values must be specified, and the algorithm will train and assess a model for each set of parameter values in the grid. GridSearch can be computationally expensive, particularly when there are many possible parameter value combinations.

On the other hand, Hyperopt is a more advanced library for tuning hyperparameters. It searches for the ideal set of hyperparameters using an optimization approach called

Tree-structured Parzen Estimator (TPE). To narrow the search to the most promising areas of the hyperparameter space, TPE builds a probability distribution over the hyperparameters and updates it based on the model's historical performance.

In plain English, GridSearch is an exhaustive search approach that tries all conceivable parameter combinations, whereas Hyperopt employs a smart search method that learns from the model's prior evaluations and attempts to more effectively explore the hyperparameter space.

It is important to note that Hyperopt is typically more computationally effective than GridSearch and is especially helpful when there are several hyperparameters or a complex search space.

Sequence-to-sequence learning

For the validation and holdout sets, potential concerns regarding the LSTM's requirement for preceding datapoints are mitigated. Specifically, for the initial data point of a stock in the validation period, the model uses the terminal instances from the training set to form a complete sequence. This ensures continuity in the data fed into the LSTM, preserving the sequential context that is critical for its predictive performance.

The LSTM model necessitates that data for each stock be chronologically ordered, maintaining the temporal integrity essential for sequence-to-sequence learning. Thus, sequences for each stock are grouped together within the dataset:

All sequences for Stock A (Sequence 1, Sequence 2, ..., Sequence N) Followed by all sequences for Stock B (Sequence 1, Sequence 2, ..., Sequence N) And so on for Stock C, and subsequent stocks.

As the LSTM processes sequences from different stocks, its hidden states are reset to avoid information leakage. This reset is crucial; it ensures that the predictions for each stock are generated independently, based on its own historical data, without influence from the data of other stocks.

In contrast, the TabTransformer applied in our study is not a sequence-to-sequence learning model. It does not model temporal dependencies in the data. Instead, it focuses on capturing complex relationships and interactions between features within individual data points, applying self-attention to learn which features are most important without considering their order in a sequence. This characteristic makes the TabTransformer particularly well-suited for tabular data where the primary goal is to discern the influence of various independent features on the target variable.

4.3 Experiments

This section briefly discusses the experiments conducted and the motivation behind them. The experiments each focus on a different component of the machine learning architecture.

Imputation

The dataset has a large number of N/A values, which occur when there is no available data for a particular stock feature on a certain date. Many machine learning algorithms cannot run or have reduced performance with N/A values in a dataset. Thus, they must be removed or substituted for a value that the ML algorithms can process. To deal with N/A values, one must use an imputation technique. In the imputation experiment, a variety of simple and multivariate imputers are evaluated on four arbitrarily selected features from the dataset. These features were chosen, purged of all existing N/A values, and subsequently reintroduced with artificial N/A values at different proportions: 10%, 20% and 50%. This approach allows for a robust assessment of the effectiveness of imputation techniques to accurately impute known data. The performance comparison used the R-squared metric to measure the imputers ability to reproduce the original dataset.

Feature Selection

There are often redundant features that do not have any predictive power in relation to the target. These features make it more difficult for a model to learn meaningful relationships between the features and the labels, which may result in an increase in model complexity and a decrease in performance. Using feature selection algorithms to remove redundant features allows the models to focus on more important features, thereby reducing computational cost and model complexity. This process often results in an increase in performance [76]. In this experiment, the performance of the focus algorithms are measured across four different datasets. Three of these datasets are subsets reduced based on feature importances produced by LightGBM and random forest algorithms. The objective is to determine the impact of feature selection on model performance, measured through F05 score.

Hyperparameter Tuning

Hyperparameter tuning is a crucial step when optimising machine learning models and is particularly important when working with imbalanced data. Finding suitable hyperparameters can be challenging in these scenarios as common parameter settings often perform poorly in the presence of a skewed class distribution [77]. This is because most

algorithms expect a a balanced class distribution. Hyperparameter tuning allows models to mitigate overfitting and underfitting through the optimisation of the models' complexity and regularisation parameters. The comparison of different machine learning models with various hyperparameter configurations enables the selection of the best model for the specific problem and dataset.

Data Sampling

Data sampling techniques are a valuable tool when dealing with highly imbalanced datasets as they address the class imbalance problem [77]. This experiment compares standard data sampling techniques, including random oversampling, random undersampling, ADASYN, SMOTE, OSS and NCR, to investigate their potential advantages over non-sampled data. The evaluation will be performed using the F05 metric and PRAUC, which are commonly used to measure a model's ability in imbalanced classification tasks.

Holdout Performance

This is an evaluation, which incorporates all the insights from the preceding experiments to compare the performance of the most optimal models. The comparison is done using F05 scores, a consistent metric throughout the validation experiments. Additionally, ROC curves and Precision-Recall curves are leveraged to give a comprehensive understanding of the algorithms' performances on unseen data. Moreover, the evaluation also explores a reduced subset of data to assess if comparable results are achievable with a smaller volume of data. The evaluation also ensembles all the models trained on the full dataset to test the real-world applicability of these models in the construction of an investment strategy.

Chapter 5

Imputation and Feature Selection

5.1 Imputing Missing Data

The dataset that has been constructed has many N/A values, so it is worth-while to find an imputation method that is both accurate and computationally efficient. Due to the high dimensionality of the dataset, listwise deletion is not a viable option as too much important information will be lost.

Simple Imputer Comparison

The simple imputers chosen for comparison include:

- Mean imputer
- Median imputer
- Rolling mean imputer by group
- Rolling median imputer by group
- Forward fill imputer by group (negligible amount of N/As at beginning of database are backward filled by group)

The 'group' element in some of the imputers refers to data being grouped into its RIC-Code before implementing any imputation. The experiment was conducted on a single year of the database 2017-2018. Four features were randomly selected from the database, with any N/A rows being removed, as this generates a complete database to try and predict. Then on the complete data, N/As were randomly added to the database at

different percentages 10%, 20% and 50%. The tables below display the performance of each imputer on each of the features according to R^2. R^2 evaluates linear regression functions (perfect is 1.0) and describes the variation of the predicted target values and the true values.

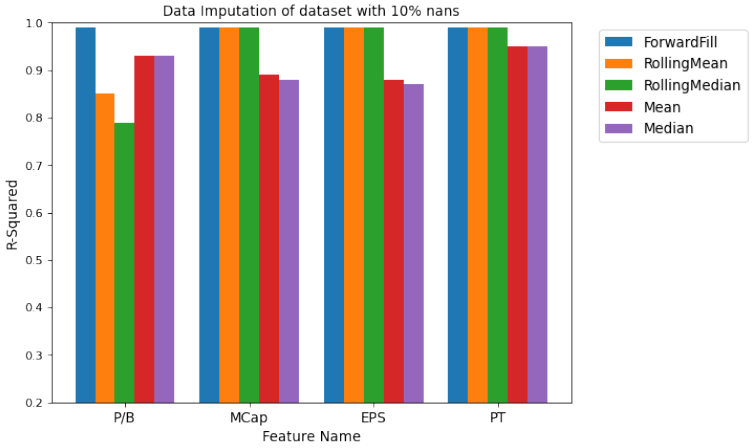


FIGURE 5.1: Data Imputation on dataset with 10% nans

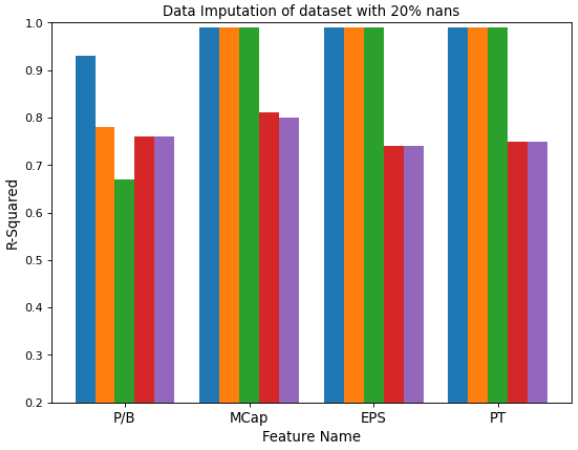


FIGURE 5.2: Data Imputation on dataset with 20% nans

Discussion on Simple Imputers

The performance of the simple imputers is evaluated thoroughly using four randomly selected features and measuring their performance through the R^2 metric. It is clear through this investigation that the simple imputer with the highest performance is 'FFill' by group. Achieving above 0.9 R^2 on all features with any percentage of N/As that it was tested on. 'RollingMean' and 'RollingMedian' by group both replaced N/As accurately. However, the imputation of the feature 'Price To Book' showed a significant difference between 'FFill' and the rolling imputations which got wider as the N/A percentage increased. The gap was most prominent in the dataset with 50% N/As as every method

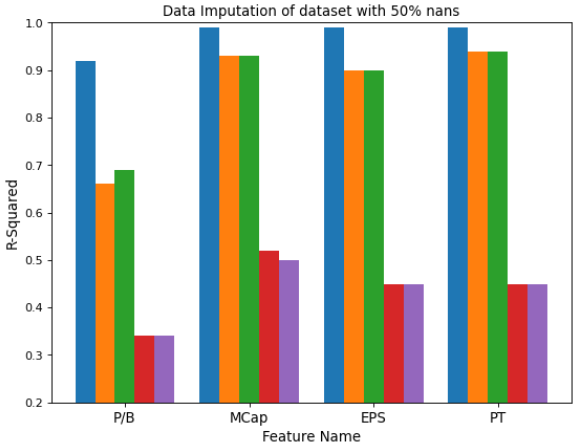


FIGURE 5.3: Data Imputation on dataset with 50% nans

was inferior to 'FFill' on all features. 'Mean' and 'Median' also fell significantly behind on this dataset.

Multiple Imputer Comparison

The multivariate imputers include:

- K-Nearest Neighbours Imputer
- MICE
- MissForest Imputer

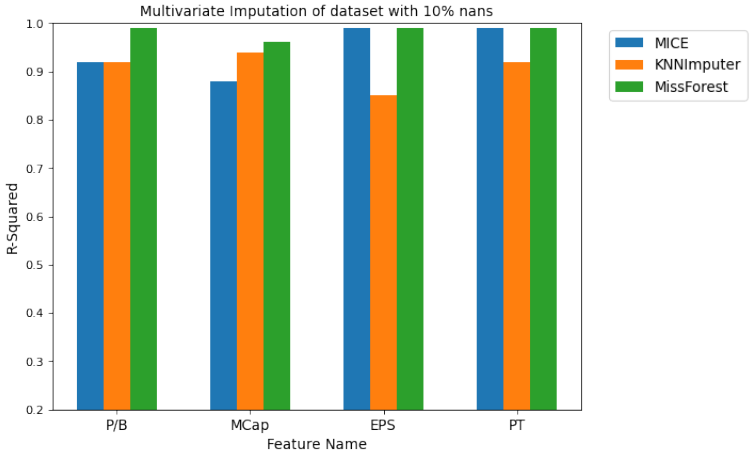


FIGURE 5.4: Data Imputation on dataset with 10% nans

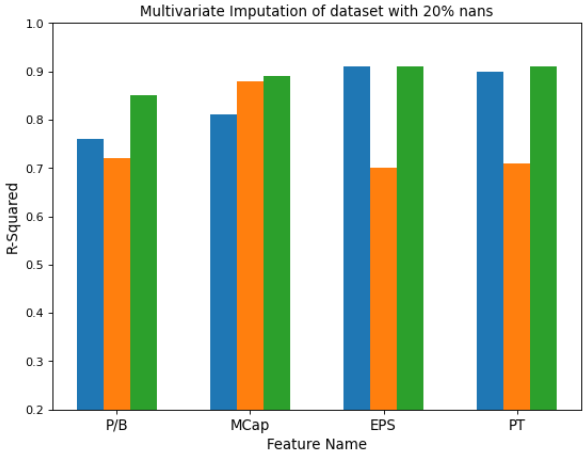


FIGURE 5.5: Data Imputation on dataset with 20% nans

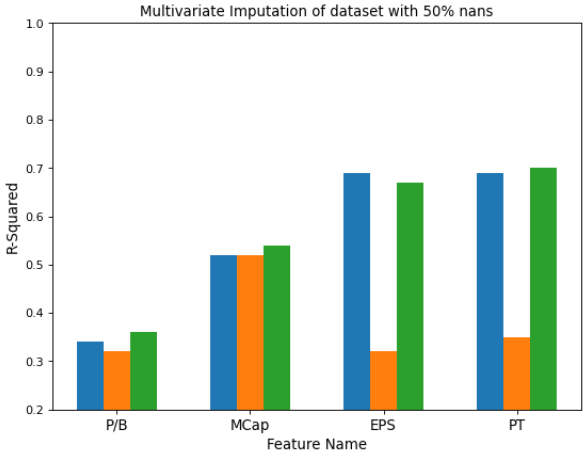


FIGURE 5.6: Data Imputation on dataset with 50% nans

Discussion on Multivariate Imputers

The highest performing multivariate imputer was the MissForest algorithm. This is essentially a random forest regressor that predicts the missing values of a feature using all other features it has available. Although the performance of MissForest and MICE was very similar, MissForest produced a more accurate imputed dataset in all the test cases. The KNNImputer performed relatively well when the percentage of missingness was 10% and 20%, however in the 50% test case it was found to be significantly worse compared to MICE and MissForest.

Conclusion

The simple imputers using the 'groupby' methodology (FFill, RollingMean, RollingMedian) had the best attempt at recreating the original dataset. However, these methodologies rely on the company with missing feature data to have past records of the same features. In such cases MissForest will be used to impute whatever data the 'groupby'

methods could not, as it outperformed mean, median and all other multivariate techniques tested.

5.2 Feature Selection

This experiment is conducted with the aim of reducing the dimensionality of the dataset, without sacrificing performance. The experiment includes dropping features/columns that produced low feature importance across a multitude of models. Reducing the feature vector subset can reduce overfitting, improve performance and reduce the computational cost of running algorithms. This is an approach for dealing with 'the curse of dimensionality' [41], this describes the phenomena that occurs when adding more dimensions to a feature space whereby the number of samples required increases exponentially in relation to the number of dimensions.

Five-fold walk forward cross validation was used in conjunction with Hyperopt Bayesian optimisation library to find which dataset produced the best validation performance on key metrics for imbalanced datasets. The number of M&As in the dataset is very low and thus the training folds all included 7+ years of data to give the algorithms more data to identify trends that lead to a company being a M&A.

Reducing dimensionality

This experiment compares the performance of the focus algorithms on the full dataset, and subsequently smaller versions (in terms of dimensions) of the dataset. The original database has around 200 features (Figure A.5), its performance is measured against three smaller versions of the database which are 75%, 50% and 25% of the original 200 features.

Tree-based methods were used for feature selection due to their computational efficiency and recorded performance in similar imbalanced problems. In fact, PayPal [76] incorporates these methods when trying to detect fraudulent transactions which is an inherently imbalanced problem.

Dimension reduction was conducted by running 50 runs of LightGBM and random forest separately on the full dataset and choosing the top three models in terms of F05 score for both. Then the average feature importance across the six chosen models was used to cull the number of features in the dataset. Features with the lowest average feature importance were dropped at different thresholds, firstly dropping 25% of features then

50% and finally 75% of the original features. The analysis is conducted on each of the algorithms individually to find its preferred feature set.

F05 The F-score metric is a weighted harmonic mean of the imbalanced specific metrics precision and recall. The standard F-score metric is the F1-score which is an exact mean of precision and recall. For this study, F05 has been deemed more appropriate as it assigns double the weight to precision compared to recall. This is due the fact that it would be better to detect less M&A events if the ratio of correct positive predictions is higher (precision).

Logistic Regression Logistic regression is the baseline algorithm and has been the subject of many papers investigating its efficiency on M&A events [12, 25, 63].

