Predicting Major Donor Prospects using Machine Learning

by

Aishwarya Vaishali Sathyamurthi

Thesis
submitted in partial fulfillment of the requirements
for the Degree of Master of Science(Computer Science)

Acadia University
Fall Graduation 2022

© by Aishwarya Vaishali Sathyamurthi, 2022

This thesis by Aishwarya Vaishali Sathyamurthi was defended successfully in an oral examination on May 03, 2022.

The examining committee for the thesis was:

Dr. Corinne Haigh, Chair

Dr. Hai Wan, External Examiner

Dr. Daniel L. Silver, Internal Examiner

Dr. Greg Lee, Supervisor

Dr. Darcy Benoit, Head

The author retains copyright in this thesis. Any substantial copying or any other actions that exceed fair dealing or other exceptions in the Copyright Act require the permission of the author.

Contents

List of Tables

List of Figures

Abstract

An important concern for many not-for-profits is their major gift fundraising. Major gifts are large gifts (typically $10,000+) and donors who give these gifts are called major donors. Depending upon the charity type, major gifts can constitute as much as 80% of donation dollars received by a charity. Thus, being able to predict who will give a major gift is crucial for charities. In the for-profit sector, using machine learning to target customers has been a long-standing strategy. Charities (i.e., not-for-profits) have been slower to implement machine learning to aid with donor identification and retention. To implement this, we used random forest classifier which predicted with high accuracy (94.47%) and 31 prospects. We tried different experiments using deep learning techniques, Adaboost, decision trees, extra trees classifiers, LASSO, ElasticNet, XGBoost and Light Gradient Boosting regression in order to develop models that can accurately predict future major donors who will donate $10,000 or more. We also experimented with using only donation and behavioural data, which saw increase in false positive values than false negatives for most of the charities. Furthermore, we forecast how much money major donor constituents will contribute to the charity, in which Light GBM regression model was performing well with lowest RMSE values and standard deviations.

Definitions of Abbreviations and Symbols

- AI — Artificial Intelligence
- BDLSTM — Bi-Directional Long Short-Term Memory
- CF — Constituent Features
- CNN — Convolutional Neural Network
- DL — Deep Learning
- EDA — Exploratory Data Analysis
- EFB — Exclusive Feature Bundling
- GBDT — Gradient Boosting Decision Tree
- GD — Gradient Descent
- GRU — Gated Recurrent Unit
- LASSO — Least Absolute Shrinkage and Selection Operator
- LGBM — Light Gradient Boosting Model
- LSTM — Long Short Term Memory
- LYBUNT — Last Year But Unfortunately Not This Year
- LR — Logistic Regression
- MAE — Mean Absolute Error
- MD — Major Donor
- ML — Machine Learning
- MG — Major Gift
- MGOs — Major Gift Officers
- NaN — Not a Number
- NN — Neural Network

- ReLU — Rectified Linear Unit
- RMSE — Root Mean Squared error
- RNN — Recurrent Neural Network
- SMOTE — Synthetic Minority Oversampling Technique
- St.Dev — Standard Deviation
- SYBUNT — Some Year But Unfortunately Not This Year
- XGBoost — eXtreme Gradient Boosting

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Greg Lee, whose expertise, understanding, care, and patience played a vital role in my successful completion of graduate studies. I would also like to thank Dr. Elhadi Shakshuki and Dr. Daniel L. Silver, who guided me through the course units. I am thankful to the faculty at Acadia's Jodrey School of Computer Science for their support. I would also like to thank Ms. Theresa Starratt.
I want to express my gratitude towards Mr. Mark Hobbs, CEO of Fundmetric and his team for all their help and support, who have supported me throughout the research journey.
I am grateful to my family for constantly motivating me to achieve the research milestone. I especially want to extend my thanks and appreciation to my mother Rajana Dharmaraj, my brother Karthikeyen and my love Ashok Preetham who have always supported me during good and bad times.

Chapter 1

Introduction

Charities rely on major gifts for a significant portion of their budget. University foundations in particular receive about 80% of their donation dollars from major gifts [14]. The threshold for a major gift varies by charity, but typically ranges from $10,000 to $50,000. While these are typical minimum thresholds, major gifts can be in the range of millions of dollars.

Major donors are donors who have either given a gift that meets the charity's major giving threshold or who have an official pledge to do so with the charity. Since these donations can be 500x to 50000x larger than the average donation to a charity, charities spend much more time with each potential major donor than non-major donors in order to increase the likelihood of a gift. Thus, having a precise list of likely major donors is critical for a charity's success. The aim of the research presented in this paper is to predict potential major donors. Major donors are important because their gifts make up a large chunk of the organisations overall fundraising revenue. It is crucial to prioritize the relationships with them. Major donors are more inclined to give to nonprofits that have a dedicated stewardship strategy to cultivate their relationships [28].

Charities employ major gift officers (MGOs) strictly to seek out and 'convert' major donor prospects. These MGOs can spend years developing a relationship with potential major donors and thus the decision concerning with whom to begin a relationship is an important one [14]. Typically, MGOs maintain a spreadsheet of prospects, hand-chosen based on wealth alone. Wealth is not the only factor in an individual's decision to make a major gift, and so MGOs consider demographic, education, behavioural, and donation data [14].

MGOs are the organisation go-to people for all aspects of major giving.

They can help nonprofits with major gift efforts such as identification, cultivation, solicitation, and stewardship [14].

1.1 Major Donors Strategies

In this section, we will discuss some of the most effective strategies used by nonprofits for identifying major donors.

- Create a donor recognition wall: Donor recognition walls allow charities to honor major donors in a permanent, meaningful way. By building a lasting testament to the major donors, charities will be able to convey the depth and longevity of their impact and appreciation [28].

- Perform a prospect screening: Prospect research is a research process performed by a nonprofit's fundraising and development teams to gather more data about their donors and prospects to determine a donor's ability and desire to donate to a specific cause [28].

- Look into planned giving potential: Planned gifts are donations that are decided on in the present and then allocated to the nonprofit in the future. This type of contribution is often made when donors leave charitable donations as a part of their wills after they have passed away. Historically, most planned gifts are of equal size to major gifts, sometimes even larger. However, they are also less prevalent.

- Start a Major Donor Society: When charities request a major gift, they are asking for a big commitment. Major donors need to be carefully cultivated leading up to the solicitation and just as carefully stewarded after the gift has been made. One way to do that is to create a branded major donor society [1].

- Hiring a Major Gifts Officer (MGO): MGOs are permanent employees. They are responsible for handling major donor prospect files, preparing educational materials, collaborating with the marketing team to create promotional materials, presenting major gift appeals to prospects, making the major gift proposal, following up with major donors to maintain

[1] Major donor societies are groups set up by a non-profit to help cultivate and steward donors, keep them loyal to the organization, and encourage them to give more (and larger) gifts in the future.

the relationship and finally seeking to upgrade opportunities when appropriate.

- Host events catered to Major Donor Acquisition: An event catered to major donors gathers people capable of making a large gift together so that they can learn more about the organization, the constituents charities serve, and the kind of impact their gifts could have, all while the fundraisers simultaneously become familiar with the prospects.

- Engage Major Donors as Volunteers: Volunteerism helps donors bond with the organization. This is true for all small and mid-level donors as well as major donors [28].

With machine learning (ML), nonprofits can identify individuals who are more likely to engage and donate to their cause to support their mission [13]. Machine learning algorithms can quickly analyze data (such as donation, demographic, behavioural and educational datasets), making the learning process much more efficient [10]. Humans have limited time and resources to process and analyze huge data sets. Machine learning has the ability to extract insights from these complex data sets in minutes.

1.2 Scope

This research develops machine learning models that predict major donors across various charities, as well as estimates of how much money a major donor constituent [2] will contribute to the charity. The exact dollar amount that is considered a major gift varies from organization to organization. My research will help the industry perform better in raising more money for charities. Machine learning models facilitate the work of charities by predicting lists of potential major donors for raising maximum donations. Due to their size, major gifts can alter the "path of charity". They are a substantial funding for a nonprofit. One major-gift-size donation can be the difference between meeting a fundraising goal or not.

1.3 Thesis Structure

This thesis consists of seven chapters. Chapter 1 includes the introduction and structure of this thesis. Chapter 2 introduces the problem and its scope, related research, along with the objectives. Domain knowledge on machine learning (ML) and deep learning models, including supervised learning, long short-term memory (LSTMs), bidirectional long short-term memory (BDLSTMs), convolutional neural networks (CNNs), gated recurrent unit (GRUs), recurrent neural networks (RNNs) are presented in Chapter 3. Chapter 4 presents the data sources, data preparation and the hardware and software environment used in our experiments. Chapter 5 explains the theory behind different approaches used for predictions. Chapter 6 presents the empirical evaluation of the discussed approaches for predicting major donor prospects. And Chapter 7 presents the conclusions of this research and suggests probable future work.

[2] A constituent who donates $10,000 or more to a charity are considered as an major donor constituent.

Chapter 2

Problem Formulation

The business problem at hand is to generate a ranked list of constituents who have never given a major gift as prospects, so that MGOs can focus their time and effort on them. To do so, we solve the problem of determining which machine learning algorithm can best learn to distinguish between major donors and non-major donors and then use that algorithm to predict future major donors.

We expected that the most valuable insight in the data would be to predict when a minor donor is likely to become a major donor. Then such persons might be actively solicited for major donations. We will make use of actual charitable data, including demographic, donation, educational, and behavioural data. This data will be fed into machine learning models in order to predict who will become future major donors. Furthermore, we forecast how much money major donor constituents will contribute to the charity. As any donor who donates $10,000 is considered a major donor, but a donor who donates $1,000,000 has much more impact than someone who donates $10,000, and the fundraising team must focus more on such donors.

The aforementioned goal can be achieved by training machine learning models using labelled data and predicting major donor prospects for a charity. The model is trained on a balanced dataset. The unbalanced dataset includes a greater number of non-major donors (people who have not made a major donation) than major donors (people who have made a major donation) as described in Table 4.2. If the unbalanced data is fed to a machine learning model, it affects the performance metrics of the model as mentioned in Section 4.2. So, to avoid that, we balance the data which has equal amount from major donor and non-major donor while training and testing, which leads the

ML models to perform better across several performance metrics (e.g., accuracy, precision and F1-score) in order to predict the major donors in the test set.

2.1 Related Research

Many suggestions for fundraising emails can be found on the World Wide Web [16]. As early as 1994, research was conducted on the automatic discovery of major donors [26]. Lindahl and Winship used a logit model to demonstrate that low past giving levels are generally associated with a lower likelihood of making a major gift. While charities have been slow to adopt machine learning, for-profit organisations have long used these methods [2, 8].

Most of the past research done predicting donor giving behavior makes use of linear techniques. Connolly and Blanchette (1986) [30] used discriminant analysis, and Gerlinda Melchiori (1988) [29] used classification analysis to predict donor behavior, both of which are types of linear regression. These techniques are inappropriate when the object is to predict rare events (such as giving over $10,000) or when the dependent variable has an upper or lower bound and there are a large number of individuals at the bound (as with giving, where there are numerous individuals with zero giving).

Fundraising has played a prominent role in the development history of higher education in North America [46]. In 2017, research was conducted that tackled the two major fundraising challenges of identifying potential donors who can upgrade the number of their pledges and identifying new donors. Personal attributes like age, wealth, marital status, and gender were considered for the research. Gaussian naive Bayes, random forest, and support vector machine algorithms were used and evaluated. The test results showed that the models can successfully distinguish between donors and non-donors with the best model achieving an F-Score of 60%. For a robust test, a ten-fold cross-validation method was used for validation purposes. In the validation section, the SVM performed the best in terms of F1-score, accuracy, and recall rate. In contrast, different types of deep learning methods were employed and assessed in this research. The test findings showed that the deep learning methods were able to accurately find major donor prospects according to F1-score.

Brittingham and Pezzullo note that certain current characteristics of alumni were found to be predictors for major gift giving in some studies, but not others [6]. Income, age, number of degrees from the institution, emotional attach-

ment to the school, participation in alumni events, and participation in and donation to other voluntary and religious groups were found to be predictors.

Wesley and Christopher (1992) used logit analysis in 1992 to predict the individuals who would give higher (e.g., $100,000) or lower ($1,000) donations based on the data from the alumni database as well as the geo-demographic information [45]. Their result showed that 92% of the dollars could be collected with 36.5% prospects selected in the annual fund model. Later with their upgraded model (1994) [26], a slightly better performance was achieved for major gift prediction. In this research, the test results using deep learning models showed accurate results when using large data sets for certain charities, compared to logistic regression model as described in empirical studies.

In the United States, charitable giving was $307 billion in 2008 (Giving USA Foundation, 2008) [3]. Gifts from individuals are always the largest source of nonprofit gifts and usually represent about 80% of all annual giving to the estimated 1.4 million U.S. nonprofits. Major gifts are called "transformative" in the charitable world and comprise only a very small percentage of household giving but are often critically important to a nonprofit's growth and success [22].

Machine learning models like artificial neural networks (ANNs) and random forests (RFs) have been utilized to predict donor interactions with a large charity [12], such as, "can we predict when a minor donor is likely to become a major donor for targeted donation requests?". This is comparable to our work, in which we use demographic, donation, educational, and behavioural data to train models such as Adaboost, decision trees, random forest classifiers, extra trees classifiers, LSTMGRUs, SimpleRNNs, GRUs, BDLSTM-GRU-TDLs and BDLSTMCNNs to predict whether or not constituents will make a major donation to support a specific cause. We also experiment using only the donation and behavioural data and removing demographic, and educational data to see how well the model predicts. Further, we built a regression model that predicts how much money major donor constituents will contribute to the charity.

Chapter 3

Background Studies

This chapter provides the basic concepts of Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning along with the setup of our empirical work.

3.1 Artificial Intelligence

Artificial intelligence (AI) is a field which combines computer science and robust datasets, to enable problem-solving. It has as sub-fields of machine learning and deep learning. These disciplines are comprised of AI algorithms which seek to create expert systems which make predictions or classifications based on input data.

3.2 Machine Learning

Machine learning (ML) is the study of statistical theory and mathematical algorithms that computers use to optimize performance relying on example data or past experiences. Machine learning algorithms build a model to make predictions using training data rather than using explicit instructions. Machine learning provides solutions to many problems in image recognition, speech recognition, medical diagnosis and donor predictions. Machine learning is categorized as supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning.

- **Supervised learning**

  Supervised learning algorithms know the target outputs of inputs in the training data and try to model the relationships between target outputs and input features. Learning algorithms produce a function to predict output values for any new inputs after learning from sufficient labeled data sets. Learning algorithms can also compare predicted values with target output values and use the errors to improve performance.

  Supervised learning can be split into two subcategories: classification and regression.

  - **Classification** uses an algorithm to accurately assign test data into specific categories. It recognizes specific entities within the dataset and attempts to draw some conclusions on how those entities should be labeled or defined. Common classification algorithms are linear classifiers, support vector machines, decision trees, k-nearest neighbor, and random forest.
  - **Regression** is used to understand the relationship between dependent and independent variables. The output variable must be of continuous nature or real value. Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

- **Unsupervised Learning**

  For unsupervised learning, the model is trained with unlabeled data sets. The system may not figure out the correct output values for any inputs. However, it can try to determine the hidden patterns and rules from input data, summarize and group data points, which gives us meaningful data information. Unsupervised learning models are utilized for three main tasks: clustering, association, and dimensionality reduction.

- **Semi-supervised learning**

  Semi-supervised learning is between supervised learning and unsupervised learning. Algorithms use both labeled data and unlabeled data in the training process. Semi-supervised learning is the best choice when there is limited training data.

- **Reinforcement Learning**

Reinforcement learning algorithms interact with the environment by taking actions and acquiring reward feedback. The goal of reinforcement learning is to find a suitable action model that would maximize the total cumulative reward of the agent.

We compared various machine learning and deep learning techniques and evaluated the mean accuracy for each of them by a Stratified K-fold cross-validation to prevent overfitting. In this basic approach, K-fold CV, the training set is split into k smaller sets :

- The model is trained using the k-1 folds as training data.
- It uses the last fold to compute the model performance.

3.2.1 Gaussian Naive Bayes

Naive Bayes classifiers are a group of supervised machine learning classification algorithms based on the Bayes theorem. It is a simple classification technique, but has high functionality [27]. Complex classification problems can also be implemented by using Naive Bayes Classifier.

When working with continuous data, an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{3.1}$$

Sometimes assume variance is independent of Y (i.e., σi), or independent of Xi (i.e., σk) or both (i.e., $\sigma(z)$)

Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution. An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all what is needed to define such a distribution.

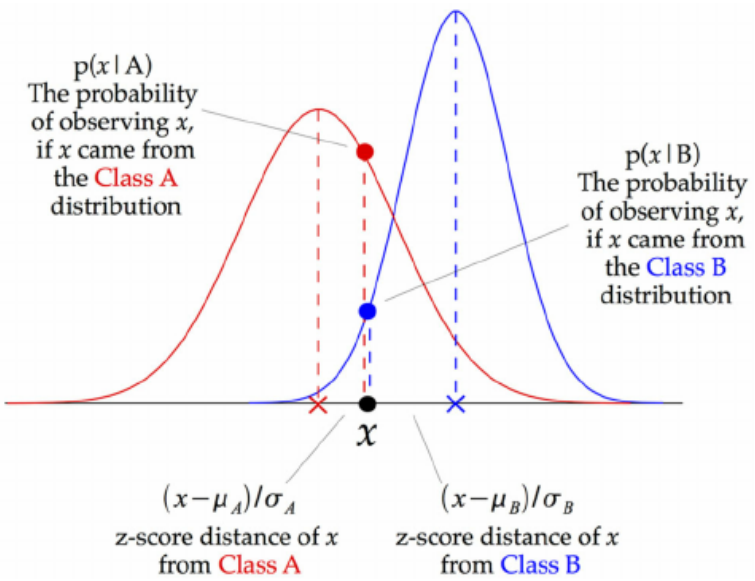

Figure 3.1: Illustration of how a Gaussian Naive Bayes classifier works[27]

3.2.2 Decision Trees

A decision tree is the powerful statistical tool for classification, prediction, interpretation, and data manipulation. Decision trees consists of three different elements:

- Root node: The top level node represents the objective the model.
- Branches: They are the stem from the root represent different options or courses of action that are available when making a particular decision. They are indicated with an arrow line and include associated costs, as well as the likelihood to occur.
- Leaf node: The leaf nodes which are attached at the end of the branches represent possible outcomes for each action. There are two types of leaf nodes: square leaf nodes, which indicate another decision to be made, and circle leaf nodes, which indicate a chance event or unknown outcome.

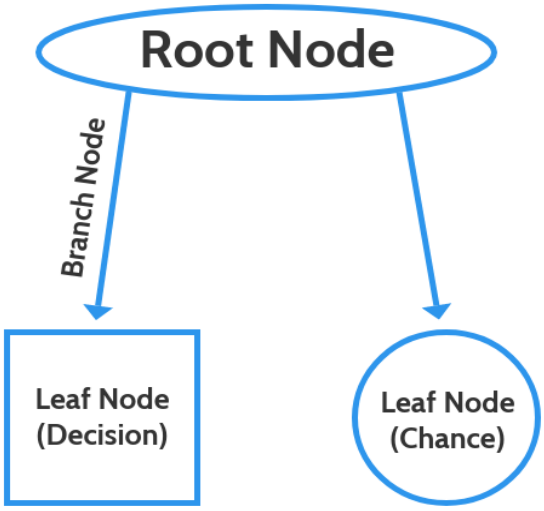

Figure 3.2: Elements in a decision tree[21]

Decision trees are fundamentally recursive, the algorithm learns through repetition [21]. The algorithms attempts different splits and determines the split that achieves the correct classification as many times as possible. The root node is selected based on the attribute selection measure (ASM) and is repeated until there is a leaf node (cannot split anymore).

ASM is a technique used in data mining processes for data reduction. The two main ASM techniques are Gini Index and Information Gain (ID3).

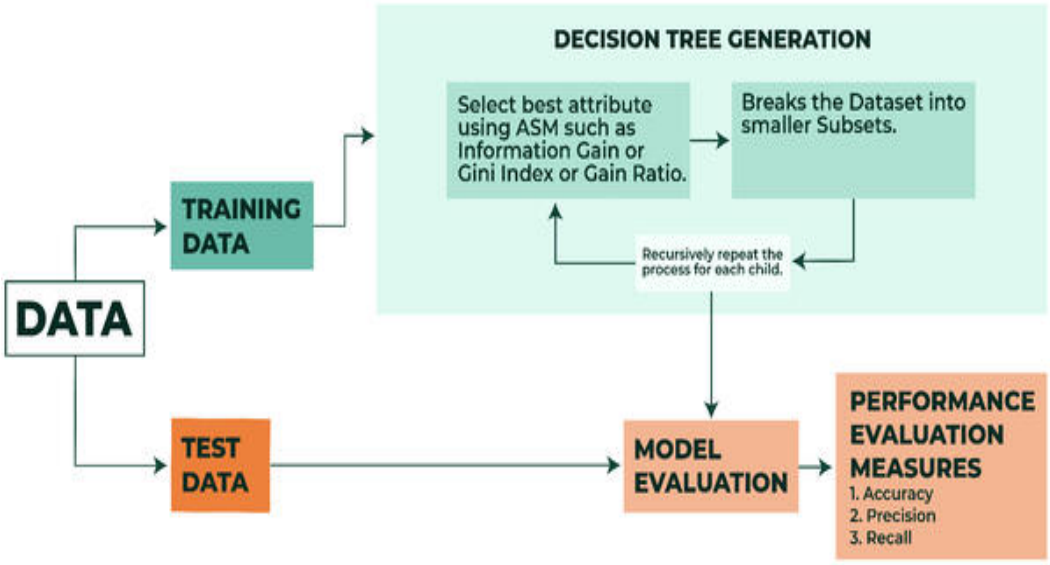

Figure 3.3: Illustration of how a Decision tree model works[21]

The ID3 algorithm builds decision trees using a top-down greedy search approach through the space of possible branches with no backtracking. The steps in ID3 algorithm are as follows:

- It begins with the original set S as the root node.
- On each iteration of the algorithm, it iterates through the very unused attribute of the set S and calculates Entropy(H) and Information Gain (IG) of this attribute.
- It then selects the attribute which has the smallest entropy or largest information gain.
- The set S is then split by the selected attribute to produce a subset of the data.
- The algorithm continues to recur on each subset, considering only attributes never selected before.

