Norwegian School of Economics

Bergen, Fall 2020

Modelling Probability of Default with Machine Learning

How well does machine learning perform and can it replace the standard methods?

Ruben Jæger-Pedersen

Supervisor: Jonas Andersson

Master Thesis, Economics and Business Administration

Major in Business Analytics

NORWEGIAN SCHOOL OF ECONOMICS

Acknowledgement

Working with this thesis has been a difficult, but an incredible learning experience by allowing me to get a better understanding of both machine learning and modelling of default in credit risk.

First, I would like to thank my supervisor Jonas Andersson, for helpful advice and conversation to guide me through this thesis. I would also like to thank Ingvar Ersland for valuable information and conversation regarding the current landscape of machine learning and modelling of default in Norway. This has been extremely helpful, and I believe has added an extra layer of depth to my thesis. I would also like to thank Lars Petter Haugen for valuable contribution.

Abstract

In this master thesis we apply a variation of different machine learning techniques on a dataset for credit card clients in Taiwan to model the probability of default.

In this master thesis, we apply machine learning techniques on a dataset for credit card clients in Taiwan to model the Probability of Default (PD). The machine learning methods used were the Logistic Regression, Decision Tree, Random Forest, XGBoost, K-Nearest Neighbor (KNN) and Neural Network. We use Receiver Operating Curve Area Under the Curve (ROC AUC) and Confusion Matrix to assess the performance of each of the models, where the ROC AUC is used as our main performance measurement.

We look into the standard methods of assessing credit and how the General Data Protection Regulation (GDPR) affects machine learning now and in the future.

Random Forest performed the best followed by XGBoost and Neural Network. The difference in ROC AUC score between the top four models were only 0.023, while the worst performers KNN and Decision Tree were far behind.

Keywords – Probability of Default, PD Modelling, Machine Learning, GDPR, IRB, Standard method

Contents

List of Figures

List of Tables

1. Introduction

Being able to accurately predict whether a customer is going to default or not is vital to the survival of credit lending companies. If the predictions are too strict, they will reject customers who would not default and therefore miss out on income. On the other hand, if the predictions are too lenient, they might lose money due to accepting customers who is going to default. The accuracy of credit companies is therefore incredibly important to how well they fare.

Different methods of reporting credit have been used for over 100 years and the earliest uses dates back to 1869. Conceivably, these methods were far simpler, and the earliest methods often revolved around the "gut feel" of the lender. (Marketplace.org, n.d.). These methods for reviewing credit continued to evolve and in 1989 one of the most widely used credit scoring systems to date, FICO, was created. The credit scoring system revolved around the gathering of financial related statistics in order to classify the borrower (FICO, n.d.).

Machine learning is by no means a new topic and was used as early as the 1950s, possibly even earlier (Kononenko, 2001). Despite being discovered this early, the use of machine learning did not "take off" until later. This was largely due to two factors: available data and computational power. In order for machine learning to be most effective the input data needs to be of a substantial size and quality. Secondly, machine learning in general requires a lot of computational resources, which were lacking back then. Today, companies store much more of their data, which opens up for of using tools like machine learning is there. Further, computational power has increased exponentially, while the prices have even decreased. Several studies have undertaken on this particular subject and Nordhaus (2001) found that the computational power increased by 55% on an annual basis from 1940 till around 2000.

In this thesis we will be looking at a few different machine learning methods. While some methods may prove to be better in terms of giving a better prediction, they may suffer from being subsequently harder to interpret. For banks and credit card companies, this may not be a problem, however as new laws aimed to protect the consumers such as GDPR, the requirement for transparency has increased with it.

1.1 Literature Review

Machine learning has not yet been used as a standalone method for extending credit or loans. Depending on the type of loan and whether the loan is targeted towards consumers or firms, a various of different credit scoring methods have been employed. For consumers, the most known credit scoring model is the FICO score. FICO score is calculated by assessing five components: Amount of Debt, Payment History, Length of Credit History, New Credit and Credit Mix. The weighting of each of the components is different, based on the estimated importance. The end result is a credit score, which is usually between 350 and 850. A higher end score indicates that the consumer has a low credit risk, whereas a lower score indicates a higher risk (FICO, pg. 4, 2018). For private firms, Moody's RiskCalc has been a commonly used tool in the United States, since it was released in 2000 (Falkenstein et al., 2000).

There have been several studies regarding credit scoring and machine learning. Ong et al. (2005) reviewed the performance of a series of machine learning techniques, including Decision Tree, Neural Network and Logistic Regression. The best performing technique was General Programming (GP), but that method is out of the scope for this thesis. Both the Logistic Regression and Neural Network resulted good predictions, while the decision tree performed poorly.

Hand and Henley (1996) performed a similar study of Neural Networks and Logistic Regression, albeit nine years earlier. While their paper indicated that both the Logistic Regression and Neural Network were good performers, their conclusion was more reserved; There is no best overall method and that what method to use is highly dependent on the data used (Hand & Henley, pg. 535, 1996).

Xiao et al. (2006) investigates similar machine learning methods to the two previous papers, but also includes K-Nearest Neighbor (KNN) and various variations of Neural Network. The study was done on three credit datasets from Germany, Australia and United States. Once again, the Logistic Regression were among the best performers, only slightly behind the top performer, Support Vector Machines. KNN performed poorly on two of the three datasets used. A total of three variations were used for the Neural Network, were two of the variations performed well.

Overall, the Logistic Regression and Neural Network were the two most commonly reviewed machine learning methods for these papers, where the performance of the two techniques were

similar. The explanatory ability of the techniques was a vocal discussion point in the paper of Xiao et al. (2006). The best model for overall interpretability was the Decision Tree, followed by the Logistic Regression. The Neural Network were among the worst in terms of interpretability due to its complexity in determining the output (Xiao et al., pg 431, 2006).

1.2 Thesis Structure

This Master Thesis is divided into 7 chapters, including the introduction. The 2nd chapter introduces theory regarding credit risk, machine learning and the specific machine learning methods used. Chapter 3 presents the dataset used for the empirical part of this thesis. It includes a brief description of the dataset itself and its variables. In chapter 4, we go through the methodology for the empirical part, where we do some changes to the dataset and the tuning of relevant hyperparameters. Chapter 5 presents the empirical results, while we in chapter 6 will discuss machine learning and general questions around it. We draw a conclusion of this thesis in chapter 7.

2. Theory

2.1 Credit Risk

Credit risk is the risk a lender takes due to the uncertainty whether a borrower will repay the amount of money borrowed plus other agreed fees, such as interest. The goal of the lender, as in any financial business, is to maximize their revenue and reduce costs. The income and costs for credit businesses stems largely from maximizing the volume of loans and interest revenue while minimizing the loss, or defaults, on each of these loans. This is the business rationale among the credit card companies, which the dataset in this thesis is based on.

2.2 Definintion of Default

When a customer is not able to repay a loan, it is defined as a default. While the definition of default is simple enough, it may be difficult to determine exactly when the customer actually has defaulted. After how many days or months can a loan be considered default?

The Basel Committee, which serve as a banking supervisor for its member, has released three accords, often referred to as The Basel Accords I to III, in 1998, 2004 and 2013 (BIS, n.d.). The purpose of these accords is to serve as regulations for banks and financial institutions. In the 2nd accord, Basel II, they define a default when either of two events have occurred:

1. “The bank considers that the obligor is unlikely to pay its credit obligations to the banking group in full, without recourse by the bank to actions such as realising security (if held)”.

2. “The obligor is past due more than 90 days on any material credit obligation to the banking group. Overdrafts will be considered as being past due once the customer has breached an advised limit or been advised of a limit smaller than current outstandings” (BIS, 2019b).

2.3 Assessment of Credit Risk as of Today

As of today, none of the banks or financial institutions use machine learning when extending credit or measuring credit risk. We briefly introduced two methods in chapter **1.2**, RiskCalc and FICO score, which are two commonly used method in United States. In Norway, they currently use two main approaches: The Standardized Approach and Internal-Rating-Based Approach (IRB) (Finanstilsynet, 2017). Both of these methods are based on the Basel Accords, more specifically Basel II. These two methods are used to calculate the amount of capital required by the bank.

The Standardized approach works by assigning risk weights to different exposures (BIS, 2019a). The risk weights are usually assigned in one of two ways. The first option is to use the Capital Requirements Regulations (CRR) template values to assess the borrowers' risk. The second option is to use external ratings of the borrowers' risk from a selected number of approved external rating bureaus (Finanstilsynet, 2017). However, this can only be done for some exposure classes defined by the national supervisor (BIS, n.d.).

For the IRB approach, the calculation of the capital requirement is done by one of two methods: The Foundation Approach and the Advanced Approach. The Foundation Approach requires the bank to compute only the probability of default (PD). The last two risk weights, Exposure At Default (EAD) and Loss Given Default (LGD) is calculated by the national supervisor (in Norway, "Finanstilsynet"). The Advanced Approach requires the bank to compute every risk weight in addition to PD.

In order to apply either of the IRB methods, one required approval from Finanstilsynet. If approved, the bank can use their own models to compute the risk weights, which is then applied to arrive at the capital requirement needed. Very few banks are granted allowance to use the IRB models, and in 2018 Finanstilsynet made this even stricter by adding a new requirement where the value of the corporate portfolio need to be at least 30 billion NOK to for the use IRB Approach (Finanstilsynet, 2018). Even after getting approval to create IRB models, there are strict requirements to the data used and its validation methods.

It is important to emphasize that the two approaches of IRB introduced above are used to identify the capital requirement for a bank. They are not used as a standalone decision tool based on predicted probability of default in order to grant or deny loans. Today, the banks still

use traditional financial data, such as equity and income, to decide whether or not to grant a loan and size of the loan.

2.4 Data and GDPR

Over the past few years, the amount of data has increased tremendously. According to the International Data Corporation, the amount of data increased by 16.6% last year and they expect the annual growth from 2019-2024 to be 17.8% (IDC, n.d.). As data simply is a form of information, this translates into more information being available and collected. Whether using machine learning algorithms or other means of decision making, more information is likely to improve the end result, assuming the data can be used. Depending on the source of the data, it can either be structured or unstructured data. Structured data is data that can be used without any need of heavy transforming, whereas instructed data needs to be processed to gain any valuable insight from it.

