HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MASTER THESIS

Miss
Safa Binte Ayaz

Lead Scoring with Machine Learning

2023

Faculty of **Applied Computer Sciences and Biosciences**

MASTER THESIS

Lead Scoring with Machine Learning

Author:
Safa Binte Ayaz

Study Programme:
Applied Mathematics in Networks and Data Science

Seminar Group:
MA18w1-M

First Referee:
Prof. Dr. Thomas Villmann

Second Referee:
Michael Olschimke

Mittweida, Febuary 2023

Bibliographic Information

Binte Ayaz, Safa: Lead Scoring with Machine Learning, 61 pages, 24 figures, Hochschule Mittweida, University of Applied Sciences, Faculty of Applied Computer Sciences and Biosciences

Master Thesis, 2023

Abstract

This thesis investigates the efficacy of four machine learning algorithms, namely linear regression, decision tree, random forest and neural network in the task of lead scoring. Specifically, the study evaluates the performance of these algorithms using datasets without sampling and with random under-sampling and over-sampling using SMOTE. The performance of each algorithm is measure using various performance metrics, including accuracy, AUC-ROC, specificity, sensitivity, precision, recall, F1 score, and G-mean. The results indicate that models trained on the dataset without sampling achieved higher accuracy than those trained on the dataset with either random under-sampling or random over-sampling using SMOTE. However, the neural network demonstrated remarkable results on each dataset compared to the other algorithms. These findings provide valuable insights into the effectiveness of machine learning algorithms for lead scoring tasks, particularly when using different sampling techniques.

The findings of this study can aid lead management practices in selecting the most suitable algorithm and sampling technique for their needs. Furthermore, the study contributes to the literature by providing a comprehensive evaluation of the performance of machine learning algorithms for lead scoring tasks. This thesis has practical implications for businesses looking to improve their lead management practices, and future research could extend the analysis to other machine learning algorithms or more extensive datasets.

I. Contents

II. List of Figures

III. List of Tables

IV. Nomenclature

CRM	Customer Relation Management, Seite 4
$x = [x_1, x_2, ..x_n]$	row vector of the inputs, Seite 13
MQL	Marketing Qualified Leads, Seite 7
$\beta_1, ..., \beta_n$	coefficient values that control the change in y for a unit change in x_1, Seite 13
SQL	Sales Qualified Lead, Seite 7
d	Dataset, Seite 13
n	number of datapoints in input data, Seite 13
x	input variable, Seite 13
$\hat{y}$	target variable, Seite 13
MSE	Mean Squared Error, Seite 13
MLE	Maximum Likelihood Estimator, Seite 14
$L(\theta)$	Log loss, Seite 14
dtrue	Class of datapoints in DT that support decision rule, Seite 15
dtrue	Class of datapoints in DT that does not support decision rule, Seite 15
p_j	proportion of data points of class dtrue, Seite 16
$\Delta w_{ij} = C x_{ij} y_j$	Hebb rule, Seite 18
C	Learning rate constant in ANN, Seite 18
$w = [w_1, w_2, .., w_n]$	The row vectors of the weight, Seite 19
L	Loss function/Cost function, Seite 19
$H = h_1, h_2, ..., h_n$	a set of hyperparameters, Seite 22
G	Cartesian product of the ranges of all hyperparameters, Seite 22
f(h)	performance metric of the model trained with hyperparameters h, Seite 22
h*	optimal hyperparameters, Seite 22
M	Majority class samples, Seite 32
N	Minority class samples, Seite 32
$N^{'}$	One of the k-nearest neighbour of N, Seite 32
N_{new}	a new sample as a result of linearly interpolating between N and $N^{'}$, Seite 32
r	Pearson's correlation coefficient, Seite 35
F	features importance score, Seite 36
P	The number of independent variables in a linear regression model, Seite 36
$R2(x_i)$	coefficient of determination in a regression model, Seite 36

V. Preface

This thesis represents the culmination of my studies in Applied mathematics in network and data science at Hochschule Mittwedia University of Applied Sciences. It is the result of months of research, analysis, and writing, and it represents my original work unless otherwise referenced.