3.2.3 Adaboost Classifier

Adaboost or Adaptive Boosting is one of ensemble boosting classifier proposed by Yoav Freund and Robert Schapire in 1996 [32]. It is a meta-estimator that

begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as a base classifier if it accepts weights on individual training examples. Adaboost should meet two conditions:

- The classifier should be trained interactively on various weighed training examples.
- In each iteration, it tries to provide an excellent fit for these examples by minimizing training error.

It works in the following steps:

- Initially, all observations are given equal weights.
- A model is built on a subset of data.
- Using this model, predictions are made on the whole dataset.
- Errors are calculated by comparing the predictions and actual values.
- While creating the next model, higher weights are given to the data points which were predicted incorrectly.
- Weights can be determined using the error value. For instance, the higher the error the more is the weight assigned to the observation.
- This process is repeated until the error function does not change, or the maximum limit of the number of estimators is reached.

The "strength" of the "weak" learners: In this research, we use simple weak learners, such as decision stumps (1-level decision trees), to overcome the problem of overfitting in Adaboost algorithms.

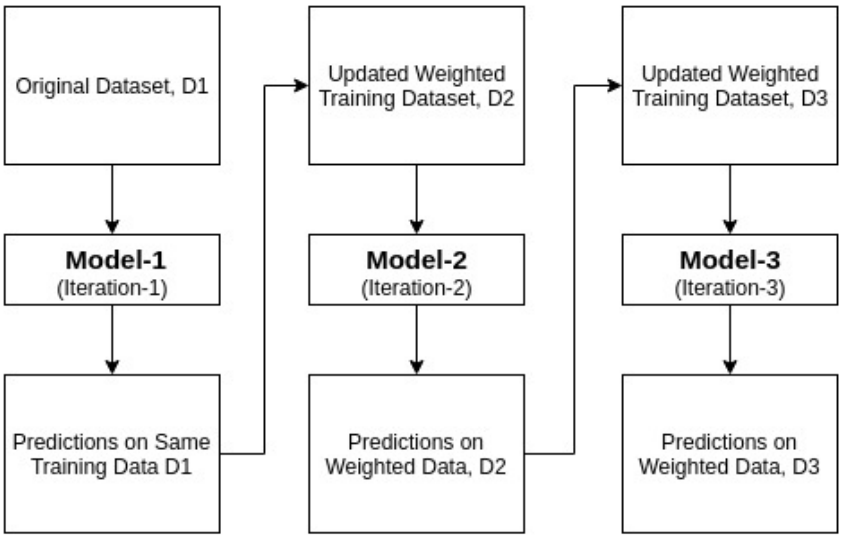

Figure 3.4: Adaboost Model[32]

3.2.4 Random Forest Classifier

A random forest is a collection of decision trees whose results are aggregated into one final result. They limit overfitting without substantially increasing error due to bias. It is also one of the most used algorithms, because of its simplicity and diversity (it can be used for both classification and regression tasks).

A random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. With random forest, we can also deal with regression tasks by using the algorithm's regressor. Random forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. We can also make trees more random by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does).

The hyperparameters in random forest are either used to increase the predictive power of the model or to make the model faster. The hyperparameters of sklearns[1] built-in random forest function:

[1]sklearn is an indispensable part of the python machine learning toolkit. It is widely used across classification, predictive analytics, and many other machine learning tasks.

- Increasing the predictive power: $n\_estimators$ hyperparameter, which is the number of trees the algorithm builds before taking the maximum voting or taking the averages of predictions. In general, a higher number of trees increases the performance and makes the predictions more stable, but it also slows down the computation. Another important hyperparameter is $max\_features$, which is the maximum number of features random forest considers to split a node. The last important hyperparameter is min_sample_leaf, which determines the minimum number of leaves required to split an internal node.

- Increasing the model's speed: The n_jobs hyperparameter tells the engine how many processors it is allowed to use. If it has a value of one, it can only use one processor. A value of "-1" means that there is no limit. The $random\_state$ hyperparameter makes the model's output replicable. The model will always produce the same results when it has a definite value of $random\_state$ and if it has been given the same hyperparameters and the same training data. Lastly, there is the oob_score (also called oob sampling), which is a random forest cross-validation method. In this sampling, about one-third of the data is not used to train the model and can be used to evaluate its performance. These samples are called the out-of-bag samples.

3.2.5 Extra Trees Classifier

An extra trees classifier also known as extremely randomized trees is an ensemble machine learning algorithm that combines predictions from many decision trees. It is related to random forests. It uses a simpler algorithm to construct the decision trees than random forests do to use as members of the ensemble. Each decision stump will be built with the following criteria:

- All the data available in the training set is used to built each stump.

- To form the root node or any node, the best split is determined by searching in a subset of randomly selected features of size square root(number of features). The split of each selected feature is chosen at random.

- The maximum depth of the decision stump is one.

3.3 Artificial Neural Networks

ANNs are one of the most commonly used ML techniques. They are built based on inspiration from biological neural systems which have many small computing nodes, neurons, that are interconnected [41]. In a similar fashion, ANNs are made up of simple nodes or neurons connected by weights w (Figure 3.5).

An activation function θ determines the output y by computing the sum of the products of the weights w_{ij} and its input node values. A threshold or bias θ_j , is used to restrict the activation of a neuron, as shown in Figure 3.5. The output y of a neutron should be equal the target value t.

$$o_j = \varphi(\theta_j + \sum_i x_i w_i) \tag{3.2}$$

If it does not, the weights are adjusted as

$$w_{ij} = w_{ij} + \Delta w_{ij} \tag{3.3}$$

$$\Delta w_{ij} = \eta \delta_j x_i \tag{3.4}$$

where w_{ij} is the value of the connection weight from the unit in layer i to the unit in layer j, x_i is the value of input from the unit in layer i to the unit in layer j, η is the learning rate, Δw_{ij} is the value of the weight update for the connection weight from the unit layer i to the unit in layer j, δ_j is the derivative of error at the unit j with respect to weight w_{ij} .

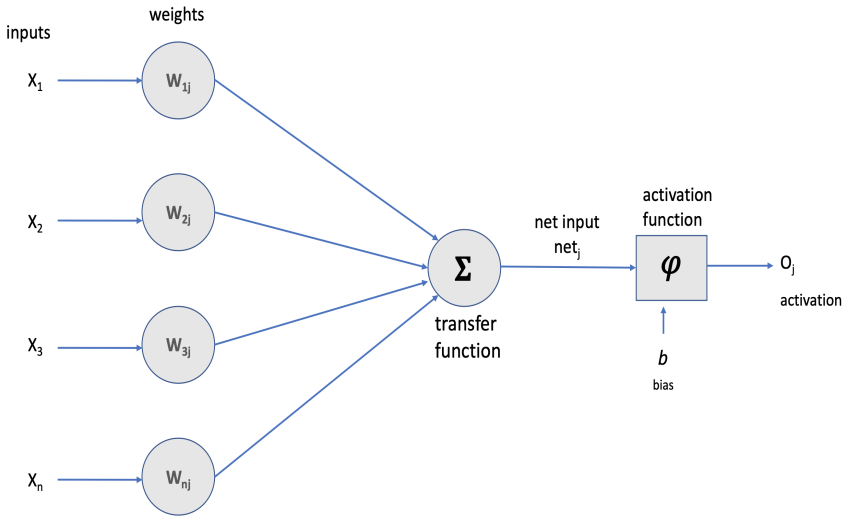

Figure 3.5: The structure of the artificial neuron [18]

3.3.1 Multi-Layer Perceptron

A Multi-Layer Perceptron (MLP), is a non-linear function developed by using multiple layers of neurons in an ANN model. In an MLP network, the output of a neuron in the first layer is used as the input for the next layer (Figure 3.6).

MLP does not allow interconnections within a layer or to previous layers, so it is an acyclic feed-forward graph. The layer between the input layer and the output layer is called the hidden layer. As the number of hidden nodes and the number of hidden layers are increased, the MLP is capable of modelling increasingly complex hypothesis spaces. However, with each additional layer it becomes increasingly challenging to adjust the lower weights of the network [24].

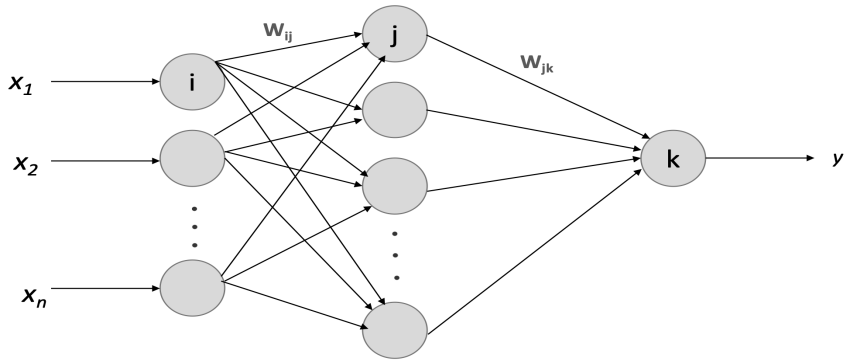

Figure 3.6: An architecture of Multi-layer perceptron (MLP) model [42]

3.4 Back-Propagation

Back-Propagation (BP) is a supervised algorithm that solves the challenge of updating weights in an MLP model. BP is one of the most important ANN learning algorithms. It aims to find the global minimum on an error surface to network node weights. In BP, weights are modified proportional to their contribution by using a gradient descent (GD) technique to minimize the error. The GD technique reduces the sum of squared error (SSE) over all given examples:

$$SSE = \frac{1}{2}\sum_{k=1}^{n}(t_k - y_k)^2 \tag{3.5}$$

where the output is the set of output nodes, and t_k and y_k are the given target values from the example and the output of the network for that target value, respectively [44]. The BP algorithm starts at the output layer and propagates the error through each node to the next lower layer and so on until reaching the input layer. The weights between the hidden and output layers are modified prior to modifying the weights between the input and hidden layers. The BP algorithm may get stuck in a local minimum as the weights are updated. To moderate this, the BP tunes parameters like the learning rate η and momentum α. The actual output of hidden neuron j for input signal i is given by:

$$h_j = f[\sum_i x_i w_{ij} + \theta_j] \tag{3.6}$$

where x_i is the input signal of the input neuron i, w_{ij} is the synaptic weight between the input neuron i and the hidden neuron j, θ_j is the bias of the hidden node j, and f is the activation function. Equation 3.6 shows the formula for the i^{th} weight, which relies on a continuous non-linear activation function for communication between layers [44].

$$\Delta w_{ij} = -\eta \frac{\delta E}{\delta w_{ij}} \tag{3.7}$$

where Δw_{ij} is value of the weight update for the connection weight from the unit in layer i to the unit in layer j, η is the learning rate, and δ is the derivative of error at unit j with respect to weight w_{ij} and error E. The formula to update the weight w_{ij} between nodes j and i for the current iteration n is:

$$\Delta w_{ij}(t) = \eta \delta_j x_i + \alpha \Delta w_{ij}(t-1) \tag{3.8}$$

where $\Delta w_{ij}(t-1)$ is the weight update of the previous iteration, δ_j is the derivative of the error at node j with respect to the weight w_{ij}, and x_i is the input over the connection associated with w_{ij}. The learning rate η controls the amount of weight update. A larger learning rate means a larger weight update while a smaller learning rate has smaller weight updates [44]. To understand the impact of momentum α, consider a ball in the physical world. When a ball rolls down a steep grade it will have a high momentum. This enables the ball to move through small dips in the ground's surface and carry on moving downhill. In a similar manner, momentum α in Equation 3.8 increases with the magnitude of the prior weight update and this prevents the BP from getting stuck in shallow local minima.

3.4.1 Overfitting and Underfitting models

Overfitting is possible in every machine learning problem [37]. A fundamental issue in machine learning is the tension between optimization and generalization. *Optimization* refers to the process of adjusting a model in such a way that the performance on the training data (the learning in machine learning) can be improved. *Generalization* refers to how well the trained model performs on data it has never seen (test data) [37].

At the beginning of training, optimization and generalization are correlated: the lower the loss on training data, the lower the loss on test data. This is when the model is said to be *underfit* and the network has not yet modeled

all relevant patterns in the training data. After a certain number of iterations on the training data, generalization stops improving and validation set [2] metrics begin to degrade. This means that the model is starting to *overfit* and starting to learn patterns that are explicit to the training data but that are misleading or irrelevant with regards to new data [37].

3.4.2 Preventing Overfitting

To keep a model from learning misleading or irrelevant patterns found in the training data, the best solution is to obtain additional training data. A model trained on more data will normally generalize better. If that is not possible, the next best solution is to adjust the quantity of data the models allowed to store. When a network can only afford to memorize a small number of patterns, the optimization process will force it to focus on the most prominent patterns that have better possibility of generalizing well. This process of battling overfitting is called *regularization*. One of the regularization technique used in this research were adding dropout layers.

Reducing the network's size

The most straightforward approach to prevent overfitting is to diminish the size of the model: the number of learnable parameters in the model (which is determined by the number of layers and the number of units per layer). In deep learning, the number of learnable parameters in a model is frequently referred to as the model's capacity [43]. Intuitively, a model with more parameters has a larger memorization limit and therefore can easily learn a perfect dictionary like mapping between training samples and their targets - a mapping with no generalization power.

Deep learning models in general are good at fitting the training data, yet the real test is generalization to an independent test set, not fitting. Bigger networks start overfitting almost immediately after just one epoch, and overfit more severely. Their validation loss is also noisier as the model is not properly regularized. A few possible reasons for the noisy validation loss are:

- Insufficient number of training samples.

[2] The fraction of training data used to evaluate loss.

- Large learning rates which leads stochastic gradient descent (SGD) to jump over the local minima. This would be an extreme case of "underfitting".

The possible solutions to reduce noise on validation loss are:

- Obtain more data.
- Alter the hyper-parameters.
- Implement weight regularization methods.

Adding weight regularization

A common method to mitigate overfitting is to put constraints on the complexity of a network by forcing its weights to accept only small values that make the distribution of weights more regular. This is called *weight regularization* [43]. This way we try to limit its flexibility, but also encourage it to build solutions based on multiple features. Two popular versions of this method are: L1 - Least Absolute Deviations (LAD) and L2 - Least Square Errors (LS).

- L1 regularization: A regression model that uses L1 regularization is called *Lasso* [3] *Regression*. The cost added is proportional to the absolute value of the weight coefficients (the L1 norm of the weights).
- L2 regularization: The model which uses the L2 regularization technique is called *Ridge Regression*. The cost added is proportional to the square of the value of the weight coefficients (the L2 norm of the weights). It is also called *weight decay*.

Adding dropout

Dropout is one of the most utilized regularization techniques for neural networks used in this research. Dropout applied to a layer means to arbitrarily drop out (set to zero) a number of output features of the layer during training. The dropout rate is the fraction of the features that are zeroed out and usually set between 0.2 and 0.5. While testing, no units are dropped out, instead, the layer's output values are scaled down by a factor equal to the dropout rate, to balance for the fact that more units are active than at training time.

[3] Least Absolute Shrinkage and Selection Operator

3.4.3 Activation function

An activation function is used to determine the output of neural network like yes or no. It maps the resulting values in between 0 to 1 or -1 to 1 etc. The activation functions can be divided into 2 types:

- **Linear activation function:** The linear activation is a simple unit which transforms the input as "y=w.x + b". These units have a linear behavior and the output will not be confined between any range (-infinity to infinity) [40].

- **Non-linear activation function:** The non-linear activation functions makes it easier for the neural networks to generalize with variety of data and differentiate between the output [40]. Some of the non-linear activation functions are:

  1. Rectified Linear Unit (ReLU)
  2. Sigmoid or Logistic Activation Function
  3. Tanh or hyperbolic tangent Activation Function

Sigmoid or Logistic activation function

Sigmoid function values range between 0 to 1. Therefore, it is used for models to predict the probability as an output. The function is differentiable [4]. The function is monotonic but function's derivative is not. The softmax function is a more generalized logistic activation function which is used for multiclass classification [40].

[4] we can find the slope of the sigmoid curve at any two points.

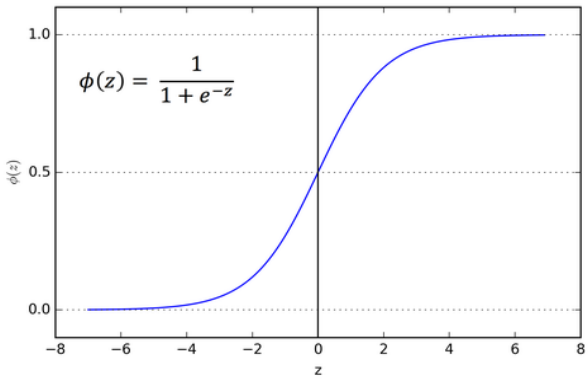

Figure 3.7: Sigmoid activation function [40]

Tanh or hyperbolic tangent activation function

The range of the tanh function is from -1 to 1. tanh is also sigmoidal (s - shaped). The advantage is that the negative inputs will be mapped strongly negative and the zero inputs will be mapped near zero in the tanh graph. The function is differentiable. The function is monotonic while its derivative is not monotonic. The tanh function is mainly used classification between two classes [40].

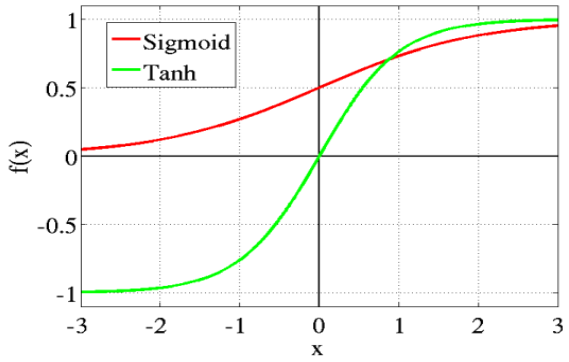

Figure 3.8: Tanh activation function [40]

ReLU (Rectified Linear Unit) activation function

The ReLU is the most used activation function in almost all the convolutional neural networks or deep learning [40]. The function and its derivative both

are monotonic [5]. ReLUs are half rectified (from bottom) as seen in Figure 3.9, the range is between "0" and "infinity". ReLU solves the vanishing gradient problem. If the input value is positive, the ReLU function returns it, if it is negative, it returns 0. The ReLU's derivative is 1 for values larger than zero. Because multiplying 1 by itself several times still gives 1, this addresses the vanishing gradient problem.

The ReLU function can be represented using the following equation:

$$y(x) = max(0, x) \tag{3.9}$$

In the above equation, "x" is the input tensor or variable and is not differentiable at x=0.

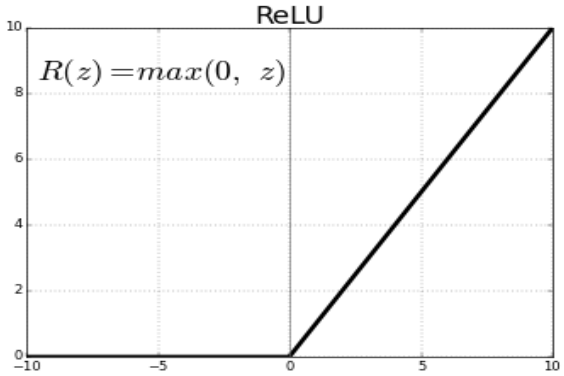

Figure 3.9: The rectified linear unit (ReLU) activation function [40]

3.4.4 Batch Normalization

Training deep neural networks with multiple layers can be challenging as they can be sensitive to the initial random weights and configuration of the learning algorithm. Batch normalization is a technique designed to standardize the inputs to a layer in a deep neural network. When implemented, batch normalization has the effect of accelerating the training process and in some cases improving the performance of the neural network. Sometimes, this layer reduces the number of training epochs required to train deep neural networks.

A batch normalization layer can be used at most points in a model and with most types of neural networks. This layer can be added to the model to

[5] A function which is either entirely increasing or decreasing.

standardize raw input variables or the outputs of a hidden layer, but cannot be used as an alternative for data preparation.

3.5 Deep Learning

Deep Learning (DL) is an area of ANN research where algorithms are developed that can learn representations of the input comprised of many hidden layers of neurons [20]. Each layer represents a certain level of abstraction, with each layer being more abstract than the layer below it.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made. Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly.

Deep learning distinguishes itself from classical machine learning by the type of data that it works with and the methods in which it learns [11]. Machine learning algorithms leverage structured, labeled data to make predictions. Deep learning eliminates some of data pre-processing that is involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction.

3.5.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a specialized type of neural network model designed for working with two-dimensional image data, although they can be used with one-dimensional and three-dimensional data.

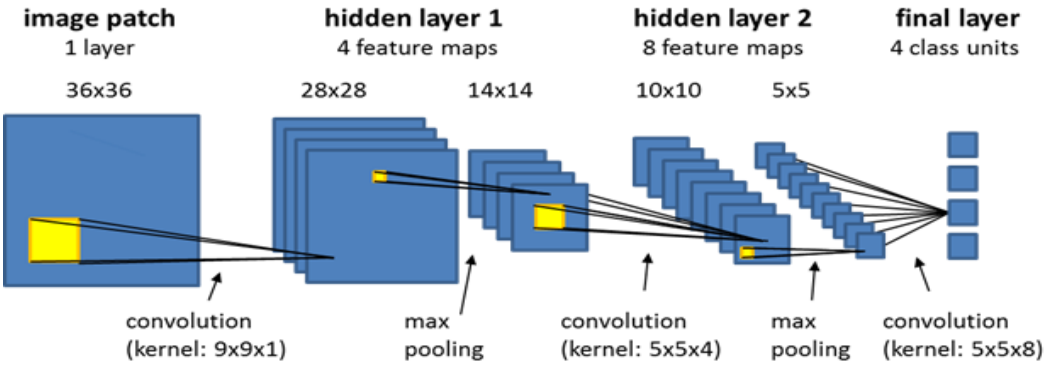

Figure 3.10: Convolutional neural networks[38]

Basics of Convolutional Neural Network

CNNs have the same functionality irrespective of their dimensionality. The only difference is the structure of the input data and how the filter, also known as convolutional kernel or feature detector, moves across the data. Each layer of CNNs (Figure 3.10) conduct different tasks.

- **Convolutional layer**

  It has two key parameters. One is the kernel size, and the other is the number of filters. The layer first divides the input into fixed-size patches that are the same size in all filters. A layer contains many filters with varying weight values, allowing each filter to learn as many features as possible. The network also applies an activation function. Since the network itself extracts features from the input, the layer outputs a series of feature maps which enter the next layer.