As the data and its availability has steadily increased over time, so has the amount of data that is being collected. The 25th of May 2018 the European Union released the new law "General Data Protection Regulation" (GDPR). The law is aimed at protecting the consumers and heavily punishing corporations for violating laws regarding consumer privacy (GDPR, n.d.d). Consumer Protecting laws is nothing new and was already implemented in 1995 under the name "Data Protection Directive". GDPR differentiates from this by being more specific and violation of the law results in severe fines of up to €20 million (GDPR, n.d.c). Protecting consumer privacy is not the only focus of GDPR. Article 22 of GDPR states that a consumer cannot be rejected to a decision solely based on automated processing (GDPR, n.d.b). If the consumer is rejected by an automated system, they have the right for the process to be manually checked instead. Many of the key points in GDPR are aimed at protecting the consumers from being mistreated by corporations. Some of the articles also focus on flexibility for the consumers. "The Right to Data Portability" under article 20, allows consumers to transfer their personal data between providers (GDPR, n.d.a).

2.5 The Bias-Variance Tradeoff

When dealing with any kind of machine learning or statistical model, the concept of bias-variance is an important aspect. The bias of the model can be seen as the difference in the

predicted values of a model versus the observed values due to assumptions made by the model. The most classical example is how a linear regression model will assume that there is a linear relationship between the predictors, when in fact the relationship *could* be non-linear. This results in high bias for the predictions made by the linear model. On the other hand, imagine a model with inputs that does not have to be linear and consequently follows the training data very well. While this model would have significantly lower bias it would also have an increase in the amount of variance. The issue with having high variance is that the predictions will vary significantly if we change our training set but keep the same test set.

Both variance and bias leads to errors in predictions, which is why the Total Error is displayed as the U-curve below.

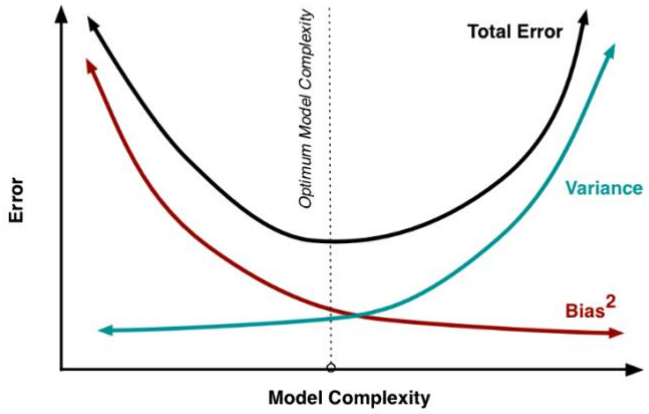


Figure 2.1: *Bias-Variance Trade-off (Fortman-Roe, 2012)*

As we can see there is a tradeoff between bias and variance. Low variance can lead to high bias and low bias to high variance. The ideal tradeoff between the two can be seen at the minimum point of the test error, shown by the stippled line.

2.6 Supervised vs. unsupervised learning

As this thesis exclusively will be dealing with supervised learning as opposed to unsupervised learning, we will dig a bit deeper into this subject. Supervised learning can be divided into regression and classification. For regression problems, one is typically dealing with quantitative variables, while classification problems usually deal with qualitative variables.

However, it is important to note that this is not black and white, and some regression problems deals with qualitative variables and vice-versa (James et al., 2017). The logistic regression, which will be discussed later in this thesis, is an example of this. The logistic regression uses quantitative variables, but the use case of it is often for dealing with binary problem, such as default vs. non-default (James et al., 2017).

2.7 Machine Learning and AI

Machine learning and artificial intelligence (AI) are two words that have gained a massive increased traction in recent years. While they are often used interchangeably it is important to know the difference between them. Machine learning is where the machine is able to benefit from its experiences and adapt or develop to produce a better result. AI on the other hand is able to make its own decisions. Machine learning can be seen as a subset of AI (Pathmind, n.d.).

Machine learning is typically further divided into two main categories: supervised learning and unsupervised learning. In supervised learning, for every observation *i* there is a response *y* which is associated with this observation. This means that the goal in supervised learning is to accurately predict future *y* values using relevant predictors and in turn will often explain the relationship between the predictors and the response (James et al., 2017). A linear regression is a good and simple example of a method using supervised learning, where we try to fit the model of predictors to predict our response variable. Unsupervised learning is a bit more difficult compared to supervised learning. For every observation *i* there is vector of measurement and **not** a response *y*. We can no longer use statistical tools such as the linear regression as there is no response variable that can supervise the procedure and it is therefore unsupervised (James et al., 2017). A typical unsupervised learning tool is clustering, where the goal is to assign each observation to a certain number of clusters, in the best possible way.

2.8 Machine Learning Methods

In this subsection, we will present the machine learning methods used in this thesis and the general theory behind them.

2.8.1 Logistic Regression

The logistic regression is often viewed to be a special case of linear regression, but with a categorical response variable instead of a continuous (Abonazel & Ibrahim, pg. 80, 2018). While linear regression can be used to solve classification problems, or more specifically credit default probabilities, it is not the most ideal tool to use. The problem arises as there is no upper or lower limit of our response variable. If we were to assume a bank that accepted or rejected loans based on the balance of the client. For individuals with extreme balance values in either ends, the predicted value could result in being less than 0 or higher than 0. As the goal of credit card default problems usually are to try to predict probabilities of default this creates a problem, as probabilities have to be within the range of 0 and 1.

When dealing with a binary categorical response variable, the logistic regression is a better tool to use. Instead of using a straight line to predict the outcome, the logistic regression uses the Logistic Function and will always yield a result between 0 and 1. The Logistic Function is given by:

$$p(\bar{X}) = \frac{e^{\beta_0+\beta_1*X_1+\cdots+\beta_K*X_K}}{e^{\beta_0+\beta_1*X_1+\cdots+\beta_K*X_K}+1} \tag{2.1}$$

As we can see in the figure below, the Logistic Function is what gives the Logistic Regression its distinctive S-Curve.

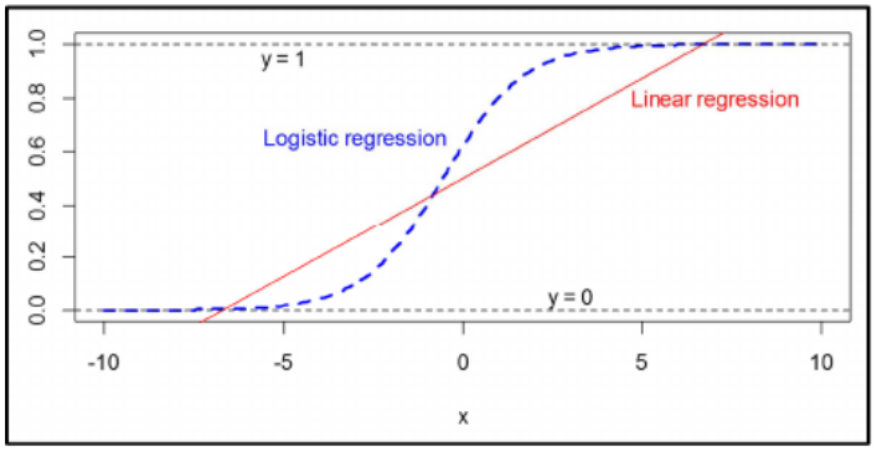


Figure 2.2: *Linear vs. Logistic Regression (Abonazel & Ibrahim, 2018)*

From the Logistic Function we can continue to find the odds for any given data point, by transforming the function above to the log odds:

$$Log\left(\frac{p(\bar{X})}{1-p(\bar{X})}\right) = \beta_0 + \beta_1 * X_1 + \cdots + \beta_K * X_K \qquad (2.2)$$

As seen from the formula above, we can find the log odds from the parameters on the right-hand side. However, while we for regular linear regression do this by using Least Sum of Squares, we instead use the Maximum Likelihood to determine the best fit (Abonazel & Ibrahim, pg. 81, 2018). The Maximum Likelihood estimates the parameters of each observation and will predict values closer to 1 for client that are more likely to default and 0 for clients who is not likely to default.

2.8.2 Decision Tree

Similarly to the logistic regression, Decision Trees can be used both for regression and classification problems. The Decision Tree has a simple layout, which starts with the root node. The root node is the first and main node and represents the whole population. The tree is then split into daughter nodes by recursive partitioning. The number of daughter nodes

depends on what type of classification tree we want. The most used type is the binary recursive partitioned tree, where the number of daughter nodes per split is two. One can use more than two daughter nodes by using multiway splits, however evidence suggests that these types of trees does not necessarily provide better accuracy (Ishwaran & Rao, 2009). Tree impurity is measured to determine what the split should be and how good it is. The more similar each observation of the nodes are, the higher the decrease in impurity is, which leads to a better fit. Finding the tree impurity can be done by various functions, where the Gini Index is the most popular (Ishwaran & Rao, 2009)

The Gini Index is given by:

$$Gini\ Index = 1 - \sum_{j=1}^{c} p_j^2 \tag{2.3}$$

The Gini Index ranges from 0 to 1. And as we want to decrease the impurity, the lower the value the better each observation fits into a split.

While many of the machine learning methods have a great track record for predicting probability of default, they often lack in terms of explainability. One may think that the accuracy of the predictions is the most important, which is correct to a large degree. However, one should not forget that explainability is also fairly important. It is critical for the firm itself as it can easier interpret why or why not customers have been rejected by the algorithm. And it is also important in order to easier explain to a potential customer why their loan was rejected, as opposed to just referring to a complex algorithm.

While the simplicity of the decision tree model is one of its strengths it also introduces a weakness. The decision tree suffers from high variance, which in turn may results in very varying performance on the training and test data set. As a result of this, the prediction of the model on the test data may be significantly lower compared to other machine learning algorithms.