The thesis topic was inspired by the interest of Mr.Michael Olschimke, The Cofounder Scalefree International GmbH and Professor Thomas Villmann from Hochschule Mittweida, which made me able to choose it as my thesis topic. Their expertise and experience helped me in shaping the direction of my research and in ensuring the quality of my work.

I sincerely is grateful to Professor Thomas Villmann for supporting me in the writing of my thesis, proofreading the text and providing me with insightful feedback timely. Additionally, I would like to thank Mr. Michael Olschimke and individuals in the Information Technology department at Scalefree International GmbH for giving their valuable time and feedback that helped me determine the appropriate steps to take during this research. Also, I would like to express my gratitude to all those who have supported me in this endeavor especially my family and academic mentors.

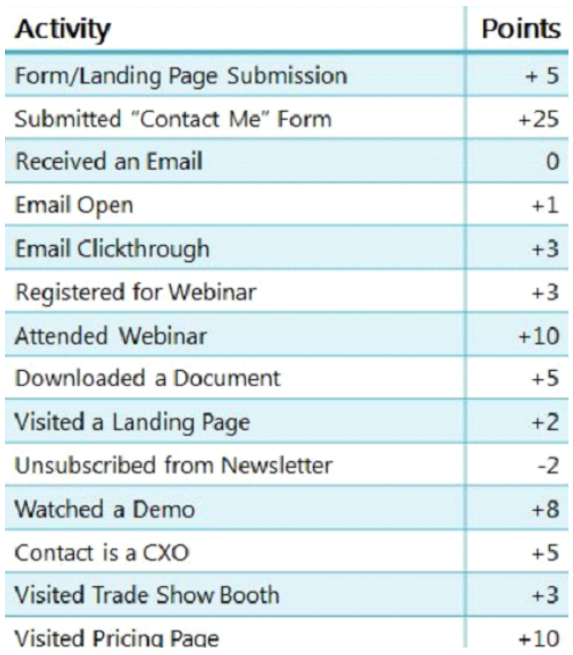
Finally, I hope that this thesis will contribute to the ongoing research about lead scoring with machine learning and provide a foundation for future research in this area. I am grateful for the opportunity to contribute to this field.

Safa Binte Ayaz - Mittweida, Feb-28-2023

1 Introduction

Over the past few years, in the competitive business world, companies need to take important business decisions related to customer possession [10]. Lead Scoring is among some of the newly emerging techniques used by the sales and marketing departments of various companies that assist to identify potential leads by awarding points against their actions [11]. It helps in understanding qualified lead profiles and producing more sales for the company by tracking prospects' online activities over time and rewarding value points for each activity they perform on a company platform.

Every company has a different framework for assigning points to score their leads. Some of the common online activities against which a lead is judged may include email clicking, watching a webinar, chatting with a company agent, subscribing to a newsletter for updates or reading a white paper, etc. [10]. All these activities might show the interest of a person in a product but no one can be sure enough that he or she will eventually purchase it. So, the whole sales process of different leads needs to be monitored and the score is defined against each activity. It helps determine lead intent and how intense the lead intent over a particular product is and to understand which lead deserves more points, the benchmark of the sale process is the ultimate purchase of a particular product or service. For example, the figure below defines the point-assigning procedure for each attribute of leads as activities.



Activity	Points
Form/Landing Page Submission	+ 5
Submitted "Contact Me" Form	+25
Received an Email	0
Email Open	+1
Email Clickthrough	+3
Registered for Webinar	+3
Attended Webinar	+10
Downloaded a Document	+5
Visited a Landing Page	+2
Unsubscribed from Newsletter	-2
Watched a Demo	+8
Contact is a CXO	+5
Visited Trade Show Booth	+3
Visited Pricing Page	+10

Figure 1.1: Manual lead scoring matrix [1].