- **Pooling layer**

  It reduces the size of a feature map. Once the feature maps from previous convolutional layer enter a pooling layer, the network divides the input feature map into a fixed number of regions (determined by the pool size) and summarises the value in each region into a single maximum or average value. This is called max-pooling or average-pooling which can reduce model complexity and prevent overfitting while allowing some loss of information.

- **Dense layer**

  It is also known as fully connected layer and is the last part of the network. Following completion of all convolutional-pooling computations, the network arranges the values of final feature maps in a row. The final dense layer outputs the absolute error for regression using *relu* method.

Understanding Conv1D parameters

A 1-Dimensional convolutional layer (Conv1D) creates a kernel that passes over a single spatial (temporal) dimension to produce a tensor of outputs. The parameters in Conv1D layers are:

- filters: Integer, the dimensionality of the output space (i.e. the number of output filters in the convolution).
- kernel-size: An integer or tuple/list of a single integer, specifying the length of the 1D convolution window.
- activation function: It determines the output of a deep learning model, its accuracy and computational efficiency of training a model.
- kernel-initializer: A function applied to the kernel weights matrix. Used mainly to initialize all values prior to training [47].

3.5.2 Recurrent Neural Networks

Standard neural networks such as feed forward neural networks do not have memory to store what they learn. For every iteration, the network starts fresh as it does not remember the data in the previous iteration while processing the current set of data, which is a disadvantage when identifying correlations and data patterns. This is where recurrent neural networks (RNNs) come into picture. RNNs have a unique architecture that enables data to persist and models short term dependencies. So, RNN are neural networks that are designed for the effective handling of sequential data but are also useful for non-sequential data [39].

RNN Architecture

The input of an RNN must be three dimensional in the following format : batch size, the number of steps and the number of features. The number of

steps depicts the number of time-steps/segments we feed in one line of input in the entire batch of data to RNN. The RNN unit in tensorflow is called an "RNN cell". The RNN cell refers to the whole layer (not just a single cell) as the connections are recurrent and thus follow the "feeding to itself" approach. The RNN layer is comprised of single rolled layer that unrolls according to "number of steps" provided.

Tensorflow is the primary open source software library for dataflow and differentiable programming across a range of tasks. We use mainly for: classification, perception, understanding, discovering, prediction and creation of the models.

As mentioned earlier, RNNs have the special ability to model short term dependencies due to their hidden state. They retain information from one time step to another flowing through the unrolled RNN units. Each unrolled RNN unit has a hidden state. The current time step's hidden state is calculated using information from the previous time step's hidden state and current input. This helps in retaining information on what the model saw in the previous time step when processing the current time step's information. Also, all the connections in RNN have weights and biases (optional in some architectures).

When first feeding the data into RNN, it will have a rolled architecture as shown in Figure 3.11:

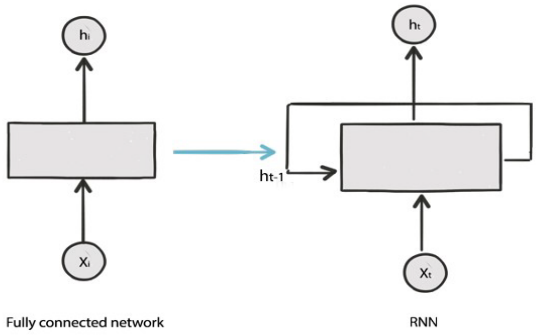

Figure 3.11: RNN:A Simple Architecture[17]

When the RNN starts to process the data, it will unroll and produce outputs as shown in Figure 3.12:

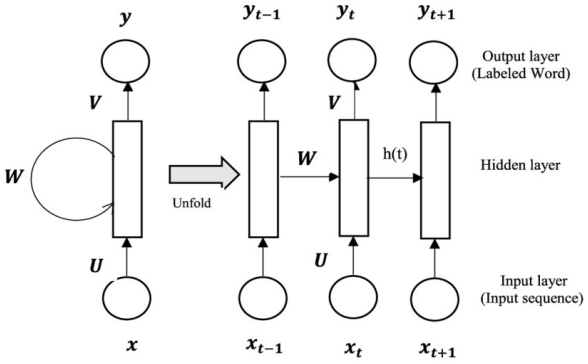

Figure 3.12: RNN:Unfolded Architecture with single hidden layer[7]

Vanishing Gradient Problem

Training neural networks involves the addition of many layers as this increases the capacity of network [15]. But a problem with training networks with many layers (e.g. deep neural networks) is that the gradient diminishes as it propagates backward through the network. The error might have very little effect on the model as it is so small and this problem is refered to as the "vanishing gradient" problem. Sometimes, this error can be unstable and also explode, where the gradient exponentially increases as it propagates backward through the network. This is referred to as the "exploding gradient" problem [15].

Vanishing gradients are particularly common in RNNs as the update of the network involves unrolling the network for each time step (Figure 3.12). Training continuously does not improve the performance of the ML models [15].

To overcome the vanishing gradient problem, several methods have been proposed. Some of them are:

- Using Long Short-Term Memory networks(LSTMs) [35].
- Using activation functions such as a rectified linear unit (ReLU), as this function allows more gradient to flow backward through the model during training which improves performance of the model.

3.5.3 Gated Recurrent Unit

Gated Recurrent Unit is a type of Recurrent Neural Network that addresses the issue of long term dependencies which can lead to vanishing gradients larger

vanilla RNN networks experience. To solve the vanishing gradient problem of a standard RNN, GRU uses update gate and reset gate. These are two vectors which decide what information should be passed to the output. GRUs address this issue by storing "memory" from the previous time point to help inform the network for future predictions [23].

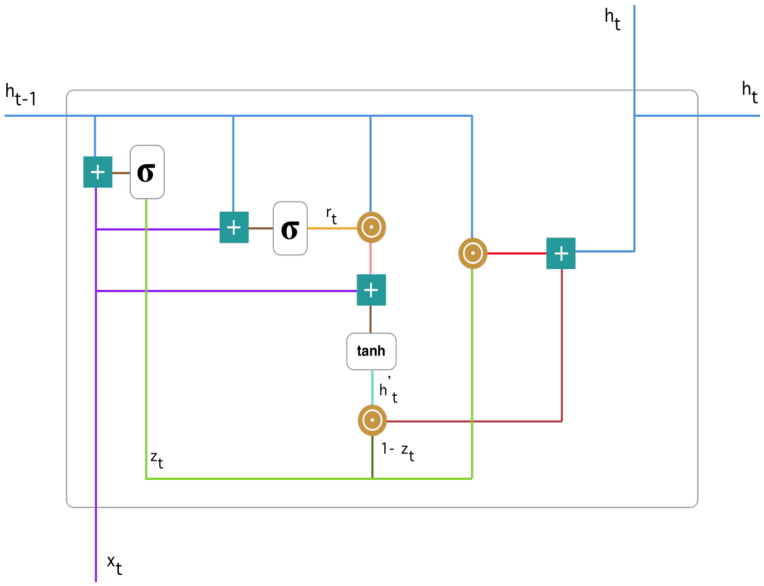

Figure 3.13: Gated Recurrent Unit[36]

Update gate

Update gate calculates z_t for time step t using the formula:

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{(t-1)}) \tag{3.10}$$

When x_t is plugged into the network unit, it is multiplied by its own weight $W^{(z)}$. $h_{(t-1)}$ which holds the information for the previous $t-1$ units and is multiplied by its own weight $U^{(z)}$. Both results are added together and a sigmoid activation function is applied to squash the result between 0 and 1.

The update gate helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future. This is important because the model can decide to copy all the information from the past and eliminate the risk of vanishing gradient problem.

Reset gate

This gate is used from the model to decide how much of the past information to forget. To calculate it, we use:

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{(t-1)}) \tag{3.11}$$

We plug in $h_{(t-1)}$ and x_t, multiply them with their corresponding weights, sum the results and apply the sigmoid function.

Current memory content

A new memory content which will use the reset gate to store the relevant information from the past. It is calculated as follows:

$$\tilde{h_t} = tanh(Wx_t + r_t \bigodot Uh_{(t-1)}) \tag{3.12}$$

1. Multiply the input x_t with a weight W and $h_{(t-1)}$ with a weight U.
2. Calculate the Hadamard (element-wise) product between the reset gate r_t and $Uh_{(t-1)}$.
3. Sum up the results of step 1 and 2.
4. Apply the nonlinear activation function tanh.

Final memory at current time step

1. Apply element-wise multiplication to the update gate z_t and $h_{(t-1)}$.
2. Apply element-wise multiplication to $(1 - z_t)$ and $\tilde{h_t}$.
3. Sum the results from step 1 and 2.

$$h_t = z_t \bigodot h_{(t-1)} + (1 - z_t) * \tilde{h_t} \tag{3.13}$$

3.5.4 Long Short-Term Memory Networks

Long Short-Term Memory networks (LSTMs) is a type of RNNs, which are capable of learning long-term dependencies and they work effectively on a large variety of problems. LSTMs remember information for a long period of time and are designed explicitly to solve long-term problems. LSTMs have similar structure though the internals have different components when compared to a single tanh (activation) layer in RNN. The four layers in the architecture interact with each other.

The cell state C allows information to flow through the entire LSTM unchanged, which enables the LSTM to remember context for a long period of time (See Figure 3.14). The horizontal line has several inputs and outputs which is controlled by *gates* that allows information to be added to or removed from the cell state. The sigmoid layers output numbers between 0 and 1, describing how much should be let through from each component. An LSTM has three of these gates to control the cell state.

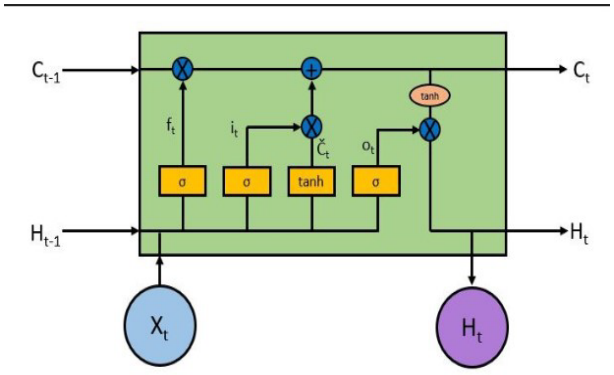

Figure 3.14: Long Short-term Memory[5]

Forget Gate

This gate decides what information should be thrown away or kept. Information from the previous hidden state and information from the current input is passed through the sigmoid function. This gate return a value between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep. The forget state equation is:

$$f_t = \sigma(W_{xf}x^{(t)} + W_{hf}h^{(t-1)} + b_f) \tag{3.14}$$

Input Gate

To update the cell state, we have the input gate (i_t). We pass the previous hidden state and current input into a sigmoid function. This decides which values will be updated by transforming the values to be between 0 and 1: 0 means not important, and 1 means important. The hidden state and current input are also into the tanh function to squish values between -1 and 1 to help regulate the network. The tanh output is then multiplied with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output. The input state equation is:

$$i_t = \sigma(W_{xi}x^{(t)} + W_{hi}h^{(t-1)} + b_i) \tag{3.15}$$

Output Gate

The output (o_t) gate decides what the next hidden state (contains information on previous inputs) should be. First the previous hidden state and the current input are passed into a sigmoid function and then the modified cell state is passed to the tanh function. The tanh output is multiplied with the sigmoid output to get the required output (information a hidden state should carry). The output state equation is:

$$o_t = \sigma(W_{xo}x^{(t)} + W_{ho}h^{(t-1)} + b_o) \tag{3.16}$$

3.5.5 Bi-directional Long Short-Term Memory Networks

A Bi-directional LSTM, or BDLSTM, is a sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction. BDLSTMs effectively increase the amount of information available to the network, improving the context available to the algorithm.

BDLSTM adds one more LSTM layer, which reverses the direction of information flow. It means that the input sequence flows backward in the additional LSTM layer. Then it combines the outputs from both LSTM layers in several ways, such as average, sum, multiplication, or concatenation.

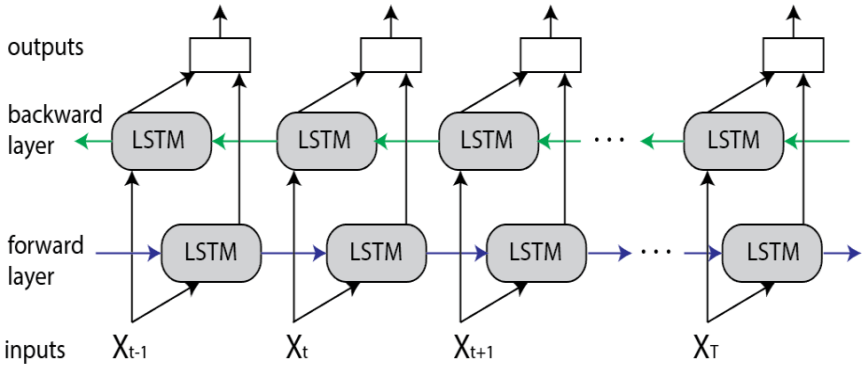

Figure 3.15: Bi-directional long short-term memory[1]

3.6 Regression Analysis

Regression analysis is the process of estimating the relationship between a dependent variable and independent variables. Regression analysis is one of the most basic tools in the area of machine learning used for prediction. Using regression we can fit a function to the available or training data and try to predict the outcome for the future or hold-out data points. Regression serves two purposes.

- To estimate missing data within the training data range (Interpolation)
- To estimate future data outside the training data range (Extrapolation)

3.6.1 Logistic Regression

Logistic regression is a process of modeling the probability of a discrete outcome given an input variable. The most common logistic regression models is a binary outcome; something that can take two values such as true/false or yes/no.

Logistic regression is a simple and efficient method for binary and linear classification problems, which achieve very good performance with linearly separable classes. It is an extensively employed algorithm for classification in industry. The logistic regression model, like the Adaline and perceptron can be generalized to multiclass classification. Scikit-learn has a highly optimized version of logistic regression which supports multiclass classification task [19].

To create a probability, we will pass z (input features) through the sigmoid function, $\sigma(z)$. The sigmoid function is also called the logistic function, and gives logistic regression its name. The sigmoid has the following equation,

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.17}$$

The sigmoid function $\sigma(z)$ takes a real value and maps it to the range [0,1]. It is nearly linear around 0 but outlier values get squashed toward 0 or 1.

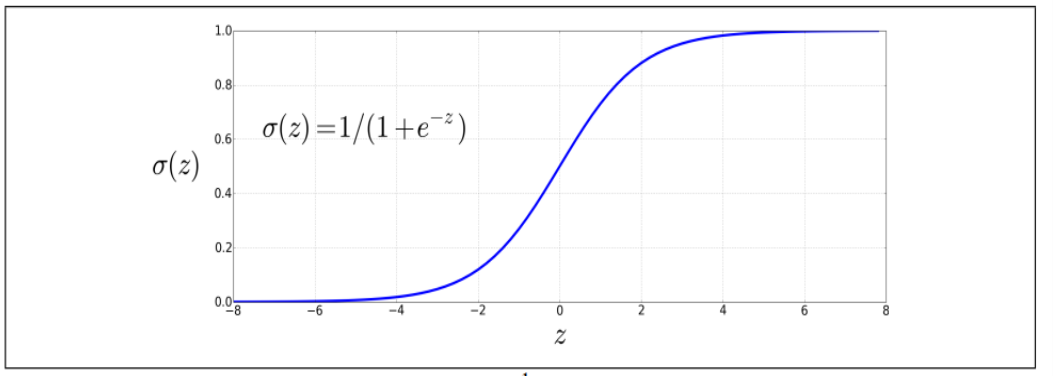

Figure 3.16: Sigmoid function[19]

3.6.2 LASSO Regression

LASSO or Least Absolute Shrinkage and Selection Operator regression regularize against overfitting on the training data points and enforces sparsity on the learned weights.

L1 Regularization

If a regression model uses the L1 regularization technique, then it is called Lasso Regression. L1 regularization adds a penalty that is equal to the absolute value of the magnitude of the coefficient. This regularization type can result in sparse models with few coefficients. Some coefficients might become zero and get eliminated from the model. Larger penalties result in coefficient values that are closer to zero.

The LASSO model can be shown as:

$$\sum_{i=1}^{n}(y_i - \sum_{j} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{3.18}$$

Where, β is a vector of coefficients, λ denotes the amount of shrinkage. If $\lambda = 0$ implies all features are considered and it is equivalent to the linear regression where only the residual sum of squares is considered to build a predictive model. If $\lambda = \infty$ implies no feature is considered i.e, as λ closes to infinity, it eliminates more and more features. The bias increases with increase in λ and the variance increases with decrease in λ.

3.6.3 ElasticNet Regression

ElasticNet Regression is used to address the limitations of LASSO regression such as it is non-convex nature. It uses a penalty function based on:

$$||\beta||_1 = \sum_{j=1}^{p} |\beta_j| \tag{3.19}$$

Use of this penalty function has several limitations [48]. For example, if there is a group of highly correlated variables, then LASSO tends to select one variable from a group and ignore the others. To overcome these limitations, the elastic net adds a quadratic part to the penalty ($||\beta||^2$), where β is a vector of coefficients, which when used alone is ridge regression [6]. The estimates from the elastic net method are defined by:

$$\beta \equiv argmin_{\beta}(||y - X\beta||^2 + \lambda_2||\beta||^2 + \lambda_1||\beta||_1) \tag{3.20}$$

The quadratic penalty term makes the loss function strongly convex, and it therefore has a unique minimum. The elastic net method includes the LASSO and ridge regression, in which each of them is a special case where $\lambda_1 = \lambda$, $\lambda_2 = 0$ or $\lambda_1 = 0$, $\lambda_2 = \lambda$. The naive version of elastic net method finds an estimator in a two-stage procedure: For each fixed λ_2 it finds the ridge regression coefficients, and then does a LASSO type shrinkage. This type of estimation incurs a double amount of shrinkage, which leads to increased bias

[6] Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where linearly independent variables are highly correlated.

and poor predictions. To improve the prediction performance by rescaling the coefficients of the naive version of elastic net by multiplying the estimated coefficients by $(1 + \lambda_2)$ [48].

3.6.4 Gradient Boosting Regression

Gradient boosting approach can used for both regression and classification problems. These techniques generate a prediction model in the form of a series of weak prediction models, usually decision trees. Three components are involved in gradient boosting:

- A loss function to be optimized.
- A weak learner to make predictions.
- An additive model to add weak learners to minimize the loss function.

3.6.5 XGBoost for Regression

XGBoost is a powerful approach for building supervised regression models. The validity of this statement can be inferred by knowing about its (XGBoost) objective function and base learners. The objective function contains a loss function and a regularization term. The most common loss functions in XGBoost for regression problems is $reg : linear$, and that for binary classification is $reg : logistics$.

Ensemble learning involves training and combining individual models to get a single prediction, and XGBoost is one of the ensemble learning methods. XGBoost expects to have the base learners which are uniformly bad at the remainder so that when all the predictions are combined, bad predictions cancels out and better one sums up to form final good predictions.

The output of regression is either continuous or real values. There are several metrics involved in regression like root-mean-squared error (RMSE) and mean-squared-error (MAE). They are defined as follows:

- RMSE: Root Mean Square Error is the square root of the average of the squared differences between the estimated and the actual value of the variable/feature.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Predicted_i - Actual_i)^2} \tag{3.21}$$

- MAE: It is an absolute value of the sum of actual and predicted differences.

$$MAE = (\frac{1}{n}) \sum_{i=1}^{n} |y_i - x_i| \tag{3.22}$$

3.6.6 Light Gradient Boosting Model

A Gradient Boosting Decision Tree or a GBDT is a very popular machine learning algorithm that has effective implementations like XGBoost and many optimization techniques are actually adopted from this algorithm [31]. The main features of the LGBM model are as follows :

- Higher accuracy and a faster training speed.
- Low memory utilization.
- Comparatively better accuracy than other boosting algorithms and handles overfitting much better while working with smaller datasets.
- Parallel learning support.
- Compatible with both small and large datasets.

Chapter 4

Data Preparation and Model Architecture

This chapter provides the data source and details the data preparation technique used in the thesis.

4.1 Data Source

The data used in the experiments is collected based on the interactions of constituents with clients of *Fundmetric*, a Halifax based company whose objective is to help non-profit organizations raise more money by focusing on turning one time donors into lifetime supporters. *Fundmetric* works with organizations such as universities and disease related charities. They create personalized emails and develop donor profiles based on their interaction with the software. This approach generates a huge amount of data, more than the competing communication platforms, which is provided to machine learning algorithms to help achieve the objective of this research.

According to the Fundraising Effectiveness Project [4], more than 50% of all donors stop donating after one year. This costs massively for the charities to sustain the level of operations as they have to invest in new donor acquisition. *Fundmetric* has been innovative in their approach to generate data needed by charities to identify the most engaged donors as opposed to depending on siloed information (features directly provided from donor data) and demographic data.

The major donor data generated by *Fundmetric* is based on constituent

interaction with charities. For our experiments, we collected data from 8 charities as shown in the Table 4.1.

Representation	**Type of charity**
AlzC	Alzheimer's charity
CC	Cancer charity
EC-1	Educational charity-1
EC-2	Educational charity-2
EC-3	Educational charity-3
EC-4	Educational charity-4
RC-1	Religious charity-1
RC-2	Religious charity-2

Table 4.1: Data sets from various charities.

These data sets have fewer major donors than non-major donors as seen in Table 4.2. This means the major donor data is heavily skewed towards non-major donors and needs to be balanced before training a model [25]. As some of the features are categorical, we use one-hot encoding to convert them to numerical format.

	AlzC	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
Major Donors	46	82	2080	4393	658	3226	1856	309
Non-Major donors	90859	52123	104677	76155	54121	211519	64843	101459

Table 4.2: Number of samples of each type in each data set.

4.1.1 Data Set Used

Charities have been gathering data on their constituents for tax purposes, but there is also a database that can be used to find future donations. This information includes the constituent's address, as well as the donation amount and date. As data analysis and machine learning technologies become more common, charities recognize the value of data and begin to collect more data to help differentiate between constituents. This data can be broken down roughly into the following categories:

Demographic data

Demographic data include age, gender, income, and job title, but most charities do not keep track of these values for many constituents. Instead, address information can be used to infer some of this information, and the method of request is recorded to determine which solicitation methods and modes of communication are acceptable to a constituent. Machine learning algorithms can learn to distinguish constituents from each other based on this data, but more is needed. Below is a list of subset features with name and description for demographic data.