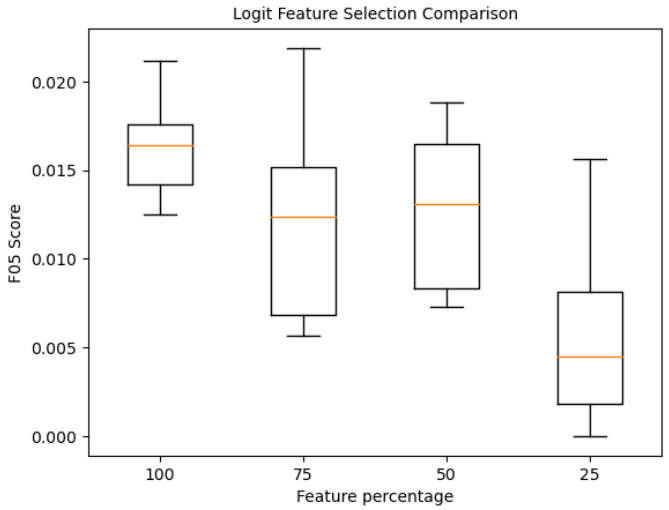


FIGURE 5.7: F05 Validation Results

The best feature percentage for logistic regression is the full dataset which produces the best max, min, median and overall spread of data (Figure 5.7). The performance of the algorithm remains good across the different feature percentages, with a significant dropoff only observed in the lowest feature percentage subset. The spread of performance shown by the interquartile range across the logistic regression models demonstrates the consistency of the algorithm. Logistic regression is useful for comparison purposes against the state-of-the-art algorithms such as recurrent neural nets and gradient boosted trees due to its prevalence in this particular field.

Random Forest The first algorithm tested was the random forest, where 50 trials were run on generally good hyperparameters.

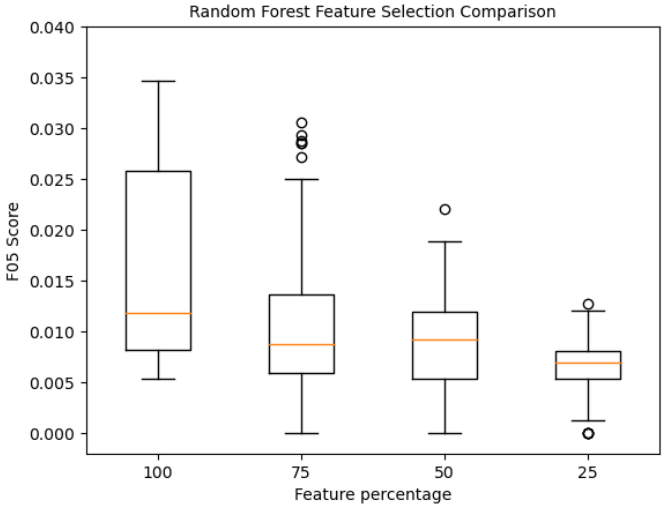


FIGURE 5.8: F05 Validation Results

The box and whisker clearly shows that the full dataset produces the best validation results for the random forest classifier (Figure 5.8), with it producing the highest median, maximum and minimum F05 score of all the feature percentages over 50 trials. The performance of the random forest trends downwards the fewer features it is given. Only a few outlier models from the 75% feature subset compete with the top 25 models trained on the full dataset. This observation is not surprising as random forests inherently train on a subset of the features with a subsample of the training data for each tree and thus do not usually require feature pruning [78, 79]. They are extremely robust models and thus providing more features should be better in theory, as long as the features are not highly correlated. The dataset using only 25% of the original features demonstrates the problem with too few features. With fewer features available for the random forest to subsample, the distribution becomes more constrained. This restriction limits the amount of information the model can draw from, increasing the likelihood of underfitting.

LightGBM Microsoft's LightGBM was the second algorithm to be tested on different feature percentages.

Similar to the random forest the LightGBM Classifier performs best when it is given the full feature set as it has the highest maximum, minimum and median value. The full dataset's top 25 models were all higher than the best trial in each of the other three datasets, thus it has quite a significant performance benefit compared to the models trained on lesser features. The trials on 75% of the data maintain a relatively strong performance, but when the features are reduced to 50% and 25% the result of an average

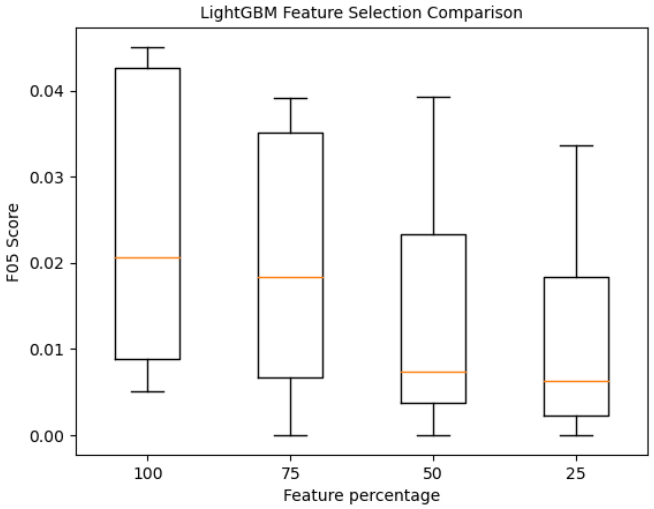


FIGURE 5.9: F05 Validation Results

trial tends to be worse. To show this we compare the results of the feature subset of 75% to the feature subset of 25%, 75's median to upper quartile (models ranked 26-50) produce a similar performance to 25's models between the upper quartile and the maximum. The models created on 50% of the data still demonstrate the ability to outperform models that have access to more features. This is most notable where the maximum of the 50% feature subset models has a higher F05-score than the best model from the 75% feature subset.

LightGBM has two important parameters which help it create weak learners to later be ensembled. The parameters include 'subsample' which is the fraction of samples provided in the input to train each tree and 'colsample_by_tree' which is the fraction of features to be used to train each tree [40]. In other words, it is robust to 'weak features' and often will find more performance when additional useful features are added to the database.

LSTM A LSTM is a deep learning method which takes into account the time-series aspect of the data. It has been historically good at language recognition and forecasting time series data [80]. The long-short term memory neural network can process sequences of time-series data, giving it the ability to find key patterns that occur through time and avoids the vanishing gradient problem associated with standard recurrent neural networks.

The LSTM models seem to prefer models with fewer features, this is likely because smaller datasets allow the neural network to converge faster. This is the point where model training has reached a minimum error rate on the training set, additional training will not benefit the model. The best performing feature set is the 50% subset which has the best minimum, median and the top 25 models (Q3 to maximum) of any of the feature

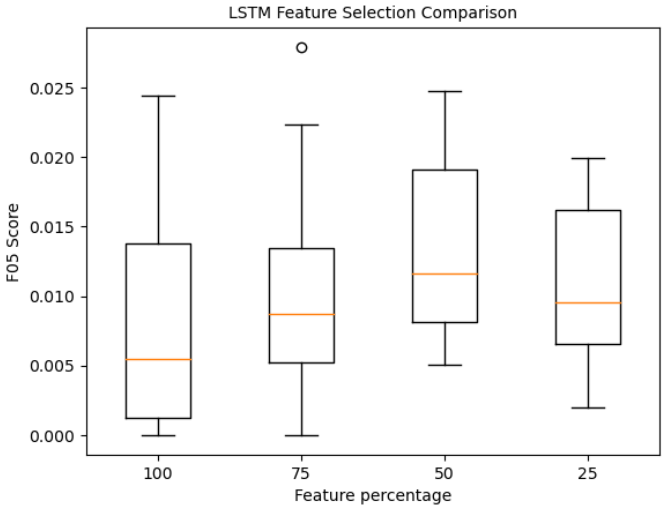


FIGURE 5.10: F05 Validation Results

sets. There is a case to be made that the higher feature set could perform better if the computation time was not an issue, however, to complete 50 trials of each feature set many models did not converge. The LSTM uses input normalisation (Min-Max scaler) to help the neural network converge faster. The LSTM uses 'tanh' as its activation function for its hidden layer as this is the default for LSTMs. The 'tanh' activation function is slower than the popular 'ReLU' function but is considered more appropriate for the LSTM framework [81] as 'ReLU' can produce large varying outputs causing it to diverge.

TabTransformer The TabTransformer is a tabular based deep learning model which uses the attention architecture to make categorical variables more useful by increasing their expressive power in a structured dataset.

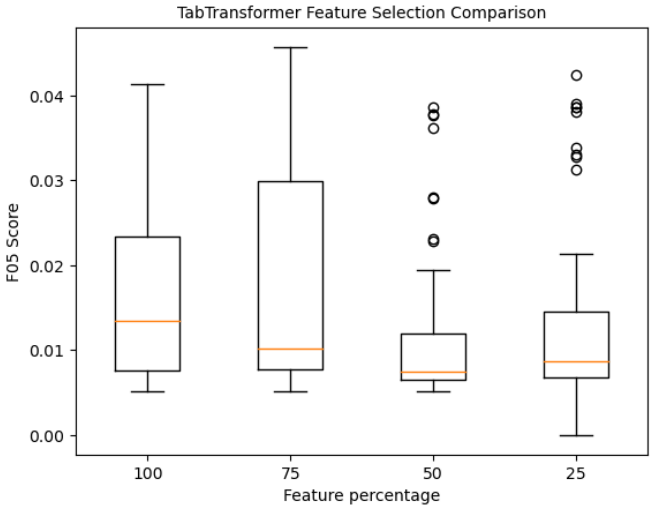


FIGURE 5.11: F05 Validation Results

The results in Figure 5.11 show the TabTransformer algorithm had its highest performing models (Q3 to maximum) when using 75% of features. This was followed closely in performance by the models using all of the features. There is not a significant difference in the performance of these two feature spaces, however it could be speculated that the models using 100% of the features could be using too many features (noise) leading to overfitting. Conversely, the models using only 50% and 25% of the features led to the loss of important information thus resulting in poorer performance in terms of F05 score. It is worth noting that outliers in the 50% and 25% feature percentage groups performed well, meaning that certain hyperparameter combinations can perform well even with a lesser feature set.

5.3 Summary

The validation results from the focus algorithms have produced some noteworthy conclusions. The tree-based models: random forest classifier and LightGBM classifier both produce their best results when given the full feature set to work with as they are robust in eliminating noise from the dataset and are best left to choose their own feature subsets. The deep neural nets: LSTM and TabTransformer tend to perform better when feature selection has removed 'irrelevant features' (noise) which allows the algorithm to converge faster, and is less likely to overfit [82]. Overall deep neural networks (DNNs) have more complex architectures compared to tree-based models and are more likely to overfit to noisy data. Feature reduction helps reduce the 'noisiness' making it easier for the DNNs to learn the important patterns within the dataset. Tree-based models show great performance when handling data with a high dimensionality as they split on the most important features.

Chapter 6

Hyperparameter Tuning

In this chapter the individual hyperparameters of each algorithm will be analysed and optimised to produce the greatest validation performance which should, in theory help the holdout performance in the final models. These hyperparameters control the model's function and have a big effect on performance. Hyperparameter tuning becomes even more relevant in the event of a significant class imbalance, when the number of samples in one class vastly outweighs the number of samples of the other class. An imbalanced dataset can present several difficulties when training the model, including bias in favour of the more common class and poor predictive accuracy on the minority class. Therefore, to overcome these challenges and optimise the models careful selection and tuning of hyperparameters is required.

6.1 Tuning Process

The Hyperopt package has been shown to be the most efficient technique in terms of accuracy and computation time when compared to standard techniques such as Grid search and Random search [83]. The major advantage of using a Bayesian optimisation (informed search) method like Hyperopt is that it improves from previous trials. This is often a less tedious and time-consuming technique compared to standard techniques.

In the previous feature selection experiment, Hyperopt was used to run through a space of hyperparameters for every algorithm with smaller and smaller feature subsets. The process can be summarised as follows:

- Create four datasets from the original dataset containing 100%, 75%, 50% and 25% of features.

- Define a parameter space for machine learning algorithm of choice.
- On each of the four datasets run 50 trials of 5-fold cross validation using the Hyperopt package which uses Bayesian Optimisation to find the most optimal parameters within the parameter space.

The trials from the best performing feature subset of each algorithm are analysed in this section to propose optimal parameters.

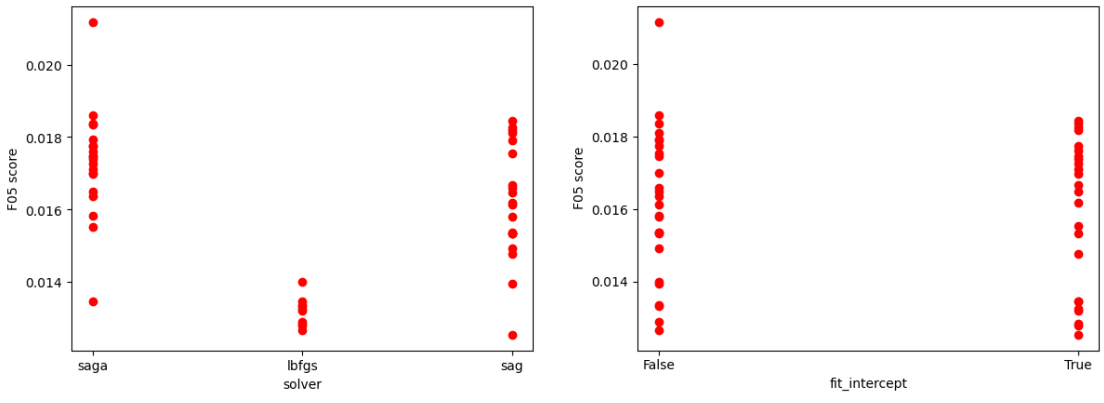
6.2 Algorithms

Logistic Regression

The hyperparameters adjusted for logistic regression are as follows:

solver	'sag', 'saga' and 'lbfgs'
C	Range between 0.05 and 3
tol	Range between 0.01 and 0.1
max_iter	Range between 100 and 10000
fit_intercept	True or False

TABLE 6.1: Logit Hyperopt Space



(A) solver parameter (B) fit_intercept parameter

FIGURE 6.1: (A) SAGA and SAG solvers produce better performance compared to L-BFGS (B) The 'fit_intercept' bias does not seem to impact performance

Each point in the scatter plots Figure 6.1a and Figure 6.1b correspond to one of 50 runs of hyperparameter tuning, where the model's performance is evaluated using F05 score. These parameters are analysed independently due to their distinct nature, which allows

clearer understanding of the individual impact of each parameter on the model's performance. The only hyperparameters that changed the performance of logistic regression noticeably included the solver and 'fit_intercept' parameters. The solver (optimizer) parameter is much better using 'sag' or 'saga' compared to the default solver 'lbfgs', Figure 6.1a. The stochastic gradient descent (SGD) solvers 'sag' and 'saga' are both effective for large datasets[84, 85]. On such datasets, it has been demonstrated that these solvers converge more quickly and are more resilient to improper initialisation compared to 'lbfgs'. The two methods both incorporate regularisation, 'sag' employs a recollection of previous gradients to enhance the optimisation, 'saga' uses an adaptive step size that modifies the learning rate based on gradient information. Both these methods lessen the risk of overfitting and increase the model's generalisability. The computation of the Hessian matrix is necessary for the second-order optimisation algorithm 'lbfgs', this is often computationally expensive for big datasets. However, 'sag' and 'saga' simply need to compute the gradient, which is far less expensive in terms of compute power.

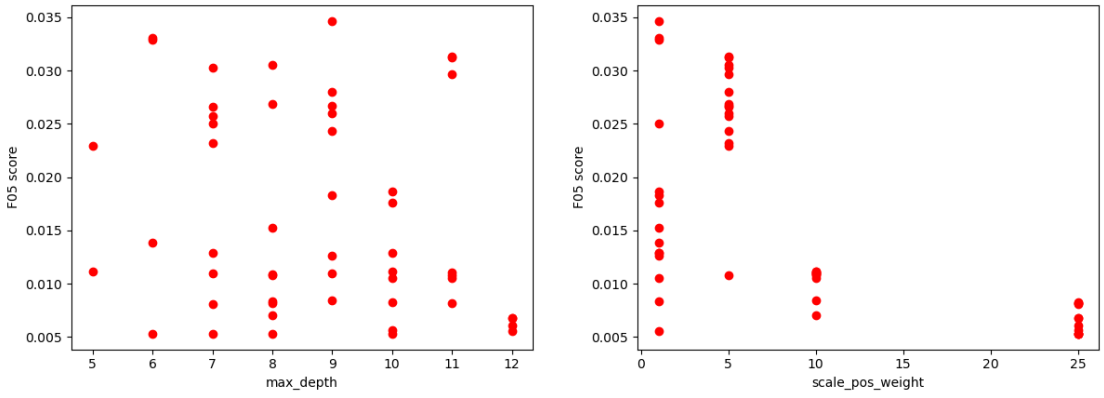
The fit_intercept parameter (Figure 6.1b) adds a bias to the decision function. The scatter plot does not show a clear advantage in choosing to use a bias for the logit model.