The figure below shows the simple intuition behind a Decision Tree. If the observations fail to meet the "requirement" of the root node, i.e. being less than 4.5 years, we moves to the left. On the contrary, if the requirement is fulfilled, we move to the right and the process is repeated for the next daughter node.

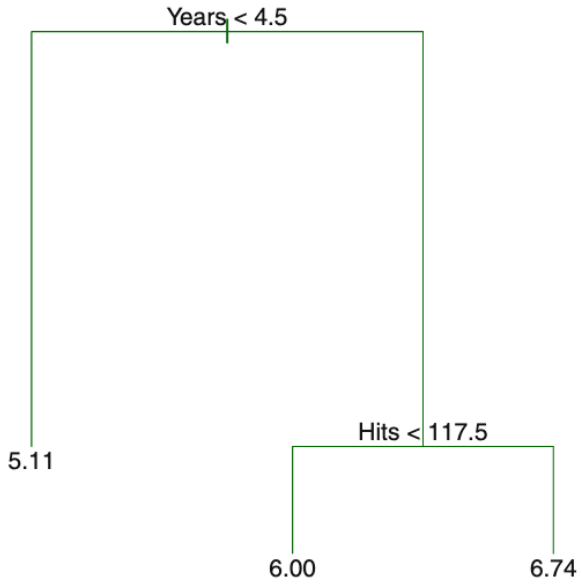


Figure 2.3: *Simple Decision Tree (James et al., 2017)*

2.8.3 Random Forest

Random Forest is, like the name suggests, also a "tree" method. However, while we in Decision Trees only use one tree, the Random Forest consists of multiple trees that merges together into a single tree. This is done by using a technique called Bootstrap Aggregation, or often referred to as "Bagging". The idea behind Bagging is to reduce the variance introduced by averaging multiple samples. While we ideally would like to average training data over multiple datasets, this is not necessary as we can take multiple samples from one dataset and then average these out (James et al., 2017):

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(x) \tag{2.4}$$

While Bagging improves the prediction accuracy this comes at the cost of a new problem; correlation. In Bagging many of the trees produced will be highly correlated. By averaging many highly correlated trees the reduction in variance will not be as significant as if these trees are correlated. To solve this, Random Forest will not select predictors solely based on their estimated prediction power (Pretorius et al., 2016). Instead, it selects a random number of predictors for each split and then choses the best predictor among these. This method of choosing predictors will greatly reduce the variance of all the trees as it allows for some of the perceived weaker predictors to be taken into account. The numbers of random predictors chosen per split can be defined by the user.

2.8.4 XGBoost

XGBoost has quickly become one of the most used machine learning algorithms in the past few years. On Kaggle, a website that regularly holds competitions with sizeable rewards and recognitions, 15 of the 29 winners in 2015 used XGBoost as either their main model or in combination with other models (Chen & Guestrin, 2016). The scalability of the model, the fast computation time and its accurate predictions on several types of data has been the key factors for the heavy success and use of XGBoost.

XGBoost is a machine learning algorithm that uses Gradient Boosting based on the research from Friedmann et al. (Chen & Guestrin, 2016). The Gradient Boosting uses an ensemble tree method to create a number of regular Decision Trees based on any given data sample, where the prediction will be the sum of all the predictions from every tree made. As this alone could easily result in overfitting the data, XGBoost aims to minimize the regularized objective model given by this formula (Chen & Guestrin, 2016):

$$\mathcal{L}(\phi) = \sum_{i} l\left(\hat{y}_i, y_i\right) + \sum_{k} \Omega\left(f_k\right) \tag{2.5}$$

$$\Omega(f) = \gamma T + \ 1/2\ \lambda ||\omega||^2 \tag{2.6}$$

The first term measures the difference between the predicted $\hat{y}_i$ and the observation y, while the second term omega acts as a penalty the more complex the model is.

2.8.5 K-Nearest Neighbor (KNN)

The K-Nearest Neighbor is a non-parametric machine learning method mainly used for classification problems. KNN classifies the test observation x_0 based on K numbers of neighbors by determining its probability of being class j given by the formula (James et al., 2017):

$$\Pr(Y = j|X = x_0) = \frac{1}{K}\sum_{i\in\mathcal{N}_0} I(y_i = j) \tag{2.7}$$

The distance from the test observation x_0 and its' closest neighbor can be determined by various ways of measuring distance, where the Euclidean Distance is the most widely used metric (Hu et al., 2016). The Euclidean Distance is given by:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.8}$$

If we were to classify if x_0 defaults or not with K being 11, the 11 closest neighbors would be identified. If 7 out the 11 closest neighbors were classified as having defaulted, KNN would predict x_0 to default as 7/11 > 4/11.

The downsides of KNN are that the method is a lazy learner, and that the outcome is heavily biased and dependent on the number of neighbors K (Guo et al., 2004). This results in higher computation time and cost, due to the way new observations are handled and classified. The Bias-Variance tradeoff discussed earlier in this chapter is present when determining the appropriate number of neighbors K. If K is set too low, one will encounter low bias but in turn higher variance. With K set too high, the variance is on the lower side with bias being consequently higher. Therefore, tuning the K parameter to hit the sweet spot between bias and variance is important for this machine learning method.

2.8.6 Neural Networks (NN)

Neural Networks is one of the more complicated machine learning tools to be implemented in this thesis. The model tries to replicate how the human brain works by using neurons that connect with each other in various ways (Du & Swamy, pg. 1, 2013). Because of the way the neural networks connect neurons together, it is referred to as a connectionist model. The mapping of the neurons depends on the architecture the neural network used. The figure below shows a layered feedforward network.

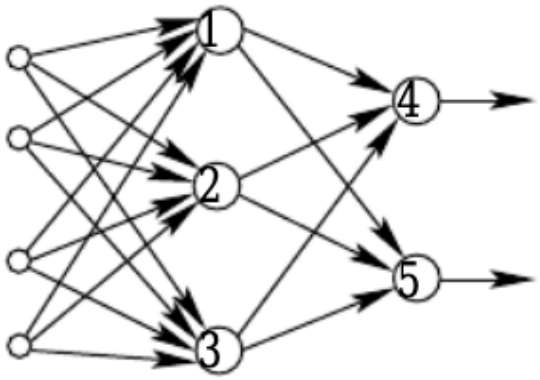


Figure 2.4: *Feedforward Neural Network (Du & Swamy, 2016)*

As the name suggest, the neurons in this architecture will only connect neurons from forward-hidden layers and there is no feedback given from the hidden layer (Du & Swamy, 2013). The four nodes at the very beginning of the figure are our input nodes, which is where our predictors are. There is only one hidden layer, which is where the three neurons are located. The last two nodes at the right, are the output nodes.

The number of neurons per layer and total number of hidden layers are hyperparameters that can be set by the user. In line with other models, this also need to be done by tuning or a similar method to avoid overfitting of the data.

Neural Networks usually provides good prediction accuracy, but it needs to be tuned well to avoid overfitting. In addition to problems with overfitting, the Neural Network also suffers from being hard to interpret and is often labelled as a black-box because of this (Du & Swamy, 2013).

3. Data

In this part, we will introduce the dataset used for the empirical part of this thesis.

3.1 Data Decsription

The data set used for the empirical part of this thesis is from a publicly available dataset by UCI Machine Learning Repository. The dataset is based on the default of credit card customers in Taiwan from 2004. Of the 30 000 included observations, roughly 22 % of the customers defaulted on their payment. There is a total of 25 variables in the dataset, including the response variable indicating default or non-default.

3.2 Variables

The table below shows all the variables in this dataset, including a brief description of them as well as what values these can be.

Variable Name	**Description**	**Value**
DEFAULT (original name default.payment.next.month)	Whether the customer defaulted or not	1 = Default, 0 = Non-default
ID	Customer ID	1-30 000
LIMIT_BAL	Amount of credit given	Any amount, positive
SEX	Gender	1 = Male, 2 = Female
EDUCATION	Level of education	1 = Graduate school 2 = University 3 = High School 4 = Others
MARRIAGE	Martial status	1 = Married 2 = Single 3 = Others
AGE	Age of the customer in years	Their age (min. recorded = 21, max. recorded = 79)
PAY_1 (original name PAY_0)	History of payment in September 2005	-1 = paid duly 1 = 1 month late … 8 = 8 months late
PAY_2	History of payment in August 2005	-1 = paid duly 1 = 1 month late … 8 = 8 months late
PAY_3	History of payment in July 2005	-1 = paid duly 1 = 1 month late …

		8 = 8 months late
PAY_4	History of payment in June 2005	-1 = paid duly 1 = 1 month late … 8 = 8 months late
PAY_5	History of payment in May 2005	-1 = paid duly 2 = 2 months late … 8 = 8 months late
PAY_6	History of payment in April 2005	-1 = paid duly 1 = 1 month late … 8 = 8 months late
BILL_AMT1	Amount of bill statement in September 2005 (in NT Dollars)	Any value, both positive and negative
BILL_AMT2	Amount of bill statement in August 2005 (in NT Dollars)	Any value, both positive and negative
BILL_AMT3	Amount of bill statement in July 2005 (in NT Dollars)	Any value, both positive and negative
BILL_AMT4	Amount of bill statement in June 2005 (in NT Dollars)	Any value, both positive and negative
BILL_AMT5	Amount of bill statement in May 2005 (in NT Dollars)	Any value, both positive and negative
BILL_AMT6	Amount of bill statement in April 2005 (in NT Dollars)	Any value, both positive and negative
PAY_AMT1	Amount of previous payment in April 2005 (in NT Dollars)	0 to any value
PAY_AMT2	Amount of previous payment in May 2005 (in NT Dollars)	0 to any value
PAY_AMT3	Amount of previous payment in June 2005 (in NT Dollars)	0 to any value
PAY_AMT4	Amount of previous payment in July 2005 (in NT Dollars)	0 to any value
PAY_AMT5	Amount of previous payment in August 2005 (in NT Dollars)	0 to any value
PAY_AMT6	Amount of previous payment in September 2005 (in NT Dollars)	0 to any value

Table 3.1: *Variables, descriptions and possible values*

3.3 Software

The programming language used for this Master Thesis has been R alongside the integrated development environment RStudio. All the data processing, cleaning and making of models has been done with RStudio on a local machine with various packages depending on the tasks performed.