- Prefix - The constituent's prefix.
- Gender - The constituent's gender.
- Address - Address of residence.
- Age - How old is a constituent.
- Contact Email - Does the constituent allow email contact?
- Contact Phone - Does the constituent allow phone contact?
- City - The constituent's city on mailing address.
- Province - The constituent's province on mailing address.
- Country - The constituent's country on mailing address.

Donation data

Donation data is recorded by charities not only for receipting purposes, but also to track revenue. Donation dates and amounts can aid machine learning algorithms, but they do not directly provide trend data to them. As a result of these two simple features, we create new donation features such as largest gift, smallest gift, and date of last donation. Below is a list of subset features with name and description for donation data.

- Maximum Donation - Biggest donation made by constituent.
- Minimum Donation - Smallest donation made by constituent.
- Donation Count - Number of donations made be constituent.

- Total Donations - Lifetime giving of constituent.
- Average Donation - Average of gift given by constituent.
- Standard Deviation - Standard deviation of gifts given by constituent.
- Variance - Variance of gifts given by constituent.
- Donation Lifetime - Total number of days from the constituent's first gift to last gift.
- Best Donation Type - Method of payment of largest donation made by the constituent.
- Span Frequency - How often they gave while they were giving.
- True Frequency - How often they gave since they started giving.
- Days Since First/Last Donation - Number of days since first or last donation.
- LYBUNT - Donor who gave Last Year But Unfortunately Not This Year.
- SYBUNT - Donor who gave Some Year But Unfortunately Not This Year.

Educational data

University foundations benefit from a more in-depth understanding of their constituents' activities while they were students. These foundations use club memberships, degree numbers, and graduation dates to determine what materials to send to their alumni, and machine learning can use these features as well. Below is a list of subset features with name and description for education data.

- Latest Graduation Year - Year the alumni finished their (last) degree.
- Latest School - What school the alumni (last) graduated from.
- Latest Degree Type - Type of degree/diploma (last) earned.
- Latest Degree Code - Institution specific code for (last) degree.
- Latest Major Code - Secondary category of (last) degree.

Behavioural data

The interactions of constituents with a given charity are frequently not recorded by the charity, but they can be an indicator of future giving. Whether a constituent attends events, volunteers, opens charity emails, or watches charity videos, both humans and computers can learn how much affinity a constituent has for a charity. Below is a list of subset features with name and description for behavioral data.

- Number of Emails Received - Number of emails sent that were actually received.
- Percent Emails Opened - What percentage of emails does the constituent open?
- Total Opens - Total number of email opens overall.
- Number of Clicks - Number of links clicked in all emails.
- Clicks per open - How many links are click when an email is opened?
- Clicks per received - How many links are clicked per received email?
- Total Engagement - Total number of engagement records.
- Unique Engagement Type Count - Total number of unique engagement types performed.
- Top Engagement Type - Most performed engagement "type" (name).
- Unique Engagement Service Count - Total number of unique engagement services used.
- Top Engagement Service - Most performed "service" used.
- Unique Engagement Action Count - Total number of unique actions performed.
- Top Engagement Action - Most performed action.
- Total video engagement - Total number of engagement records.

- Number of donation page views - Number of times the constituent viewed one of our donation portal pages.
- Number of video page views - Number of times the constituent viewed one of our video portal pages.
- Days Since Last Interaction - Number of days since last interaction in any form.

Data of these four types can be used to analyze charities' history, but more importantly in this research, to help predict who is likely to give a major gift. In the next chapter, we describe how we use this data to provide charities with accurate lists of potential major donors, so that they can focus their time on developing a relationship with potential major donors instead of trying to build useful lists of potential donors themselves.

4.2 Data Preparation

One of the first things to do is check the shape and size of the data. We feed the data in the form of comma separated value (CSV) files to machine learning models whose dimension along the X-axis is the number of constituents and the dimension along the Y-axis is the number of features. Once the dataset has been loaded, the next step is to check for any missing values, as described in the next section.

There are two different datasets fed to the machine learning model, *major donors* (data which has major donations made by constituents) and *non-major donors* (data where no major donations are made by constituents). The non-major donors data have more samples (negatives) compared to major donor data (positives) which makes the data unbalanced.

As the data in Table 4.2 are unbalanced, we balance the data of major donors and non-major donors and then split into train (70%) and test (30%) to feed it to the model and calculate the accuracy, precision, recall, and F1-score. We first over-sample the major donors dataset and then balance using the following approach.

We have over-sampled the minority class using synthetic minority over-sampling technique (SMOTE). We define a SMOTE instance with default parameters that will balance the minority class and then fit and apply it in one step to create a transformed version of the dataset. Once transformed, we

summarize the class distribution of the new transformed dataset, which would now be balanced through the creation of many new synthetic examples in the minority class.

The quality and size of data used are the key factors to determine the performance of a machine learning model. So, it is important to examine and preprocess the data set before feeding it to the machine learning algorithm. The essential data preprocessing techniques used in this research were:

- Cleaning data by removing missing values for data set.
- Data transforms where attributes are scaled in order to best expose the structure of the problem later to learning algorithms.

4.2.1 Dealing with missing values in dataset

It is common in real-world applications to have one or more values missing in the samples for different reasons. This could have happened due to error in data collection process or particular fields could have been simply left blank and are represented as NaN (not a number) or *null*. Most statistical modeling are unable to handle missing values and may produce unpredictable results if not taken care of. In this research, all the null values are replaced with *zero* because with neural networks, it is safe to input missing values as *zero*, with the condition that *zero* is not already a meaningful value. The network will learn from exposure to the data that the value *zero* means missing data and will start ignoring the value.

4.2.2 Handling Categorical Data

Categorical data is also known as nominal and ordinal data. Categorical data expresses an attribute that cannot be measured and it assumes values in a finite or infinite set of values, often named levels. Many machine learning algorithms require that categorical features/columns are encoded as integer values.

The features used in the experiments are listed in Section 4.1. Some features, such as "title" (e.g.,"Ms") are *nominal* and thus need to be transformed for most machine learning algorithms. We use one-hot encoding and create a new feature for every value of each nominal feature, with exactly one of these newly created features having value 1 for each parent feature, and the rest of the values being 0.

4.2.3 Handling Giveaway features

Giveaway features are features that can tell machine learning algorithms immediately whether someone is a major donor. Intuitively, we would want to use these features in our predictions to increase the accuracy of a model differentiating between major donors and non-major donors, but a perfectly accurate model (one that correctly classifies each major donor and non-major donor) is not useful in terms of finding *potential* donors, since all non-major donors will be classified as non-major donors by this model.

Instead, we seek accurate, but not perfect, models capable of identifying non-major donors who resemble major donors and that are thus worth approaching. Giveaway features for major giving include *maximum donation*, *average donation*, *intercept*, *slope*, *total donations* and *standard deviation of gifts*, since values of these that are larger than the major giving threshold for a charity can immediately reveal to a model who the major donors are, and thus create a perfect, yet useless, model. We remove giveaway features from the data in order to build useful models.

4.2.4 Hardware and Software Environment

The following are the hardware and software configurations used for model development and testing.

Hardware configuration for local computer:

- CPU - Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz
- GPU - Intel(R) UHD Graphics
- RAM - 8GB

Software configuration:

- Python 3.8.5
- TensorFlow 2.4.1
- Keras 2.4.3
- Numpy 1.19.2

- Matplotlib 3.3.2
- Scikit-Learn 0.24.2

The model is trained and tested on Jupyter Notebook 6.1.4 (Anaconda IDE).

Chapter 5

Theory and Approach

In this chapter, we describe the methods and approaches used in our experiments to build the machine learning models. The data used to implement these ML approaches are described in Chapter 4 and the empirical studies in Chapter 6.

5.1 Problem Refinement

The goal of this study is to generate a list of major donors so that major gift officers can focus their efforts on the most likely major donor constituents. We use machine learning algorithms in order to try to accurately model major giving, so that we can feed a model a constituent and get an accurate idea of whether that constituent is likely to give a major gift.

The objectives of this research are:

- To build a model which accurately predicts major donor prospects for charities.

- To build a model that predicts how much money major donor constituents will contribute to the charity.

5.2 Approach

For this study, we used real charitable data, such as demographic, donation, educational, and behavioural data. This data was fed into machine learning algorithms and tested on the non-major donor data in order to predict who will become future major donors.

We were able to extract information about the variables using our understanding from the trained machine learning models and extensive exploratory data analysis (EDA) [33]. EDA also aided in the statistical analysis, which was then followed by the machine learning (ML) pipeline. It was also necessary to merge major and non-major donors to see the differences in the patterns/behavior of the donors. In the step of data processing, we need to prepare our data in specific ways before feeding into a machine learning model. One of the examples is to perform a one-hot encoding on categorical data since, a lot of algorithms cannot work directly with categorical data. Therefore, we need to convert categorical data into a numerical form and our machine learning algorithm can take in that as input.

We also experiment with only the donation and behavioural data and removing giveaway features, demographic, and educational data to see how well the models predict. Furthermore, we forecast how much money major donor constituents will contribute to the charity.

5.3 Machine Learning Approaches

To achieve the objectives presented above, we built deep learning models that learns from historical data. Here, the data is technically not time series, neither sequential, these are individual observations LSTMs/RNNs is good for time series data or sequential data as they learn the previous states of data, but these can be used for more such as we are using them for classification. As seen in [9], RNN performed well on non-sequential data with highest accuracy when compared to other machine learning models. Our data is observations which are not depending on time, there order does not matter, and one is totally independent of others. It is like independent records. The following section describes the deep learning models used for performing various experiments using the LSTMs, RNNs, CNNs, GRUs and BDLSTM algorithms.

Input dimension to Deep learning models

For all the machine learning models used in this research, the performance was evaluated using a confusion matrix, accuracy, precision, recall and F1-score. One of the input data[1] provided to ML models is from an Educational Charity-1 which has 2080 positive samples and 104677 negative samples that looks back at 525 columns/features after one-hot encoding of the original features.

This input data has to be balanced as there are more negative samples than positive samples (Table 4.2). After oversampling the dataset (Table 4.2) using SMOTE technique, the shape of training and testing data is 4160 $\times$ 525. Next, we used the *train_test_split* [2] function to split data arrays into two subsets for training and testing data, then printing the size of the new dataset which is transformed into 2675 $\times$ 525 (samples, features) and 1485 $\times$ 525. We set the train size to 70% and test size to 30%. The input dimension for all the models used must be three-dimensional, so, we reshape 2675 $\times$ 525 and 1485 $\times$ 525 to 2675 $\times$ 1 $\times$ 525 (samples, time steps, features) and 1485 $\times$ 1 $\times$ 525. To evaluate the performance of the model, the experiments were conducted by altering the network parameters of the models used in this research.

The input dimension mentioned in this section is provided to the ML models to perform various experiments seen in Chapter 6.

The RMSProp [3] and Adam [4] optimization technique were used to train the networks for all the experiments and the network was trained on various charities data (Table 4.1) for 100 epochs. RMSprop learning rate of 0.001 were used because it exponentially decays average of squared gradients and a learning rate of 0.01 for Adam because, in addition to exponentially decaying average of past squared gradients like RMSprop, it also stores an exponentially decaying average of past gradients, and it is one of Keras default parameters.

[1] The input data used is from all the charities mentioned in Table 4.1

[2] The *train_test_split* procedure is used to estimate the performance of machine learning algorithms to make predictions on data not used to train the model.

[3] RMSprop is a gradient-based optimization technique used in training neural networks This normalization balances the step size (momentum), decreasing the step for large gradients to avoid exploding and increasing the step for small gradients to avoid vanishing.

[4] Adam is an adaptive learning rate method, it computes individual learning rates for different parameters. Its name is derived from adaptive moment estimation.

5.3.1 LSTM-GRU

The Long Short-Term Memory-Gated Recurrent Unit (LSTM-GRU) architecture is comprised of a sequence of long short-term memory, gated recurrent unit, activation, dense, and dropout layers. LSTM-GRU uses fewer training parameters and therefore uses less memory and executes faster with more accurate results on a larger dataset when compared to supervised learning models. The models were programmed in python, using the frameworks TensorFlow and Keras. For training, a batch size of 64 was used.

The input data was transformed into 3D matrix form which is maintained throughout the various layers of network. The network used for the experiments consists of two long short-term memory layers, 1 gated recurrent unit layer with fully connected dropout hidden layer. The dense layers were placed at the output end of the network, concluding with an output layer utilizing a *ReLU* and *Sigmoid* activation functions as seen in the Figure 5.1.

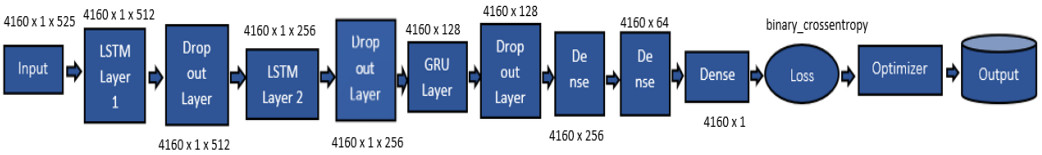

Figure 5.1: LSTM-GRU Network architecture used in experiments[5].

After performing numerous experiments by varying network configurations and learning parameters, the best architecture was an LSTMGRU model that consists of two LSTM layers, 1 GRU Layer, 3 dropout layers and a fully connected block (Figure 5.1).

LSTM(512) reads the input data and outputs 512 features with 1 timestep for each because $return\_sequences = True$. LSTM(256), takes the 1x512 input from Layer 1 and reduces the feature size to 256. Since $return\_sequences = True$, it outputs a feature vector of size 1x256. GRU(128), $return\_sequences = False$ it returns the last output.

5.3.2 SimpleRNN

The SimpleRNN architecture is comprised of a sequence of SimpleRNN, activation, dropout and dense layers. The input data was transformed into a 3D matrix form which is maintained throughout the various layers of network.

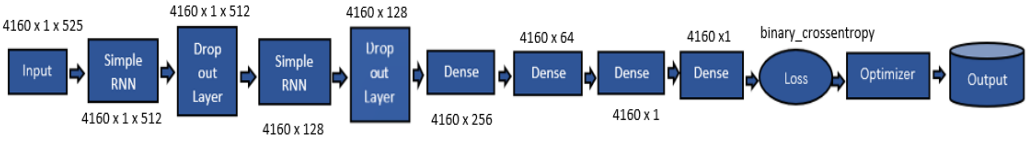

Figure 5.2: SimpleRNN Network architecture used in experiments[7].

After performing numerous experiments by varying network configurations and learning parameters, the best architecture was a SimpleRNN model that consists of 2 SimpleRNN blocks, 2 dropout layers and a fully connected block (Figure 5.2) which is used for all the experiments in Chapter 6.

The initial SimpleRNN layer consisted of 512 units (dimensionality of output space) which reads the input data and outputs 512 features with 1 timesteps for each because $return\_sequences = True$ and $kernel\_initializer = glorot\_normal$, a dropout layer and second SimpleRNN layer with 128 units followed by a dropout layer and a fully connected dense layer.

5.3.3 GRU

The Gated Recurrent Unit (GRU) architecture is comprised of a sequence of GRU, activation, dropout and dense layers. The input data was transformed into 3D matrix form which is maintained throughout the various layers of network. The network used for the experiments consists of two gated recurrent unit layers and two dropout layers. The dense layers were placed at the output end of the network, concluding with an output layer utilizing a $ReLU$ and $Sigmoid$ activation functions as seen in the Figure 5.3.

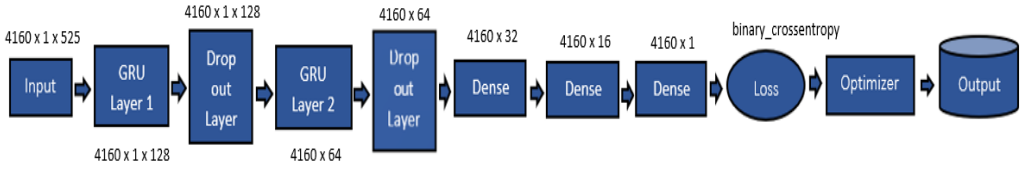

Figure 5.3: GRU Network architecture used in experiments[34].

After performing numerous experiments by varying network configurations and learning parameters, the best architecture was an GRU model that consists

of 2 GRU blocks, 2 dropout layers and a fully connected block (Figure 5.3) which is used for all the experiments in Chapter 6.

The initial GRU layer consisted of 128 units which reads the input data and outputs 128 features with 1 timesteps for each because $return\_sequences = True$ and $kernel\_initializer = glorot\_normal$, a dropout layer and second GRU layer with 64 units followed by a dropout layer and a fully connected dense layer.

5.3.4 BDLSTM-GRU-TDL

The Bidirectional Long Short Term Memory-Gated Recurrent Unit-Time Distributed Layer (BDLSTM-GRU-TDL) architecture is comprised of a sequence of BDLSTM, GRU, TDL, activation, dropout and dense layers. The input data was transformed into 3D matrix form which is maintained throughout the various layers of network. The network used for the experiments consists of 1 bidirectional long short term memory layer, 1 gated recurrent unit layer, 1 time distributed layer with fully connected *ReLU* hidden layer and two dropout layers. One of the dense layers were placed at the output end of the network, concluding with an output layer utilizing a *Sigmoid* activation function as seen in the Figure 5.4.

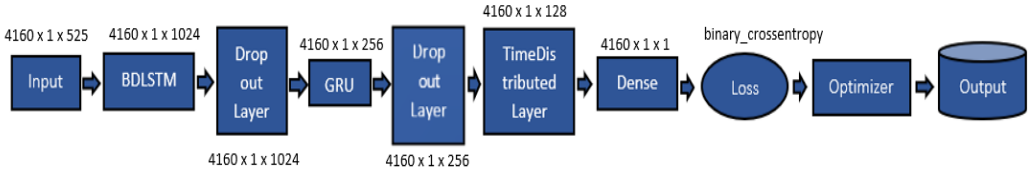

Figure 5.4: BDLSTM-GRU-TDL Network architecture used in experiments[34].

After performing numerous experiments by varying network configurations and learning parameters, the best architecture was an BDLSTM-GRU-TDL model that consists of 1 BDLSTM layer, 1 GRU layer, 1 TDL layer, 2 dropout layers and a fully connected block (Figure 5.4) which is used for all the experiments in Chapter 6.

The initial BDLSTM layer consisted of 512 units which reads the input data and outputs 1024 features with 1 timesteps for each because $return\_sequences = True$ and $kernel\_initializer = glorot\_uniform$, a dropout layer and followed by a GRU layer with 256 units, a dropout layer and TDL layer with 128 units

with fully connected *ReLU* hidden layer, a fully connected dense layer with *Sigmoid* activation function.

5.3.5 BDLSTM-CNN

The Bidirectional Long Short Term Memory-Convolutional Neural Network (BDLSTM-CNN) architecture is comprised of BDLSTM, convolutional, activation, dense, and dropout layers. The input data was transformed into 3D matrix form which is maintained throughout the various layers of network. The network used for the experiments consists of 1 bidirectional long short term memory layer, 1 convolutional layer with fully connected *ReLU* hidden layer and 3 dropout layers. The dense layers were placed at the output end of the network, concluding with an output layer utilizing a *ReLU* and *Sigmoid* activation functions as seen in the Figure 5.5.

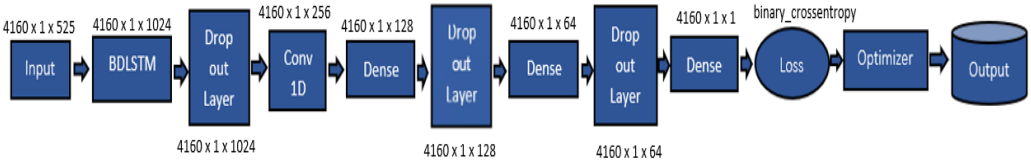

Figure 5.5: BDLSTMCNN Network architecture used in experiments[38].

After performing numerous experiments by varying network configurations and learning parameters, the best architecture used in the experiments in Chapter 6 was an BDLSTM-CNN that consists of 1 BDLSTM and 1 convolutional blocks, 3 dropout layers and a fully connected block (Figure 5.5).

The initial BDLSTM layer consisted of 512 units which reads the input data and outputs 1024 features with 1 timesteps for each because $return\_sequences = True$ and $kernel\_initializer = glorot\_uniform$, a dropout layer and a convolutional layer consists of 256 units, 1 kernel size and a hidden *ReLU* activation function, which is followed by a dense layer with a hidden *ReLU* and *Sigmoid* activation functions and 2 dropout layers.

5.4 Other Machine Learning Approaches

To achieve the objectives presented in Section 5.2, we built a machine learning model that learns from historical data. The following section describes

the machine learning models used for performing various experiments using gaussian naive-bayes, support vector machines, decision trees, random forest, extra trees, Adaboost, gradient boosting and logistic regression.

Supervised learning models

K-fold is a cross-validator that divides the dataset into k folds. Stratified is to ensure that each fold of dataset has the same proportion of observations with a given label. The folds are made by preserving the percentage of samples for each class. It is used for estimating the performance of a machine learning algorithm on the dataset. It uses a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. When a specific value for k is chosen, it is used in place of k in the reference to the model, such as k=3 becoming 3-fold cross-validation.

Input dimension to Machine learning models

For all the machine learning models used in this research, the performance was evaluated using a confusion matrix, a classification report and the mean recall score. One of the input data[5] provided to ML models is from a Religious charity-1 which has 1856 positive samples and 64843 negative samples that looks back at 134 columns/features.

This input data has to be balanced as there are more negative samples than positive samples (Table 4.2). We first over sample the major donor dataset (Table 4.2) using SMOTE technique, the shape of training and testing data is 3712×134. Further, we used the *train_test_split* function to split data arrays into two subsets for training and testing data, then printing the size of the new dataset which is transformed into 2598×134 and 1114×134 (samples, features). We set the train size to 70% and test size to 30%. To evaluate the performance of the model, the experiments were conducted by altering the network parameters of the models used in this research.

The input dimension mentioned in this section is provided to the ML models to perform the various experiments seen in Chapter 6.

[5]The input data used is from all the charities mentioned in Table 4.1

Building cross-validation models

K-fold cross-validation is a method for estimating the performance of a model on an unseen data. It is a technique used for hyperparameter tuning such that the model with the most optimal value of hyperparameters can be trained.

The following is done in this technique for training and testing the model:

1. Instance of StratifiedKFold is created by passing number of folds (nsplits=3). Since, we have large datasets, we can use the value of K from 3 to 5 folds to obtain an accurate estimate of the average performance of the model while reducing the computational cost of refitting and evaluating the model on different folds. The number of folds increases if the data is relatively small. However, larger values of k results in increase of the runtime of the cross-validation algorithm.