Random Forest

The hyperparameters adjusted for random forest are as follows:

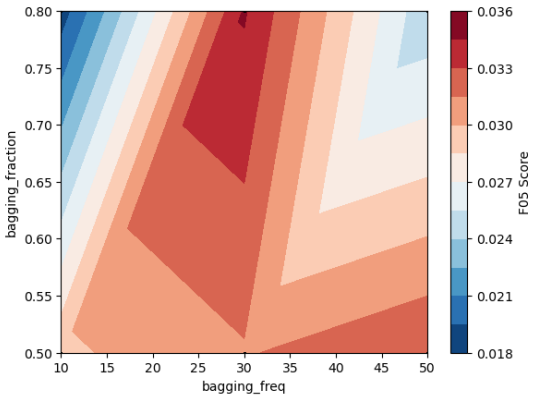
max_depth	Range between 5 and 12
bagging_freq	Range between 10 and 50
bagging_fraction	Choice of 0.5 or 0.8
reg_alpha	Choice of 0.0001, 0.1, 1 or 10
n_estimators	50
scale_pos_weight	Choice of 1, 5, 10 or 25

Table 6.2: Random forest Hyperopt Space



(A) max_depth parameter

(B) scale_pos_weight parameter



(C) bagging_freq and bagging_fraction

FIGURE 6.2: (A) max_depth parameter seems to favour the 7-9 range (B) scale_pos_weight is significantly better when set to 5 (C) Contour plot showing the 'sweet spot' for bagging_freq and bagging_fraction parameters

The most obvious performance enhancing parameter is scale_pos_weight which is a parameter that determines how important the minority class is in comparison to the majority. When the target variable is unbalanced, the 'scale_pos_weight' parameter is used to balance the class distribution in the training set [86]. Setting the weight to 5 produces the best F05-score (see Figure 6.2b), with a value of 5 the random forest treats one instance of the positive class as equivalent to five instances of the negative class during the model training process. When the positive class is relatively uncommon and the model is forced to train on a limited subset of positive samples then up-weighting the positive class can be effective. Thus up-weighting the positive instances helps the model discriminate better between positive and negative instances. When the 'scale_pos_weight' is set to higher values such as 10 or 25 the training set may be geared too much towards the positive class, which results in overfitting and poor performance on the validation set. In the case where 'scale_pos_weight' is set to 1, the positive and negative samples are treated equally. This often causes a machine learning model to struggle with differentiating positive and negative samples and thus exhibiting a bias in favour of the

dominant class. Essentially the models tend to not generalise well when the positive samples in the training set have significantly different characteristics to the positive samples in the validation set. The F05-score is a metric derived from precision and recall, upweighting the positive class often coincides with an increase in the recall metric. Thus 'scale_pos_weight' = 5 helps to increase the recall of the model without sacrificing too much of the precision of the model.

The Bayesian optimizer favours the 7-9 range of max_depth (Figure 6.2a) as it produces most of the models with a high F05-score. The random forest algorithm is an ensemble of decision trees whose individual predictions are combined to create a final output. The maximum depth parameter determines the number of splits that each decision tree can undergo before encountering the leaf node (deepest node). A deeper tree can often capture more intricate connections between feature vectors and the binary target. It can also cause overfitting, in which the model too closely resembles the training set and thus does not generalise to unseen data. The maximum depth between 7-9 is a good compromise between over-training and the complexity of the random forest model [87, 88].

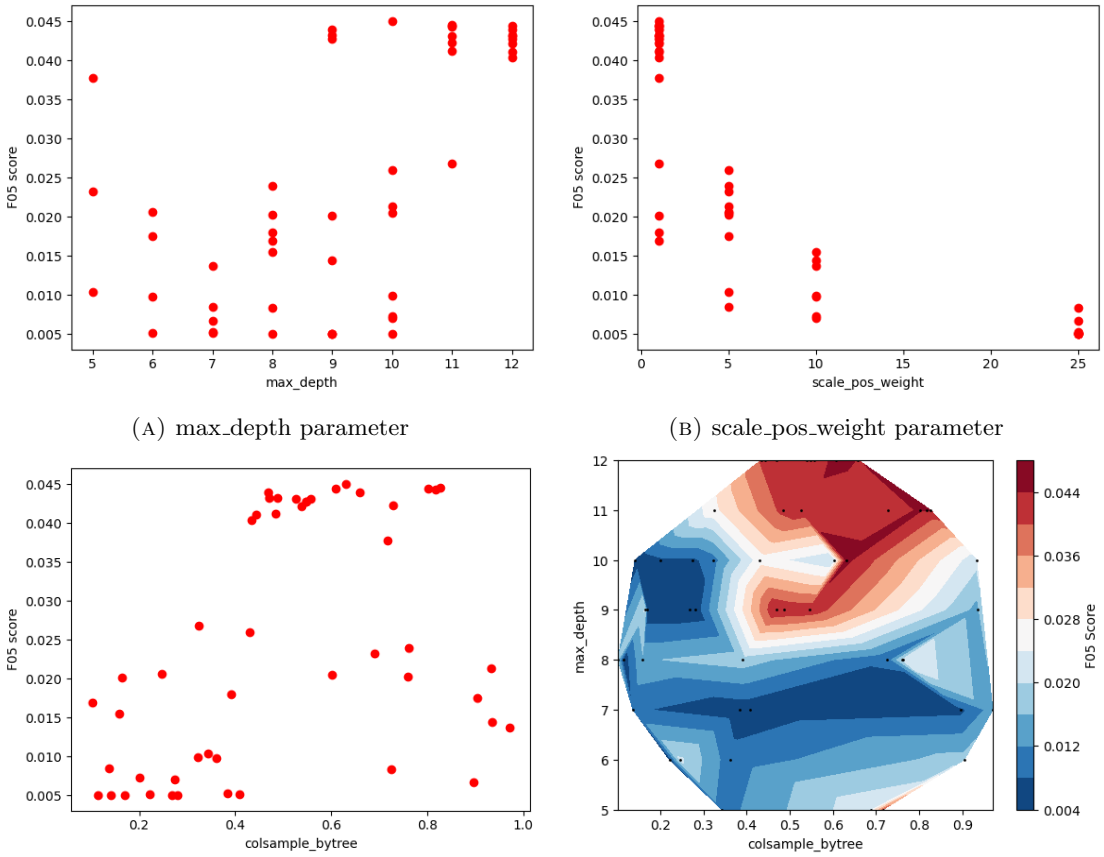
Figure 6.2c presents a contour plot that simultaneously evaluates the interaction between the bagging frequency and bagging fraction parameters in a random forest model. Unlike the previous scatter plots, this two-dimensional contour plot is necessary to illustrate the co-dependence of these two parameters. Bagging frequency and bagging fraction both fall under the bootstrap aggregating technique used in specific ensemble machine learning models. Bootstrap aggregating trains many models on various subsets of the training set and then combines their individual predictions to create a final prediction. Bagging frequency regulates how frequently the bagging technique is applied to the data, and bagging fraction regulates the percentage of the data used in each bagging phase. The contour plot shows the optimal range for the random forest model when referring to bagging frequency and bagging fraction is between 25-35 and 0.65-0.8 respectively, see Figure 6.2c. A bagging frequency that is too high leads to overtrained models [89], a value between 25 and 35 indicates that the bagging process is being applied somewhat frequently. A bagging fraction value between 0.65-0.8 denotes that a sizable chunk of the data is selected when bagging. Within these two ranges there is a balance in the diversity of the trees and the computational complexity of the model. At the extremes of the contour plot the bagging frequency of 40-45 might be too high, causing the trees to be less diverse due to the random forest being limited to a certain set of features. When this is coupled with high bagging fraction it worsens the overfitting issue [90] resulting in subpar performance on the validation set.

LightGBM

The hyperparameters adjusted for LightGBM are as follows:

max_depth	Range between 5 and 12
colsample_bytree	Range between 0 and 1
scale_pos_weight	Choice of 1, 5, 10 or 25
reg_alpha	Choice of 0.0001, 0.1, 1 or 10
n_estimators	50

TABLE 6.3: LightGBM Hyperopt Space



(A) max_depth parameter

(B) scale_pos_weight parameter

(C) colsample_bytree

(D) colsample_bytree and max_depth

FIGURE 6.3: LGBM hyperparameter scatter plots

Max_depth , scale_pos_weight and colsample_bytree are the parameters that significantly impact performance of the LGBM models. The LightGBM models improve significantly with a deeper tree, with maximum depths of 11 and 12 (see Figure 6.3a), this is consistent with the findings in other studies [87, 88]. The relatively deep decision trees can help identify more intricate patterns in large datasets which seems to pay dividends in the

study. With a maximum depth of 11 or 12 the model builds reasonably deep trees without overfitting.

Scale_pos_weight does best when left on the default value of 1, with all the models with an F05-score above 0.035 corroborating this theory. This is unexpected as a higher 'scale_pos_weight' shifts the focus to the positive class (minority). In some cases it can lead to overfitting as it makes the model too sensitive to the minority class leading to sub-optimal performance.

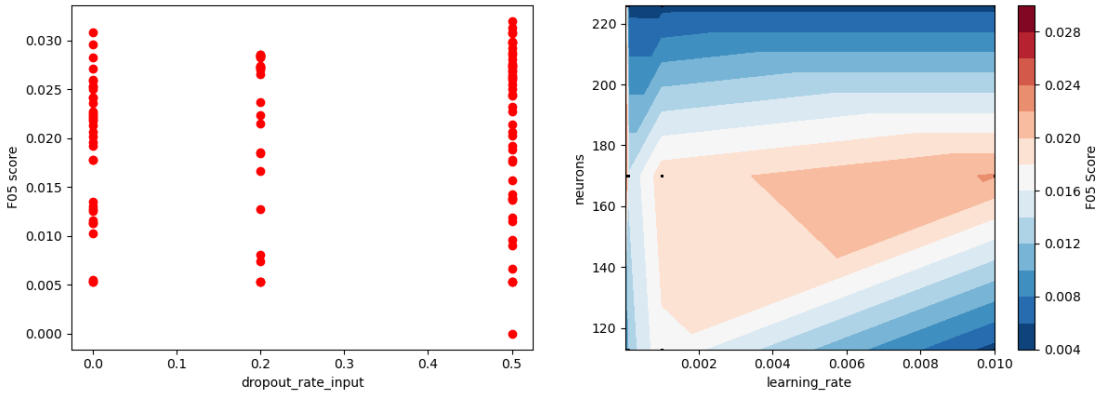
Figure 6.3d presents a contour plot that combines the two parameters max_depth and colsample_bytree offering a two-dimensional view of their interaction. This allows a visualisation of the optimal regions where both parameters work together to achieve the best F05 scores. This gives a more comprehensive view of optimal parameter combinations compared to one dimensional plots. The contour plot (Figure 6.3d) shows the optimal range for the LGBM models fall between 0.4 and 0.85 for colsample_bytree and a maximum depth between 9-12. Colsample_bytree is the fraction of features used to build each individual tree, this ideal range combats underfitting and overfitting with a single hyperparameter [91]. A value between 0.4 and 0.85 means that a significant fraction of the columns are being taken into account, this often helps capture more information from the data. It is important to note that this means the model is not using all features for each tree which may help prevent overfitting. A very low colsample_bytree (less than 0.4) also causes the models to perform poorly. Meanwhile the maximum depth parameter residing between 9-12 is able to capture complex relationships between feature vectors and the target without overfitting or having to high a model complexity [87, 88]. On the other hand, any model considered shallow (max_depth below 9) shows a significant drop-off in performance no matter the colsample_by_tree value used. The shallow LGBM can restrict the model's capacity to learn intricate and nonlinear correlations between the feature vectors and the target, this can often result in an underfit model.

LSTM

The hyperparameters adjusted for LSTM are as follows:

learning_rate	Range between 0.01 and 0.00001
dropout_rate_input	Choice of 0, 0.2 and 0.5
neurons (first layer)	Range between 113 and 226
neurons (second layer)	Half the neurons as the first layer

Table 6.4: LSTM Hyperopt Space



(A) dropout_rate parameter (B) learning_rate and neurons

FIGURE 6.4: LSTM hyperparameter scatter plots

The dropout_rate parameter (Figure 6.4a) has slightly better performance when set to 0.5 compared to the lower values. The dropout_rate regulates the percentage of neurons that are 'switched off' at random during each training phase. Half of the neurons in the LSTM model are turned off during each training step when the dropout rate is 0.5. This reduces the complexity of the model and enables it to learn more robust feature vectors, which can help minimise overfitting. Due to their complex design and numerous parameters, LSTM models are prone to overfitting, especially when working with big sequences. A dropout rate of 0.5 aids in regularising the model to a large extent and thus preventing overfitting.

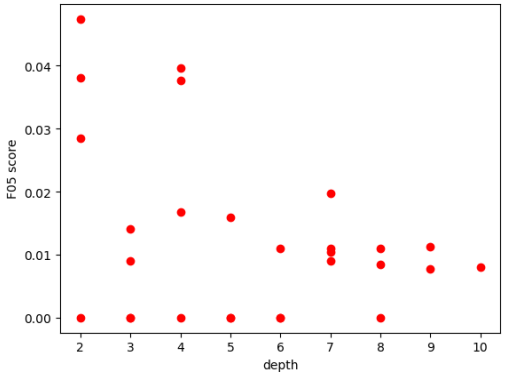
Figure 6.4b is a contour plot that combines both learning rate and neurons in a two-dimensional view. This plot helps to comprehend their interdependent effects on the LSTM model performance. The LSTM produces the best results when selecting a model between 140 to 180 neurons for the first hidden layer and between 70 and 90 neurons for the second hidden layer (half of the first). The low-level features are extracted by the first hidden layer. The model may learn from a wide range of features due to the relatively high number of neurons in this layer. The second hidden layer combines the features extracted from the first hidden layer to form more complex representations of the data. Using half the neurons of the first hidden layer will reduce model complexity and thus combat overfitting further. The learning rate determines how frequently the optimisation algorithm updates the model parameters. A learning rate between 0.004 to 0.01 represents a relatively small step size, allowing the optimiser to converge on a solution without overtraining.

TabTransformer

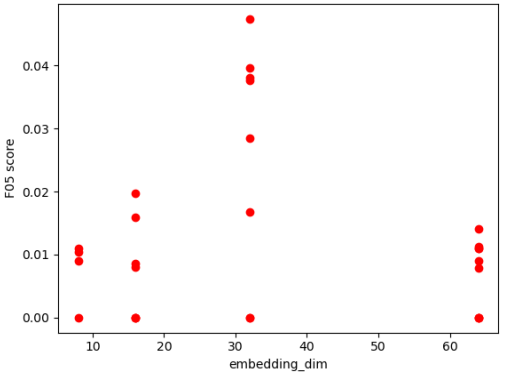
The hyperparameters adjusted for TabTransformer are as follows:

learning_rate	Range between 0.01 and 0.00001
embedding_dim	Choice of 8, 16, 32 or 64
depth	Range between 2 and 10
heads	Range between 4 and 16
attn_dropout	Range between 0 and 0.5
ff_dropout	Range between 0 and 0.5

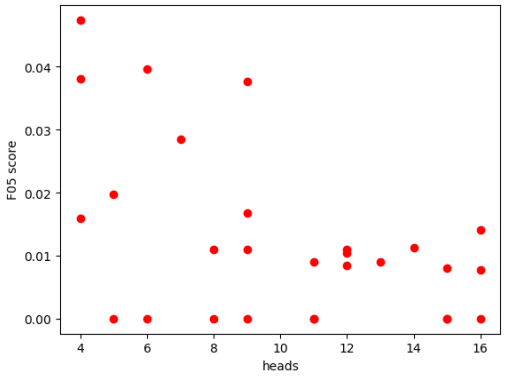
TABLE 6.5: TabTransformer Hyperopt Space



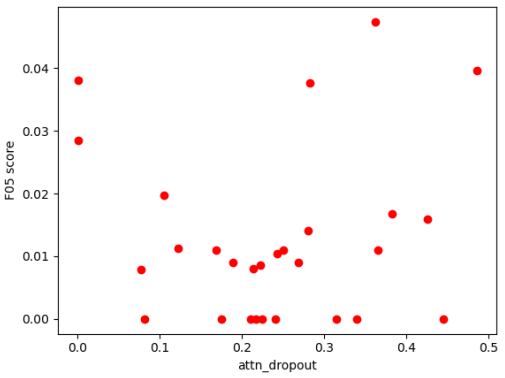
(A) depth parameter



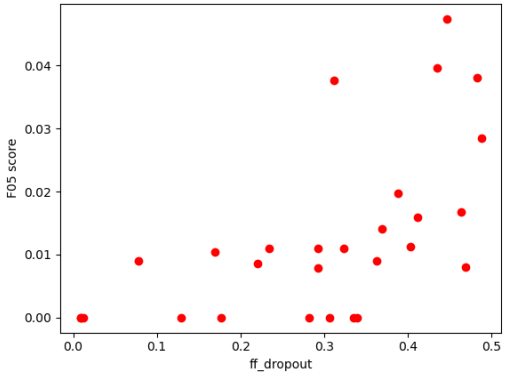
(B) embedding_dimension parameter



(C) heads parameter



(D) attn_dropout parameter



(E) ff_dropout parameter

FIGURE 6.5: TabTransformer hyperparameter scatter plots

TabTransformer is a deep learning tabular algorithm which has a tendency to memorise the training data causing the model to overfit [92], thus a low 'depth' parameter (Figure 6.5a) is closer to optimal. The embedding_dim parameter sets the dimensionality of the feature embeddings that the model learns for every feature vector, the original paper on TabTranformers [8] uses an embedding dimension of 32. A larger embedding feature increases the capacity of the model to learn meaningful representations of input feature vectors but can also increase the risk of overfitting. The converse is true for a model with a low embedding dimension where the models have a reduced capacity to learn complex relationships and can thus result in an underfit model. In this case the standard embedding dimension of 32 is the best for the TabTransformer (Figure 6.5b).