4. Methodology

The methodology part will explain the various processes done of the dataset and the machine learning methods used in this thesis.

4.1 Data Cleaning

In order to get the most accurate predictions by the model implemented, it is important that the data are thoroughly inspected and cleaned. Missing variables and extreme values can, somewhat depending upon the context, heavily skew the data and influence the predictions being made.

The dataset has no missing or extreme values and was in general very clean. The first change we made was to first remove the variable "ID". This column was simply just an identifcation for the customers, ranging from 1st observation to the 30 000th, and therefore served no purpose as a predictor. We also made some small name changes to some of the variables to fit the rest of the dataset better. PAY_0 was changed to PAY_1, to better fit in with BILL_AMT1 and PAY_AMT1. The dataset already contains our response variable, and it is therefore no need for us to define this ourselves based on a certain criterion. However, the response variables name was changed from "default.payment.next.month" to "DEFAULT".

However, the dataset had a few inconsistencies regarding some of the variable values. The variable "MARRIAGE" is an integer with values ranging from 1-3, as seen in the table from the previous chapter. The dataset had 54 instances of this variable being 0 and these were moved to the 3rd category "Other". Further inspecting the remaining variables, shows that in all the PAY_1-6 variables there are values of -2 and 0. Given the data description, these should range from -1 when the customer has paid in time and from 1-8 depending on how late the payment was made. There is no information regarding these values and we have to look more into this.

By further inspecting these variables, we can see that the value -2 and 0 makes up a significant part of our observations. Combining these two together (-2 and 0) makes up for almost 80% of our dataset and therefore simply removing these would heavily decrease the size of the data and most likely the prediction accuracy of our models. Our hypothesis is that the PAY_1 variable has a lot of predictive power when determining whether a customer is going to default

or not. By subsetting the dataset we can see that the distribution for defaults vs. non-default when PAY_1 is -1, i.e. paid in time, is 84%. When doing the same for -2 and 0, we get 86% and 87% respectively. We suspect that -2 and 0 might be mislabelled and should instead be labelled as -1. The next step in this process to determine this is to see how the distribution of default is when the PAY_1 is 1, meaning payment is made 1 month late. The result of this was a distribution of 33 % non-defaults. While this alone does not necessarily confirm our hypothesis regarding the predictability of the variable, it certainly strengthens it. We repeated the process from above on all the PAY_X variables and the numbers we got from this was similar. As a result of this, we decided to change the PAY_X values from -2 and 0 to -1. While this is not a perfect way of solving this problem, the suspected predictability of the predictor and the distribution of the default vs non-default, we believe this is a satisfactory way of dealing with it.

4.2 Data partitioning

When building a statistical model, one will usually split the data into a training set and a test/validation set. The relevant models will first be applied and will learn from the training set, before they are being run on the new and unseen data in the test set. By doing this, one can ensure that models with good predictions on the training set actually predicts well on unseen data and is not just a case of overfitting. If the model follows the training data too well and predicts poorly on the test set, it is usually a case of overfitting and the model has not been able to identify the underlying relationship in the data set.

There are many methods for choosing a split and in this thesis, we have chosen an 80/20 split. 80% of the dataset will be used for training and the remaining 20% will be held out and used as a test set.
We used the “set.seed” function in R and set this to “123” to get reproducible results. This function ensures that the same random numbers are being used and the splits of test and training data, models and results will be the same if ran multiple times or on different computers.

4.3 Tuning of Hyperparameters

Many of the machine models presented and used in this thesis require carefully tuned hyperparameters in order to get optimal predictions. The hyperparameters are defined as parameters than can be changed manually when "building" machine learning models. However, not all the models we use needs to be tuned or even have any hyperparameters at all. The tuning process can be a long and slow process depending on the size of the data and the complexity of the model. What makes the tuning process even harder is that there usually is no "one size fits all" solution and we often need to brute force an optimal solution. This can be done in various ways and in this thesis we have used a grid search in RStudio. This is still not a perfect method as it can be computationally expensive the smaller the increments of each hyperparameter in the grid search is. Therefore, even tuning with a method like grid search will not yield the perfect solution, as it is done in increments and not in a continuous search. Despite the fact that tuning can increase the computation time, it can vastly increase the prediction accuracy.

4.3.1 Tuning of Decision Tree

In our model for the Decision Tree, we only have one hyperparameter that we tune; number maximum depth. The maximum depth defines how deep the tree should be. We tune the number of trees by doing a cross-validation and selecting the tree with the lowest error rate.

4.3.2 Tuning of Random Forest

For Random Forest there are three hyperparameters we are looking to tune: number of random predictors, maximum depth and number of trees.

The number of random predictors is the number of predictors to be randomly selected at each split. As stated in the theory part, this is usually set to the square root of the total number of predictors. This value should, alongside the other hyperparameters, be tuned to increase the prediction power. The square root of our 24 predictors is between 4-5 and we therefore start our grid search with a value slightly lower than this. We start with 2 and increase in increments of 5 up to a maximum of 22.

There is no similar general rule for the maximum depth of the trees made by Random Forest. What the max depth should be heavily depend on the number of predictors and the size of the

dataset. As this is not the biggest dataset and with our 24 predictors, we set the max depth to range from 5 to 30.

One of the proclaimed features of Random Forest is that it supposedly handles overfitting in regard to the number of trees well. However, a study from 2012 found the optimal number of trees to be 128 (Oshiro et al., pg. 166, 2012). This study was done on several types of data and with numbers of trees ranging from very low values to over 4000 trees. The results showed that there were at most very little increase in performance when increasing the number of trees and in some scenarios fewer trees performed better. The computational time also increased exponentially the more trees were used for the model. With this in mind, our grid search for number of trees started at 100, increasing by 20 until the maximum of 200 trees was reached.

4.3.3 Tuning of XGBoost

XGBoost is the model that required the most tuning in terms of numbers of hyperparameters. We tuned the model by these hyperparameters: Loss reduction, Maximum depth, learning rate, minimum sum of instance weight, subsample ratio and subsample of each tree. While XGBoost is known for being a fast algorithm, the number of hyperparameters heavily increased the computation time for this method, as these needed to be tuned.

The loss reduction function defines how strict or loose the model branches out to new daughter nodes. Higher value will result in a less complex tree. There is no specified default value for this, and we search from 0 to 5.

Maximum depth functions the same way as for our previous model, Random Forest. However, the maximum depth is usually set to a smaller value for XGBoost. Chen & Guestrin (2016) used 8 as maximum depth for all the trees. We set our maximum depth to be between 3 and 11, to cover well below and above the value of 8 used in the study.

The learning rate determine how much each tree should contribute to the final prediction made, where the value ranges from 0 to 1. A lower value can be used to prevent overfitting but will in turn increase the computational time of the model. Our grid search values has been set between 0.01 and 0.3 as we want to avoid the potential of overfitting.

The hyperparameter minimum sum of instance weight regulates the sum of instance weight required to further partition each tree, where a higher value increases the threshold and results in fewer nodes. The default value is 1 and for our search we look at values between 1 and 5.

Subsample for each tree decides how big the percentage of predictors to be chosen for the building of each tree. This value can be set to be between 0 and 1, where 0 indicates not choosing any predictors and 1 for choosing all. This method is somewhat similar to how Random Forest operates, but the difference is that XGBoost does this at a **tree-level** instead of per split in each tree.

Subsample ratio works similarly to the previous hyperparameter, but instead of subsampling our predictors we are subsampling a percentage of our training data. It is set between 0 and 1 and choosing a lower value can reduce the chance of overfitting the data.

Below is a table that summarizes the hyperparameters we searched in the tuning for XGBoost.

Hyperparameter	Grid search
Loss Reduction (Gamma)	0 to 20
Maximum Depth (Max_depth)	3 to 8
Learning Rate (ETA)	0.0 to 0.3
Minimum Sum of Instance Weight (Min_child_weight)	1 to 5
Subsample Ratio (Subsample)	0.5 to 1
Subsample for each tree (Colsample_bytree)	0.5 to 1

Table 4.1: *XGBoost Hyperparameters and Grid Search*

4.3.4 Tuning of KNN

As introduced in the theory chapter, setting the right number of neighbours, K, is crucial for getting a good result with this method. K is also the only hyperparameter that is set and the final value used is closely tied to the bias-variance trade-off.

In order to find the optimal number of K, we used cross validation with different values for K to find the iteration with the lowest error rate.

4.3.5 Tuning of Neural Network

The Neural Network is the most complex method applied on the dataset. For this model, we have chosen to only tune one hyperparameter, the number of neurons. In other words, the Neural Network will have a single hidden layer, where we tune the number of neurons in this layer. The type of Neural Network used for our model is a layered feedforward, as shown in figure 2.4 in chapter 2. We tuned the network with neurons ranging from 1 and 6 as our highest value.

4.4 Evaluation of Model Performance

When looking at the performance of a model, there are several different methods to choose from. Depending on the type of problem you are dealing with, some evaluation metrics may perform better than others. As we in our case are handling a binary classification problem, we have chosen two popular metrics within this field; Accuracy derived from a Confusion Matrix and Receiver Operatic Characteristic Area Under the Curve, also known as ROC AUC. While ROC AUC will be our main evaluation metric of choice in this thesis, the Confusion Matrix will provide us with information on how well each model performed specifically on the dataset given a certain threshold.

The Confusion Matrix displays how well each of the models were able to classify the predictions made on the dataset. This is done by dividing the results into four categories, where two of these classes are correctly predicted. “True positive” and “True negative” is where our model will correctly have predicted from the dataset, whereas “False positive” and “False negative” shows us where our model made the wrong predictions. The two types of errors are often referred to as Type-I and Type-II error respectively.