2. Split method is invoked on the instance of StratifiedKFold to gather the indices of training and test splits for 3 folds.

3. The *cross_val_score*() function is used to perform the evaluation, taking the dataset and cross-validation configuration and returning a list of scores calculated for each fold.

4. We created multiple models and averaged the results to check for means and standard deviations in the model metrics.

Exhaustive grid search for classification

The grid search [6] exhaustively generates candidates from a grid of parameter values specified with the *param_grid* parameter. It iterates over the classifiers, fits the classifiers on the training set, making predictions on the training set and finally evaluates the classifiers. The apparent best classifier is then used to make predictions on the test set. The *cv_results_* attribute contains useful information for analysing the results of this search.

[6] It is a tuning technique that attempts to compute the optimum values of hyperparameters and was performed on a specific parameter values for the model provided by *GridSearchCV*.

Parameterizations for each algorithm

- Decision Tree: We have 4 values/experiments for one parameter which is max_depth.
- Adaboost: We have 5 values/experiments for one parameter which is $learning\_rate$.
- Random Forest: We have 7 values/experiments for 3 parameter which are max_depth, $max\_features$ and $n\_estimators$.
- Gradient Boosting: We have 7 values/experiments for 3 parameters which are max_depth, $learning\_rate$ and $n\_estimators$.
- Extra Trees Classifier: We have 10 values/experiments for 3 parameters which are $criterion$, $max\_features$, $min\_samples\_leaf$ and $min\_samples\_split$.

The performance was evaluated using confusion matrix, accuracy, precision, recall and F1 score.

5.5 Evaluation metrics

Confusion Matrix:

It is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with four different combinations of predicted and actual values. It is useful for measuring accuracy, precision, recall and specificity.

To evaluate the *major donor model*, we have taken four evaluation metrics into consideration, where TP stands for true positive, FP stands for false positive, TN stands for true negative, and FN stands for false negative.

Classification Accuracy:

The base metric used for model evaluation is accuracy, describing the number of correct predictions over all predictions:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5.1}$$

We use *accuracy_score* function of sklearn.metrics to compute accuracy of our classification model.

Classification Report:

This report consists of the scores of precision, recall, and F1-score. They are explained as follows:

Precision:

Precision is a measure of how many of the positive predictions made are correct (true positives).

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

Recall or Sensitivity:

Recall is a measure of how many of the positive cases the classifier correctly predicted, over all the positive cases in the data.

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

F1-Score:

F1-Score is a measure combining both precision and recall.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.4}$$

Chapter 6

Empirical Studies

This chapter describes five experiments performed using 16 machine learning algorithms: Random forest classifier, Adaboost, gradient boosting, extra trees classifier, gaussian naive-bayes, decision tree, logistic regression, LSTM-GRUs, SimpleRNNs, GRUs, BDLSTM-GRU-TDLs, BDLSTMCNNs, LASSO regression, ElasticNet, XGBoost and Light GBM. The experiments performed are as follows:

- Build a model to predict major donor prospects for charities, which will later be described as two experiments.
- Build a model using only donation data to predict major donors for charities.
- Build a model using only donation and behavioural data to predict major donors for charities.
- Build a model that predicts how much money major donor constituents will contribute to a charity.

6.1 Initial Experiments

The initial experiments were carried out on balanced datasets using random forest classifiers, Adaboost, extra trees classifiers, gaussian naive-bayes, decision trees and logistic regression to see how well the machine learning models

can predict the potential major donors, since supervised machine learning algorithms requires less computing power and takes few seconds to few hours to train rather than deep learning algorithms which are more complex to set up and demand more powerful hardware and resources which requires increased use of graphical processing units. GPUs are useful for high bandwidth memory and ability to hide latency (delays) in memory transfer due to thread parallelism (the ability of many operations to run efficiently at the same time).

Non-major donors (negative cases) outnumber major donors (positive cases) for all charities. Data was balanced in all experiments in order to not bias the model strongly towards negative cases. Testing data was kept separate from training data and all results shown are on testing data.

We used confusion matrices as a metric to evaluate our models, since we seek a model that will predict with high accuracy and some false positives. A model that is perfectly accurate is useless since it gives no prospects. However, a model with too many false positives is likely to be erroneous. As a result, they must be balanced. Models that give no false positives find no potential major donors, which is not what we want. False positives are what we seek for. False negatives (major donors classified as non-major donors) are what we want to avoid, since we know this is wrong (a major donor is a major donor and cannot not be a major donor, but a non-major donor could become a major donor).

6.2 Experiment 1: Predicting major donor prospects using supervised learning models

6.2.1 Objective

The goal of this experiment was to demonstrate supervised learning models ability to predict major donor prospects for charities. If the supervised learning model can predict a major donor with high accuracy and some false positive values, it can be inferred that the supervised learning models can be utilized to predict future major donors.

6.2.2 Data and Approach

The data we used for this experiment is from 8 charities as provided in the below Table 6.1: an Alzheimer's charity, a cancer charity, 2 religious charities and 4 educational charities. The input dimension for training and testing the ML models for this experiment is provided in Section 5.4 and the data preparation are provided in Section 4.2. In this experiment the ML models were built using the best architectures mentioned in Section 5.4.

	AlzC	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
Major Donors	46	82	2080	4393	658	3226	1856	309
Non-Major donors	90859	52123	104677	76155	54121	211519	64843	101459

Table 6.1: Number of samples of each type in each data set.

6.2.3 Results and Discussion

From the results produced by the above listed models, the test accuracy of the model for classifying the predicted value is shown in Table 6.7.

The results shown in Tables 6.2, 6.3, 6.4, 6.5 and 6.6 are from the predictions using random forest classifier, Adaboost classifier, extra trees classifier, Gaussian Naive-Bayes, decision tree and logistic regression respectively.

Table 6.2 shows the results on EC-1 data for predicting major donors. From the results produced for charity EC-1, the best performing model was a random forest classifier with test accuracy of 94.47%. Table 6.2 shows values for classifying 1485 donors as seen in Section 5.4 (which includes 719 major donors and the remaining 766 are non-major donors) as major or non-major donors. Out of 719 major donors, 668 of them are classified correctly with precision of 95.56% (Table 6.8) and the remaining 51 major donors are classified as non-major donors by random forest classifier. F1-Score (Table 6.10) measures a balance between precision and recall and if there is an uneven class distribution (large number of actual negatives). Out of 766 donors in the non-major donors category, 735 are classified as non-major donors with recall of 92.90% (Table 6.9) and the remaining 31 are classified as major donors. These are the constituents we are looking for, who might give a major gift, but we are looking for more false positive values, without comprising too much accuracy for the random forest model to be useful to predict major donors.

	TP	**FP**	**FN**	**TN**
Logistic Regression	649	28	70	738
Gaussian Naive Bayes	570	306	149	460
Decision Tree	633	150	86	616
Random Forest	668	31	51	735
AdaBoost	552	77	167	689
ExtraTrees	679	71	40	695

Table 6.2: Results for predicting major donor prospects using supervised learning for EC-1 charity with 1485 constituents.

Table 6.3 shows the results on EC-2 data for classifying 2120 (with similar calculations as seen in Section 5.4) donors as major donors or non-major donors. The best performing model for EC-2 charity was Adaboost (as seen in Table 6.7) with test accuracy of 91.83%. Out of 908 major donors, 822 major donors are classified correctly with precision of 91.25% (Table 6.8) and the remaining 86 major donors are classified as non-major donors by Adaboost. Conversely, out of 1212 donors in the non-major donors category, 1125 are classified as non-major donors with recall of 90.52% (Table 6.9) and the remaining 87 are classified as major donors.

	TP	**FP**	**FN**	**TN**
Logistic Regression	745	165	163	1047
Gaussian Naive Bayes	775	1193	133	19
Decision Tree	524	508	384	704
Random Forest	605	58	303	1154
AdaBoost	822	87	86	1125
ExtraTrees	672	90	236	1122

Table 6.3: Results for predicting major donor prospects using supervised learning for EC-2 charity with 2120 constituents.

The results on EC-3 charity for classifying 374 donors (with similar calculations as seen in Section 5.4) as major donors or non-major donors is shown in Table 6.4. The best performing model for charity EC-3 was decision trees (as seen in Table 6.7). The model's test accuracy in classifying the predicted value

is 96.79%. Out of 222 major donors, 213 major donors are classified correctly with precision of 98.61% (Table 6.8) and the remaining 3 major donors are classified as non-major donors by decision trees. In the non-major donors category, 149 are classified as non-major donors with recall of 95.94% (Table 6.9) and the remaining 3 are classified as major donors.

	TP	**FP**	**FN**	**TN**
Logistic Regression	212	16	10	136
Gaussian Naive Bayes	168	101	54	51
Decision Tree	213	3	9	149
Random Forest	195	31	27	121
AdaBoost	161	31	61	121
ExtraTrees	186	13	36	139

Table 6.4: Results for predicting major donor prospects using supervised learning for EC-3 charity with 374 constituents.

The RC-1 charity results for classifying 1114 donors (with similar calculations as seen in Section 5.4) as major or non-major donors is shown in Table 6.5. The best performing model for charity RC-1 was logistic regression (as seen in Table 6.7). The test accuracy of the model when classifying the predicted value is 93.35%. Out of 512 major donors, 480 major donors are classified correctly with precision of 91.95% (Table 6.8) and the remaining 32 major donors are classified as non-major donors by logistic regression. In the non-major donors category, 560 are classified as non-major donors with recall of 93.75% (Table 6.9) and the remaining 42 are the donors we are looking for who could give major gifts, that the model actively predicted positive.

	TP	**FP**	**FN**	**TN**
Logistic Regression	480	42	32	560
Gaussian Naive Bayes	271	172	241	430
Decision Tree	387	115	125	487
Random Forest	439	90	73	512
AdaBoost	445	134	67	468
ExtraTrees	443	132	69	470

Table 6.5: Results for predicting major donor prospects using supervised learning for RC-1 charity with 1114 constituents.

Table 6.6 shows the results for a cancer charity for classifying 51 donors (with similar calculations as seen in Section 5.4) as major or non-major donors. The best performing model for a cancer charity was logistic regression (as seen in Table 6.7). The test accuracy of the model when classifying the predicted value is 96.07%. Out of 22 major donors, 21 major donors are classified correctly with precision of 95.45% (Table 6.8) and the remaining 1 major donors are classified as non-major donors by logistic regression. In the non-major donors category, 28 are classified as non-major donors with recall of 95.45% (Table 6.9) and the remaining 1 are the donors we are looking for who could give major gifts, however, we need more false positive values for the logistic regression model to be useful in predicting major donors.

	TP	**FP**	**FN**	**TN**
Logistic Regression	21	1	1	28
Gaussian Naive Bayes	15	9	7	20
Decision Tree	17	8	5	21
Random Forest	21	2	1	27
AdaBoost	20	5	2	24
ExtraTrees	18	2	4	27

Table 6.6: Results for predicting major donor prospects using supervised learning for a cancer charity with 51 constituents.

Models	Logistic Regression	Gaussian Naive Bayes	Decision Tree	Random Forest	AdaBoost	ExtraTrees
AlzC	53.57%±0.060	78.57%±0.396	85.71%±0.226	**92.85%±0.046**	78.57%±0.299	85.71%±0.302
CC	**96.07%±0.170**	68.62%±0.246	74.50%±0.029	94.11%±0.162	86.27%±0.068	88.23%±0.148
EC-1	93.40%±0.037	69.36%±0.217	84.10%±0.057	**94.47%±0.127**	83.56%±0.092	92.52%±0.117
EC-2	84.52%±0.111	37.45%±0.009	57.92%±0.035	82.97%±0.023	**91.83%±0.045**	84.62%±0.042
EC-3	93.04%±0.031	58.55%±0.198	**96.79%±0.020**	84.49%±0.017	75.40%±0.020	86.89%±0.011
EC-4	69.42%±0.088	51.70%±0.159	70.97%±0.072	**90.44%±0.159**	88.94%±0.110	83.21%±0.155
RC-1	**93.35%±0.264**	62.92%±0.284	85.90%±0.027	85.36%±0.168	81.95%±0.135	81.95%±0.202
RC-2	87.63%±0.015	52.68%±0.015	72.04%±0.056	**96.77%±0.014**	93.54%±0.005	82.79%±0.014
Mean	83.87%±0.097	59.98%±0.190	78.49%±0.065	**90.85%±0.089**	85%±0.096	87.15%±0.123

Table 6.7: Accuracies for major donor prediction using supervised learning for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
Logistic Regression	54.16%	**95.45%**	**95.86%**	81.86%	92.98%	63.93%	91.95%	91.30%	83.29%
Gausian Naive Bayes	72.56%	62.50%	65.06%	39.38%	62.45%	53.24%	75.27%	53.14%	58.79%
Decision Tree	**92.30%**	68%	80.84%	50.77%	**98.61%**	76.29%	**93.62%**	91.22%	81.45%
Random Forest	82.34%	91.30%	95.56%	**91.25%**	86.28%	**98.37%**	82.98%	92.43%	89.68%
AdaBoost	80%	80%	87.75%	90.42%	83.85%	86.03%	76.85%	93.93%	85.12%
ExtraTrees	92.20%	90%	90.53%	88.18%	93.46%	89.82%	77.04%	**98.55%**	**89.97%**

Table 6.8: Precision for major donor prediction using supervised learning for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
Logistic Regression	86.56%	**95.45%**	90.26%	49.40%	95.49%	**87.79%**	**93.75%**	84.84%	85.35%
Gausian Naive Bayes	60%	68.18%	79.27%	**85.35%**	75.67%	32.50%	45.54%	90.23%	65.69%
Decision Tree	80%	77.27%	88.03%	57.70%	95.94%	60.06%	78.99%	52.52%	73.81%
Random Forest	**86.66%**	95.23%	92.90%	66.62%	87.83%	82.06%	85.74%	**93.93%**	87.34%
AdaBoost	80%	90.90%	76.77%	90.52%	72.52%	86.23%	86.91%	92.83%	**95.77%**
ExtraTrees	80%	81.81%	**94.43%**	74%	**83.78%**	74.55%	86.52%	68.68%	82.17%

Table 6.9: Recall for major donor prediction using supervised learning for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
Logistic Regression	66.63%	**95.45%**	92.97%	61.61%	94.21%	73.98%	**92.84%**	87.95%	83.09%
Gausian Naive Bayes	65.68%	65.21%	71.46%	53.89%	59.67%	40.36%	56.74%	66.88%	59.50%
Decision Tree	**85.71%**	72.33%	84.28%	54.01%	**97.25%**	67.21%	85.68%	66.66%	76.64%
Random Forest	84.44%	93.22%	**94.21%**	77.01%	89.01%	**89.48%**	84.33%	92.57%	76.90%
AdaBoost	80%	85.10%	81.89%	**90.46%**	80.58%	88.54%	81.57%	**93.37%**	85.18%
ExtraTrees	85.66%	85.70%	92.43%	80.47%	95.37%	81.48%	81.50%	80.95%	**85.44%**

Table 6.10: F1-Score for major donor prediction using supervised learning for all charities.

6.2.4 Summary

We experimented with six different ML models for 8 charities in order to accurately predict future major donor prospects. For the ML models used in this research, we are looking for bigger false positive values than false negatives, which indicates who might give a major gift. Based on the mean accuracy and confusion matrices values on the training set, the best performing model was random forest classifier (as seen in Table 6.7) to predict major donors on the test data. However, the false positive values for educational charities (EC-1 (Table 6.2) and EC-3 (Table 6.4)) and a cancer charity (Table 6.6) are less than false negative values and need to be increased. Gaussian Naive Bayes, predicts with more false positives for educational charities (EC-1, EC-2 and EC-3), religious (RC-1) and a cancer charity than other models, but with less accuracy as seen in Table 6.7. In the upcoming experiment, we will evaluate various deep learning techniques looking to improve the false positives.

Based on the analysis of each models during training, we used the best performing model obtained for each charities to predict future major donors. For example, for EC-1 charity, the best performing model was random forest classifier, we used that model to make predictions on the test data. As for some of the charities the major donors ranges from 100, and charities generally does not contain major donors in the range of 25,000/50,000. Table 6.11 shows the final [1] predicted major and non-major donors on the test set.

Labels	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
1 (Predicted Major donors)	1491	955	408	2346	144	686	1402	1883
0 (Predicted Non-Major donors)	89368	51168	104269	73809	53977	210833	63441	99576

Table 6.11: Major donors for all the charities using supervised learning algorithms.

[1] A final machine learning model is a model that we use to make predictions on new/test data.

6.3 Experiment 2: Predicting major donor prospects using deep learning

6.3.1 Objective

The false positives for educational charities (EC-1 and EC-3) and a cancer charity are less than false negatives, as tested in previous experiments. In order to predict major donors, we need high accuracy and more false positives. This experiments objective is to improve the false positive values while maintaining similar accuracies to Experiment 1 for most of the charities using deep learning techniques for predicting major donor prospects.

6.3.2 Data and Approach

The data we used for this experiment is the same data used in Experiment 1 which is from 8 charities: an Alzheimer's charity, a cancer charity, 2 religious charities and 4 educational charities (Table 4.2). The input dimension for training and testing the ML models for this experiment is provided in Section 5.3 and the data preparation are provided in Section 4.2. In this experiment the ML models were built using the best architectures mentioned in Tables 6.12, 6.13, 6.14, 6.15, and 6.16 with the *learning_rate* of 0.001 and 0.01 for RMSprop and Adam optimizers.

LSTM-GRU Parameters	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3, EC-4, RC-1 and RC-2**
LSTM Layer 1	1024	512	512	512	512
Dropout	0.2	0.2	0.2	0.2	0.2
LSTM Layer 2	512	256	64	256	256
Dropout	0.2	0.2	0.2	0.2	0.2
GRU	512	128	32	128	128
Dropout	0.2	0.2	0.2	0.2	0.2
Activation Function	ReLU	ReLU	ReLU	ReLU	ReLU
Activation Function	ReLU	ReLU	ReLU	ReLU	tanh
Activation Function	Sigmoid	ReLU	Sigmoid	Sigmoid	Sigmoid
Optimizer	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop

Table 6.12: Best performing LSTM-GRU architectures for all the charities.

RNN Parameters	AlzC	CC, EC-1, EC-2, EC-3, EC-4, RC-1 and RC-2
SimpleRNN Layer 1	512	512
Dropout	0.2	0.3
SimpleRNN Layer 2		128
Dropout		0.3
Activation Function	ReLU	ReLU
Activation Function	ReLU	ReLU
Activation Function	Sigmoid	Sigmoid
Activation Function	Sigmoid	Sigmoid
Optimizer	RMSprop	RMSprop

Table 6.13: Best performing SimpleRNN architectures for all the charities.

GRU Parameters	AlzC	CC	EC-1	EC-2, EC-3, EC-4, RC-1 and RC-2
GRU Layer 1	512	512	128	512
Dropout	0.3	0.3	0.3	0.3
GRU Layer 2	156	156	64	156
Dropout	0.3	0.3	0.3	0.3
Activation Function	ReLU	ReLU	ReLU	ReLU
Activation Function	ReLU	ReLU	ReLU	ReLU
Activation Function	Sigmoid	Sigmoid	Sigmoid	Sigmoid
Optimizer	RMSprop	RMSprop	Adam	RMSprop

Table 6.14: Best performing GRU architectures for all the charities.

BDLSTM-GRU-TDL Parameters	AlzC	CC	EC-1	EC-2, EC-3, EC-4, RC-1 and RC-2
BDLSTM Layer	512	512	512	512
Dropout	0.3	0.3	0.2	0.3
GRU Layer	256	256	256	256
Dropout	0.3	0.3	0.2	0.3
TDL Layer				
Activation Function	ReLU	ReLU	ReLU	ReLU
Activation Function	Sigmoid	Sigmoid	Sigmoid	Sigmoid
Optimizer	RMSprop	RMSprop	RMSprop	RMSprop

Table 6.15: Best performing BDLSTM-GRU-TDL architectures for all the charities.

BDLSTM-CNN Parameters	AlzC, CC, EC-1, EC-2, EC-3, EC-4, RC-1 and RC-2
BDLSTM Layer	512
Dropout	0.2
Conv1D	256
Activation Function	ReLU
Kernal Size	1
Activation Function	ReLU
Dropout	0.2
Activation Function	ReLU
Dropout	0.2
Activation Function	Sigmoid
Optimizer	RMSprop

Table 6.16: Best performing BDLSTM-CNN architectures for all the charities.

6.3.3 Results and Discussion

The results shown in Tables 6.17, 6.18, 6.19, 6.20 and 6.21 are from the predictions using LSTM-GRUs, SimpleRNNs, GRUs, BDLSTM-GRU-TDLs and BDLSTMCNNs respectively. We split the data into 2 sets training and test set and ran the models for 5 trials and averaged all the values to get mean and standard deviation values through cross validation and calculated the accuracy, precision, recall and F1-score using deep learning models, which saw increase in false positive values with slight decrease in accuracies when compared to Experiment 1 for predicting future major donors.

Table 6.17 shows the results on EC-1 data for predicting major donors using the best performing LSTM-GRUs, SimpleRNNs, GRUs, BDLSTM-GRU-TDLs and BDLSTMCNNs architectures as seen in Tables 6.12, 6.13, 6.14, 6.15 and 6.16. Table 6.17 shows values for classifying 1485 donors as seen in Section 5.3 (which includes 719 major donors and 766 are non-major donors) as major donors or non-major donors. Out of 719 major donors, 669 of them are classified correctly with precision of 85.76% (Table 6.23) and the remaining 50 major donors are classified as non-major donors by LSTM-GRU. Out of 766 donors in the non-major donors category, 655 are classified as non-major donors with recall of 93.04% (Table 6.24) and the remaining 111 are classified as major donors. From the results produced for charity EC-1, the best performing model was LSTM-GRU. The average number of false positives after 5 trials is 111±9 for the LSTM-GRU model. The test accuracy of the model for classifying the predicted value is 89.15%±0.058, which when com-

pared to Experiment 1 shows slight decrease in ML models accuracy (Best accuracy: 94.47%±0.127), but higher false positive (False positive: 31) and lower false negative (False negative: 51) values for EC-1 charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values when compared to Experiment 1 for EC-1 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	669	111	50	655
RNN	606	136	113	630
GRU	650	132	69	634
BDLSTM-GRU-TDL	643	113	76	653
BDLSTM-CNN	646	121	73	645

Table 6.17: Results for predicting major donor prospects for EC-1 charity with 1485 constituents.