The number of transformer heads (Figure 6.5c) controls the degree of parallelism in the self-attention mechanism and can affect the capacity of the model and its ability to learn complex relationships between inputs. A larger number of heads can increase the model's capacity to capture fine-grained patterns, but can also increase the risk of overfitting. A smaller number of heads can reduce the model's capacity but may also limit its ability to capture complex patterns. All the superior models have the number of heads lying between 4-9 transformer heads.

The dropout rate for attention (attn_dropout) and feed-forward (ff_dropout) layers in TabTransformer can affect the model's performance and its ability to generalize to new data. Dropout is a regularization technique that randomly drops out some activations during training to prevent overfitting.

Setting a higher dropout rate above 0.3 (Figure 6.5d, 6.5e) can increase the regularization strength and improve the model's generalization ability, but it can also reduce the model's capacity to learn complex patterns. Conversely, setting a lower dropout rate between 0 and 0.3 can increase the model's capacity but may also increase the risk of overfitting.

6.3 Summary

This chapter provides an analysis of the hyperparameter tuning that each algorithm underwent to find the most optimal models for each algorithm. The beginning of the chapter explains the process of how the hyperparameter tuning is conducted. Then each of the focus algorithms and their hyperparameter tuning space is presented, followed by scatter plots and contour plots which show where the optimal parameters lie.

Classifier Algorithm	**Ideal Parameters**
Logistic Regression	max_iter=5736, fit_intercept='False', tol=0.086, C=1.5, solver='saga'
Random Forest	n_estimators=50, max_depth=9, num_leaves=31, reg_alpha=0.0001, bagging_fraction=0.8, bagging_freq=30, scale_pos_weight=5
LightGBM	n_estimators=50, max_depth=12, num_leaves=40, learning_rate=0.1, colsample_bytree=0.47, scale_pos_weight=1
LSTM *both LSTM layers have the same dropout and recurrent dropout, the second layer has half the units of the first (90 units)	number of layers=3, units=180, dropout=0.5, recurrent_dropout=0.2, sequence_length=13, optimiser='adam', learning_rate=0.01, epochs=5, batch_size=128
TabTransformer	embedding_dim=32, depth=2, heads=4, attn_dropout=0.35, ff_dropout=0.45, mlp_hidden_factors=[4, 8], optimiser='adamw', learning_rate=0.007, weight_decay=0.006, num_epochs=50

Table 6.6: Ideal Parameters for Various Classifier Algorithms

Chapter 7

Data Sampling

The dataset is highly imbalanced with a ratio of 0.4 : 99.6, thus oversampling and undersampling techniques are tested on each of the focus algorithms to see if they improve validation performance. The dataset used in this experiment has imputed its N/A values using the best imputation methods found in the imputation experiment.

Sampling Process The sampling comparison uses the same five-fold cross validation split as done in the feature selection experiment. In each fold, the train portion is oversampled/undersampled by date to reduce bias incurred when fitting the algorithm.

Oversampling and undersampling by date is essential, in that it preserves the sequential nature of the data. This means that the data samplers cannot introduce future information into the training set, as this would result in data leakage and compromise the models. Thus, any synthetic samples or reduction in the dataset must take into consideration the order of the dataset. The LSTM model had a more tailored approach due to its sequential nature. Specifically, undersampling was performed by stock, grouping all datapoints for each stock before any data reduction was applied, oversampling was avoided due to the complexity of proper implementation for a LSTM model. This retained the temporal sequence within each stock's data. The validation portion of each fold is left the same, as then it can be used to give a realistic evaluation of the algorithm. The 'imbalanced-learn' package is used to apply well researched oversamplers: random oversampler, SMOTE and ADASYN as well as the undersamplers: random undersampler, NCR and OSS. After the sampling technique has been applied each focus algorithm is then run on a 5-fold cross validation using Bayesian optimisation for a total of 50 models. The average of the top 5 models (in terms of F05 and PRAUC) for each algorithm is used to display the effectiveness of the sampler.

The aim of this experiment is to find the best sampler to use for each algorithm and not to find the best sampler across all algorithms, thus algorithms will be analysed on an individual basis.

7.1 Oversamplers

The oversamplers to be compared include:

- Random Oversampler
- Synthetic Minority Oversampling Technique (SMOTE)
- Adaptive Synthetic (ADASYN)

As shown in Table 7.1 the comparison of sampling techniques was measured using the F05 metric and AUCPR.

Classifier	Metric	No oversampling	Random oversampling	SMOTE	ADASYN
LR	F05	0.0190	0.0184	0.0187	0.0156
RF	F05	0.0326	0.0336	0.0330	0.0311
LGBM	F05	0.0438	0.0442	0.0430	0.0402
LSTM	F05	0.0335	-	-	-
TT	F05	0.04	0.0456	0.0409	0.0402
LR	AUPRC	0.0165	0.0158	0.0159	0.0143
RF	AUPRC	0.0194	0.0245	0.0184	0.0149
LGBM	AUPRC	0.0199	0.0226	0.0221	0.0219
LSTM	AUPRC	0.0178	-	-	-
TT	AUPRC	0.0224	0.0235	0.0231	0.0211

TABLE 7.1: Average validation performance of top 10% of models for each algorithm

Logistic Regression

None of the oversampling techniques improve the logistic regression algorithm in the chosen metrics: F05 and AUPRC. Oversampling may add to the complexity of the model due to the increase in minority samples in the dataset, this can cause overfitting making it challenging to predict on unseen data.

Random Forest

The random forest seems to reap some benefit from the application of oversamplers, with the only exception being the ADASYN oversampler. ADASYN is a more sophisticated method of oversampling which generates synthetic samples by adapting the density

distribution of samples near the class boundary. It may be less effective when the number of minority samples is extremely small, making the generated synthetic samples unrealistic and thus providing no new information for the model to capture.

The simpler oversampling methods random oversampling and SMOTE can be more effective when the imbalance is extreme. Random oversampling duplicates existing samples from the minority class whilst SMOTE creates samples using interpolation of minority samples. These simpler methods might produce more realistic samples that are closer or the same as 'real' minority class examples. The downside of ADASYN is it can create samples that quite different from the original minority examples.

LightGBM

The LightGBM has similar traits to the random forest as they are both tree-based models that are built by ensembling many decision trees. The ADASYN oversampler is significantly worse off in terms of F05 score for the LightGBM this is likely for the same reason as in the random forest. The better methods were not oversampling or random oversampling where neither emerged as superior methods in terms of F05 score. It is noteworthy that the random oversampler method led to an increase in the AUPRC metric.

LSTM

In the context of financial time series forecasting, particularly when applying a LSTM, the integrity and sequential nature of the dataset is crucial. LSTMs are designed to find underlying temporal dependencies, and are thus sensitive to the inherent sequence within the dataset. Implementing oversamplers such as SMOTE, random oversampling and ADASYN is risky as it creates non-existent stocks which may not appear in enough timestamps to be useful in training. Therefore no oversamplers were not used in conjunction with the LSTM, due to the complexity involved in proper implementation.

TabTransformer

TabTransformer achieves the best results when utilising random oversampling followed by SMOTE and ADASYN, respectively. The TabTransformer models that used no oversampling also produced a relatively good F05 score, but it was lower than the oversampled models. The random oversampling technique fared the best, which may suggest that the duplicate samples are more representative of real-world data and are more beneficial than the synthetic samples generated by SMOTE or ADASYN.

7.2 Undersamplers

The undersamplers compared include:

- Random Undersampler
- Nearest Cleaning Rule (NCR)
- One-Sided Selection (OSS)

As shown in Table 7.2 the comparison of sampling techniques was measured using the F05 metric and AUCPR.

Classifier	Metric	No undersampling	Random undersampling	NCR	OSS
LR	F05	0.0190	0.0339	0.0292	0.0345
RF	F05	0.0326	0.035	0.0309	0.0339
LGBM	F05	0.0438	0.0325	0.0312	0.0367
LSTM	F05	0.0335	0.0311	0.0435	0.0395
TT	F05	0.04	0.0388	0.0475	0.0437
LR	AUPRC	0.0165	0.0164	0.0161	0.0153
RF	AUPRC	0.0194	0.0203	0.019	0.0196
LGBM	AUPRC	0.0199	0.021	0.0203	0.0203
LSTM	AUPRC	0.0177	0.0173	0.0176	0.0181
TT	AUPRC	0.0224	0.0226	0.0229	0.0219

TABLE 7.2: Average validation performance of top 10% of models for each undersampler-algorithm combination

Logistic Regression

In contrast to the oversampling methods, the undersamplers resulted in a substantial increase in F05 score. The disparity in AUPRC metric between the undersampling techniques and no sampling was negligible. The substantial increase in F05 score may be due to the large reduction in model complexity by having it train on far fewer samples. Logistic regression is a relatively simple model in the field of machine learning, and therefore it finds significant benefit in the noise reduction in the training data.

Random Forest

There is a small increase in F05 score when using random undersampling and OSS. The balancing of the data distribution likely enhances the feature importance estimation. The random forest model then assigns a higher feature importance to features that are important to both the majority and minority class, instead of exhibiting bias in favour of the majority class. Based on the observed improvement in F05 score, the

undersampling methods may help learn more defined decision boundaries often yielding better classification results [93–95].

LightGBM

Undersampling does not improve the F05 score of the LightGBM models. This could be attributed to the loss of a large number of samples, leading to a potential loss of information. This loss of samples may adversely impact the boosting process [40], where decision trees are built sequentially on the residuals of the previous decision tree. The reduction in training data for intermediate decision trees may result in weaker intermediate learners, thereby affecting the overall performance of the boosting process. It is important to note that the LightGBM algorithm is effective as a cost-sensitive algorithm and this could be a reason that it performs so well with no undersampling [96].

LSTM

Table 7.2 indicates that the LSTM performs better with the application of NCR and OSS undersamplers, which is in line with surveys on the topic [97]. A LSTM model can become too complex due to its large number of learnable parameters and thus bias towards the majority class. The implementation of these undersampling methods may enable the LSTM to identify the more important discriminate features, resulting in improved feature importance from the training process. Another advantage of the undersampled dataset in conjunction with the LSTM is that it can converge on an optimal epoch much faster as the smaller dataset can update weights more efficiently. The Nearest Cleaning Rule and One-Sided Selection are more selective in their removal of samples as these techniques try to create a more representative sample distribution of the majority class [61, 62]. This may be why these methods outperform the random undersampler in the case of the LSTM models.

TabTransformer

Similar to the LSTM the TabTransformer models improve substantially on their F05 score when using NCR or OSS to undersample the data. Overall, the use of NCR and OSS to selectively undersample the majority class likely reduces overfitting thus leading to a better performance on the validation set.

7.3 Discussion

The results of the resampling experiment provide insight into the effectiveness of different oversampling and undersampling techniques on the mergers and acquisition dataset. Random oversampling was found to be the most effective oversampling technique for

improving the performance of the random forest, LightGBM and TabTransformer algorithms. SMOTE and ADASYN were found to be less effective, with ADASYN consistently being the worst oversampling technique. In contrast, the LSTM and logistic regression models did not benefit from oversampling. Thus, oversampling is not conductive to optimal model performance for the logistic regression and LSTM models.

Undersampling techniques were observed to be effective in terms of increasing the F05 score and AUPRC for every algorithm tested except LightGBM as evidenced in Table 7.2. The optimal undersampler varied among the different algorithms. Random undersampling produced the best performance for the random forest classifier, while NCR improved the deep learning algorithms (LSTM and TabTransformer) the greatest amount. Finally, OSS was the optimal undersampler for the logistic regression models. This highlights the importance of considering the specific algorithm being used when deciding whether to use oversampling or undersampling techniques. It also shows that often different techniques need to be tried and tested to find the most effective approach necessary for a specific algorithm and dataset combination.

7.4 Summary

This chapter describes a comparative analysis that was conducted to assess the performance of different resampling methods in the forecasting of mergers and acquisitions. The emphasis was on identifying the most effective sampling-algorithm combinations in terms of F05 score and AUCPR. The findings of this experiment offer insightful suggestions for the practical application of these resampling methods in the next chapter.

Chapter 8

Results and Analyses

This chapter compares the most optimal validation models for each algorithm on the holdout dataset. The dataset is common across all the algorithms and incorporates the imputation techniques found to be most effective in the imputation experiment. The number of features used for the algorithms differs according to what produced the best F05 score for the specific algorithm in the feature selection experiment. Hyperparameter tuning and data sampling techniques were used to overcome the highly imbalanced nature of the data.

8.1 Holdout Performance of Algorithms

The thesis focused on enhancing the validation performance of the machine learning models, with the end goal of creating an optimised version of each algorithm.

In an effort to examine the impact of training set size on model performance, this study employed two distinct sets of data for holdout analysis. The first training set was the complete training dataset which spans from the start of 2000 to the end of 2017. The second set consisted of a reduced subset using the most recent four years of training data spanning from the beginning of 2014 to the end of 2017. This approach is designed to evaluate if an expansive dataset is necessary for yielding valuable results. This comparison has implications on computational efficiency, as a smaller dataset with similar results would reduce computational costs significantly. The investigation further provides insights into the relevance of temporal proximity of the training data in the context of predicting mergers and acquisitions.

Additionally, an equally weighted ensemble of all the models are also created for both datasets. It is a well-established phenomenon in the field of machine learning that the

aggregation of predictions generated by diverse models often results in a superior forecast than any of the individual models predictions [98]. The ensemble consists of an equal weighting of each model which is referred to as 1/n. This simple robust system often outperforms complex weighting schemes when developing an ensemble [12, 99].