	ACTUAL POSITIVE CLASS	ACTUAL NEGATIVE CLASS
PREDICTED POSITIVE CLASS	*True positive (Tp)*	*False negative (Fn)*
PREDICTED NEGATIVE CLASS	*False positive (Fp)*	*True negative (Tn)*

Table 4.2: *Example of a Confusion Matrix*

From our Confusion Matrix we are able to get an accuracy of the correct percentage of predictions made by the model, given by:

$$Accuracy = \frac{Tp + Tn}{Tp + Tn + Fn + Fp} \tag{4.1}$$

The advantages of using this metric for evaluation is that the requirement for computational power is low and it is easy to understand for everyone, not just professionals within the field. Another reasoning for using the Confusion Matrix in conjunction with the accuracy derived from it is to validate that the models are actually predicting both defaults and non-defaults. The importance of this can be illustrated by imagining a heavily imbalanced dataset. Assume we have a dataset with 20 000 observations and 19 000 of them being non-default. Simply by predicting every observation to be non-default, we would achieve an accuracy of 95%.

The ROC AUC is widely used to assess the performance of classification problems. The ROC graph can be plotted with Specificity on the X-axis and Sensitivity on the Y-Axis, which are given by (Fawcett, 2006):

$$Specificity = \frac{True\ Negatives}{False\ Positives + True\ Negatives} \tag{4.2}$$

$$Sensitivity = \frac{True\ Positives}{False\ Negatives + True\ Positives} \tag{4.3}$$

The advantage of using the ROC AUC curve compared to only using the Confusion Matrix is that it provides a better overall view for the model. From the Confusion Matrix alone, you can only see the Sensitivity and Specificity for a single point, while the ROC AUC curve shows the Specificity and Sensitivity for various thresholds. When plotting the ROC AUC curve, the X axis is usually labelled 1-Specificity. In essence, we are plotting the True Positives versus the True Negatives.

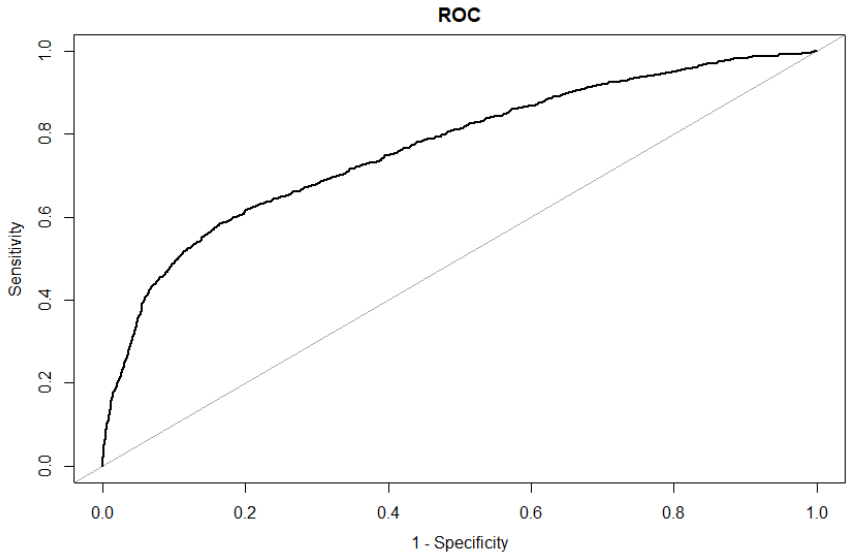


Figure 4.1: *Example of ROC AUC plot*

The figure above shows the plotted ROC AUC curve in black. The highest possible value to achieve is up in the left corner, where we would correctly predict everything and in turn have no False Positives. The grey line is where X = Y and can be classified as a random performance. In other words, this can occur if the model is just blindly guessing (Fawcett, 2006). Any curve that is below this grey line will therefore perform worse than guessing and could be a case of not being able to properly handle the information (Fawcett, 2006).

The reasoning for using the ROC AUC curve as our main deciding tool is due to the nature of the credit market. Instead of just giving a snapshot of a singular threshold, ROC AUC displays multiple thresholds. The consequence of issuing credit to a consumer who defaults has a greater cost to the bank compared to denying credit for a customer who does not the default. Therefore, we believe that ROC AUC will be a better tool to rank our models.

5. Results

In this part of the thesis, we will present the results from each model in an increasing order based on the ROC AUC value. As mentioned earlier, the ROC AUC will be the evaluation metric we use to evaluate the performance of the model. The Confusion Matrix will be a complementary tool. The main reason for this is that the Confusion Matrix provides a snapshot of how the model performed on this training and test data, while ROC AUC gives a better overall view of the model performance, as it looks at the Specificity and Sensitivity across various thresholds.

Additionally, we will show the final values of the tuning of the hyperparameters for each model.

5.1.1 Decision Tree

The Decision Tree performed the worst out of all the models, slightly behind the KNN. This is to be expected as the Decision Tree is only made up by one single tree, unlike Random and Forest XGBoost. This translates to a high amount of variance as even small changes in the training data can highly affect the outcome of this model.

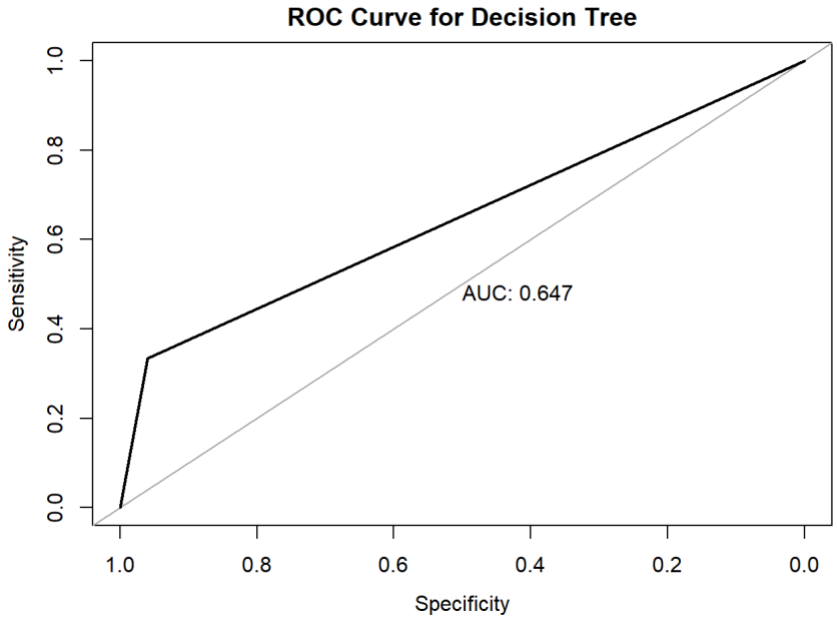


Figure 5.1: *ROC for the Decision Tree*

5.1.2 KNN

KNN performed similarly to the Decision Tree and was only slightly better measured by the ROC AUC. While one would naturally expect the Decision Tree to perform bad, it might come as a bit of a surprise how KNN performs close to the same and being way behind the other machine learning methods One reason for this is that the dataset is somewhat unbalanced, and this can affect KNN to some extent. While the dataset is not *heavily* imbalanced, the number of non-defaults makes up for around 77% of the dataset, which still is a significant amount. A study done in 2015 by Beckmann et al., confirmed that imbalanced datasets indeed affects the performance of classifier algorithms such as KNN (Beckmann et al., 2015).

By repeated cross-validation of KNN, we arrived at the value **21** for number of neighbours **K**. The value of K is chosen based on the highest accuracy achieved, which can be seen in the plot below.

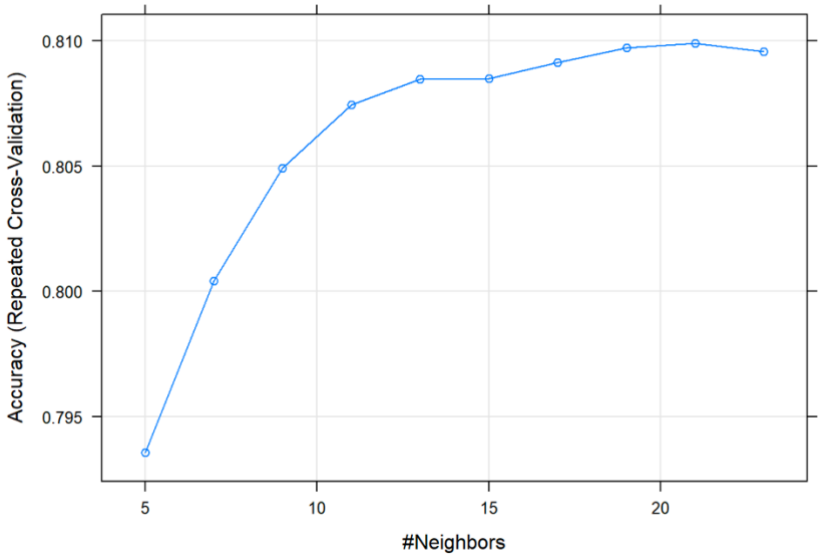


***Figure 5.2**: Cross-validation for KNN*

Despite KNNs modelling simplicity, the computational time is significant, and even on par or slower compared to some of the more complex models. Combined with its low predicting power based on our empirical results, the overall performance is low.

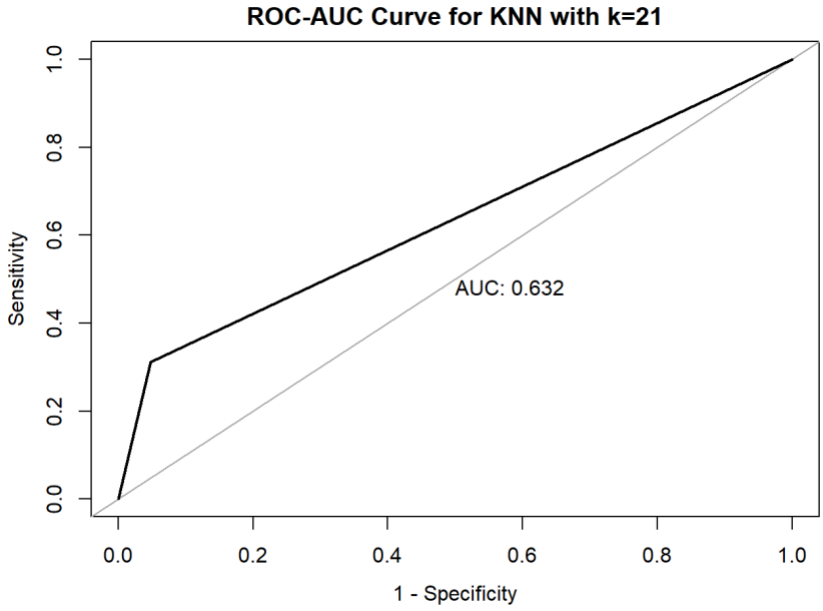


Figure 5.3: *ROC curve for KNN*

5.1.3 Logistic Regression

The Logistic Regression performs substantially better than both KNN and the Decision Tree and is very close to the top three models of this thesis. The simplicity and speed of the model is a clear advantage, and the model is still able to perform among the best ones.