Figure 1.1 is a table of lead attributes and their points based on activities.

Determining the right value point for action is compiled under lead qualification criteria, often known as a scoring matrix. It needs to be handled by authorities on the basis of assumptions. An example of a scoring matrix is shown above.

This type of lead scoring is typically known as point-based lead scoring or simply manual lead scoring. The biggest barrier in this type of lead scoring is to design of a scoring matrix based on user demographics. This typically falls under the task of statisticians and is done by stats software [12].

Another method that evolves over time and requires ML techniques to score the leads is known as predictive lead scoring. It typically requires the available data of leads to analyze the purchase probability using predictive analysis of historical data and understand the similarities and patterns in the buying process of different leads [12].

Our thesis aims to acquire a comprehensive understanding of lead scoring through the application of machine learning (ML) techniques.

1.1 Related Work

Take a simple example of book shops, booksellers can keep track of what books are sold and which customers bought it and that's all the information they can access. Which customers looked into which books and how they navigated through the buying process can not be monitored for every customer, but once the online market came into existence, the understanding of customer interest and need altered completely. It became effortless to monitor not only customers' purchases but also the books they searched for, how much time they spent online, what are their interests, how they reacted to website layout, ads, reviews, promotions, etc, additionally, the similarities and correlations among customers. ML algorithms would help the retailers to find out what may be the interest of customers, and what they might find helpful for their consumers in the future, for e.g, what books they would like to read next. Traditional buying and selling methods simply couldn't access this kind of information [13].

The big data revolution has made life easier as it is the job of big data to do the hard work [13]. Since Information Technology(IT) evolved, companies are heavily investing their time and energy to utilize the available data to upgrade their business [11]. By monitoring the behavior and interest of prospects while in the buying process, it is possible to find hidden patterns and relationships in their actions, and understanding ideal profile and behavioral characteristics as they proceed, ultimately helps in focusing on the right leads for the sales team.

Since data-driven decisions typically result in better decisions, using data-driven methodologies to make critical decisions and resolve various business problems will soon become the new normal [13]. Another important saying from a review [13] that depicts the importance of big data analysis in making smarter decisions is "You can't manage what you don't measure." Rather than the following gut, the data does the job for smarter future predictions. The more we have the volume of data, the better we can expect from predictive models.

From an overview in [13] it can be concluded that companies should follow some data-driven intensive methods as a replacement for manual methods as it claims that pure domain knowledge is not enough for big data analysis. Also, traditional methods are vulnerable. Researchers have been focusing on the application of ML to lead scoring in recent years. There's a clear requirement for further investigation of these methods. This thesis points to constructing a predictive model by looking at these methods.

Companies are spending major revenues on marketing [14]. For example, by emailing them, asking them to fill out forms, sending them free subscription letters, or just simply call them. They try to build customer relations and trust over time through various strategies and platforms to attract them for business but still the consumer journey is vague to them. Lead data that include their personal information, such as name, country, email address, the channels through which leads are acquired, and explicit data such as contact's ID, activity data that may include website visits, email opens and click, form submission, etc. can be used to make the digital data beneficial for the advertising market. Marketing agents try to engage as many consumers to try to convert them into leads. Executing lead scoring as a portion of your overall marketing technique can assist you in superior marketing endeavors while making you more efficient. 74 percent of top-performing businesses, according to Gleanster, use automated lead nurturing [14].

G. Cui et al in [14] worked on a large data set using a Bayesian network with evolutionary programming and compared the results with deep learning models and tree-based models and claimed that Bayesian works better in terms of accuracy. Whereas, Benhaddou et al in [15] experimented with small data sets using the Bayesian Networks as a lead-scoring model.

Moreover, a couple of studies have been conducted to analyze the performance of stochastic and non-stochastic models to handle lead scoring. One of many studies in [10] represents how the combination of linear regression with tree-based models such as Random forest performs in this domain and compares and contrasts the result to gain potential business insights important for decision-makers.