The following optimised algorithms (according to F05 score) were evaluated on the holdout set:

- Logistic Regression with One-Sided-Selection
- Random Forest with Random Undersampling
- LightGBM with Random Oversampling
- LSTM with Nearest Cleaning Rule
- TabTransformer with Nearest Cleaning Rule
- Ensemble of all models

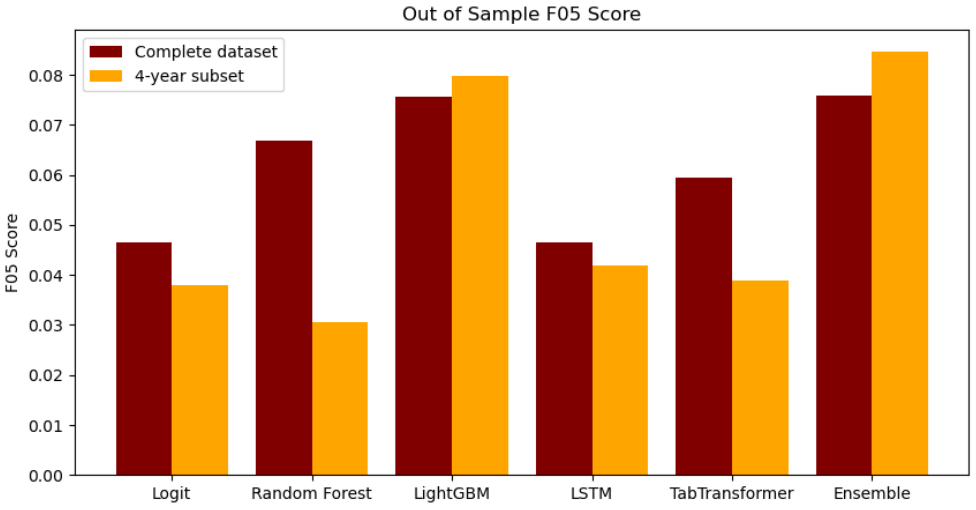


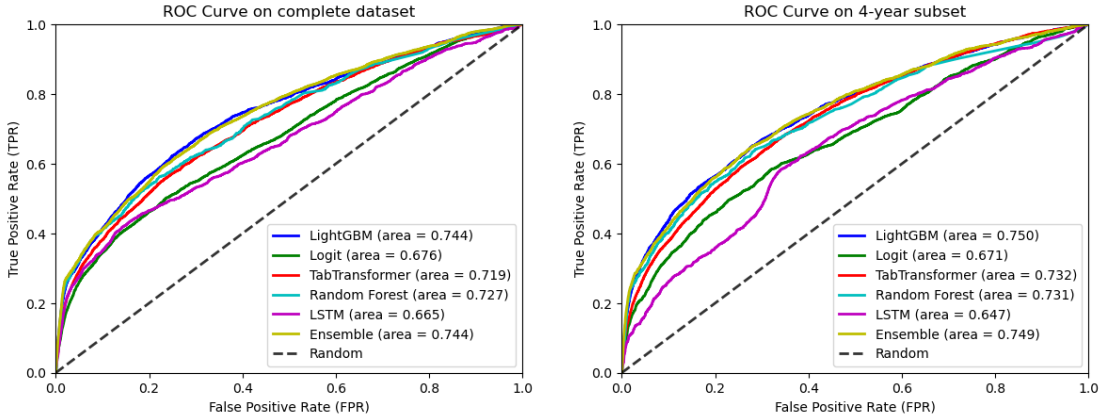
FIGURE 8.1: F05 scores on holdout set validation optimised model for each algorithm studied in the thesis and a 1/n ensemble. There is two bars for each algorithm indicating the dataset used to train the models.

Figure 8.1 presents a comparison of the highest-performing validation models for each algorithm, including logistic regression, random forest, LightGBM, LSTM and TabTransformer. The models were evaluated using the F05 score, a single metric that incorporates precision and recall (favouring precision). The comparison includes models trained on the full dataset and a more recent subset, as well as 1/n ensembles for both. This figure gives insight into the relative performance of these models and ensembles and their suitability for predicting M&As.

A notable trend in the graph is that all the individual algorithms, except for the LightGBM, achieve better F05 scores on the full dataset over the 4-year subset. However, a surprising result is that the ensemble of the 4-year subset models outperforms the ensemble on the complete dataset.

On average, the F05 scores of the individual algorithms exhibit a modest absolute improvement of 1.3% on the complete dataset compared to the 4-year subset. Notably, the random forest and TabTransformer algorithms show the largest absolute increase, averaging 2.8% higher F05 scores on the larger dataset. The observed differences in performance indicate that training on the complete dataset provides a slight advantage in predicting M&A activity for most individual algorithms.

Access to a larger dataset and a longer time period generally improves a model's ability to capture stable patterns in the data [100], leading to better generalisation and improved performance of the holdout set. This trend is evident in all the individual algorithms excluding the LightGBM. The ensemble and the LightGBM model did better when on the smaller dataset, this could be because of non-stationarity in time series data. In a dynamic learning adaptive system like the financial markets where efficiency should improve over time, the data closest to the present is often most valuable. Thus, the most recent data (2014-2017) may be more relevant to the holdout set (2018-2019), as underlying patterns in the data have changed over time [101]. However, this effect is often not strong enough to outweigh the benefits of training on a larger dataset for most models. LightGBM which performs better on the smaller dataset, is the only algorithm that uses oversampling. This, in turn, suggests that oversampling may have a more pronounced effect on the performance of models trained on the 4-year subset, whereas the use of undersampling by the other classifiers may have contributed to their relatively lower performance on the 4-year subset. The 4-year subset ensemble yields a higher F05 score compared to the ensemble on the full dataset, but both are better than any of the individual models on their respective dataset, confirming Clement's theory [99] that a combination of predictions generally beats an individual forecast. Ensembling individual models that have different strengths and weaknesses and uncorrelated errors will complement one another creating a robust ensemble.

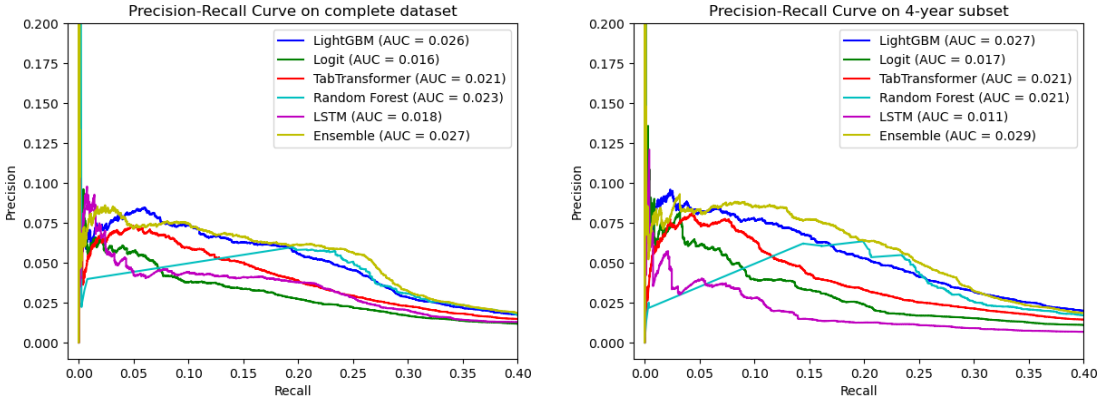


(A) Receiving operating characteristic curve of all models trained on the complete dataset

(B) Receiving operating characteristic curve of all models trained on the 4-year subset

FIGURE 8.2: Comparison of ROC curves (A) and ROC curves (B) which are trained on two distinct time periods with identical hyperparameters and resampling strategies. The curves illustrate the true positive rate plotted against the false positive rate that was achieved on the holdout set.

To further interrogate the results, ROC curves are provided for each model. A ROC curve provides an overall measure of a classifier's performance across all possible classification thresholds. Figure 8.2 provides results that are consistent with the findings of Figure 8.1 in terms of F05 scores and AUC-ROC. For example, both metrics suggest LightGBM and the Ensemble are the best performers and logistic regression and LSTM are worse. This consistency in results reinforces the conclusions drawn in the F05 analysis of the holdout set. Notable discrepancies in the patterns between the F05 score analysis and the ROC curves were identified for the TabTransformer and random forest algorithms. These models exhibited more pronounced variations in their F05 scores (Figure 8.1) between the complete dataset and the 4-year subset; however, their AUC-ROC scores in Figure 8.2 remained relatively consistent across both datasets. The observation indicates that the models maintain a similar balance between sensitivity (recall) and specificity across datasets, even if their F05 scores (which favour precision) differ [102]. The results of the ROC curves suggest the models' ability to discriminate between positive and negative classes remains relatively stable, regardless of the amount of training data available.



(A) Precision-recall curve of all models trained on the complete dataset

(B) Precision-recall curve of all models trained on the 4-year subset

FIGURE 8.3: Comparison of precision-recall curves (A) and precision-recall curves (B) which are trained on two distinct time periods with identical hyperparameters and resampling strategies. The curves illustrate the precision plotted against the recall that was achieved on the holdout set.

ROC curves are a powerful instrument for visualising a classifier's performance. However, they often provide an overly optimistic assessment when applied to highly imbalanced datasets [77, 102, 103]. For highly skewed datasets, such a predicting mergers and acquisitions, the precision-recall curve is often more appropriate, as it is sensitive to the classifier's performance on the minority class.

The precision-recall (PR) curves in Figure 8.3 reveal a similar pattern across all models, with LightGBM and the Ensemble achieving the highest PR-AUC, followed by the random forest and TabTransformer, while logistic regression and LSTM lag behind. Comparing the results across the datasets, we observe that the Ensemble model demonstrates improved performance on the 4-year subset, which aligns with the observations from the F05 score analysis in Figure 8.1. Furthermore, the difference between the AUCs of the PR curves for the full dataset and the 4-year subset is generally smaller than the differences observed in the F05 scores. This implies that the models' ability to maintain a balance between precision and recall is more consistent across datasets than their F05 scores, which prioritise precision. When comparing the PR AUCs (Figure 8.3) to the ROC AUCs (Figure 8.2), it is evident that the PR AUCs are generally lower. This is expected, as ROC curves tend to create an optimistic view on skewed datasets. The consistency in the patterns assessed across F05 scores, ROC AUCs and PR AUCs reinforces the conclusion that LightGBM and the Ensemble models are the best performing models for predicting M&A activity in this study.

8.1.1 Summary of Holdout Analysis

The thorough evaluation of the F05 scores, ROC AUCs and PR AUCs for the different models enables us to determine the best-performing algorithms for predicting M&A activity. Across the three evaluation metrics, LightGBM and the Ensemble models consistently exhibit superior performance. This robust performance can be attributed to their ability to capture complex patterns in the data and maintain a stable balance between precision and recall in the presence of class imbalance.

The F05 scores highlight the models' precision-oriented performance, the ROC AUCs provide an assessment of the models' discriminative ability between positive and negative classes. The PR AUCs offer a more sensitive and informative evaluation in the context of imbalanced data.

Overall, the quantitative data clearly shows that LightGBM and the Ensemble models as the most effective models for predicting mergers and acquisitions in the holdout period 2018-2019.

8.2 Performance of M&A model on a portfolio level

To test the utility of our Ensemble M&A model in a manner that reflects a possible real-world application, we analyse its performance on a portfolio level. The portfolio is setup to equally weight investments across all stocks that the model predicts to be M&A events on a weekly basis. This is done using the Ensemble model trained on the full dataset as this was the model that was planned to be used as the portfolio test. While the Ensemble trained on the 4-year subset had better overall performance, using it would be a form of data leakage. The leakage would be a result of selecting the model after seeing its performance on the holdout set and thus compromise the integrity of findings. The selection of the original Ensemble model trained on the complete dataset robustly aligns with our initial research questions.

Figure 8.4 presents the absolute returns (not accounting for transaction costs) of the Ensemble model using different probability thresholds (0.5%, 1% and 10%) to predict M&A events. This translates to approximately 50, 100 and 1000 stocks being bought and sold every week, respectively. It is important to note that the portfolios have a 100% turnover of stocks every week. This means that every week all the stocks are sold and then the same number of stocks is bought according to what are the most confident takeover predictions for the following week. The 'Universe' is showing the cumulative performance if you bought every share in the Thomson Reuters global stock index. It

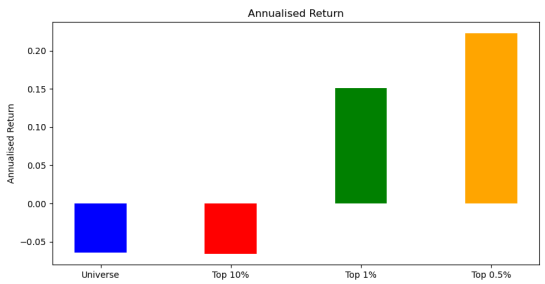
FIGURE 8.4: Cumulative performance with portfolios with increasingly higher probabilities of being M&A events according to the Ensemble model.

is also important to note that the dip in performance at the end of 2018 coincides with a mistake made by the Federal Reserve (Fed) to raise interest rates. The Fed raised its interest rate for the fourth time within the year in December 2018. The U.S. economy was weakening and the trade war between the U.S. and China added uncertainty to the global economy. This rate hike was seen as far too aggressive by many financial analysts as increasing interest rates leads to increased borrowing costs (less growth), selling equities to buy higher interest bonds (lowering stocks prices).

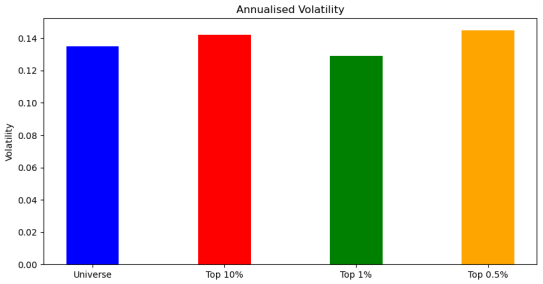
FIGURE 8.5: S&P 500 from Jan 2018 to August 2019 [11]

Collectively, these issues led to negative investor sentiment and large-scale selling of stocks, triggering a significant downturn in the market. This was reflected in the S&P

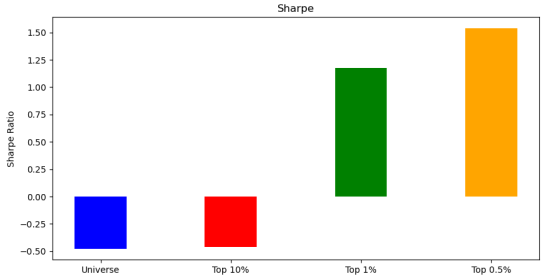
500 see Figure 8.5, which fell approximately 20% from its peak in late September.



(A) Annualised return determines the amount an investment has increased each year as a percentage. A higher annualised return indicates more money made.



(B) Annualised volatility, also known as annualised standard deviation, is a statistical measure of the dispersion of returns in a portfolio. A Higher volatility indicates higher risk.



(C) The Sharpe Ratio is the measurement that evaluates the risk-adjusted return of a portfolio. A higher Sharpe Ratio indicates better risk-adjusted performance.

(D) The worst weekly return is the lowest return (most money lost) the portfolio has experienced in a week. The worst weekly return indicates the potential downside risk that investors can expect.

FIGURE 8.6: Detailed performance of portfolios with increasingly higher M&A probability stocks

The Ensemble model demonstrates profitable returns when using thresholds that only predict the most confident predictions as mergers or acquisitions. Specifically, when only the top 0.5% and top 1% of probabilities are predicted as M&As, the models yield annualised returns of approximately 15% and 22% respectively. These portfolios also have superior Sharpe Ratios of 1.1 and 1.5 respectively, on top of this their volatility and worst weekly return is similar to that of the universe. It's notable that the performance of the portfolio focusing on the top 10% most confident predictions closely mirrors that of the universe. These results suggest that in order to create an effective portfolio, it might be advantageous to only use the most confident predictions.

The effects of transaction costs for a portfolio with 100% weekly stock turnover are an important factor to consider. The portfolio assessment did not integrate these transactions costs, which can be significant given the high turnover rate of the portfolio. For instance, according to Fidelity [104], a portfolio with starting capital of $100 million that is based on the top 1% of the most confident predictions (approximately 100 stocks bought and sold each week), could potentially accrue transaction costs of close to $1 million dollars annually. As a result, there is some overconfidence in the returns of the

portfolio described in this study. Another factor would be the need for a robust trading structure as a portfolio selling and buying 100 stocks per week requires a large amount of work and would add significant overhead cost to the firm. Subsequent research endeavours should incorporate transaction costs to produce a more realistic approximation of portfolio performance.

Chapter 9

Conclusion

The primary aim of this research is to conduct a comprehensive evaluation of optimised machine learning algorithms and their ability to predict mergers and acquisitions accurately. This involved integral parts of the machine leaning pipeline, involving data imputation to handle missing values inherent to high-dimensional financial data, feature selection, hyperparameter tuning and data sampling methods.

The M&A dataset, with a high-dimensionality and many missing values required the specific consideration of imputation techniques. A range of techniques were assessed covering both simple and multivariate imputers, to deal with missing values without a significant loss of valuable information. After thorough testing, we found that the simple imputer 'Forward Fill' (grouped by company name) produced the best results, where a company had a past value for a feature. However, when a company had no past values available for a feature, the multivariate MissForest algorithm emerged the most effective.

The next step is to reduce dimensionality of the dataset, to minimise overfitting, enhance performance and decrease computational cost. The performance (in terms of F05 score) is compared against reduced versions, retaining 75%, 50% and 25% of the original feature set. The results reveal that different algorithms perform better with different size datasets. The decision tree-based algorithms (random forest and LightGBM) and logistic regression perform optimally with the full feature set, whereas the deep neural networks like LSTM and TabTransformer did better with reduced feature subsets.

Optimising the hyperparameters of algorithms is a well-established methodology for enhancing the algorithms capability in handling unseen data. Using the Hyperopt package, Bayesian optimisation is utilised to improve the validation performance of the models.