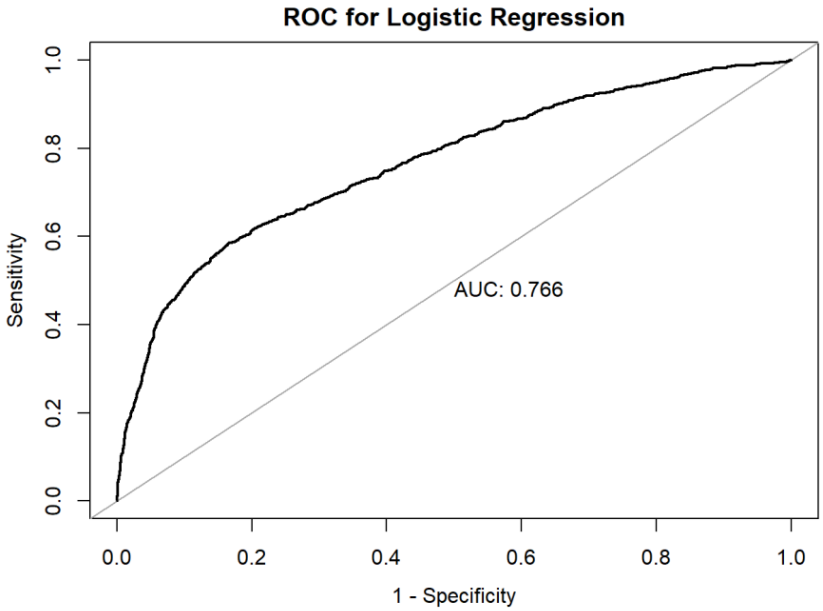


Figure 5.4: *ROC for the Logistic Regression*

5.1.4 Neural Network

The Neural Network performed almost identical to the Logistic Regression. Neural Network is often praised for its great ability to classify; however, the model did perform the best out of our chosen methods. It is important to note that this was a neural network with only one hidden layer. Increasing the number of hidden layers could potentially have increased its prediction power, but due to the computational time and power required to properly do this it was not a feasible solution. The computational time with two hidden layers and 3 neurons in each layer reached over 24 hours before we decided to stop. With 1-4 neurons in layer 1 and up to 2 neurons in the second layer, the computational time was shorter, but the result worse compared to a single layer with 3 neurons. Consequently, we decided to tune the neural network only using a single layer and grid search the optimal number of neurons in this layer. The final number of neurons was set to **3**.

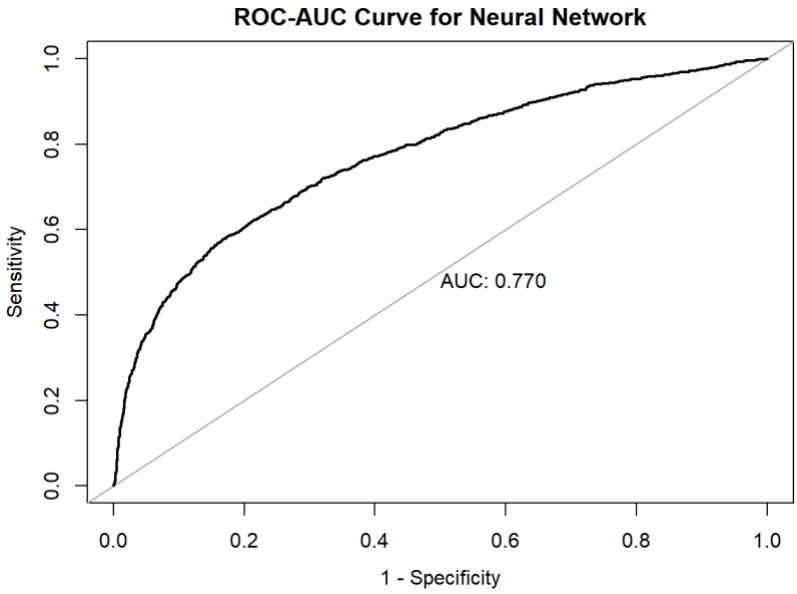


Figure 5.5: *ROC for the Neural Network*

5.1.5 XGBoost

XGBoost was the method with the highest number of hyperparameters to tune. The model performance was almost perfectly in-between the Neural Network and Random Forest, with 0.009 ahead and 0.01 behind respectively.

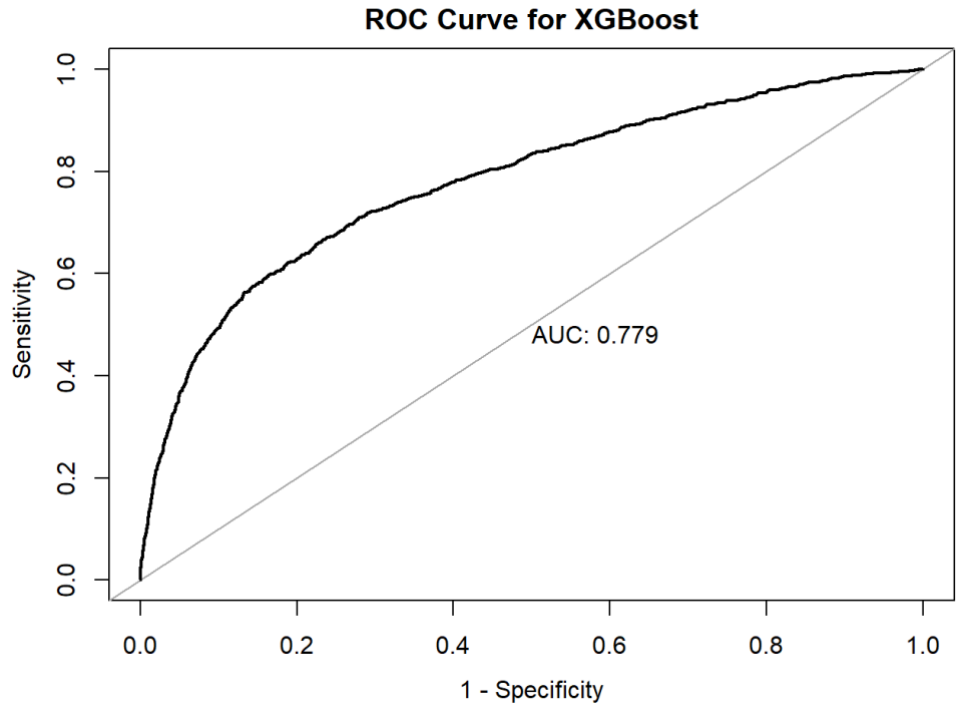


Figure 5.6: *ROC for XGBoost*

5.1.6 Random Forest

The best performing model on our dataset was the Random Forest. The method had an ROC AUC score of 0.789. We believe that the reasoning for Random Forest performed the best might be due to its ability to reduce overfitting by selecting only a random number of predictors per split. This in turn, as presented in chapter **2.8.3** , reduce the correlation. As only a few predictors are chosen per split, the variation is also decreased, contributing to better predictions as well.

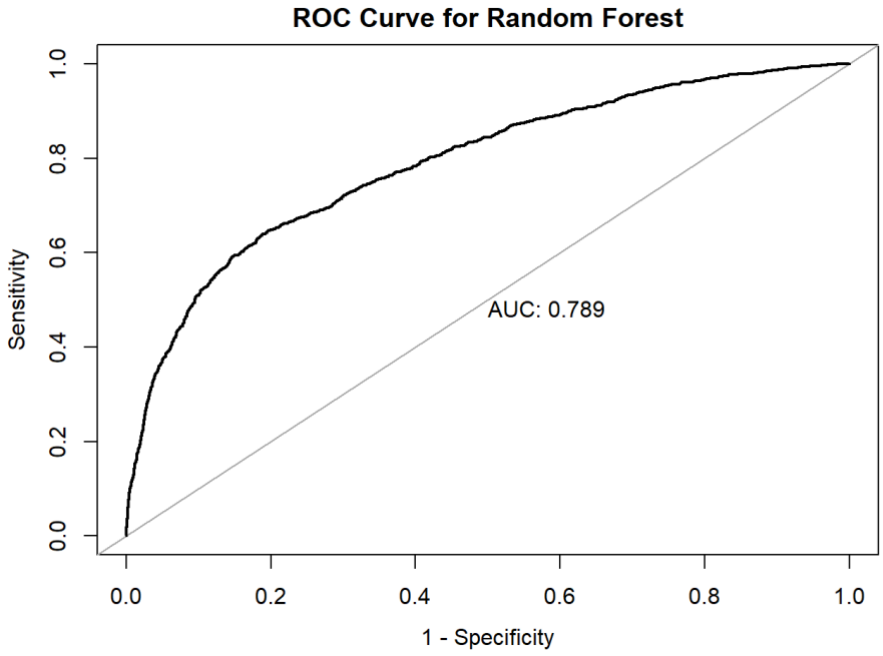


Figure 5.7: *ROC for Random Forest*

5.2 Summary of Performance

The table 5.1 resents the final results of all the models, ranked in an increasing order by ROC AUC score. We have also added the final values of the hyperparameters used for each of the models. The accuracy is gathered from the Confusion Matrix from each of the methods, with the best score in **bold**.