1.2 Problem Statement

With the emergence of digital marketing, B2B(business-to-business) companies have started to reach prospects more than ever before and generate more leads. This also means businesses will acquire a mix of leads with different qualities. Some will be relevant and some won't be as relevant or 'qualified'. Getting potential customers out of their audience is a constant issue for most businesses.

Another problem if a company has thousands of leads to work with, which is often the case, working with them on an individual level becomes impossible. The purpose is to lessen the labor cost of continuously interacting with the leads and estimate the results based on those interactions for each prospect.

The ideal technique is to think about lead scoring to improve the productivity of the sales force's process. Genuine browsing data from the contacts are used to prepare the data for classification, and a special buy moment is used as the final step for the contacts who have previously bought the product.

This thesis main goal is to examine how machine learning (ML) models can be used to improve lead scoring in B2C (business-to-customer) and B2B (business-to-business) scenarios. Data Manipulation Language (DML) operations are used for data access and prepossessing. In addition, 70% of the input data is used for training ML models and comparing ML classifiers to calculate lead scores, with the remaining 30% being used for testing. Furthermore, the other purpose is to serve the CRM by its special application case of customer conversion and what type of audience to target by analyzing the lead scoring results and an effort to unveil business cognizance, for instance, what types of leads to target.

The extensive research objective of this thesis is to answer the following questions

- How ML can be integrated to carry out effective lead scoring and what is the role of ML in automating lead scoring tasks?
- What ML algorithm to select that gives better results in achieving lead scoring?
- How the results and interpretation will help the business to understand lead behaviors and what the business needs to interpret from results?
- How do the sampling techniques when applied to our data set produce results for evaluating a classifier and which classifier combined with sampling technique outperforms the others?
- How does the model tuning affect the performance of the classifier?

By contrasting the models, we will assess how well deterministic and non-deterministic models perform in this domain. We will deal with Logistic Regression, Decision Trees under the category of the deterministic model, Random Forests, and Neural Networks under the category of non-deterministic models as our ML techniques. By contrasting the execution results of both approaches, the model selection reveals how well ML models handle this particular challenge. The outcomes will then be assessed to determine which ones perform best in terms of accuracy when implemented without the application of sampling methods and after the application of under-sampling and oversampling methods.

1.3 Thesis Structure

The following six chapters make up the thesis:

In chapter 2, we introduce and examine two distinct lead scoring methodologies along with the issues and associated challenges.

In chapter 3, we define the working of machine learning algorithms with their advantages and disadvantages. Moreover, we discuss the model tuning method. Furthermore, the evaluation techniques to access the classifier performance are discussed in detail.

In chapter 4, we define the data preprocessing methodologies, including the missing data issues and how to overcome them. We discuss data splitting and resampling techniques and some well-known and important data transformation and feature selection techniques to improve the model performance.

In chapter 5, we provide an overview of our dataset and a comprehensive discussion of the experiment as a whole. We also present the results of the exploratory data analysis we do in this thesis. Moreover, we show the results of each classifier in terms of the evaluation techniques. Finally, we compare classifier and estimate methodologies to provide our findings with and without sampling techniques using graphs and tables.

In chapter 6, we conclude the thesis by summarizing the key findings and contributions of the study, while also acknowledging its limitations and areas for future research.

2 Theory

2.1 Lead Scoring

Manual lead scoring is a widely used lead scoring technique for identifying, prioritizing, and qualifying specific users based on their actions, demographics, and purchase probability, as briefly covered in the first chapter. These users are known as leads in the business sector. As to begin, the organization may score the lead on the basis of the data available to the team [10].

For example, if a user, let's call it a contact, spends five minutes on the web page reading a blog post and does no further activity will be assigned five points but the contact that chatted with the team or downloaded an ebook could be assigned points five. The reliable scoring criteria as per activity performed by clients need to be built by marketing experts keeping in mind the business need. The contact propensity to purchase, also known as lead readiness, is measured by the sum of the scores given to a single lead. In such a case assuming that contact with a high score also called hot leads has a high purchase probability. The idea here is to reduce the efforts of salespeople to utilize less time on leads that score lower than others also called cold leads.