Table 6.18 shows the results on EC-2 data for classifying 2120 donors as major donors or non-major donors. Out of 908 major donors, 764 major donors are classified correctly with precision of 76.24% (Table 6.23) and the remaining 144 major donors are classified as non-major donors by LSTM-GRU. Conversely, out of 1212 donors in the non-major donors category, 974 are classified as non-major donors with recall of 84.14% (Table 6.24) and the remaining 238 are classified as major donors. As per the results from Table 6.22 for charity EC-2, the best performing model was LSTM-GRU. The average number of false positives after 5 trials is 238±2 for the LSTM-GRU model. The test accuracy of the model for classifying the predicted value is 81.98%±0.027, which when compared to Experiment 1 shows decrease in ML models accuracy (Best accuracy: 91.83%±0.045), but higher false positive (False positive: 87) and false negative (False negative: 86) values for EC-2 charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values when compared to Experiment 1 for EC-2 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	764	238	144	974
RNN	755	254	153	958
GRU	744	289	164	923
BDLSTM-GRU-TDL	748	288	160	924
BDLSTM-CNN	727	273	181	939

Table 6.18: Results for predicting major donor prospects for EC-2 charity with 2120 constituents.

The results for EC-3 charity for classifying 374 donors as major donors or non-major donors is shown in Table 6.19. The best performing model for charity EC-3 was RNN (as seen in Table 6.22). Out of 222 major donors, 201 major donors are classified correctly with precision of 85.89% (Table 6.23) and the remaining 21 major donors are classified as non-major donors by RNN. In the non-major donors category, 119 donors are classified as non-major donors with recall of 90.54% (Table 6.24), while the remaining 33 are classified as major donors. The average number of false positives after 5 trials is 33±5 for the RNN model. The model's test accuracy in classifying the predicted value is 85.56%±0.016, which when compared to Experiment 1 shows decrease in ML models accuracy (Best accuracy: 96.79%±0.020), but higher false positive (False positive: 3) and false negative (False negative: 9) values for EC-3 charity. From the analysis, RNN model is performing well with respect to higher false positive values when compared to Experiment 1 for EC-3 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	208	43	14	109
RNN	201	33	21	119
GRU	188	50	34	102
BDLSTM-GRU-TDL	184	44	38	108
BDLSTM-CNN	188	40	34	112

Table 6.19: Results for predicting major donor prospects for EC-3 charity with 374 constituents.

The RC-1 charity results for classifying 1114 donors as major or non-major donors is shown in Table 6.20. The best performing model for charity RC-1 was LSTM-GRU (as seen in Table 6.22). Out of 512 major donors, 472 major donors are classified correctly with precision of 81.09% (Table 6.23), while the other 40 major donors were classified as non-major donors by LSTM-GRU. In the non-major donors category, 492 donors are classified as non-major donors with recall of 92.18% (Table 6.24), and the remaining 110 are the donors we are looking for who could give major gifts, that the model actively predicted positive. The average number of false positives after 5 trials is 110±12 for the LSTM-GRU model. The test accuracy of the model when classifying the predicted value is 92.19%±0.012, which when compared to Experiment 1 shows slight decrease in ML models accuracy (Best accuracy: 93.35%±0.264), but higher false positive values (False positive: 42) and false negative (False negative: 32) values for RC-1 charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values when compared to Experiment 1 for RC-1 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	472	110	40	492
RNN	459	132	53	470
GRU	473	122	39	480
BDLSTM-GRU-TDL	456	89	56	513
BDLSTM-CNN	480	102	32	500

Table 6.20: Results for predicting major donor prospects for RC-1 charity with 1114 constituents.

Table 6.21 shows the results for a cancer charity for classifying 51 donors as major or non-major donors. The best performing model for a cancer charity was BDLSTM-GRU-TDL (as seen in Table 6.22). Out of 22 major donors, 20 major donors are classified correctly with precision of 95.23%, while the other 2 major donors were classified as non-major donors by BDLSTM-GRU-TDL. In the non-major donors category, 28 donors are classified as non-major donors with recall of 90.90% (Table 6.24), and the remaining 1 are the donors we are looking for who could give major gifts. The average number of false positives after 5 trials is 1±1 for the BDLSTM-GRU-TDL model. The test accuracy of the model when classifying the predicted value is 94.11%±0.017, which

when compared to Experiment 1 shows slight decrease in ML models accuracy (Best accuracy: 96.07%±0.170), but same false positive (False positive: 1) and higher false negative (False negative: 1) values for a cancer charity. From the analysis, logistic regression model from Experiment 1 is performing well with respect to higher accuracies and same false positive values for a cancer charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	19	2	3	27
RNN	16	3	6	26
GRU	19	1	3	28
BDLSTM-GRU-TDL	20	1	2	28
BDLSTM-CNN	19	9	3	20

Table 6.21: Results for predicting major donor prospects for a cancer charity with 51 constituents.

Models	**LSTM-GRU**	**RNN**	**GRU**	**BDLSTM-GRU-TDL**	**BDLSTM-CNN**
AlzC	85.38%±0.042	82.30%±0.026	88%±0.015	88.66%±0.013	**89.74%±0.032**
CC	90.19%±0.026	82.35%±0.048	92.15%±0.009	**94.11%±0.017**	76.47%±0.005
EC-1	**89.15%±0.058**	83.23%±0.012	86.46%±0.022	87.27%±0.014	86.93%±0.017
EC-2	**81.98%±0.027**	80.80%±0.031	78.63%±0.036	78.86%±0.032	78.58%±0.029
EC-3	84.75%±0.042	**85.56%±0.016**	77.54%±0.041	78.07%±0.056	80.21%±0.039
EC-4	90.50%±0.028	90.84%±0.005	**91%±0.008**	90.66%±0.035	**90.88%±0.029**
RC-1	**92.19%±0.012**	83.39%±0.029	85.54%±0.028	86.98%±0.031	87.97%±0.041
RC-2	95.02%±0.005	**96.86%±0.011**	**96.94%±0.016**	95.50%±0.029	96.54%±0.057
Mean	**88.64%±0.030**	85.66%±0.022	87.03%±0.020	87.51%±0.042	85.91%±0.035

Table 6.22: Accuracies for major donor prediction for all charities.

Models	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**	**Mean**
LSTM-GRU	86.83%	90.47%	**85.76%**	**76.24%**	82.86%	91.16%	81.09%	95.63%	86.25%
RNN	86%	84.21%	83.12%	74.82%	**85.89%**	90.49%	77.66%	**97.79%**	84.99%
GRU	86.13%	95%	83.12%	72.02%	78.99%	90.70%	79.49%	97.64%	**86.63%**
BDLSTM-GRU-TDL	85.75%	**95.23%**	85.05%	72.20%	80.70%	90.55%	**83.66%**	95.69%	86.10%
BDLSTM-CNN	**89.14%**	67.85%	84.22%	72.70%	82.45%	**92%**	82.47%	95.90%	83.34%

Table 6.23: Precision for major donor prediction for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	82%	86.36%	**93.04%**	**84.14%**	**93.69%**	91.09%	92.18%	94.72%	**89.65%**
RNN	82.66%	72.72%	84.28%	83.14%	90.54%	91.77%	89.64%	96.12%	86.35%
GRU	**90.66%**	86.36%	90.40%	81.93%	84.68%	**91.86%**	92.38%	96.43%	89.28%
BDLSTM-GRU-TDL	88%	**90.90%**	89.42%	82.37%	82.88%	91.29%	89.06%	95.65%	88.69%
BDLSTM-CNN	86.66%	86.36%	89.84%	80.06%	84.68%	89.24%	**93.75%**	**97.51%**	88.51%

Table 6.24: Recall for major donor prediction for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	84.34%	88.36%	**90.26%**	**81%**	87.94%	91.12%	86.28%	95.17%	**88.05%**
RNN	84.29%	78.04%	83.69%	78.76%	**88.15%**	91.12%	83.22%	96.94%	85.42%
GRU	**88.33%**	90.47%	86.60%	76.65%	81.73%	**91.27%**	85.45%	**97.03%**	87.19%
BDLSTM-GRU-TDL	86.83%	**93.01%**	87.18%	76.95%	81.77%	90.91%	86.27%	95.66%	87.32%
BDLSTM-CNN	87.88%	75.99%	86.93%	76.20%	83.55%	90.59%	**87.74%**	96.69%	85.69%

Table 6.25: F1-Score for major donor prediction for all charities.

6.3.4 Summary

From this experiment, the false positives have been improved for educational (EC-1, EC-2 and EC-3) and religious (RC-1) charities (as seen in Table 6.22) using deep learning models. We experimented with five different ML models for 8 charities. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU to predict major donors on the test data. The confusion matrix values for LSTM-GRU model have been improved for EC-1 charity when comparing to Experiment 1 with slight decrease in accuracies (Table 6.22). But with a cancer charity, the false positive values remain the same (Table 6.21) as of Experiment 1. RNN and BDLSTM-GRU-TDL model are performing better for EC-3 and cancer charities when comparing to other models in Experiment 1 with respect to more false positive values, but with slight decrease in accuracies. This suggests that ML models using deep learning models are performing better with higher false positive values for most of the charities data compared to Experiment 1. As seen in Table 6.22 for EC-4 and RC-2 charities, the accuracies overlap with including standard deviations, the best model with more false positive values, without comprising too much accuracy was used (GRU (False positive: 65) and RNN (False positive: 9) for EC-4 and RC-2 charities) for predicting future major donor prospects.

Based on the analysis of each models during training, we used the best performing model obtained for each charities to predict future major donors. For

example, for cancer charity, the best performing model was BDLSTM-GRU-TDL, we used that model to make predictions on the test data. Table 6.26 shows the final predicted major and non-major donors on the test set.

The results from this experiment suggests that, LSTM-GRUs (as seen in Table 6.22), predicts more accurate results for more than one educational and religious charities using deep learning models, in particular with respect to false positives.

In further experiments, we investigate the deep learning models with only donation data, to improve both accuracies and false positive values to predict major donors.

Labels	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
1 (Predicted Major donors)	9281	9137	6672	3761	1176	10986	82	754
0 (Predicted Non-Major donors)	81578	42986	98005	72394	52945	200533	64761	100705

Table 6.26: Major donors for all the charities.

6.4 Experiment 3: Predicting major donor prospects using only donation data

6.4.1 Objective

In previous experiments, for predicting major donors using deep learning models, which saw an increase in false positive values but slight decrease in accuracies when compared to Experiment 1. This experiments objective is to input only donation data to deep learning models and remove the behavioral data, educational data, demographic data and giveaway features to improve the accuracies for 8 charities to predict future major donors. Since, it may contain fewer misleading data in which the model's accuracy can be improved.

6.4.2 Data and Approach

The data we used for this experiment is the same donation data used in Section 6.2 (Table 4.1). The donation data used in this experiment are shown in the Section 4.1.1. In this research, we compared the performance of different algorithms, using all the features (unpruned) in the dataset as seen in Experiment 1 and 2 to pruned dataset with including only donation data. When compared to Experiment 1, Experiment 2 for predicting major donor prospects using deep learning models found an increase in false positive values but a minor fall in accuracies. The goal was to understand whether an algorithm (say A) on an unpruned dataset performs better than another algorithm (say B), will algorithm B perform better in terms of false positives and accuracies on the pruned data or vice-versa. For this experiment, the best ML models were built using the architectures mentioned in Section 6.3 and Section 5.3. The input dimension for training and testing the ML models for this experiment is provided in Subsection 5.3 and the data preparation are provided in Section 4.

6.4.3 Results and Discussion

The results shown in Tables 6.27, 6.28, 6.29, 6.30 and 6.31 using LSTMGRUs, SimpleRNNs, GRUs, BDLSTM-GRU-TDLs and BDLSTM-CNNs respectively.

Table 6.27 shows the results values for classifying 1485 donors (which includes 719 major donors and 766 are non-major donors) as major donors or

non-major donors. Out of 719 major donors, 649 major donors are classified correctly with precision of 81.94% (Table 6.33) and the remaining 70 major donors are classified as non-major donors by LSTM-GRU. Out of 766 donors in the non-major donors category, 623 are classified as non-major donors with recall of 90.26% (Table 6.34) and the remaining 143 are classified as major donors. From the results produced for EC-1 charity using donation data, the best performing model was LSTM-GRU. The test accuracy of the model for classifying the predicted value is 85.65%±0.016, which when compared to Experiment 1 (Best accuracy: 94.47%±0.127, False positive: 31, False negative: 51) and Experiment 2 (Best accuracy: 89.15%±0.058, False positive: 111, False negative: 50) with all features, shows slight decrease in ML models accuracy, but higher false positive values for EC-1 charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values than false negatives when compared to Experiment 1 and 2 for EC-1 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	649	143	70	623
RNN	565	152	154	614
GRU	588	227	131	539
BDLSTM-GRU-TDL	626	144	93	622
BDLSTM-CNN	625	145	94	621

Table 6.27: Results for predicting major donor prospects using donation data for EC-1 charity with 1485 constituents.

Table 6.28 shows the results using donation data on EC-2 data for classifying 2120 donors as major donors or non-major donors. Out of 908 major donors, 772 major donors are classified correctly with precision of 89.83% (Table 6.33) and the remaining 136 major donors are classified as non-major donors by BDLSTM-GRU-TDL. Conversely, out of 766 donors in the non-major donors category, 1055 are classified as non-major donors with recall of 85.75% (Table 6.34) and the remaining 157 are classified as major donors, as these are the constituents we are looking for, who might give a major gift which the model predicted positive. As per the results from Table 6.32 for charity EC-2, the best performing model using donation data was BDLSTM-GRU-TDL. The test accuracy of the model for classifying the predicted value

is 86.17%±0.023, which when compared to Experiment 1 (Best accuracy: 91.83%±0.045, False positive: 87, False negative: 86) and Experiment 2 (Accuracy: 81.98%±0.027, False positive: 238, False negative: 144) with all features, shows slight decrease in ML models accuracy than Experiment 1, but higher false positive values, where as increase in accuracy, but lower false positives than Experiment 2 for EC-1 charity. From the analysis, LSTM-GRU model from Experiment 2 is performing well with respect to higher false positive values when compared to Experiment 1 and 3 for EC-2 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	822	216	86	996
RNN	716	264	192	948
GRU	649	150	259	1062
BDLSTM-GRU-TDL	772	157	136	1055
BDLSTM-CNN	817	216	91	996

Table 6.28: Results for predicting major donor prospects using donation data for EC-2 charity with 2120 constituents.

The results on EC-3 charity for classifying 374 donors as major donors or non-major donors is shown in Table 6.29. Out of 222 major donors, 146 major donors are classified correctly with precision of 83.47% (Table 6.33) , while the other 76 major donors are classified as non-major donors by GRU. In the non-major donors category, 104 donors are classified as non-major donors with recall of 75.25% (Table 6.34), while the remaining 48 are classified as major donors. The test accuracy of the model for classifying the predicted value is 66.84%±1.110, which when compared to Experiment 1 (Best accuracy: 96.79%±0.020, False positive: 3, False negative: 9) and Experiment 2 (Best accuracy: 85.56%±0.016, False positive: 33, False negative: 21) with all features, shows decrease in ML models accuracy than Experiment 1 and 2, but higher false positive values for EC-3 charity. From the analysis, there is an overlap between accuracies (as seen in Table 6.32) between RNN, GRU, BDLSTM-GRU-TDL and BDLSTM-CNN with including standard deviations, in which RNN model is performing well when compared to other deep learning models with respect to more false positives than false negatives, but decrease in accuracies when compared to Experiment 1 and 2 for EC-3 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	135	44	87	108
RNN	166	73	56	79
GRU	146	48	76	104
BDLSTM-GRU-TDL	160	67	62	85
BDLSTM-CNN	141	44	81	108

Table 6.29: Results for predicting major donor prospects using donation data for EC-3 charity with 374 constituents.

The RC-1 charity results for classifying 1114 donors as major or non-major donors is shown in Table 6.30. Out of 512 major donors, 481 major donors are classified correctly with precision of 93.76% (Table 6.33), while the other 31 major donors were classified as non-major donors by BDLSTM-CNN. In the non-major donors category, 570 donors are classified as non-major donors with recall of 93.94% (Table 6.34), and the remaining 32 are the donors we are looking for who could give major gifts, that the model actively predicted positive. The test accuracy of the model for classifying the predicted value is 94.34%±0.019, which when compared to Experiment 1 (Accuracy: 93.35%±0.264, False positive: 42, False negative: 32) and Experiment 2 (Accuracy: 92.19%±0.012, False positive: 110, False negative:40) with all features, shows increase in ML models accuracy than Experiment 1 and 2, but lower false positive values for RC-1 charity. Also there is an overlap between accuracies (as seen in Table 6.32) between LSTM-GRU and BDLSTM-CNN with including standard deviations, in which BDLSTM-CNN model is performing well when compared to LSTM-GRUs with respect to more false positives than false negatives. From the analysis, LSTM-GRU model from Experiment 2 is performing well with respect to higher false positive values when compared to Experiment 1 and 3 for RC-1 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	479	31	33	571
RNN	436	43	76	559
GRU	475	126	37	476
BDLSTM-GRU-TDL	462	36	50	566
BDLSTM-CNN	481	32	31	570

Table 6.30: Results for predicting major donor prospects using donation data for RC-1 charity with 1114 constituents.

Table 6.31 shows the results for a cancer charity for classifying 51 donors as major or non-major donors. Out of 22 major donors, 16 major donors are classified correctly with precision of 69.56% (Table 6.33), while the remaining 6 major donors were classified as non-major donors by BDLSTM-CNN. In the non-major donors category, 22 donors are classified as non-major donors with recall of 72.72% (Table 6.34), and the remaining 7 are the donors we are looking for who could give major gifts, which the model actively predicted positive. The test accuracy of the model for classifying the predicted value is 74.50%±0.009, which when compared to Experiment 1 (Best accuracy: 96.07%±0.170, False positive: 1, False negative: 1) and Experiment 2 (Best accuracy: 94.11%±0.017, False positive: 1, False negative: 2) with all features, shows decrease in ML models accuracy than Experiment 1 and 2, but higher false positive values for a cancer charity. From the analysis, BDLSTM-CNN model is performing well with respect to higher false positive values when compared to Experiment 1 and 2 for a cancer charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	15	8	7	21
RNN	13	11	9	18
GRU	14	19	8	10
BDLSTM-GRU-TDL	15	8	7	21
BDLSTM-CNN	16	7	6	22

Table 6.31: Results for predicting major donor prospects using donation data for a cancer charity with 51 constituents.

Models	LSTM-GRU	RNN	GRU	BDLSTM-GRU-TDL	BDLSTM-CNN
AlzC	**88.33%±0.031**	78.33%±0.031	82.50%±0.028	85.83%±0.042	86.66%±0.040
CC	70.58%±0.015	60.78%±0.011	47.05%±0.028	70.58%±0.018	**74.50%±0.009**
EC-1	**85.65%±0.016**	79.39%±0.016	54.88%±0.003	84.04%±0.020	83.90%±0.029
EC-2	85.75%±0.008	78.49%±0.006	80.70%±0.004	**86.17%±0.023**	85.51%±0.015
EC-3	64.97%±0.007	**65.50%±0.001**	**66.84%±1.110**	**65.50%±0.012**	**66.57%±0.005**
EC-4	**93.99%±0.016**	84.52%±0.018	79.22%±0.003	89.90%±0.024	92.07%±0.016
RC-1	**94.25%±0.017**	89.31%±0.011	85.36%±0.010	92.28%±0.021	**94.34%±0.019**
RC-2	**83.57%±0.041**	81.32%±0.026	74.30%±0.004	80.92%±0.031	80%±0.030
Mean	**83.38%±0.018**	77.20%±0.015	71.35%±0.148	81.90%±0.023	82.94%±0.020

Table 6.32: Accuracies for major donor prediction using donation data for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	**94.84%**	65.21%	**81.94%**	89.49%	**85.85%**	**95.69%**	**93.92%**	**85.06%**	**86.50%**
RNN	81.74%	54.16%	78.80%	83.09%	62.83%	89.67%	91.02%	83.81%	78.14%
GRU	86.38%	42.42%	72.14%	88.84%	83.47%	77.94%	79.03%	70.29%	75.06%
BDLSTM-GRU-TDL	92.47%	65.21%	81.29%	**89.83%**	85.84%	94.71%	92.77%	81.62%	85.46%
BDLSTM-CNN	94.54%	**69.56%**	81.16%	89.53%	85.84%	92.51%	93.76%	79.52%	85.80%

Table 6.33: Precision for major donor prediction using donation data for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	**81.66%**	68.18%	**90.26%**	79.19%	75.41%	**92.07%**	93.55%	**79.43%**	**82.46%**
RNN	75%	59.09%	78.58%	73.06%	69.45%	78.10%	85.15%	75.77%	74.27%
GRU	81.65%	63.63%	81.78%	81.22%	75.25%	81.25%	92.77%	78.59%	79.51%
BDLSTM-GRU-TDL	78.33%	68.18%	83.10%	**85.75%**	70.48%	84.39%	90.23%	77.18%	79.70%
BDLSTM-CNN	78.33%	**72.72%**	86.92%	79.09%	**76.21%**	91.43%	**93.94%**	77.74%	82.04%

Table 6.34: Recall for major donor prediction using donation data for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	**87.20%**	66.66%	**85.89%**	84.02%	80.29%	**93.82%**	93.73%	**82.08%**	**84.21%**
RNN	77.65%	56.51%	78.68%	77.75%	65.97%	83.21%	87.98%	79.27%	75.87%
GRU	81.65%	50.90%	76.65%	84.85%	79.14%	79.50%	85.35%	74.20%	76.53%
BDLSTM-GRU-TDL	84.35%	66.66%	82.18%	**87.74%**	77.90%	89.22%	91.48%	79.25%	82.34%
BDLSTM-CNN	85.09%	**71.10%**	83.94%	83.98%	**80.73%**	91.95%	**93.84%**	78.56%	83.64%

Table 6.35: F1-Score for major donor prediction using donation data for all charities.

6.4.4 Summary

From this experiment, the false positive values have been improved for educational charities (EC-1, EC-2 and EC-3) and a cancer charity, but there was a slight decrease in accuracies (as seen in Table 6.32) when compared to Experiment 1 and 2 including all the features. We experimented with five different ML models for 8 charities. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU to predict major donors on the test data. The results from this experiment suggests that, LSTM-GRUs (as seen in Table 6.32), predicts with more false positives than false negative values for educational charities (EC-1, EC-2 and EC-3) and a cancer charity using only donation data. RNN and GRU, predicts with more false positives for educational charities (EC-1, EC-2 and EC-3), religious (RC-1) and a cancer charity than other models, but with less accuracy as seen in Table 6.32. Based on the analysis of each models during training, we used the best performing model obtained for each charities using donation data to predict future major donors. For example, for RC-1 charity, the best performing model was BDLSTM-CNN, we used that model to make predictions on the test data. Table 6.36 shows the final predicted major and non-major donors on the test set.