Random Forest performed best out of all the models, with XGBoost close behind. Logistic Regression and Neural Network performed almost equally and were not far behind the top two models.

Machine Learning Method	ROC AUC Score	Accuracy	Hyperparameters used
Random Forest	0.789	81.85%	Number of Trees = **140** Number of predictors per split = **7** Max. depth = **10**
XGBoost	0.779	81.95%	Loss Reduction = **3** Max. depth = **4** Learning Rate = **0.0752** Min. Sum of Instance W. = **4.7** Subsample Ratio = **0.8** Subsample for each Tree = **0.772**
Neural Network	0.774	81.56%	Number of Neurons = **3**
Logistic Regression	0.766	81.05%	
Decision Tree	0.647	**82.02%**	Max depth = **1**
KNN	0.632	80.56 %	Number of K = **21**

Table 5.1: *Summary of Performance*

5.3 Variable Importance

The variable importance reveals which of the variables the models relied most upon in their predictions. We have found the most important features for the following four models: Decision Tree, Random Forest, XGBoost and Neural Network. Variable importance cannot be derived directly from the KNN and the Logistic Regression, and these have not been included. As the dataset only contains 24 variables, we will list the top three most important ones for each model. Pay_1 and Pay_2 were the variables that appeared the most, by making the top three most important predictors in three of the four models.

Model	Most important variables
Decision Tree	1. Pay_1 2. Pay_2 3. Pay_3
Random Forest	1. Pay_1 2. Pay_2 3. Bill_amt1
XGBoost	1. Pay_1 2. Pay_2 3. Pay_amt3
Neural Network	1. Pay_amt2 2. Pay_amt1 3. Pay_amt3

Table 5.2: *Variable Importance*

6. Discussion

From the results we can see that the top performing machine learning methods performs well on a somewhat difficult dataset. The main question still remains unanswered: Can machine learning be implemented to determine whether or not to grant credit? One of the known drawbacks of machine learning is the interpretability of some of the more advanced models. While the methods have increased significantly in popularity over the years, the interpretability has remained the same. Under GDPR article 22, if the consumer is denied by an automated process without a clear reason as to what caused the rejection. If a prediction is made by a Random Forest or XGBoost model, it can be difficult to interpret why the application was denied or granted.

The increasing amount of data being created and collected is a positive sign from an analytical standpoint. Still, one should note that more data does not necessarily increase accuracy on predictions alone. For the data to be good, it should both be of a significant quantity and quality to ensure good predictions are being made.

We would have liked to create a reference score based on the regular methods as well for this dataset, but this was unfortunately not feasible due to the variables the dataset is made up by. Traditional scoring methods rely on, as presented in the theory chapter, on financial data. This includes data such as income, equity, solidity and so on. Our dataset consists mostly of behavioral variables, e.g. late payment and amount paid last bill. Another factor is that we simply do not know the exact formula behind the decision of extending credit, and even if we did, we do not think its method would be applicable due to the nature of the variables. From the variable importance analysis, these variables also turned out to be the most predictive variables in most of the models. Therefore, not being able to implement them in a regular model would heavily decrease prediction accuracy. As machine learning has been widely used in many professional fields such as science, healthcare and finance, we see no reason as to why it should not predict as good or even better.

7. Conclusion

Random Forest, XGBoost and the Neural Network were the top three performers of the dataset with ROC AUC scores of 0.789, 0.779 and 0.776 respectively. The Logistic Regression followed closely behind these with a score of 0.765, while KNN and the Decision Tree were far behind at 0.632 and 0.647.

Machine learning algorithms have already been deployed and used in various professional fields; however, it is uncertain when or if it will be deployed in the credit market. The GDPR law does not allow banks and financial institution to deny credit applications based on an automated process. As machine learning falls under this category, the law would have to be altered or removed for machine learning to be used as a legal method to grant credit. In theory, machine learning can be applied to either of the two IRB Approaches to help determine both the capital requirement of the banks and risk assessment of the borrowers. Currently machine learning cannot be used as a standalone method to grant or deny loans.

References

Abonazel, M. R. & Ibrahim, M. G. (2018). On Estimation Methods for Binary Logistic Regression with Missing Value. *International Journal of Mathematics and Computational Science, 4*(3), 79-85.

Beckmann, M., Ebecken, N. & Lima, B. (2015). A KNN Undersampling Approach for Data balancing. *Journal of Intelligent Learning Systems and Applications, 7,* 104-116. https://doi.org/10.4236/jilsa.2015.74010

BIS (2019a). Calculation of RWA for Credit Risk: IRB Approach minimum requirements to use IRB Approach. Retrieved from: https://www.bis.org/basel_framework/chapter/CRE/36.htm?tldate=20400918&inforce=20220101

BIS. (2019b). Calculation of RWA for Credit Risk. Retrieved from: https://www.bis.org/basel_framework/chapter/CRE/20.htm?inforce=20220101

BIS. (n.d.). BIS Chronology. Retrieved from: https://www.bis.org/about/chronology.htm?m=1%7C4%7C550

Chen, T. & Guestrin, C. (2016). XGBoost: A scalable Tree Boosting System. *International Conference on Knowledge Discovery and Datamining, 2016,* 785-794. https://doi.org/10.1145/2939672.2939785

Falkenstein, E. G., Boral, A. & Carty, L. V. (2000). RiskCalc for Private Companies: Moody's Default Model. https://doi.org/10.2139/ssrn.236011

Fawcett, T. (2006). Introduction to ROC Analysis. *Pattern Recognition Letters, 27*(8), 861-874. https://doi.org/10.1016/j.patrec.2005.10.010

FICO. (2018). Frequently Asked Questions About Fico Scores. Retrieved from: https://www.ficoscore.com/ficoscore/pdf/Frequently-Asked-Questions-About-FICO-Scores.pdf

FICO. (n.d.). Learn About the Fico Score and its Long History. Retrieved from: https://www.fico.com/25years/

Finanstilsynet (2020, 17. April). Beregningsgrunnlaget. Retrieved from: https://www.finanstilsynet.no/tema/kapitaldekning/beregningsgrunnlaget/

Finanstilsynet. (2018, 11. December). Krav til banker som søker om å benytte IRB. Retrieved from: https://www.finanstilsynet.no/nyhetsarkiv/nyheter/2018/krav-til-banker-som-soker-om-a-benytte-irb/

GDPR (n.d.a). Article 20: Right to data portability. Retrieved from: https://gdpr.eu/article-20-right-to-data-portability/

GDPR. (n.d.b). Article 22: Automated individual decision-making, including profiling. Retrieved from: https://gdpr.eu/article-22-automated-individual-decision-making/

GDPR. (n.d.c). What are the GDPR Fines? Retrieved from: https://gdpr.eu/fines/

GDPR. (n.d.d). What is GDPR, the EU's new data protection law? Retrieved from: https://gdpr.eu/what-is-gdpr/

Guo, G., Wang, H., Bell, D. A., Bi, Y. & Greer, K. (2004). KNN Model-Based Approach in Classification. *Lecture Notes in Computer Science, 2888,* 986-996. https://www.doi.org/10.1007/978-3-540-39964-3_62

Hand, D. J. & Henley, W. E. (1996). Statistical Classification Methods in Consumer Credit Scoring: a Review. *Journal of the Royal Statistical Society, 160*(3), 523-541. https://doi.org/10.1111/j.1467-985X.1997.00078.x

Hu, L.-Y., Huang, M.-W., Ke S.-W. & Tsai, C.-F. (2016). The distance function effect on k-nearest neighbor classification for medical datasets. *Springerplus, 5*(1). https://doi.org/10.1186/s40064-016-2941-7

IDC. (2020, 13. May). IDC's Global StorageSphere Forecasts Shows Continued Strong Growth in the World's Installed Base of Storage Capacity. Retrieved from: https://www.idc.com/getdoc.jsp?containerId=prUS46303920

Ishwaran, H., Rao, J. (2009). Decision Tree: Introduction. *Encyclopedia of medical decision making,* 324-328. https://doi.org/10.4135/9781412971980.n97

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2017). An Introduction to Statistical Learning: with Applications in R. Springer

Kononenko, I. (2000). Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine, 23*(1), 89-109. https://doi.org/10.1016/S0933-3657(01)00077-X

Nordhaus, W. D. (2001, September). The Progress of Computing. Retrieved from: https://ssrn.com/abstract=285168

Ong, C.-S., Huang, J.-J. & Tzeng, G.-H. (2005). Building credit scoring models using genetic programming. *Expert Systems with Applications, 29*(1), 41-47. https://doi.org/10.1016/j.eswa.2005.01.003

Oshiroa, T. M., Peres, P. S. & Baranauskas, J. A. (2012). How Many Trees in a Random forest? *Machine Learning and Data Mining in Pattern Recognition,* 154-168. https://doi.org/10.1007/978-3-642-31537-4_13

Pathmind (n.d.). Artificial Intelligence (AI) vs. Machine Learning vs. Deep Learning. Retrieved from: https://wiki.pathmind.com/ai-vs-machine-learning-vs-deep-learning

Pretorius, A., Bierman, S. & Steel, S. (2016). A Meta-Analysis of Research in Random Forests for Classification. *Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference,* 2016, 1-6. https://doi.org/10.1109/RoboMech.2016.7813171

Swamy, M. N. & Du K.-L. (2013). Neural Networks and Statistical Learning. Springer London

UCI. (n.d.). UCI Machine Learning Repository. Retrieved from: https://archive.ics.uci.edu/ml/about.html

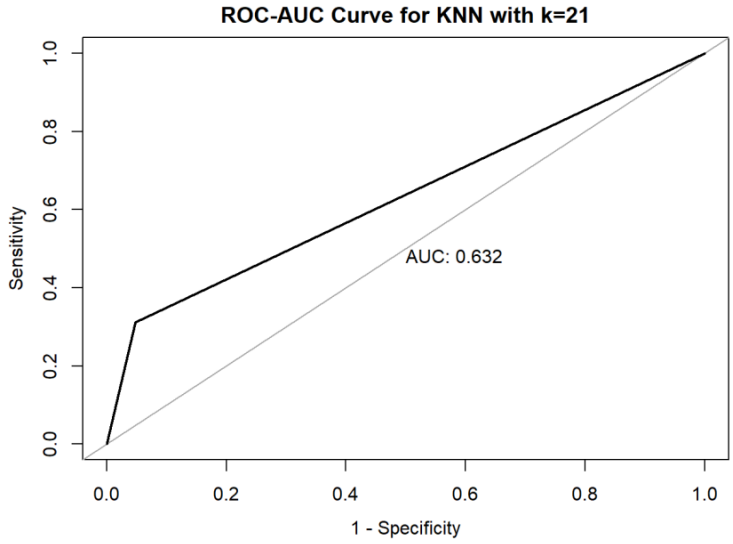
Xiao, W., Zhao, Q. & Fei, Q. (2006). A comparative study of data mining methods in consumer loans credit scoring management. *Journal of Systems Science and System Engineering, 15*, 419-435

Yeh, I. C. & Lien, C. H. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications, 36*(2), 2473-2480. https://doi.org/10.1016/j.eswa.2007.12.020

Yeh, I. C., & Lien, C. H. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications, 36*(2), 2473-2480.