A sales funnel is depicted in Figure 2.1, showing the stages of a lead from the time it is created till the time it is converted into a customer. A study by Elin Lindhal in [1] suggests how leads can be approached in different stages of the purchase funnel or in a customer journey.

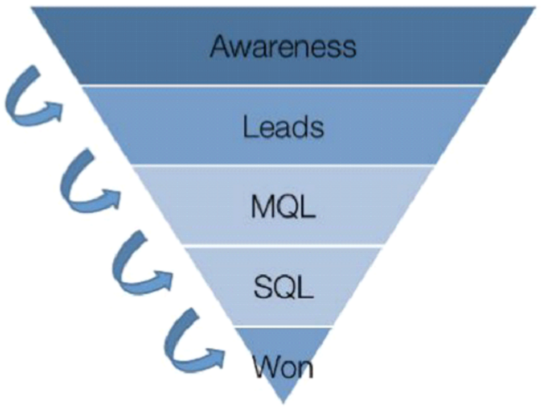


Figure 2.1: Sales Funnel defining the position of lead in a purchase cycle [2].

The horizontal blocks depict the lead current position. The width from top to bottom is reduced showing the number of contacts decreasing as the journey to purchase proceeds and fewer people reach the final stage. The first block is in the awareness stage with the lowest score, at this point lead is represented as the contact who encounters the company through different channels. One can surf and engage in activities or stop, but doing so leaves a digital trace that could be later utilized for tracking. This stage helps in understanding their customer and slicing and dicing them into different categories depending on their interest and activities. It is also known as the acquisition stage. It also identifies the best sources of lead in terms of lead acquisition. If they continue and engage in more activities, they may grow into MQL(Marketing Qualified Leads) leads with better scores. But, at this point lead may need some more nurturing campaigns by marketers for product promotion to increase lead interest. Some nurturing strategies such as sending emails, sending an offer to try the service for free, etc. can help at this stage. This phase also gives insight into what products are closely related to different types of customers and helps in product affinity. In the later stage, leads are closer to reaching a threshold point i.e won, so it can be reached directly by the sales team with a sales offer or discounts. This stage is termed SQL (Sales Qualified Lead). If the leads fulfill the demands of being considered as a sales opportunity by meeting the minimum criteria set by management and have a good demand, they are qualified for the SQL stage which is the last stage in the funnel. In addition, the historical record of lead involvement with the company in the past defines the level of interest the lead has, either low or high [1]. The sales staff follows up with the sales-qualified leads, which might lead to a successful or unsuccessful purchase [16].

However, assumptions on scoring a lead based on domain knowledge have some serious drawbacks as discussed in [17] below:

- The key concern is that manual lead scoring needs a lot of data to provide accurate behavioral, demographic, or firmographic scoring. Unfortunately, there is a severe lack of access to reliable statistical data, which is necessary for setting the appropriate value for the action.
- In manual lead scoring, the scores could be determined by a set of predetermined guidelines. Whilst manual scoring does define a set of rules, it doesn't refrain to incorporate the attributes to judge an approaching lead, some of them may be irrelevant.
- It is considered to be a more time taking process because it demands to be overhauled each time new behavioral changes are observed in leads. As processes change and new buying behaviors appear, the lead scoring model should be flexible enough to accommodate these changes. To continuously keep altering the scores the time utilized might be spent more viably somewhere else. Also, companies are abandoning this due to long processes according to Marion [17].
- The data relating to manual scoring is often complex and it is too much for a human mind to process. In this manner, all of the methods proposed to lead scoring should make use of a data-driven approach that a computer can manage, according to Bohlin [12]. This necessitates the use of predictive lead scoring, also known as automated lead scoring, which is a data mining application used in lead scoring.