This intelligent approach to the problem allows learning from previous trials and thus optimises model parameters more effectively than traditional methods such as grid search or random search. The trials were evaluated via contour and scatter plots to further refine the performance of each algorithm.

Given the highly imbalanced nature of the dataset, a variety of oversampling and undersampling techniques are explored to help the validation performance of the focus algorithms. Random oversampling, SMOTE and ADASYN are tested for oversampling, while random undersampling, NCR and OSS are used for undersampling. The effectiveness of each technique varied among the algorithms, highlighting the need for specific sampler-algorithm combinations for optimal performance. Overall the data samplers resulted in a notable increase in F05 score and AUCPR for many of the algorithms.

9.1 Main Findings

The detailed analysis of performance metrics, tailored to imbalanced classification problems, revealed that the LightGBM and Ensemble models are the best models at forecasting M&A events. A secondary objective of the thesis is to use an aggregated Ensemble model, which leverages the combined predictive capabilities of the individual algorithms as the foundation of a practical investment portfolio strategy. The selection of the Ensemble model was based on comprehensive research in the area of predicting mergers and acquisitions, which indicated that such a model can form the basis of a successful investment strategy [12].

The results imply that investors may be able to predict M&A events with a degree of reliability and thus facilitate successful investment strategies, by incorporating a machine learning approach outlined in this thesis. Furthermore, we have gathered an array of cutting-edge machine learning algorithms alongside sophisticated techniques for imputation, feature selection and data sampling. With this strategy, we are contributing to the predictive capabilities in the intricate field of predicting mergers and acquisitions, which is frequently viewed as a near unsolvable task. Many scholars and investors have refrained from studying this investment strategy due to the pronounced data imbalance, where only 0.5% of the data are M&A events, paired with the typical under-performance of probable targets. The findings offer a more optimistic viewpoint, suggesting that mergers and acquisitions can be predicted with a certain level of accuracy. In the literature review, there is a lack of advanced machine learning algorithms and methods being leveraged to address the problem. This was a key weakness which was exploited and has proven to be beneficial, especially in the results of our portfolio performance.

Additionally, this study contributes to machine learning's use in financial prediction tasks. On the holdout set, tree-based models, LightGBM and to a lesser extent random forest, outperformed more computationally demanding neural networks such as LSTM and TabTransformer. This result is consistent with existing literature that suggests tree-based models which are less sophisticated can often perform more effectively and efficiently than neural networks, particularly when dealing with high-dimensional financial data [28, 105, 106]. Our research also highlights the importance of ensemble approaches for dealing with imbalanced datasets, a prevalent problem in financial forecasting. We found that a straightforward ensemble utilising an equal weighting approach performed better than individual algorithms. This finding is consistent with increasing literature advocating for the use of ensemble learning to create more robust models [12, 98, 99].

Together, these findings offer insight into the future of machine learning approaches for imbalanced financial event prediction and possible applications in constructing investment strategies.

9.2 Limitations

One of the study's limitations is the holdout set's short duration, which spans only one year and nine months in comparison to the training set's 17-year duration. The discrepancy might have an impact on how robust the findings are, as a larger holdout dataset might have provided a more concrete proof of the model performance.

It was observed that tree-based models such as LightGBM and random forest, outperformed neural networks, including LSTMs and TabTransformer algorithms in the holdout. This is likely not universally true in every time period. Optimisation of tree-based models is relatively simple compared to that of LSTM and TabTransformer models which have seen less research in the financial world. It is worth noting that the LSTM and TabTransformer frequently outperformed the random forest on the validation sets (see Table 7.1 and Table 7.2). This indicates that the holdout set may have been particularly well-suited to tree-based models.

Another potential limitation is the selection of models. The chosen algorithms are based on financial literature and state-of-the-art techniques have yet to be widely applied to the domain of M&A prediction. An effort was made to include a diverse range of algorithms to ensure a detailed evaluation. However, it is quite possible that there are more effective algorithms which should be examined for this specific prediction task.

9.3 Future Work

Further studies should explore additional machine learning algorithms that have seen success in other imbalanced financial prediction domains. These could include well researched algorithms such as XGBoost and the Support Vector Machine. A limitation of the study is the relatively small holdout set, a more expansive holdout dataset over different time periods would provide a better temporal analysis of the model's ability to forecast M&As.

The addition of more external data, such as news feeds, sentiment analyses and filings data could further enhance forecasting accuracy. This nuanced data often contains signals of M&A events, that are not easily captured in typical financial indicators. Finally, the investment portfolio strategy is in no way designed to manage risk and can thus have significant drawdowns. Future studies should consider risk management strategies to mitigate these issues, for example making the M&A predictions equally weighted across different geographical regions or sectors (helping diversification).

Bibliography

[1] NasdaqGS. Nuance communication, inc. *Refinitiv*, 2021.

[2] Feb 2022. URL https://www.ibm.com/topics/decision-trees.

[3] Abhishek Sharma. Random forest vs decision tree: Which is right for you?, May 2020. URL https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/.

[4] Pranshu Sharma. Basic introduction to feed-forward network in deep learning, Mar 2022. URL https://www.analyticsvidhya.com/blog/2022/03/basic-introduction-to-feed-forward-network-in-deep-learning/.

[5] Sebastian Raschka. Activation functions for artificial neural networks, Mar 2022. URL https://sebastianraschka.com/faq/docs/activation-functions.html.

[6] Robert Keim. How to train a basic perceptron neural network - technical articles, Nov 2019. URL https://www.allaboutcircuits.com/technical-articles/how-to-train-a-basic-perceptron-neural-network/.

[7] Debasish Kalita. A brief overview of recurrent neural networks (rnn), Mar 2022. URL https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/.

[8] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. Tabtransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*, 2020.

[9] Arun Amballa. Feature engineering part-1 mean/ median imputation., Aug 2020. URL https://medium.com/analytics-vidhya/feature-engineering-part-1-mean-median-imputation-761043b95379.

[10] Vitor Cerqueira. 4 things to do when applying cross-validation with time series, Dec 2022. URL https://towardsdatascience.com/4-things-to-do-when-applying-cross-validation-with-time-series-c6a5674ebf3a.

[11] Federal Reserve Bank of St. Louis. Fred economic data, 2023. URL https://fred.stlouisfed.org/series/SP500#.

[12] Wang, Luo, Jussa, Alvarez, and Rohal. Machine learning takeovers. *Wolfe Research*, 2017.

[13] The Editors of Encyclopedia Brittanica. Stock exchange, Sep 2021. URL https://www.britannica.com/topic/stock-exchange-finance.

[14] Aggerholm and Lunde. Active vs. passive investing. *Copenhagen Business School*, pages 2–5, May 2019.

[15] Andrew Lo. What is an index? *MIT*, Oct 2015.

[16] Joseph Piotroski. Value investing: The use of historical financial statement information to separate winners from losers. *Journal of Accounting Research*, 38, 2000.

[17] Aswath Damodaran. Growth investing: Betting on the future? *Stern School of Business*, July 2012.

[18] Titman and Jegadeesh. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of Finance*, Mar 1993.

[19] Eggert and Jackson. Investing in change. *Mercer*, 2021.

[20] Eugene F Fama. Efficient market hypothesis. *Diss. PhD Thesis, Ph. D. dissertation*, 1960.

[21] Adam Hayes. Guide to mergers and acquisitions, Dec 2021. URL https://www.investopedia.com/terms/m/mergersandacquisitions.asp#:~:text=Key%20Takeaways%201%20The%20terms%20mergers%20and%20acquisitions,banner%20of%20one%20corporate%20name.%20More%20items...%20.

[22] Alexander Roberts, William Wallace, and Peter Moles. *Mergers and Acquisitions Edinburgh Business School*. Heriot-Watt University, 2016.

[23] Simkowitz and Monroe. A discriminant analysis function for conglomerate targets. *Southern Journal of Business*, pages 1–16, 1971.

[24] K. Palepu. Predicting takeover targets. *Journal of Accounting and Economics*, pages 3–35, 1986.

[25] Georges Geha. Use of modern machine learning techniques to predict the occurrence and outcome of corporate takeover events. *Massachusetts Institute of Technology*, Jan 2021.

[26] Guang Xiang, Zeyu Zheng, Miaoiao Wen, Jason Hong, Carolyn Rose, and Chao Liu. A supervised approach to predict company acquisition with factual and topic features using profiles and news articles on techcrunch. *Massachusetts Institute of Technology*, Jan 2021.

[27] Haykaz Aramyan. Predicting m&a targets using machine learning techniques. *Refinitiv*, September 2021.

[28] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

[29] John McCarthy. From here to human-level ai. *Artificial Intelligence*, 171(18): 1174–1182, 2007.

[30] Sepp Hochreiter. *Machine Learning: Supervised Techniques*. Johannes Kepler University Linz, 2014.

[31] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. *yann*, 2010. URL http://yann.lecun.com/exdb/mnist/.

[32] Dastan Maulud and Adnan M Abdulazeez. A review on linear regression comprehensive in machine learning. *Journal of Applied Science and Technology Trends*, 1(4):140–147, 2020.

[33] Andreas C Müller and Sarah Guido. *Introduction to machine learning with Python: a guide for data scientists*. " O'Reilly Media, Inc.", 2016. p. 24.

[34] Kaggle. Titanic - machine learning from disaster, 2020. URL https://www.kaggle.com/competitions/titanic/data.

[35] Chao-Ying Joanne Peng, Kuk Lida Lee, and Gary M Ingersoll. An introduction to logistic regression analysis and reporting. *The journal of educational research*, 96(1):3–14, 2002.

[36] Jan Salomon Cramer. The origins of logistic regression. 2002.

[37] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[38] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[39] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[40] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[41] Richard Bellman. Functional equations in the theory of dynamic programming: Viii. the variation of green's functions for the one-dimensional case. *Proceedings of the National Academy of Sciences*, 43(9):839–841, 1957.

[42] Kevin Gurney. *An introduction to neural networks*. UCL Press Limited, 1997.

[43] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[44] CM Bishop. Neural networks for pattern recognition. *Clarendon Press google schola*, 2:223–228, 1995.

[45] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[46] Larry R Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5(64-67):2, 2001.

[47] Long Short-Term Memory. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 2010.

[48] Gloria A Aguayo, Lu Zhang, Michel Vaillant, Moses Ngari, Magali Perquin, Valerie Moran, Laetitia Huart, Rejko Krüger, Francisco Azuaje, Cyril Ferdynus, et al. Comparison of deep neural networks and regularised cox regression models in the prediction of neurodegenerative diseases in the general older population. *Available at SSRN 4026085*, 2022.

[49] Andrew Briggs, Taane Clark, Jane Wolstenholme, and Philip Clarke. Missing.... presumed at random: cost-analysis of incomplete data. *Health Economics*, 12(5): 377–392, 2003.

[50] Dr. Deng. Single imputation methods for missing data: Locf, bocf, lrcf (last rank carried forward), and nocb (next observation carried backward), Jan 2021. URL http://onbiostatistics.blogspot.com/2021/01/single-imputation-methods-for-missing.html.

[51] Donald B. Rubin. *Multiple imputation for nonresponse in surveys*. John Wiley, 1987.

[52] Joseph L Schafer. *Analysis of incomplete multivariate data*. CRC press, 1997.

[53] Abdullah Z Alruhaymi and Charles J Kim. Why can multiple imputations and how (mice) algorithm work? *Open Journal of Statistics*, 11(5):759–777, 2021.

[54] Hassan Khan, Andreas P Kalogeropoulos, Vasiliki V Georgiopoulou, Anne B Newman, Tamara B Harris, Nicolas Rodondi, Douglas C Bauer, Stephen B Kritchevsky, and Javed Butler. Frailty and risk for heart failure in older adults: the health, aging, and body composition study. *American heart journal*, 166(5): 887–894, 2013.

[55] Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

[56] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[57] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE, 2008.

[58] Ivan Tomek. Two modifications of cnn. *IEEE Trans. Systems, Man and Cybernetics*, 6:769–772, 1976.

[59] Dennis L Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 408–421, 1972.

[60] Peter Hart. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3):515–516, 1968.

[61] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, page 179. Nashville, USA, 1997.

[62] Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Conference on artificial intelligence in medicine in Europe*, pages 63–66. Springer, 2001.

[63] John D'Angelo. Predicting mergers and acquisitions, 2012. Honors Undergraduate Thesis, University of Central Florida.

[64] Stewart Myers and Nicholas Majluf. Corporate financing and investment decisions when firms have information that investors do not have. *Journal of Financial Economics*, pages 187–221, February 1984.

[65] Mike Cudd and Rakesh Duggal. Industry distributional characteristics of financial ratios: An acquisition theory application. *The Financial Review*, 35(1):105–20, 2000.

[66] Michael Gort. An economic disturbance theory of mergers. *The Quarterly Journal of Economics*, 83(4):624–642, 1969.

[67] Joel Hasbrouck. The characteristics of takeover targets q and other measures. *Journal of Banking and Finance*, 9(3):351–362, February 2007.

[68] Srinivasan Ragothaman, Bijayananda Naik, and Kumoli Ramakrishnan. Predicting corporate acquisitions: An application of uncertain reasoning using rule induction. *Information Systems Frontiers*, 5:401–412, 2003.

[69] Athanasios Tsagkanos, Antonios Georgopoulos, and Costas Siriopoulos. Predicting greek mergers and acquisitions: a new approach. *International Journal of Financial Services Management*, 2(4):289–303, 2007.

[70] Chih-Ping Wei, Yu-Syun Jiang, and Chin-Sheng Yang. Patent analysis for supporting merger and acquisition (m&a) prediction: A data mining approach. In *Designing E-Business Systems. Markets, Services, and Networks: 7th Workshop on E-Business, WEB 2008, Paris, France, December 13, 2008, Revised Selected Papers 7*, pages 187–200. Springer, 2009.

[71] Guang Xiang, Zeyu Zheng, Miaomiao Wen, Jason Hong, Carolyn Rose, and Chao Liu. A supervised approach to predict company acquisition with factual and topic features using profiles and news articles on techcrunch. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 6, pages 607–610, 2012.

[72] Francisco Ramadas da Silva Ribeiro Bento. *Predicting start-up success with machine learning*. PhD thesis, 2018.

[73] Javier Arroyo, Francesco Corea, Guillermo Jimenez-Diaz, and Juan A Recio-Garcia. Assessment of machine learning performance for decision support in venture capital investments. *Ieee Access*, 7:124233–124243, 2019.

[74] Md Manjurul Ahsan, MA Parvez Mahmud, Pritom Kumar Saha, Kishor Datta Gupta, and Zahed Siddique. Effect of data scaling methods on machine learning algorithms and model performance. *Technologies*, 9(3):52, 2021.

[75] Marcos Lopez de Prado. Advances in financial machine learning (chapter 7). In *Advances in Financial Machine Learning*, chapter 7. Wiley, 1 edition, 2018.

[76] Farhan Mohammad and Samira Golsefid. Evaluation of feature selection methods, 2020. White Paper, PayPal.

[77] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[78] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[79] B Efron and RJ Tibshirani. An introduction to the bootstrap. 57 boca raton, 1993.

[80] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020.

[81] Brownlee. A gentle introduction to the rectified linear unit (relu), 2019.

[82] Matthew F Dixon, Diego Klabjan, and Lan Wei. Over sampling for time series classification. *matthew-dixon*, 2017.

[83] Sayan Putatunda and Kiran Rama. A comparative analysis of hyperopt as against other approaches for hyper-parameter optimization of xgboost. In *Proceedings of the 2018 International Conference on Signal Processing and Machine Learning*, pages 6–10, 2018.

[84] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112, 2017.

[85] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

[86] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[87] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1):3133–3181, 2014.

[88] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168, 2006.

[89] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[90] Anthony Christopher Davison and David Victor Hinkley. *Bootstrap methods and their application*. Cambridge university press, 1997.

[91] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[92] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[93] Tawfiq Hasanin, Taghi M Khoshgoftaar, Joffrey L Leevy, and Richard A Bauder. Severely imbalanced big data challenges: investigating data sampling approaches. *Journal of Big Data*, 6(1):1–25, 2019.

[94] Richard Zuech, John Hancock, and Taghi M Khoshgoftaar. Detecting web attacks using random undersampling and ensemble learners. *Journal of Big Data*, 8(1): 1–20, 2021.

[95] Tarid Wongvorachan, Surina He, and Okan Bulut. A comparison of undersampling, oversampling, and smote methods for dealing with imbalanced classification in educational data mining. *Information*, 14(1):54, 2023.