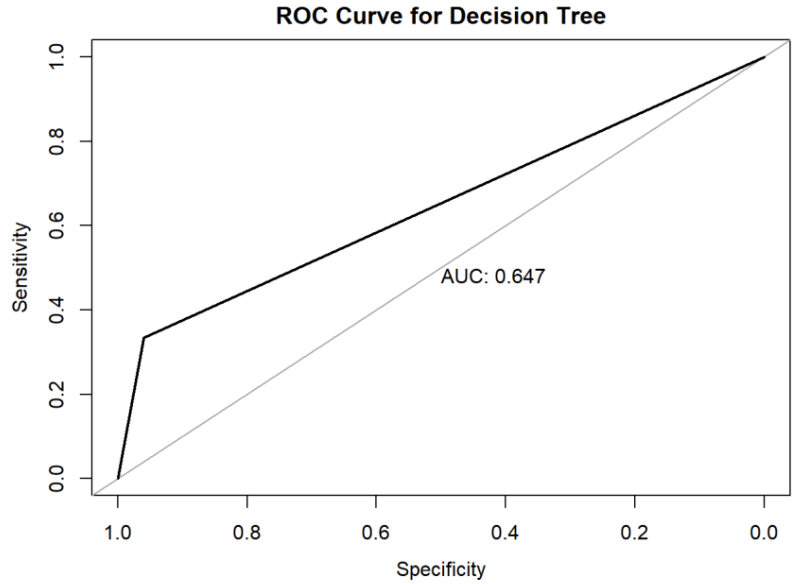
Appendix

A1 KNN – ROC and Confusion Matrix



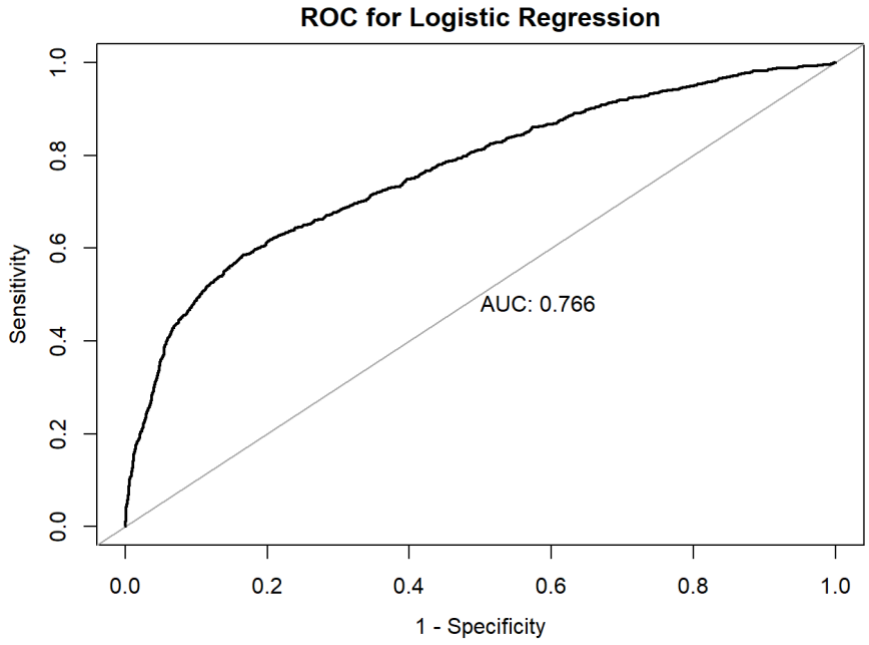

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction    0    1
##          0 4405  943
##          1  223  428
##
##                Accuracy : 0.8056
```

A2 Decision Tree – ROC and Confusion Matrix



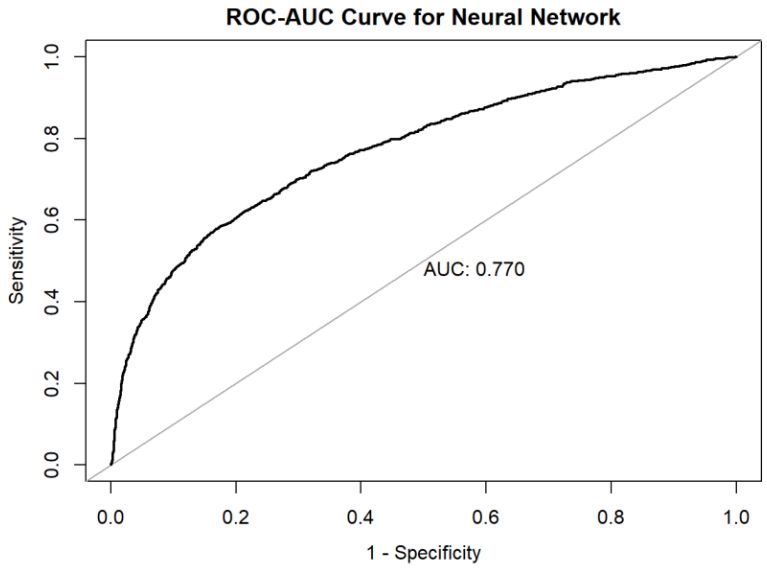

```
##            Reference
## Prediction  nodefault default
##   nodefault      74.9    14.8
##   default         3.2     7.2
##
##  Accuracy (average) : 0.8202
```

A3 Logistic Regression – ROC and Confusion Matrix



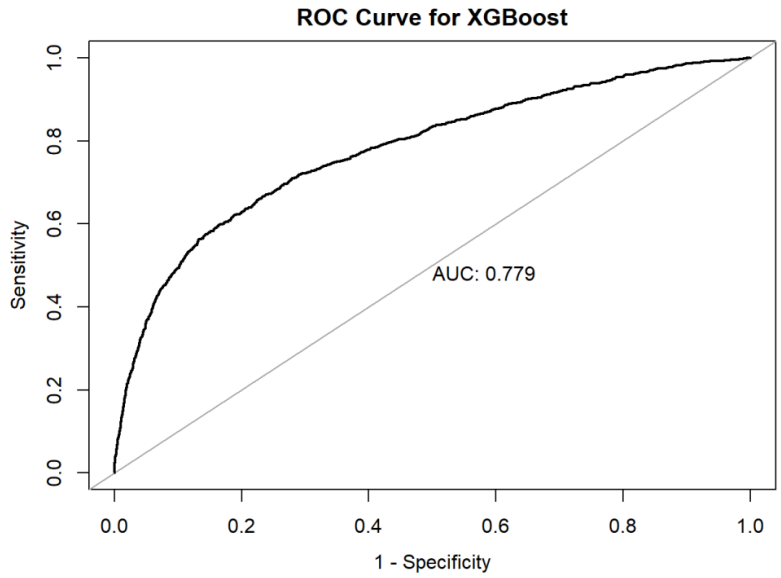
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4423  932
##          1  205  439
##
##                Accuracy : 0.8105
```

A4 – Neural Network – ROC and Confusion Matrix



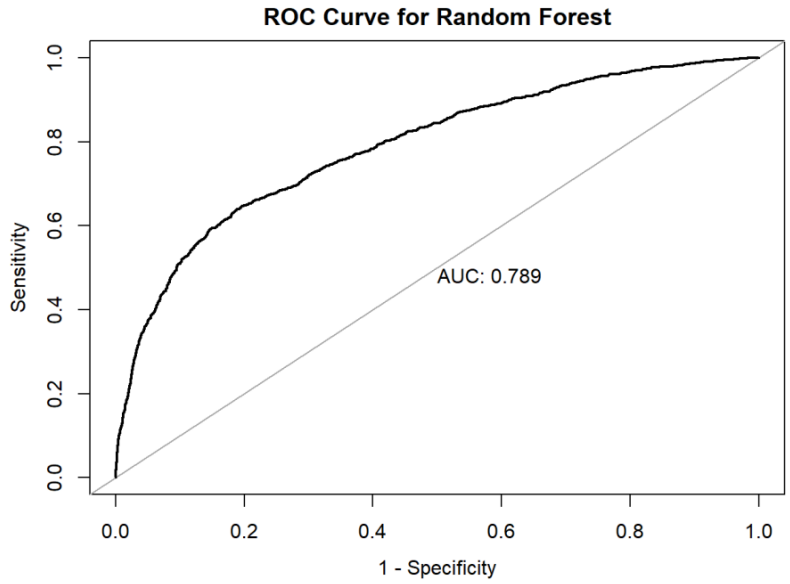

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4421  851
##          1  252  476
##
##                Accuracy : 0.8162
```

A5 – XGBoost – ROC and Confusion Matrix




```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4409  843
##          1  236  489
##
##                Accuracy : 0.8195
```

A6 – Random Forest – ROC and Confusion Matrix




```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction    0    1
##          0 4416  877
##          1  212  494
##
##                Accuracy : 0.8185
```