2.2 Predictive Lead Scoring

An Artificial Intelligence (AI) assisted lead scoring approach which defines mathematical algorithm-based scoring procedures commonly known as predictive lead scoring. Or simply we can say that these are a set of statistical techniques that work on historical data to analyze the leads' demographics and attributes to predict the probability of a lead conversion, that is why it is called predictive lead scoring. It differs from the traditional way of lead scoring as it is an automatic method to do the research and calculations for the company with a substantial decrease in human error and efforts. ML is another name for data mining techniques or pattern recognition [18].

So giant firms that own more customers to generate good amounts of data often use predictive methods so the algorithms can be fed with enough information for better results. However, if a company faces inconsistencies and huge gaps and has big data it is always a good idea to shift to predictive modeling for lead scoring rather than traditional methods to see if it is more accurate and can help the sales team improve the conversion rate. Also, the comparison of previous and present customers helps create a profile of qualified leads [18].

With predictive analysis for scoring leads comes an additional strategy called predictive marketing [16]. Predictive marketing can be defined as a customer-driven marketing approach with the objective to empower each customer's journey throughout the purchase life cycle. This strategy came into existence because it is assumed that customers nowadays contemplate highly customized walks through the journey of interacting with companies to fulfill their needs. This journey is made possible through mechanics that capture effective data through distinctive digital channels in order to carry out customized marketing campaigns and target loyal customers [18].

An important component of customer relationship management (CRM) is lead scoring. There are various CRM tools that are available in the market right now, name a few are:

- Salesforce
- Oracle
- HubSpot Sales
- SAP (Systems Applications and Products) Sales Cloud

CRM is a user-friendly tool that provides functionalities to boost a sales process. It is software that has a database to keep information of contacts, leads, and customers data that is captured through different offline and online modes and kept safely to analyze and act for business profit. These tools help optimize digital channel collaboration with the users to be able to collect effective information related to customers to target them and build customized nurturing campaigns.

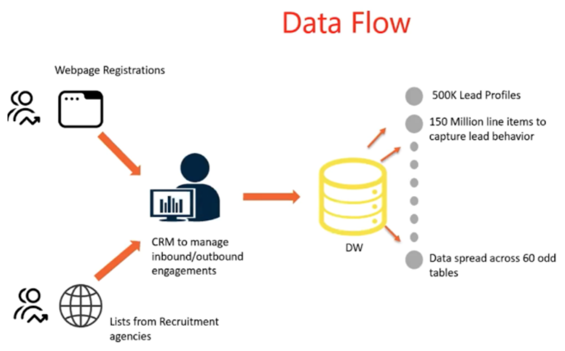

Data Flow
Webpage Registrations
CRM to manage inbound/outbound engagements
Lists from Recruitment agencies
DW
500K Lead Profiles
150 Million line items to capture lead behavior
Data spread across 60 odd tables

Figure 2.2: CRM Data Flow Chart

Figure 2.2 shows that the client got leads from 2 distinctive channels. Offline mode and online mode. Digital channels and third-party recruiters. CRM in between is managing and optimizing the big data as per the requirements of their customers.

The 5 key steps that CRM processes to keep a customer interaction with a company personalized are determining a potential prospect, acknowledging the prospect, trying hard for prospect conversion to the customer, lead retention, and being loyal to their customers.

2.3 The significance of customer attributes in the process of lead scoring

The most important step in lead scoring is the suitable selection of variables while building a predictive model that largely affects the quality of results we expect from a model. It can mainly be divided into two types of variables. This task is considered crucial and one should research properly before selecting the variables [10].

Both explicit and implicit customer attributes are important in lead scoring, as they provide different types of information that can help companies determine the likelihood of a lead becoming a customer.

Explicit customer attributes refer to information that is explicitly provided by the lead, such as job title, company size, and location. This information is typically obtained through lead generation forms, surveys, or other direct interactions with the lead. Explicit attributes are important in lead scoring because they provide a clear understanding of a lead's demographics, which can help companies determine if the lead is a good fit for their product or service [10].