[96] Mingzhu Tang, Qi Zhao, Huawei Wu, and Zimin Wang. Cost-sensitive lightgbm-based online fault detection method for wind turbine gearboxes. *Frontiers in Energy Research*, 9:701574, 2021.

[97] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):1–54, 2019.

[98] Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. Ensemble selection from libraries of models. In *Proceedings of the twenty-first international conference on Machine learning*, page 18, 2004.

[99] Robert T Clemen. Combining forecasts: A review and annotated bibliography. *International journal of forecasting*, 5(4):559–583, 1989.

[100] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

[101] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[102] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.

[103] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[104] Fidelity International. Fidelity fees calculator, 2023. URL https://www.fidelity.co.uk/fees-calculator/.

[105] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[106] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? *arXiv preprint arXiv:2207.08815*, 2022.

[107] Charu C Aggarwal et al. Neural networks and deep learning. *Springer*, 10:978–3, 2018.

[108] Vishal Maini. Machine learning for humans, part 2.1: Supervised learning, May 2018. URL https://medium.com/machine-learning-for-humans/supervised-learning-740383a2feab.

[109] Stephen Wright. Sparse optimization methods. *University of Wisconsion-Madison*, Feb 2009.

[110] Zeyi Wen, Jiashuai Shi, Bingsheng He, Jian Chen, Kotagiri Ramamohanarao, and Qinbin Li. Exploiting gpus for efficient gradient boosting decision tree training. *IEEE Transactions on Parallel and Distributed Systems*, 30(12):2706–2717, 2019.

[111] Refinitiv. MSCI ACWI [Image]. Eikon, 2021.

[112] Adam Lindberg and Gustav Gerholm. Comparison of machine learning models for market predictions with different time horizons, 2021.

Appendix A

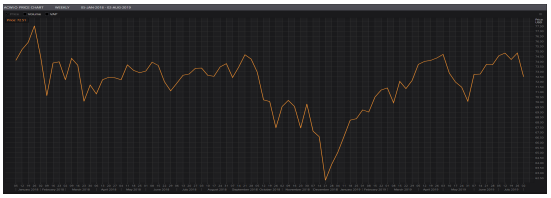
Appendix

A.1 Indexes over holdout period

(A) Dow Jones Industrial Average Index [11]

(B) NASDAQ Composite Index [11]

(C) Wilshire 5000 Total Market Index [11]



(D) MSCI ACWI Index [111]

FIGURE A.1: Performance of four popular indexes over the holdout period: Dow Jones Industrial Average, NASDAQ Composite, Wilshire 5000 and the MSCI ACWI.

A.2 Confusion Matrix of Ensemble

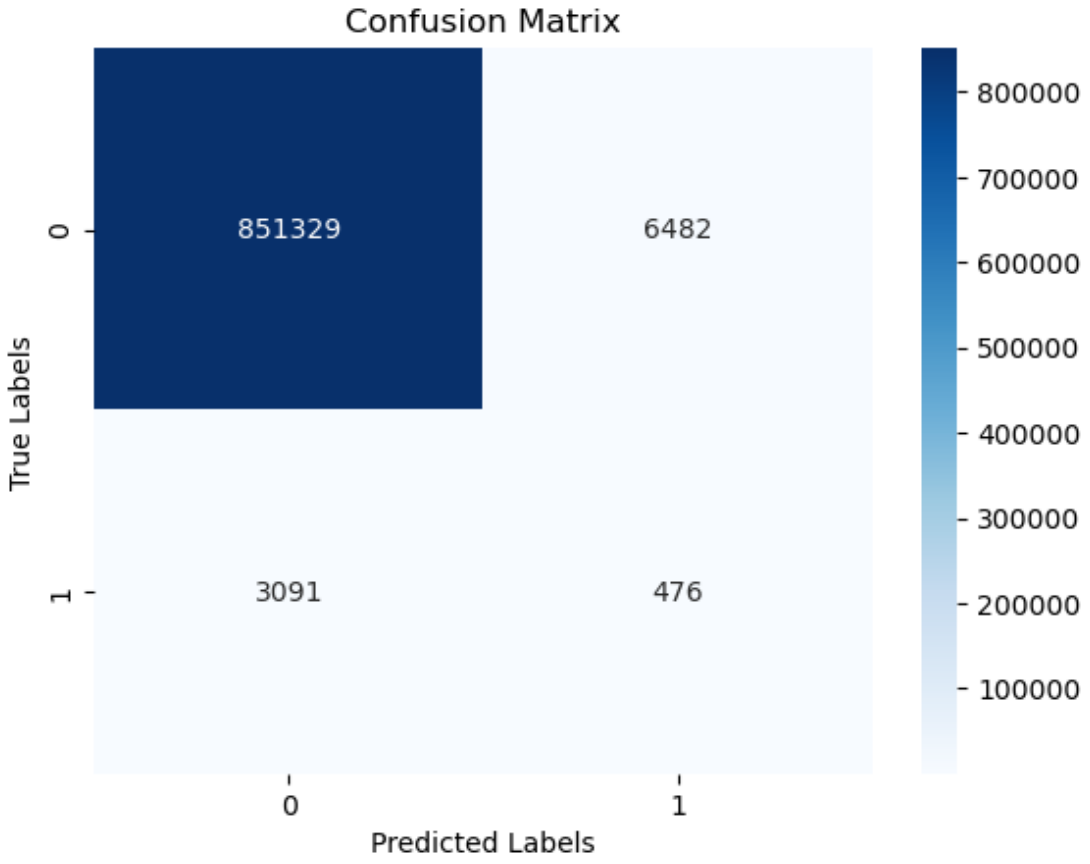


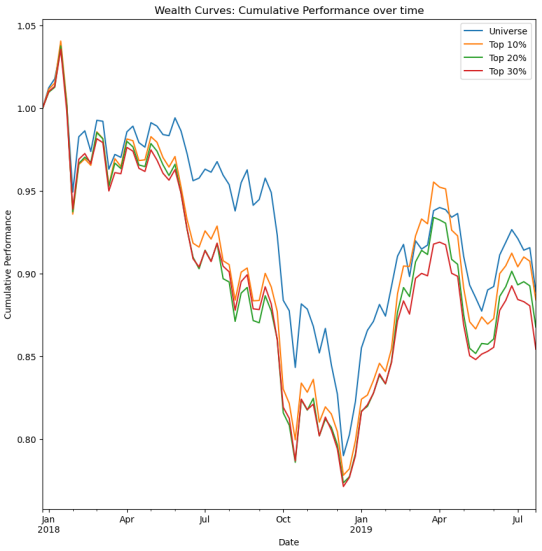
FIGURE A.2: Confusion matrix of the Ensemble model.

A.3 Portfolio Results

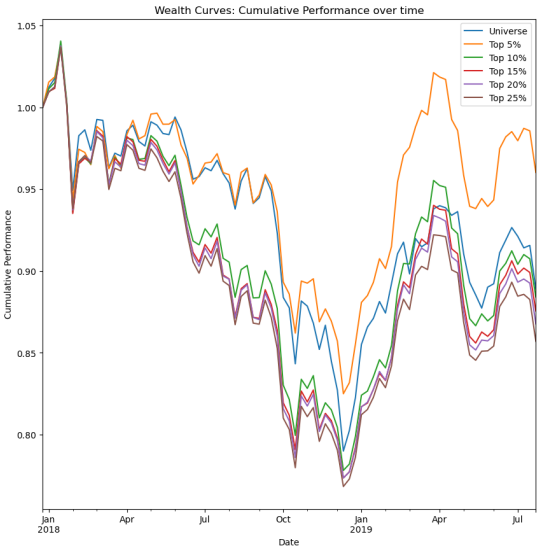
Threshold	Annualised Return	Annualised Volatility	Sharpe	Worst Weekly Return	Best Weekly Return	Beta
0.10%	32%	25%	1.26	-9%	15%	0.72
0.20%	19%	20%	0.94	-8%	11%	0.74
0.30%	40%	17%	2.29	-6%	8%	0.67
0.40%	30%	16%	1.87	-6%	7%	0.66
0.50%	22%	14%	1.54	-6%	6%	0.66
1.00%	15%	13%	1.18	-5%	6%	0.75
2.00%	13%	12%	1.07	-4%	5%	0.77
3.00%	6%	12%	0.46	-5%	5%	0.83
4.00%	0%	13%	0.02	-5%	4%	0.89
5.00%	-2%	13%	-0.13	-6%	4%	0.91
10.00%	-7%	14%	-0.46	-7%	4%	0.96
15.00%	-7%	14%	-0.51	-7%	5%	0.97
20.00%	-8%	14%	-0.55	-6%	5%	0.97
25.00%	-8%	14%	-0.61	-6%	5%	0.96
30.00%	-9%	13%	-0.65	-6%	5%	0.96

FIGURE A.3: Portfolio performance metrics for various confidence thresholds: From the 0.1% most confident predictions to the top 30%. The more confident predictions produce a higher annualised return along with a higher annualised volatility. The best portfolios in terms of Sharpe Ratio lie between the 0.3% and 0.5% most confident predictions. The beta of the portfolios trends closer to the market as the predictions used are less confident.

A.4 Wealth Curves: Cumulative Returns over Holdout Period

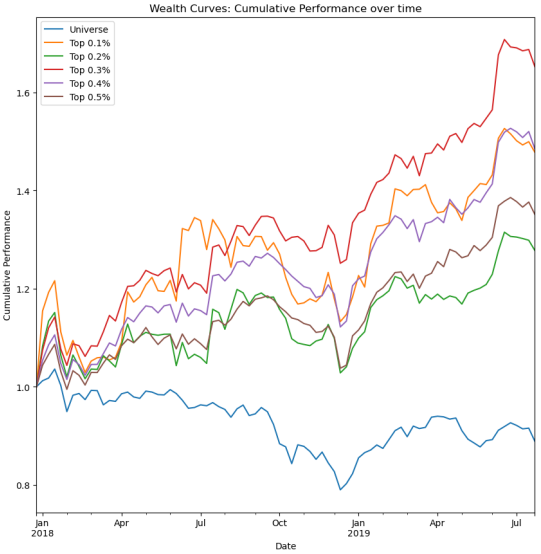


(A) No significant difference in cumulative performance between Top 10%, Top 20% and Top 30% of highest M&A probability portfolios. The Eikon universe outperforms each of these portfolios.



(B) The Top 5% M&A probability portfolio significantly outperforms the Top 10%, 15%, 20% and 25% M&A portfolios. The Eikon universe outperforms all portfolios except the Top 5%.

(C) An upward trend can be seen clearly in the cumulative returns of the Top 1%, 2%, 3%, 4% and 5% M&A portfolios, with the performance getting better as the portfolio is based on more and more confident predictions.



(D) The best performing portfolio is the Top 0.3% of most confident predictions. These targeted portfolios focusing on less stocks significantly beat the Eikon Refinitiv universe. The resulting portfolios produce high returns along with high volatility.

FIGURE A.4: Comparison of Investment Portfolios with varying confidence thresholds for merger and acquisition predictions. Each portfolio represents the same amount of investment but uses different confidence thresholds, ranging from the most confident predictions (Top 0.1%) to higher less confident thresholds (e.g. Top 5%, 10% and 30%). As the portfolio uses more confident predictions it focuses on fewer stocks, resulting in a more targeted investment strategy.

A.5 Feature Names

Category	Feature Nme
Analyst Sentiment	Recommendation - Number of Buy
Analyst Sentiment	Recommendation - Number of Buy (1 month ago)
Analyst Sentiment	Recommendation - Number of Buy (3 months ago)
Analyst Sentiment	Recommendation - Number of Buy (6 months ago)
Analyst Sentiment	Recommendation - Number of Buy (12 months ago)
Analyst Sentiment	Recommendation - Number of Sell
Analyst Sentiment	Recommendation - Number of Sell (1 month ago)
Analyst Sentiment	Recommendation - Number of Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Sell (6 months ago)
Analyst Sentiment	Recommendation - Number of Sell (12 months ago)
Analyst Sentiment	Recommendation - Number of Strong Buy
Analyst Sentiment	Recommendation - Number of Strong Buy (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Buy (3 months ago)
Analyst Sentiment	Recommendation - Number of Strong Buy (6 months ago)
Analyst Sentiment	Recommendation - Number of Strong Buy (12 months ago)
Analyst Sentiment	Recommendation - Number of Strong Sell
Analyst Sentiment	Recommendation - Number of Strong Sell (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Strong Sell (6 months ago)
Analyst Sentiment	Recommendation - Number of Strong Sell (12 months ago)
Analyst Sentiment	Recommendation - Number of Hold
Analyst Sentiment	Recommendation - Number of Hold (1 month ago)
Analyst Sentiment	Recommendation - Number of Hold (3 months ago)
Analyst Sentiment	Recommendation - Number of Hold (6 months ago)
Analyst Sentiment	Recommendation - Number of Hold (12 months ago)
Analyst Sentiment	Recommendation - Number of Total
Analyst Sentiment	Recommendation - Number of Total (1 month ago)
Analyst Sentiment	Recommendation - Number of Total (3 months ago)
Analyst Sentiment	Recommendation - Number of Total (6 months ago)
Analyst Sentiment	Recommendation - Number of Total (12 months ago)
Analyst Sentiment	Price Target - Mean
Analyst Sentiment	Price Target - Mean (1 month ago)
Analyst Sentiment	Price Target - Mean (3 months ago)
Analyst Sentiment	Price Target - Mean (6 months ago)
Analyst Sentiment	Price Target - Mean (12 months ago)
Analyst Sentiment	Dividends Per Share Mean
Analyst Sentiment	Dividends Per Share Mean (1 month ago)
Analyst Sentiment	Dividends Per Share Mean (3 months ago)
Analyst Sentiment	Dividends Per Share Mean (6 months ago)
Analyst Sentiment	Dividends Per Share Mean (12 months ago)

Analyst Sentiment	Earnings Per Share Mean
Analyst Sentiment	Earnings Per Share Mean (1 month ago)
Analyst Sentiment	Earnings Per Share Mean (3 months ago)
Analyst Sentiment	Earnings Per Share Mean (6 months ago)
Analyst Sentiment	Earnings Per Share Mean (12 months ago)
Analyst Sentiment	Dividends Per Share - Actual Surprise
Analyst Sentiment	Earnings Per Share - Actual Surprise
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (1 month ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (3 months ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (6 months ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (12 months ago)
Analyst Sentiment	Revenue - Mean
Analyst Sentiment	Revenue - Mean (1 month ago)
Analyst Sentiment	Revenue - Mean (3 months ago)
Analyst Sentiment	Revenue - Mean (6 months ago)
Analyst Sentiment	Revenue - Mean (12 months ago)
Efficiency	Asset Turnover
Efficiency	Asset Turnover (1 month ago)
Efficiency	Asset Turnover (3 months ago)
Efficiency	Asset Turnover (6 months ago)
Efficiency	Asset Turnover (12 months ago)
Growth	Long Term Growth - Mean
Growth	Long Term Growth - Mean (1 month ago)
Growth	Long Term Growth - Mean (3 months ago)
Growth	Long Term Growth - Mean (6 months ago)
Growth	Long Term Growth - Mean (12 months ago)
Growth	Captial Expenditures, Cumulative
Growth	Captial Expenditures, Cumulative (1 month ago)
Growth	Captial Expenditures, Cumulative (3 months ago)
Growth	Captial Expenditures, Cumulative (6 months ago)
Growth	Captial Expenditures, Cumulative (12 months ago)
Growth	Captial Expenditures/Total Assets
Growth	Captial Expenditures/Total Assets (1 month ago)
Growth	Captial Expenditures/Total Assets (3 months ago)
Growth	Captial Expenditures/Total Assets (6 months ago)
Growth	Captial Expenditures/Total Assets (12 months ago)
Liquidity	Liquidity Component
Liquidity	Liquidity Component (1 month ago)
Liquidity	Liquidity Component (3 months ago)
Liquidity	Liquidity Component (6 months ago)
Liquidity	Liquidity Component (12 months ago)