In further experiments, we evaluate the deep learning models using only donation and behavioural data, to increase both accuracies and false positive values to predict major donors.

Labels	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
1 (Predicted Major donors)	2013	1605	2670	1495	1572	8786	1210	6796
0 (Predicted Non-Major donors)	88846	50518	102007	74660	52549	202733	63633	94663

Table 6.36: Major donors for all the charities using donation data.

6.5 Experiment 4: Predicting major donor prospects using only donation and behavioural data

6.5.1 Objective

Experiment 2 for predicting major donors using deep learning models saw an increase in false positive values but a slight drop in accuracies for educational (EC-1, EC-2 and EC-3) and a religious (RC-1) charities, where as in Experiment 3 using only donation data BDLSTM-GRU-TDLs and BDLSTMCNNs saw slight increase in accuracies and LSTM-GRUs predicted more false positives than false negatives when compared to Experiment 2. This experiments objective is to input only donation and behavioral data to deep learning models and remove educational data, demographic data and giveaway features to improve both accuracies and false positives for 8 charities to predict future major donors.

6.5.2 Data and Approach

The data we used for this experiment is the same donation and behavioural data used in Section 6.2 (Table 4.1). The donation and behavioural data used in this experiment are shown in the Section 4.1.1.

Donation data provides past donation history about each constituent and behavioural data provides if a constituent attends events, volunteers, opens charity emails, or watches charity videos. Using this, computers can learn how much affinity a constituent has for a charity which allows us to determine whether the ML models performance improves when only donation and behavioural data are included.

Further the best ML models were built using the architectures mentioned in Section 6.2 and Section 5.3. The input dimension for training and testing the ML models for this experiment is provided in Subsection 5.3 and the data preparation are provided in Section 4.

6.5.3 Results and Discussion

Table 6.37 shows the results for classifying 1485 donors (which includes 719 major donors and 766 are non-major donors) as major donors or non-major donors. Out of 719 major donors, 568 major donors are classified correctly with precision of 82.67% (Table 6.43) and the remaining 151 major donors are classified as non-major donors by LSTM-GRU. Out of 766 donors in the non-major donors category, 647 are classified as non-major donors with recall of 78.99% (Table 6.44) and the remaining 119 are classified as major donors. From the results produced for EC-1 charity using donation and behavioural data, the best performing model was LSTM-GRU. The test accuracy of the model for classifying the predicted value is 81.81%±0.112, which when compared to Experiment 1 (Best accuracy: 94.47%±0.127, False positive: 31, False negative: 51), Experiment 2 (Best accuracy: 89.15%±0.058, False positive: 111, False negative: 50) with all features and Experiment 3 (Best accuracy: 85.65%±0.016, False positive: 143, False negative: 70) with only donation data, shows slight decrease in ML models accuracy, but higher false positive values than Experiment 1 and 2 for EC-1 charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values when compared to Experiment 1 and 2 for EC-1 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	568	119	151	647
RNN	586	205	133	561
GRU	580	167	139	599
BDLSTM-GRU-TDL	586	176	133	590
BDLSTM-CNN	594	178	125	588

Table 6.37: Results for predicting major donor prospects using donation and behavioural data for EC-1 charity with 1485 constituents.

Table 6.38 shows the results using donation and behavioural data on EC-2 data for classifying 2120 donors as major donors or non-major donors. Out of 908 major donors, 740 major donors are classified correctly with precision of 74.59% (Table 6.43) and the remaining 168 major donors are classified as non-major donors. Conversely, out of 1212 donors in the non-major donors category, 960 are classified as non-major donors with recall of 49.07% (Ta-

ble 6.44) and the remaining 252 are classified as major donors, as these are the constituents we are looking for, who might give a major gift which the model predicted positive. As per the results from Table 6.42 for charity EC-2, the best performing model using donation data was LSTM-GRU. The test accuracy of the model for classifying the predicted value is 80.18%±0.133, which when compared to Experiment 1 (Best accuracy: 91.83%±0.045, False positive: 87, False negative: 86), Experiment 2 (Best accuracy: 81.98%±0.027, False positive: 238, False negative: 144) with all features and Experiment 3 (Best accuracy: 86.17%±0.023, False positive: 157, False negative: 136) with only donation data, shows slight decrease in ML models accuracy, but higher false positive values than Experiment 1, 2 and 3 for EC-1 charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values than false negatives when compared to Experiment 1, 2 and 3 for EC-2 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	740	252	168	960
RNN	686	285	222	927
GRU	699	258	209	954
BDLSTM-GRU-TDL	754	313	154	899
BDLSTM-CNN	759	298	149	914

Table 6.38: Results for predicting major donor prospects using donation and behavioural data for EC-2 charity with 2120 constituents.

The results on EC-3 charity for classifying 374 donors as major donors or non-major donors is shown in Table 6.39. Out of 222 major donors, 154 major donors are classified correctly with precision of 75.86% (Table 6.43), while the other 68 major donors are classified as non-major donors by BDLSTM-GRU-TDL. In the non-major donors category, 103 donors are classified as non-major donors with recall of 69.36% (Table 6.44), while the remaining 49 are classified as major donors. The best performing model using donation data for charity EC-3 was BDLSTM-GRU-TDL (as seen in Table 6.42). The model's test accuracy in classifying the predicted value is 68.71%±0.097, which when compared to Experiment 1 (Best accuracy: 96.79%±0.020, False positive: 3, False negative: 9), Experiment 2 (Best accuracy: 85.56%±0.016, False positive: 33, False negative: 21) with all features and Experiment 3 (Accuracy:

66.84%±1.110, False positive: 48, False negative: 76) with only donation data, shows decrease in ML models accuracy than Experiment 1 and 2, but higher false positive values when compared to Experiment 1, 2 and 3 for EC-3 charity. From the analysis, BDLSTM-GRU-TDL model is performing well with respect to higher false positive values when compared to Experiment 1, 2 and 3 for EC-3 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	119	43	103	109
RNN	147	74	75	78
GRU	130	55	92	97
BDLSTM-GRU-TDL	154	49	68	103
BDLSTM-CNN	122	24	100	128

Table 6.39: Results for predicting major donor prospects using donation and behavioural data for EC-3 charity with 374 constituents.

The RC-1 charity results for classifying 1114 donors as major or non-major donors is shown in Table 6.40. Out of 512 major donors, 487 major donors are classified correctly with precision of 85.28% (Table 6.43), while the other 25 major donors were classified as non-major donors by RNN. In the non-major donors category, 518 donors are classified as non-major donors with recall of 95.11% (Table 6.44), and the remaining 84 are the donors we are looking for who could give major gifts, that the model actively predicted positive. The best performing model using donation data for charity RC-1 was RNN (as seen in Table 6.42). The test accuracy of the model when classifying the predicted value is 90.21%±0.034, which when compared to Experiment 1 (Best accuracy: 93.35%±0.264, False positive: 42, False negative: 32), Experiment 2 (Best accuracy: 92.19%±0.012, False positive: 110, False negative: 40) with all features and Experiment 3 (Best accuracy: 94.34%±0.019, False positive: 32, False negative: 31) with only donation data, shows slight decrease in ML models accuracy, but higher false positive values than Experiment 1 and 3 for RC-1 charity. From the analysis, LSTM-GRU model from Experiment 2 is performing well with respect to higher false positive values when compared to Experiment 1, 3 and 4 for RC-1 charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	499	104	13	498
RNN	487	84	25	518
GRU	492	120	20	482
BDLSTM-GRU-TDL	476	102	36	500
BDLSTM-CNN	495	132	17	470

Table 6.40: Results for predicting major donor prospects using donation and behavioural data for RC-1 charity with 1114 constituents.

Table 6.41 shows the results for a cancer charity for classifying 51 donors as major or non-major donors. Out of 20 major donors, 13 major donors are classified correctly with precision of 61.90% (Table 6.43), while the remaining 9 major donors were classified as non-major donors by LSTM-GRU. In the non-major donors category, 21 donors are classified as non-major donors with recall of 59.09% (Table 6.44), and the remaining 8 are the donors we are looking for who could give major gifts, which the model actively predicted positive. The best performing model using donation data for a cancer charity was LSTM-GRU (as seen in Table 6.42). The test accuracy of the model when classifying the predicted value is 81.39%±0.112, which when compared to Experiment 1 (Best accuracy: 96.07%±0.170, False positive: 1, False negative: 1), Experiment 2 (Best accuracy: 94.11%±0.017, False positive: 1, False negative: 2) with all features and Experiment 3 (Accuracy: 74.50%±0.009, False positive: 7, False negative: 6) with only donation data, shows decrease in ML models accuracy than Experiment 1 and 2, but higher false positive values than Experiment 1, 2 and 3 for a cancer charity. From the analysis, LSTM-GRU model is performing well with respect to higher false positive values when compared to Experiment 1, 2 and 3 for a cancer charity.

	TP	**FP**	**FN**	**TN**
LSTM-GRU	13	8	9	21
RNN	15	6	7	23
GRU	10	9	12	20
BDLSTM-GRU-TDL	16	10	6	19
BDLSTM-CNN	14	7	8	22

Table 6.41: Results for predicting major donor prospects using donation and behavioural data for a cancer charity with 51 constituents.

Models	**LSTM-GRU**	**RNN**	**GRU**	**BDLSTM-GRU-TDL**	**BDLSTM-CNN**
AlzC	87.50%±0.029	82.33%±0.038	82.23%±0.031	83.33%±0.034	**88.42%±0.003**
CC	**81.39%±0.135**	74.50%±0.020	58.82%±0.190	68.62%±0.215	70.58%±0.011
EC-1	**81.81%±0.112**	77.23%±0.096	79.39%±0.111	79.19%±0.104	79.59%±0.107
EC-2	**80.18%±0.133**	76.08%±0.077	77.97%±0.081	77.91%±0.080	78.91%±0.092
EC-3	60.96%±0.056	60.16%±0.053	61.69%±0.065	**68.71%±0.097**	66.84%±0.088
EC-4	86.67%±0.017	85.31%±0.013	**88.02%±0.027**	87.65%±0.021	84.46%±0.017
RC-1	89.49%±0.029	**90.21%±0.034**	87.43%±0.019	87.61%±0.020	86.62%±0.019
RC-2	83.33%±0.033	77.56%±0.002	76.28%±0.005	**86.53%±0.022**	82.05%±0.039
Mean	**81.41%±0.068**	77.92%±0.041	76.47%±0.066	79.94%±0.074	79.68%±0.047

Table 6.42: Accuracies for major donor prediction using donation and behavioural data for all charities.

Models	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**	**Mean**
LSTM-GRU	86.23%	61.90%	**82.67%**	**74.59%**	73.45%	83.70%	82.75%	78.65%	77.99%
RNN	**97.23%**	**71.42%**	74.08%	70.64%	66.51%	83.47%	**85.28%**	76.92%	78.19%
GRU	86%	52.63%	77.64%	73.04%	70.27%	84.77%	80.39%	79.41%	75.51%
BDLSTM-GRU-TDL	88.11%	61.53%	76.90%	70.66%	75.86%	**84.96%**	82.35%	84.14%	78.06%
BDLSTM-CNN	82.12%	66.66%	76.94%	71.80%	**83.56%**	80.85%	78.94%	**85.50%**	**78.29%**

Table 6.43: Precision for major donor prediction using donation and behavioural data for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	76.92%	59.09%	78.99%	49.07%	53.60%	90.71%	**97.46%**	**90.90%**	74.59%
RNN	69.23%	68.18%	81.50%	75.55%	66.21%	87.67%	95.11%	77.92%	77.67%
GRU	67.03%	45.45%	80.66%	76.98%	58.55%	92.38%	96.09%	70.12%	73.40%
BDLSTM-GRU-TDL	66.66%	**72.72%**	81.50%	83.03%	**69.36%**	91.17%	92.96%	89.61%	**85.33%**
BDLSTM-CNN	**80.55%**	63.65%	**82.61%**	**83.59%**	54.95%	**92.54%**	96.67%	76.62%	78.89%

Table 6.44: Recall for major donor prediction using donation and behavioural data for all charities.

Models	AlzC	CC	EC-1	EC-2	EC-3	EC-4	RC-1	RC-2	Mean
LSTM-GRU	81.30%	60.46%	**80.78%**	59.19%	61.97%	87.07%	89.50%	84.33%	75.57%
RNN	81.23%	**69.76%**	77.61%	73.01%	66.35%	85.52%	**89.92%**	77.41%	77.60%
GRU	**81.81%**	48.77%	79.12%	74.95%	63.87%	**88.41%**	87.54%	74.48%	**80.80%**
BDLSTM-GRU-TDL	80%	66.65%	79.13%	76.34%	**72.45%**	87.95%	87.33%	**86.79%**	79.58%
BDLSTM-CNN	81.32%	65.12%	79.67%	**77.24%**	66.30%	86.30%	86.90%	80.82%	77.95%

Table 6.45: F1-Score for major donor prediction using donation and behavioural data for all charities.

6.5.4 Summary

From this experiment, the false positives for educational charities (EC-1, EC-2 and EC-3), religious charity (RC-1) and a cancer charity has been improved, but minor drop in accuracies (as seen in Table 6.42) when comparing Experiment 1, 2 and 3 with using only donation and behavioural data for predicting major donor prospects for charities. We experimented with five different ML models for 8 charities. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU. RNN model is performing better for RC-1 charity when comparing to other models in Experiment 1 and 3 with respect to more false positive values, but with slight decrease in accuracies. This suggests that ML models using donation and behavioural data are performing better on more than one educational charities, religious and a cancer charity data compared to Experiment 1, 2 and 3, in particular with respect to accuracy and more false positives. Based on the analysis of each models during training, we used the best performing model obtained for each charities using donation and behavioural data to predict future major donors. For example, for EC-2 charity, the best performing model was LSTM-GRU, we used that model to make predictions on the test data. Table 6.46 shows the final predicted major and non-major donors on the test set.

Labels	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
1 (Predicted Major donors)	9068	9885	9261	1446	28065	1586	11887	8192
0 (Predicted Non-Major donors)	81791	42238	95416	74709	26056	209933	52956	93267

Table 6.46: Major donors for all the charities using donation and behavioural data.

6.6 Experiment 5: Predicting how much money major donor constituents will contribute to the charity

6.6.1 Objective

Some major gifts are much larger than others, as any donor who donates $10,000 is considered a major donor, but a donor who donates $1,000,000 is not the same as someone who donates $10,000. The objective of this experiment is to build a regression model for predicting how much money the major donor constituents will contribute to the charities.

6.6.2 Data and Approach

The data we used for this experiment is from 5 charities that deal with a religious charity and 4 educational charities (Table 4.1) with all the features included. Initially, a regression model was built on all the five charities. We used box-cox transformation to handle the highly skewed data [2], based on exploratory data analysis (EDA) on the dataset in the pre-processing stage and imported the scipy function $boxcox1p$ which computes the box-cox transformation of $1 + x$:

$$if(\lambda! = 0); y = \frac{((1+x)^{\lambda} - 1)}{\lambda} \tag{6.1}$$

$$if(\lambda == 0); y = \log(1+x) \tag{6.2}$$

[2]Skewness refers to a distortion or asymmetry that deviates from the symmetrical bell curve, or normal distribution, in a set of data.

We used the below five regression models in this experiment:

- LASSO regression: The Robustscaler() technique was employed on a scikit-learn pipeline to make the model more robust and reduce outliers. *alpha* set to 0.0005 and random_state = 1.

- Elastic Net regression: The Robustscaler() technique was employed on a scikit-learn pipeline to make the model more robust, as outliers are particularly sensitive to it. The scikit-learn python machine learning library provided an implementation for the Elastic Net penalized regression algorithm via the *ElasticNet* class. The *alpha* hyperparameter is set via the $l1\_ratio$ argument that controls the contribution of the L1 and L2 penalties and the *lambda* hyperparameter is set via the *alpha* argument that controls the contribution of the sum of both penalties to the loss function. Value of 0.9 is used for $l1\_ratio$ and 0.0005 is used for *alpha*.

- Gradient Boosting Regression: The parameters used for defining this model are explained as follows: n_estimators: The number of trees in the forest. max_depth = 4: The maximum depth of a tree and used to control over-fitting, it is tuned during cross validation. min_samples_split = 10: The minimum number of samples required in a node for splitting and higher values can lead to under-fitting. learning_rate = 0.05: It determines the impact of each tree on the final outcome, lower values make the model robust and thus allowing it to generalize well. max_features = sqrt: Number of features to consider while searching for a best split, square root of the total number of features works well as higher values can lead to over-fitting. min_samples_leaf = 15: Minimum samples required in a terminal node or leaf. random_state = 5: The random number seed so that same random numbers are generated every time. Loss = huber: Refers to the loss function to be minimized in each split.

Hyper parameters	Values
n_estimators	estimator
max_depth	4
min_samples_split	10
learning_rate	0.05
max_features	sqrt
min_samples_leaf	15
random_state	5
loss	huber

Table 6.47: Gradient Boosting Regression parameters.

- XGBoost Regression: It is defined by creating an instance of the XGBRegressor class. colsample_bytree = 0.4603: It is similar to max_features in gradient boosting which denotes the fraction of columns to be randomly samples for each tree and ranges from 0.5 to 1. gamma = 0.0468: A node is split only when the resulting split gives a positive reduction in the loss function, gamma specifies the minimum loss reduction required to make a split. learning_rate = 0.01: It determines the impact of each tree on the final outcome, lower values make the model robust and thus allowing it to generalize well. max_depth = 5: The maximum depth of a tree. random_state = 7: The random number seed so that same random numbers are generated every time. nthread = -1: This is used for parallel processing and takes the number of cores in the system.

Hyper parameters	Values
colsample_bytree	0.4603
gamma	0.0468
learning_rate	0.01
max_depth	5
random_state	7
nthread	-1

Table 6.48: XGBoost Regression parameters.

- LightGBM Regression: The parameters used for defining this model are explained as follows: objective = regression: It specifies the learning task and the corresponding learning objective. num_leave = 5: Maximum tree leaves for base learners. learning_rate = 0.05: It determines the impact of each tree on the final outcome. n_estimators = 720: Number of trees to fit. max_bin = 55: Used to control over-fitting and ranges from 2 to ∞. bagging_fraction = 0.8: It randomly samples the training data without re-sampling which ranges from 0 to 1. bagging_freq = 5: 0 means disable bagging, 5 means perform bagging at every 5 iteration. feature_fraction = 0.2319: The percentage of features selected. feature_fraction_seed = 9: Random seed for feature_fraction. bagging_seed = 9: Random seed for bagging. min_data_in_leaf = 6: Minimal number of data in one leaf which is used to deal with over-fitting. min_sum_hessian_in_leaf = 11: Minimal sum hessian in one leaf which is similar to min_data_in_leaf.

Hyper parameters	**Values**
objective	regression
num_leave	5
learning_rate	0.05
n_estimators	720
max_bin	55
bagging_fraction	0.8
bagging_freq	5
feature_fraction	0.2319
feature_fraction_seed	9
bagging_seed	9
min_data_in_leaf	6
min_sum_hessian_in_leaf	11

Table 6.49: LightGBM Regression parameters.

6.6.3 Results and Discussion

6.6.3.1 Cross Validation:

We evaluated the five base-models on the dataset using k-fold cross-validation and reported the root mean square error (RMSE) of all the models in the given

dataset. The function *rmse_cv* is used to train all the individual models in the 2 folds of the data created and it returns the RMSE value for the model based on the out of fold predictions compared with the actual predictions. Initially we used k=5, which saw enormous standard deviations and RMSE scores (as seen in Table 6.50 and 6.51), as we need to lower both the scores, we used k=2 on the dataset.

Regression Models	**RC-1**	**EC-1**	**EC-2**	**EC-3**	**EC-4**
Lasso Regression	$258448.63	$223652.65	$347540.59	$261499.17	$291480.58
ElasticNet Regression	$472509.48	$194308.94	$366233.13	$183405.14	$118827.32
Gradient Boosting Regression	**$45334.53**	$104758.52	$152763	$86907.56	$954214.66
XGboost Regression	$46381.82	**$12640.56**	**$20290.70**	$75250.36	**$80876.37**
Light GBM Regression	$82356.73	$102871.25	$160563.75	**$69442.64**	$108745.74
Average Stacking (GBM, Lasso, XGB)	$95474.92	$139138.75	$209959.58	$115533.74	$102453.23
Meta GBM- Average Models (LGB, GBM, XGBOOST)	$51927.97	$99384.63	$155030.92	$69602.44	$909048.92

Table 6.50: RMSE values using 5-fold for major donor amount predictions for five charities.

Regression Models	**RC-1**	**EC-1**	**EC-2**	**EC-3**	**EC-4**
Lasso Regression	$331245.01	$87686.34	**$68099.54**	$68250.12	$540843.74
ElasticNet Regression	$836605.63	**$72744.80**	$110989.59	$19765.92	$233607.43
Gradient Boosting Regression	$68215.16	$90840.35	$114423.57	$47238.25	$121439.38
XGboost Regression	$68863.28	$92200.07	$95428.72	$46606.04	**$10086.8**
Light GBM Regression	**$50111.75**	$75215.53	$95250.82	$35895.21	$894080.75
Average Stacking (GBM, Lasso, XGB)	$126886.63	$89258.97	$96129.31	**$27472.01**	$177722.97
Meta GBM- Average Models (LGB, GBM, XGBOOST)	$65024.73	$90125.47	$107883.93	$45993.62	$1055451.43

Table 6.51: Standard deviation values using 5-fold for major donor amount predictions for five charities.

6.6.3.2 Type 1: Simplest Stacking Regressor approach: Averaging Base models

We begin this simple approach of averaging base models. We build a new class to extend scikit-learn with our model and defined clones of the original models to fit the data in, trained the cloned base models, finally, computed the predictions for cloned models and averaged them.

6.6.3.3 Type 2: Adding a Meta-model

In this approach, we trained all the base models and used the predictions (out-of-fold predictions) of the base models as a training feature to the meta-model. The meta-model is used to find the pattern between the base model predictions as features and actual predictions as the target variables.

- Split the data into 2 sets training and test set.
- Train all the base models with the training data.
- Tested base models on the holdout dataset and stored the predictions (out-of-fold predictions).
- Used the out-of-fold predictions made by the base models as input features, and the correct output as the target variable to train the meta-model.

Since, k=2 we trained the model on the 1 fold and predicted on the holdout set (2^{nd} fold). Repeating this step for 2 times gives the out-of-fold predictions for the whole dataset. This is done for all the base models. Then meta-model is trained using the out-of-predictions by all the models as X and the original target variable as y. Prediction of this meta-model is be considered as the final prediction.

For this experiment, we used datasets from 5 charities (Table 4.1) with all the features included. Each charity's lowest RMSE and Standard deviation are highlighted. In conclusion, based on the RMSE values on the training set, the best performing model was Light GBM Regression to predict how much money major donor constituents will contribute to the charity on the test data.

Regression Models	**RC-1**	**EC-1**	**EC-2**	**EC-3**	**EC-4**
Lasso Regression	$51903.41	$22453.8	$19737.69	$38269.37	$22478.91
ElasticNet Regression	$44771.90	$21334.6	$19711	$38130.86	$22471.19
Gradient Boosting Regression	$17992.67	$4384.9	$4082.16	$10243.12	$4869.83
XGboost Regression	$15692.46	$12594.5	$14214.07	$18603.49	$16541.49
Light GBM Regression	$16823.69	**$3329.7**	**$2867.26**	**$9940.21**	**$3555.06**
Average Stacking (GBM, Lasso, XGB)	$21598.23	$10606.03	$9456.41	$14850.59	$11070.69
Meta GBM- Average Models (LGB, GBM, XGBOOST)	**$15130.14**	$5842.5	$5982.04	$11495.41	$7033.43

Table 6.52: RMSE values using 2-fold for major donor amount predictions for five charities.

Regression Models	**RC-1**	**EC-1**	**EC-2**	**EC-3**	**EC-4**
Lasso Regression	$28673.67	$1404.05	$2041.35	$6860.66	$845.03
ElasticNet Regression	$23589.10	**$264.76**	$2018.72	$6720.18	$877.01
Gradient Boosting Regression	**$3582.16**	$473.33	**$5.09**	$1626.28	$447.05
XGboost Regression	$8888.82	$1532.26	$449.68	**$508.90**	$534.99
Light GBM Regression	$3818	$556.75	$21.37	$1840.09	$206
Average Stacking (GBM, Lasso, XGB)	$11665.67	$610.70	$596.89	$1801.70	$279.38
Meta GBM- Average Models (LGB, GBM, XGBOOST)	$6980.59	$984.15	$308.49	$1169.17	**$122.30**

Table 6.53: Standard deviation values using 2-fold for major donor amount predictions for five charities.

Chapter 7

Conclusion and Future Work

The goal of this thesis was to accurately predict major donor prospects for a number of different non-profit organizations and investigate which machine learning algorithms and data types are best to make these predictions. In addition, we predicted the amount of money that the major donor constituents will contribute to the charity. Real-world major donor and non-major donor data is utilized to train and test the machine learning models, which are then interrogated to determine the major donors to the charity.

7.1 Contributions

Our research shows that machine learning, that we have effectively developed in terms of specific metrics, will accurately predict future major donors. Machine learning models used to model the major donor data in this research are LSTMGRUs, SimpleRNNs, GRUs, BDLSTM-GRU-TDLs, BDLSTMC-NNs, random forest classifier, Adaboost, gradient boosting, extra trees classifier, gaussian naive bayes, decision tree, logistic regression, LASSO regression, elastic net, Light GBM and XGBoost. The data types used in our research are: demographic, donation, behavioural and educational data.

In Experiment 1, we experimented with six different ML models for 8 charities in order to accurately predict future major donor prospects. For the ML models used in this research, we are looking for bigger false positive values than false negatives, which indicates who might give a major gift. Based on the mean accuracy and confusion matrices values on the training set, the best performing model was random forest classifier (as seen in Table 6.7) to predict

major donors on the test data. However, the false positive values for educational charities (EC-1 (Table 6.2) and EC-3 (Table 6.4)) and a cancer charity (Table 6.6) are less than false negative values and needed to be increased. Gaussian Naive Bayes, predicts with more false positives for educational charities (EC-1, EC-2 and EC-3), religious (RC-1) and a cancer charity than other models, but with less accuracy as seen in Table 6.7.

In Experiment 2, the false positives have been improved for educational (EC-1, EC-2 and EC-3) and religious (RC-1) charities (as seen in Table 6.22) using deep learning models. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU to predict major donors on the test data. The confusion matrix values for LSTM-GRU model have been improved for EC-1 charity when comparing to Experiment 1 with slight decrease in accuracies (Table 6.22). But with a cancer charity, the false positive values remain the same (Table 6.21) as of Experiment 1. RNN and BDLSTM-GRU-TDL model are performing better for EC-3 and cancer charities when comparing to other models in Experiment 1 with respect to more false positive values, but with slight decrease in accuracies. This suggests that ML models using deep learning models are performing better with higher false positive values on most of the charities data compared to Experiment 1.

In Experiment 3, the false positive values have been improved for educational charities (EC-1, EC-2 and EC-3) and a cancer charity, but there was a slight decrease in accuracies (as seen in Table 6.32) when compared to Experiment 1 and 2 including all the features. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU to predict major donors on the test data. The results from this experiment suggests that, LSTM-GRUs (as seen in Table 6.32), predicts with more false positives than false negative values for educational charities (EC-1, EC-2 and EC-3) and a cancer charity using only donation data. RNN and GRU, predicts with more false positives for educational charities (EC-1, EC-2 and EC-3), religious (RC-1) and a cancer charity than other models, but with less accuracy as seen in Table 6.32.

In Experiment 4, the false positives for educational charities (EC-1, EC-2 and EC-3), religious charity (RC-1) and a cancer charity has been improved, but minor drop in accuracies (as seen in Table 6.42) when comparing Experiment 1, 2 and 3 with using only donation and behavioural data for predicting major donor prospects for charities. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU. RNN model is performing

better for RC-1 charity when comparing to other models in Experiment 1 and 3 with respect to more false positive values, but with slight decrease in accuracies. This suggests that ML models using donation and behavioural data are performing better on more than one educational charities, religious and a cancer charity data compared to Experiment 1, 2 and 3, in particular with respect to accuracy and more false positives.

Of all these, including all the features turned out to be the better model with more accurate results when predicting future major donors as seen in Experiment 1 and 2. Among the deep learning models developed and from the results on Section 6.3, the best performing model chosen for all the charities was LSTM-GRU to predict major donors on the test data as shown in Table 6.22 with mean accuracy of 88.64%±0.030. Where as, using supervised learning models for predicting major donors, random forest classifier is performing better compared to other models for all charities with mean accuracy of 90.85%±0.089, but with fewer false positives when compared to Experiment 3 and 4.

Table 7.1 summarizes the best ML model for each charity in terms of best accuracy, false positive values and type of data used (with all the features, with only donation data and with donation and behavioural data).

Using the analysis from Section 6.2, 6.3, 6.4 and 6.5, we predicted on how much money the major donor constituents will contribute to the charity as mentioned in Section 6.6.

Models	**AlzC**	**CC**	**EC-1**	**EC-2**	**EC-3**	**EC-4**	**RC-1**	**RC-2**
Accuracy (All features)	89.74%±0.032	92.15%±0.017	88.88%±0.058	79.38%±0.027	85.56%±0.016	91%±0.008	92.19%±0.012	**96.94%±0.016**
False positives	2	1	111	238	33	50	110	3
ML Model Used	BCNN	BGRU-TDL	LSTM-GRU	LSTM-GRU	RNN	GRU	LSTM-GRU	GRU
Accuracy (using SL)	**92.85%±0.046%**	**96.07%±0.170**	**94.47%±0.127**	**91.83%±0.045**	**96.79%±0.020**	90.44%±0.159	93.35%±0.264	96.77%±0.014
False Positives	1	1	31	87	3	29	42	1
ML Model Used	RF	LR	RF	AB	DT	RF	LR	RF
Accuracy (only donation data)	88.33%±0.031	74.50%±0.009	85.65%±0.016	86.17%±0.023	66.84%±1.110	**93.99%±0.016**	**94.34%±0.019**	83.57%±0.041
False Positives	1	7	143	157	48	123	32	10
ML Model Used	LSTM-GRU	BCNN	LSTM-GRU	BGRU-TDL	GRU	LSTM-GRU	BCNN	LSTM-GRU
Accuracy (only donation and behavioural data)	88.42%±0.003	81.39%±0.135	81.81%±0.112	80.18%±0.133	68.71%±0.097	88.02%±0.027	90.21%±0.034	86.53%±0.022
False Positives	1	8	119	252	49	116	84	10
ML Model Used	BCNN	LSTM-GRU	LSTM-GRU	LSTM-GRU	BGRU-TDL	GRU	RNN	BGRU-TDL

Table 7.1: Summary of best ML model based on experiments performed in this research (SL: Supervised Learning, BCNN: BDLSTM-CNN, BGRU-TDL: BDLSTM-GRU-TDL, RF: Random Forest, LR: Logistic Regression, AB: Adaboost, DT: Decision Tree).

The following is a summary of the findings from this research:

1. Built various supervised learning models with all the features that predicted future major donors for eight charities. Based on the mean accuracy and confusion matrices values on the training set, the best performing model was random forest classifier (as seen in Table 6.7) to predict major donors on the test data.

2. Generated deep learning models that predicted future major donor prospects for eight different charities with all the features included. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU to predict major donors on the test data. The false positives have been improved for educational (EC-1, EC-2 and EC-3) and religious (RC-1) charities (as seen in Table 6.22) using deep learning models. But with a cancer charity, the false positive values remain the same (Table 6.21) as of Experiment 1. RNN and BDLSTM-GRU-TDL model are performing better for EC-3 and cancer charities when comparing to other models in Experiment 1 with respect to more false positive values, but with slight decrease in accuracies. This suggests that ML models using deep learning models are performing better with higher false positive values on most of the charities data compared to Experiment 1.

3. Built ML models with only donation data: The false positive values have been improved for educational charities (EC-1, EC-2 and EC-3) and a cancer charity, but there was a slight decrease in accuracies (as seen in Table 6.32) when compared to Experiment 1 and 2 including all the features. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU to predict major donors on the test data. The results from this experiment suggests that, LSTM-GRUs (as seen in Table 6.32), predicts with more false positives than false negative values for educational charities (EC-1, EC-2 and EC-3) and a cancer charity using only donation data. RNN and GRU, predicts with more false positives for educational charities (EC-1, EC-2 and EC-3), religious (RC-1) and a cancer charity than other models, but with less accuracy as seen in Table 6.32.

4. Built ML models with only donation and behavioural data: The false positives for educational charities (EC-1, EC-2 and EC-3), religious charity (RC-1) and a cancer charity has been improved, but with minor drop in accuracies (as seen in Table 6.42) when comparing Experiment 1, 2

and 3 with using only donation and behavioural data for predicting major donor prospects for charities. Based on the mean accuracy on the training set, the best performing model was LSTM-GRU. RNN model is performing better for RC-1 charity when comparing to other models in Experiment 1 and 3 with respect to more false positive values, but with slight decrease in accuracies. This suggests that ML models using donation and behavioural data are performing better on more than one educational charities, religious and a cancer charity data compared to Experiment 1, 2 and 3, in particular with respect to accuracy and more false positives.

5. Built a regression model for predicting how much money major donor constituents will contribute to the charity. Based on the lowest RMSE and standard deviation scores (as seen in Table 6.52) on the training set, the best performing model chosen was Light GBM Regression to predict how much money major donor constituents will contribute to the charity on the test data.

We propose that this research can be applied in predicting the future major donor prospects for various charities.

7.2 Understanding the Behaviour of the Models

1. Based on the performance matrices from Section 6.2, 6.3, 6.4 and 6.5, the ML models such as LSTM-GRU, BDLSTM-GRU-TDL, BDLSTM-CNN and random forest classifier can be used in predicting future major donors for a charity. Based on the analysis listed in the contributions Section 7.1, below models and data set shown in Table 7.2 can be used to create the list of potential major donors for the charities.

2. Data with only donation features improved the false positives with slight decrease in accuracies for educational charities (EC-1, EC-2 and EC-3) and a cancer charity for an LSTM-GRU model.

3. Using only donation and behavioural data, the false positives for educational charities (EC-1, EC-2 and EC-3), religious (RC-1) and a cancer

Models	Data set	ML Model
AlzC	All features dataset	BDLSTM-CNN
CC	Only donation and behavioural data	LSTM-GRU
EC-1	All features dataset	LSTM-GRU
EC-2	Only donation and behavioural data	LSTM-GRU
EC-3	All features dataset	RNN
EC-4	Only donation data	LSTM-GRU
RC-1	Only donation data	BDLSTM-CNN
RC-2	Only donation and behavioural data	BDLSTM-GRU-TDL

Table 7.2: ML models which can be used by charities for creating major donor prospects list.

charity has been improved, but with minor drop in accuracies (as seen in Table 6.42) when comparing to Experiment 1, 2 and 3.

4. A charity could choose the random forest model if they seek high accuracy with fewer false positives.
5. If a charity with a low accuracy model but get more false positives then they can choose LSTM-GRU model.
6. If a charity wants in between point (4) and (5), they can use only donation and behavioural data.
7. Light GBM Regression model with lowest RMSE values and standard deviations (as seen in Table 6.52) in Section 6.6 can be used to predict how much amount major donor constituents will contribute to the charity.

7.3 Future Work

The current research can be advanced by taking the following steps:

1. Using wealth indicators which are publicly available data points about donors that provide insights into their income and wealth status. Wealth screening technology can seek out these records and compile important information for the organization. Wealth indicators can tell which of the prospects is financially capable of making a major gift and the likely size of that gift.

2. In the future, it will be interesting to explore other models such as univariate chi-square methods for features selection. SMOTE upsampling method that perturbs some of the features during upsampling could be implemented and compared with the current results.

3. Deep ANNs could be used to develop a regression model for predicting how much money major donor constituents would contribute to a charity in the future.

Bibliography

[1] Mohammed Alhamid. Bidirectional lstm. https://towardsdatascience.com/tagged/bidirectional-lstm, Accessed: July 6, 2021, 2021.

[2] C. Apte, E. Bibelnieks, R. Natajaran, E. Pednault, F. Tipu, and D Campbell. Segmentation-based modeling for advanced targeted marketing. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 408–413, 2001.

[3] The Fundraising Authority. 4 keys to building relationships with major donors. https://thefundraisingauthority.com/donor-cultivation/building-relationships-major-donors/, Accessed: December 21, 2021, 2021.

[4] Ben Miller Bill Levis and Association of Fundraising Professionals Cathlene Williams. 2019 fundraising effectiveness survey report. https://www.givingarchitects.com/portfolio/case-study-4/, Accessed: August 2nd 2021, 2019.

[5] Colah's blog. Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/, Accessed: January 17th 2021, 2015.

[6] Barbara E. Brittingham and Thomas R. Pezzullo. Fund raising in higher education / barbara e. brittingham and thomas r. pezzullo. https://catalogue.nla.gov.au/Record/5520361, Accessed: May 21st 2020, 1990.

[7] Jason Brownlee. A gentle introduction to rnn unrolling. https://machinelearningmastery.com/rnn-unrolling/, Accessed: September 28th 2021, 2019.

[8] Jonathan Burez and Dirk Van den Poel. CRM at a pay-tv company: Using analytical models to reduce customer attrition by targeted marketing for subscription services. *Expert Systems with Applications*, 32:277–288, 2005.

[9] Shubham Jaroli Anupam Shukla Saumil Maheshwari Chahes Chopra, Shivam Sinha. Recurrent neural networks with non-sequential data to predict hospital readmission of diabetic patients. *ICCBB 2017: Proceedings of the 2017 International Conference on Computational Biology and Bioinformatics*, page 18–23, October 2017.

[10] DonorSearch. Machine learning for nonprofits: The essential guide. `https://www.donorsearch.net/machine-learning-for-nonprofits/`, Accessed: January 20th 2022, 2020.

[11] IBM Cloud Education. Deep learning. `https://www.ibm.com/cloud/learn/deep-learning`, Accessed: January 9th 2021, 2020.

[12] Leily Farrokhvar, Azadeh Ansari, and Behrooz Kamali. Predictive models for charitable giving using machine learning techniques. `https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0203928`, Accessed: July 15th 2021, 2018.

[13] Mike George. Using machine learning to help nonprofits with fundraising activities. `https://aws.amazon.com/blogs/publicsector/using-machine-learning-nonprofits-fundraising-activities/`, Accessed: March 12th 2022, 2021.

[14] Major Gift's. Donor search. `https://www.donorsearch.net/major-gifts-guide/`, Accessed: January 20th 2022, 2021.

[15] Purnasai Gudikandula. The recurrent neural network (rnns). `https://towardsdatascience.com/the-recurring-neural-networks-rnns-b2c4c088eaf`, Accessed: October 17th 2020, 2019.

[16] John Haydon. How to write your best fundraising emails — social media today. `https://www.socialmediatoday.com/content/how-write-your-best-fundraising-emails`, Accessed: June 3rd 2020, December 2014.

[17] Suleka Helmini. All you need to know about rnns. https://towardsdatascience.com/all-you-need-to-know-about-rnns-e514f0b00c7c, Accessed: November 4th 2020, 2019.

[18] Douw Boshoff Joseph Awoamim Yacim. Impact of artificial neural networks training algorithms on accurate prediction of property values. ImpactofANNtrainingalgorithms.pdf, Accessed: August 26th 2020, 2018.

[19] Daniel Jurafsky and James H. Martin. Logistic regression. https://web.stanford.edu/~jurafsky/slp3/5.pdf, Accessed: January 4th 2022, 2021.

[20] Eda Kavlakoglu. Ai vs. machine learning vs. deep learning vs. neural networks: What's the difference? https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks, Accessed: May 9th 2022, 2020.

[21] Shahrukh Khan. Data science explained: Decision trees. https://www.godatadrive.com/blog/data-science-for-business-leaders-decision-trees, Accessed: May 7th 2022, 2021.

[22] Patricia Knowles and Roger Gomes. Building relationships with major-gift donors: A major-gift decision-making, relationship-building model. page 385, 2009.

[23] Simeon Kostadinov. Understanding gru networks. https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be, 2017.

[24] Maxwell Stinchcombe Kurt Hornik. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[25] Greg Lee, Ajith Kumar Veera Raghavan, and Mark Hobbs. Machine learning the donor journey. In Cyril Goutte and Xiaodan Zhu, editors, *Advances in Artificial Intelligence - 33rd Canadian Conference on Artificial Intelligence, Canadian AI 2020, Ottawa, ON, Canada, May 13-15, 2020, Proceedings*.

[26] Wesley Lindahl and Christopher Winship. A logit model with interactions for predicting major gifts doonors. *Research in Higher Education*, 35(6):729–743, 1994.

[27] Prateek Majumder. Gaussian naive bayes. https://iq.opengenus.org/gaussian-naive-bayes/, Accessed: January 4th 2022, 2021.

[28] Nexus Marketing. Major donor fundraising: Effective strategies for 2022. https://www.donorsearch.net/major-donor-fundraising/, Accessed: March 16th 2022, 2021.

[29] Gerlinda S. Melchiori. Alumni research: An introduction. https://onlinelibrary.wiley.com/doi/abs/10.1002/ir.37019886003, Accessed: June 27th 2021, 1988.

[30] Rene Blanchette Michael S. Connolly. Understanding and predicting alumni giving behavior. https://onlinelibrary.wiley.com/doi/abs/10.1002/ir.37019865107, Accessed: July 23rd 2020, 1986.

[31] Arnab Mondal. Complete guide on how to use lightgbm in python. https://www.analyticsvidhya.com/blog/2021/08/complete-guide-on-how-to-use-lightgbm-in-python/, Accessed: January 3rd 2022, 2021.

[32] Avinash Navlani. Adaboost classifier in python. https://www.datacamp.com/community/tutorials/adaboost-classifier-python, Accessed: November 20th 2022, 2018.

[33] Prasad Patil. What is exploratory data analysis? https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15, Accessed: November 21st 2020, 2018.

[34] Michael Phi. Illustrated guide to lstm's and gru's: A step by step explanation. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21, Accessed: March 13th 2021, 2018.

[35] Anima Anandkumar Yisong Yue Rose Yu, Stephan Zheng. Long-term forecasting using higher order tensor rnns.

[36] Sparkle Russell-Puleri. Gated recurrent units explained using matrices. https://towardsdatascience.com/gate-recurrent-units-explained-using-matrices-part-1-3c781469fc18, Accessed: August 27th 2021, 2019.

[37] Abhinav Sagar. 5 techniques to prevent overfitting in neural networks. https://www.kdnuggets.com/2019/12/5-techniques-prevent-overfitting-neural-networks.html, Accessed: September 27th 2021, 2019.

[38] Sumit Saha. A comprehensive guide to convolutional neural networks. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53, Accessed: January 24th 2021, 2018.

[39] Ensar Seker. Recurrent neural networks and lstm explained. https://purnasaigudikandula.medium.com/recurrent-neural-networks-and-lstm-explained-7f51c7f6bbb9, Accessed: August 21st 2021, 2020.

[40] Sagar Sharma. Activation functions in neural networks. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6, Accessed: November 12th 2021, 2017.

[41] Sheetal Sharma. Artificial neural network (ann) in machine learning. https://www.datasciencecentral.com/artificial-neural-network-ann-in-machine-learning/, Accessed: May 4th 2022, 2018.

[42] Simplilearn. An overview on multilayer perceptron (mlp). https://www.simplilearn.com/tutorials/deep-learning-tutorial/multilayer-perceptron, Accessed: May 3rd 2022, 2022.

[43] Piotr Skalski. Preventing deep neural network from overfitting. https://towardsdatascience.com/preventing-deep-neural-network-from-overfitting-953458db800a, Accessed: October 15th 2021, 2018.

[44] Kevin Tham. The backpropagation algorithm. https://kevintham.github.io/dsblog/2018/02/26/the-backpropagation-algorithm.html, Accessed: May 3rd 2022, 2018.

[45] Christopher Winship and Wesley E. Lindahl. Predictive models for annual fundraising and major gift fundraising. *Nonprofit Management and Leadership*, 1992.

[46] Liang Ye. A machine learning approach to fundraising success in higher education. Master's thesis, University of Victoria, 2017.

[47] Byron Wallace Ye Zhang. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. `https://arxiv.org/abs/1510.03820`, 2015.

[48] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B.*, 67(2):301–320, 2005.