Price Momentum	Price Close
Price Momentum	Price Close (1 month ago)
Price Momentum	Price Close (3 months ago)
Price Momentum	Price Close (6 months ago)
Price Momentum	Price Close (12 months ago)
Price Momentum	Price 52 Week High
Price Momentum	Price 52 Week Low
Quality	Net Debt
Quality	Net Debt (1 month ago)
Quality	Net Debt (3 months ago)
Quality	Net Debt (6 months ago)
Quality	Net Debt (12 months ago)
Quality	Net Debt to EBITDA
Quality	Net Debt to EBITDA (1 month ago)
Quality	Net Debt to EBITDA (3 months ago)
Quality	Net Debt to EBITDA (6 months ago)
Quality	Net Debt to EBITDA (12 months ago)
Quality	Company Shares
Quality	Company Shares (1 month ago)
Quality	Company Shares (3 months ago)
Quality	Company Shares (6 months ago)
Quality	Company Shares (12 months ago)
Quality	ROE Common Equity
Quality	ROE Common Equity % (1 month ago)
Quality	ROE Common Equity % (3 months ago)
Quality	ROE Common Equity % (6 months ago)
Quality	ROE Common Equity % (12 months ago)
Quality	Pretax ROE Total Equity %
Quality	Pretax ROE Total Equity % (1 month ago)
Quality	Pretax ROE Total Equity % (3 months ago)
Quality	Pretax ROE Total Equity % (6 months ago)
Quality	Pretax ROE Total Equity % (12 months ago)
Value	Price to Book Value Per Share
Value	Price to Book Value Per Share (1 month ago)
Value	Price to Book Value Per Share (3 months ago)
Value	Price to Book Value Per Share (6 months ago)
Value	Price to Book Value Per Share (12 months ago)
Value	Price-to-Earnings Ratio
Value	Price-to-Earnings Ratio (1 month ago)
Value	Price-to-Earnings Ratio (3 months ago)
Value	Price-to-Earnings Ratio (6 months ago)
Value	Price-to-Earnings Ratio (12 months ago)

Value	Dividend yield
Value	Book Value Per Share
Value	Book Value Per Share (1 month ago)
Value	Book Value Per Share (3 months ago)
Value	Book Value Per Share (6 months ago)
Value	Book Value Per Share (12 months ago)
Value	Enterprise Value
Value	Enterprise Value (1 month ago)
Value	Enterprise Value (3 months ago)
Value	Enterprise Value (6 months ago)
Value	Enterprise Value (12 months ago)
Value	Enterprise Value to EBITDA
Value	Enterprise Value to EBITDA (1 month ago)
Value	Enterprise Value to EBITDA (3 months ago)
Value	Enterprise Value to EBITDA (6 months ago)
Value	Enterprise Value to EBITDA (12 months ago)
Value	Enterprise Value to Sales
Value	Enterprise Value to Sales (1 month ago)
Value	Enterprise Value to Sales (3 months ago)
Value	Enterprise Value to Sales (6 months ago)
Value	Enterprise Value to Sales (12 months ago)
Value	Company Market Cap
Value	Company Market Cap (1 month ago)
Value	Company Market Cap (3 months ago)
Value	Company Market Cap (6 months ago)
Value	Company Market Cap (12 months ago)
Value	Total Assets, Reported
Value	Total Assets, Reported (1 month ago)
Value	Total Assets, Reported (3 months ago)
Value	Total Assets, Reported (6 months ago)
Value	Total Assets, Reported (12 months ago)
Value	Total Liabilities
Value	Total Liabilities (1 month ago)
Value	Total Liabilities (3 months ago)
Value	Total Liabilities (6 months ago)
Value	Total Liabilities (12 months ago)
Value	Current Ratio
Value	Current Ratio (1 month ago)
Value	Current Ratio (3 months ago)
Value	Current Ratio (6 months ago)
Value	Current Ratio (12 months ago)
	Price Close Currency
	GICS Sector
	GICS Industry Group

FIGURE A.5: Complete List of Dataset Features

A.6 Feature Selection Subsets

Category	Feature Nme
Analyst Sentiment	Recommendation - Number of Buy
Analyst Sentiment	Recommendation - Number of Buy (3 months ago)
Analyst Sentiment	Recommendation - Number of Sell
Analyst Sentiment	Recommendation - Number of Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Sell (12 months ago)
Analyst Sentiment	Recommendation - Number of Strong Buy
Analyst Sentiment	Recommendation - Number of Strong Buy (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Buy (6 months ago)
Analyst Sentiment	Recommendation - Number of Strong Buy (12 months ago)
Analyst Sentiment	Recommendation - Number of Strong Sell
Analyst Sentiment	Recommendation - Number of Strong Sell (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Hold
Analyst Sentiment	Recommendation - Number of Hold (1 month ago)
Analyst Sentiment	Recommendation - Number of Hold (3 months ago)
Analyst Sentiment	Recommendation - Number of Hold (6 months ago)
Analyst Sentiment	Recommendation - Number of Total
Analyst Sentiment	Recommendation - Number of Total (1 month ago)
Analyst Sentiment	Recommendation - Number of Total (3 months ago)
Analyst Sentiment	Recommendation - Number of Total (6 months ago)
Analyst Sentiment	Price Target - Mean
Analyst Sentiment	Price Target - Mean (1 month ago)
Analyst Sentiment	Price Target - Mean (3 months ago)
Analyst Sentiment	Price Target - Mean (6 months ago)
Analyst Sentiment	Dividends Per Share Mean
Analyst Sentiment	Dividends Per Share Mean (12 months ago)
Analyst Sentiment	Earnings Per Share Mean
Analyst Sentiment	Earnings Per Share Mean (1 month ago)
Analyst Sentiment	Earnings Per Share Mean (3 months ago)
Analyst Sentiment	Earnings Per Share Mean (6 months ago)
Analyst Sentiment	Earnings Per Share Mean (12 months ago)
Analyst Sentiment	Dividends Per Share - Actual Surprise
Analyst Sentiment	Earnings Per Share - Actual Surprise
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (1 month ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (3 months ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (6 months ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (12 months ago)
Analyst Sentiment	Revenue - Mean
Analyst Sentiment	Revenue - Mean (1 month ago)
Analyst Sentiment	Revenue - Mean (3 months ago)
Analyst Sentiment	Revenue - Mean (6 months ago)
Analyst Sentiment	Revenue - Mean (12 months ago)

Efficiency	Asset Turnover
Efficiency	Asset Turnover (1 month ago)
Efficiency	Asset Turnover (3 months ago)
Efficiency	Asset Turnover (6 months ago)
Efficiency	Asset Turnover (12 months ago)
Growth	Long Term Growth - Mean
Growth	Long Term Growth - Mean (1 month ago)
Growth	Long Term Growth - Mean (3 months ago)
Growth	Long Term Growth - Mean (6 months ago)
Growth	Captial Expenditures, Cumulative
Growth	Captial Expenditures, Cumulative (1 month ago)
Growth	Captial Expenditures, Cumulative (3 months ago)
Growth	Captial Expenditures, Cumulative (6 months ago)
Growth	Captial Expenditures, Cumulative (12 months ago)
Growth	Captial Expenditures/Total Assets
Growth	Captial Expenditures/Total Assets (1 month ago)
Growth	Captial Expenditures/Total Assets (3 months ago)
Growth	Captial Expenditures/Total Assets (6 months ago)
Growth	Captial Expenditures/Total Assets (12 months ago)
Liquidity	Liquidity Component
Liquidity	Liquidity Component (1 month ago)
Price Momentum	Price 52 Week High
Price Momentum	Price 52 Week Low
Quality	Net Debt (1 month ago)
Quality	Net Debt (3 months ago)
Quality	Net Debt (12 months ago)
Quality	Net Debt to EBITDA
Quality	Net Debt to EBITDA (1 month ago)
Quality	Net Debt to EBITDA (3 months ago)
Quality	Net Debt to EBITDA (6 months ago)
Quality	Net Debt to EBITDA (12 months ago)
Quality	Company Shares
Quality	Company Shares (1 month ago)
Quality	Company Shares (3 months ago)
Quality	Company Shares (6 months ago)
Quality	Company Shares (12 months ago)
Quality	ROE Common Equity
Quality	ROE Common Equity % (1 month ago)
Quality	ROE Common Equity % (3 months ago)
Quality	ROE Common Equity % (6 months ago)
Quality	ROE Common Equity % (12 months ago)

Quality	Pretax ROE Total Equity %
Quality	Pretax ROE Total Equity % (1 month ago)
Quality	Pretax ROE Total Equity % (3 months ago)
Quality	Pretax ROE Total Equity % (6 months ago)
Value	Price to Book Value Per Share
Value	Price to Book Value Per Share (1 month ago)
Value	Price to Book Value Per Share (3 months ago)
Value	Price to Book Value Per Share (6 months ago)
Value	Price to Book Value Per Share (12 months ago)
Value	Price-to-Earnings Ratio
Value	Price-to-Earnings Ratio (1 month ago)
Value	Price-to-Earnings Ratio (3 months ago)
Value	Price-to-Earnings Ratio (6 months ago)
Value	Price-to-Earnings Ratio (12 months ago)
Value	Dividend yield
Value	Book Value Per Share
Value	Book Value Per Share (1 month ago)
Value	Book Value Per Share (3 months ago)
Value	Book Value Per Share (6 months ago)
Value	Book Value Per Share (12 months ago)
Value	Enterprise Value
Value	Enterprise Value (1 month ago)
Value	Enterprise Value (3 months ago)
Value	Enterprise Value to EBITDA
Value	Enterprise Value to EBITDA (1 month ago)
Value	Enterprise Value to EBITDA (3 months ago)
Value	Enterprise Value to EBITDA (6 months ago)
Value	Enterprise Value to EBITDA (12 months ago)
Value	Enterprise Value to Sales
Value	Enterprise Value to Sales (1 month ago)
Value	Enterprise Value to Sales (12 months ago)
Value	Company Market Cap
Value	Company Market Cap (1 month ago)
Value	Total Assets, Reported
Value	Total Assets, Reported (1 month ago)
Value	Total Assets, Reported (3 months ago)
Value	Total Liabilities (3 months ago)
Value	Total Liabilities (6 months ago)
Value	Current Ratio
	Price Close Currency
	GICS Sector
	GICS Industry Group

FIGURE A.6: 75% most important features

Category	Feature Nme
Analyst Sentiment	Recommendation - Number of Buy
Analyst Sentiment	Recommendation - Number of Buy (3 months ago)
Analyst Sentiment	Recommendation - Number of Sell
Analyst Sentiment	Recommendation - Number of Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Sell (12 months ago)
Analyst Sentiment	Recommendation - Number of Strong Buy
Analyst Sentiment	Recommendation - Number of Strong Buy (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Buy (6 months ago)
Analyst Sentiment	Recommendation - Number of Strong Sell
Analyst Sentiment	Recommendation - Number of Strong Sell (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Hold
Analyst Sentiment	Recommendation - Number of Hold (1 month ago)
Analyst Sentiment	Recommendation - Number of Hold (3 months ago)
Analyst Sentiment	Recommendation - Number of Total
Analyst Sentiment	Recommendation - Number of Total (6 months ago)
Analyst Sentiment	Price Target - Mean
Analyst Sentiment	Price Target - Mean (1 month ago)
Analyst Sentiment	Price Target - Mean (6 months ago)
Analyst Sentiment	Earnings Per Share Mean
Analyst Sentiment	Earnings Per Share Mean (1 month ago)
Analyst Sentiment	Earnings Per Share Mean (3 months ago)
Analyst Sentiment	Earnings Per Share Mean (6 months ago)
Analyst Sentiment	Earnings Per Share Mean (12 months ago)
Analyst Sentiment	Dividends Per Share - Actual Surprise
Analyst Sentiment	Earnings Per Share - Actual Surprise
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (1 month ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (3 months ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (6 months ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (12 months ago)
Analyst Sentiment	Revenue - Mean
Analyst Sentiment	Revenue - Mean (3 months ago)
Efficiency	Asset Turnover
Efficiency	Asset Turnover (1 month ago)
Efficiency	Asset Turnover (3 months ago)
Efficiency	Asset Turnover (6 months ago)
Growth	Long Term Growth - Mean
Growth	Long Term Growth - Mean (1 month ago)
Growth	Long Term Growth - Mean (3 months ago)
Growth	Long Term Growth - Mean (6 months ago)

Growth	Captial Expenditures, Cumulative
Growth	Captial Expenditures/Total Assets
Growth	Captial Expenditures/Total Assets (1 month ago)
Growth	Captial Expenditures/Total Assets (6 months ago)
Growth	Captial Expenditures/Total Assets (12 months ago)
Liquidity	Liquidity Component
Liquidity	Liquidity Component (1 month ago)
Price Momentum	Price 52 Week High
Price Momentum	Price 52 Week Low
Quality	Net Debt (1 month ago)
Quality	Net Debt to EBITDA
Quality	Net Debt to EBITDA (1 month ago)
Quality	Net Debt to EBITDA (3 months ago)
Quality	Company Shares
Quality	Company Shares (1 month ago)
Quality	ROE Common Equity
Quality	ROE Common Equity % (1 month ago)
Quality	ROE Common Equity % (3 months ago)
Quality	ROE Common Equity % (6 months ago)
Quality	Pretax ROE Total Equity %
Quality	Pretax ROE Total Equity % (1 month ago)
Quality	Pretax ROE Total Equity % (6 months ago)
Value	Price to Book Value Per Share
Value	Price to Book Value Per Share (1 month ago)
Value	Price to Book Value Per Share (3 months ago)
Value	Price to Book Value Per Share (6 months ago)
Value	Price-to-Earnings Ratio
Value	Price-to-Earnings Ratio (6 months ago)
Value	Book Value Per Share
Value	Book Value Per Share (1 month ago)
Value	Book Value Per Share (6 months ago)
Value	Book Value Per Share (12 months ago)
Value	Enterprise Value
Value	Enterprise Value (1 month ago)
Value	Enterprise Value to EBITDA
Value	Enterprise Value to EBITDA (1 month ago)
Value	Enterprise Value to Sales (1 month ago)
Value	Enterprise Value to Sales (12 months ago)
Value	Total Assets, Reported
Value	Total Assets, Reported (1 month ago)
Value	Current Ratio
	GICS Sector
	GICS Industry Group

FIGURE A.7: 50% most important features

Category	Feature Nme
Analyst Sentiment	Recommendation - Number of Buy
Analyst Sentiment	Recommendation - Number of Buy (3 months ago)
Analyst Sentiment	Recommendation - Number of Sell
Analyst Sentiment	Recommendation - Number of Strong Buy (1 month ago)
Analyst Sentiment	Recommendation - Number of Strong Sell
Analyst Sentiment	Recommendation - Number of Strong Sell (3 months ago)
Analyst Sentiment	Recommendation - Number of Hold
Analyst Sentiment	Recommendation - Number of Hold (3 months ago)
Analyst Sentiment	Recommendation - Number of Total
Analyst Sentiment	Price Target - Mean
Analyst Sentiment	Price Target - Mean (6 months ago)
Analyst Sentiment	Earnings Per Share Mean
Analyst Sentiment	Earnings Per Share Mean (1 month ago)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate)
Analyst Sentiment	Price / Earnings Per Share (Mean Estimate) (1 month ago)
Efficiency	Asset Turnover
Efficiency	Asset Turnover (3 months ago)
Growth	Long Term Growth - Mean
Growth	Long Term Growth - Mean (1 month ago)
Growth	Captial Expenditures, Cumulative
Growth	Captial Expenditures/Total Assets
Growth	Captial Expenditures/Total Assets (1 month ago)
Quality	Net Debt (1 month ago)
Quality	Net Debt to EBITDA
Quality	Company Shares
Quality	Company Shares (1 month ago)
Quality	ROE Common Equity
Quality	Pretax ROE Total Equity %
Quality	Pretax ROE Total Equity % (6 months ago)
Value	Price to Book Value Per Share (1 month ago)
Value	Price to Book Value Per Share (3 months ago)
Value	Price-to-Earnings Ratio
Value	Price-to-Earnings Ratio (6 months ago)
Value	Book Value Per Share
Value	Book Value Per Share (1 month ago)
Value	Enterprise Value
Value	Enterprise Value (1 month ago)
Value	Enterprise Value to EBITDA
Value	Enterprise Value to EBITDA (1 month ago)
Value	Enterprise Value to Sales (1 month ago)
Value	Enterprise Value to Sales (12 months ago)
	GICS Sector

FIGURE A.8: 25% most important features

A.7 Imputers Settings

Imputer	**Ideal Parameters**
Forward Fill	Group by stock name and forward fill by feature
Rolling Mean	Group by stock name and fill by averaging previous five instances of the specific feature
Rolling Median	Group by stock name and fill by finding the median of the previous five instances of the specific feature
MissForest	n_estimators=100, max_depth=8, max_features='sqrt'

TABLE A.1: Parameters for imputers