Implicit customer attributes, on the other hand, refer to information that is inferred from a lead's behavior or actions, such as website activity, content consumption, and email engagement. Implicit attributes can be valuable in lead scoring because they provide insights into a lead's interests, needs, and level of intent to purchase. For example, a lead who spends a significant amount of time on a company's pricing page or downloads a product demo may be more likely to convert into a customer than a lead who simply visits the company's homepage.

Incorporating both explicit and implicit customer attributes into lead scoring models can help companies develop a more complete picture of each lead and their likelihood of becoming a customer. By combining demographic, firmographic, and behavioral data, companies can identify high-value leads that are both a good fit for their product or service and highly engaged and interested in making a purchase.

Overall, the significance of both explicit and implicit customer attributes in lead scoring cannot be understated. By using a combination of these attributes, companies can improve the accuracy of their lead-scoring models, increase the efficiency of their marketing and sales efforts, and ultimately drive higher conversion rates and revenue growth.

2.4 Classification Algorithms used in Predictive Analysis

ML algorithms are divided into three major categories i.e, Supervised learning algorithms that perform prediction on new data when the training data has already known classes. It is further divided into classification and regression problems. Unsupervised learning algorithms do the prediction by looking into data to find similar patterns without knowing the outputs. Clustering and association rules are some types of unsupervised learning. However, reinforcement learning algorithms search for hidden patterns in the data in order to suggest the optimum course of action [19].

These two groups of supervised learning algorithms differ in that where regression is used to estimate continuous values, on the contrary, classification is used to predict categorical outcomes. In addition, classification models contain two outputs: the discrete category and a value between 0 and 1 that indicates how likely it is that a given example corresponds to the specified category. Lead scoring is a classification problem that can be solved. This chapter's main goal is to give readers a fundamental grasp of how to use supervised learning techniques to tackle classification problems [20].

3 Machine learning

In this section, we will discuss and compare a number of linear and non-linear classifiers in detail. We will further discuss one of the popular model tuning techniques. Finally, we will discuss and compare evaluation techniques that are important to measure the performance of the classifier.

3.1 Classifiers

3.1.1 Logistic Regression (LR)

Logistic regression generally belongs to linear regression models and defines a linear relationship between the output variable and features using an equation, mathematically given as:

$$\hat{y} = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n \tag{3.1}$$

Suppose the data set d consists of n number of observations, the row vector of the inputs are $x = [x_1, x_2, ..x_n]$.

For each observation $x_1, .., x_n$ target variable$\hat{y}$ is defined, β_0 is average of $\hat{y}$, and $\beta_1, ..., \beta_n$ are coefficient values that control the change in $\hat{y}$ for a unit change in x_1.

It uses the mean squared error (MSE) method to define a straight line in the given dataset which is the difference between the actual output and the predicted class label. Linear regression works by interpreting the continuous variable output with respect to the features (input). That's why it is good for regression problems such as predicting weight relevant to body size, mass, height, etc., or predicting a house price given the relevant data [3] [4].

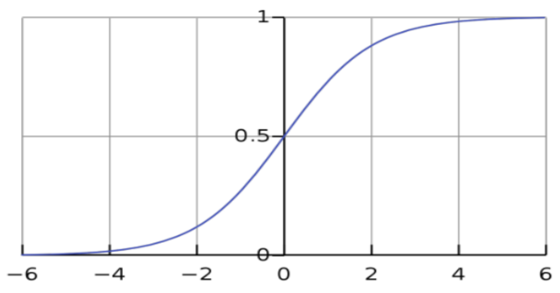


Figure 3.1: Logistic Curve [3].

To calculate conditional probabilities it uses a function called a logistic function also called a sigmoid function [21], mathematically given as,

$$P(Y|X) = \frac{e^{\beta_0+\beta_1 x_1+...+\beta_n x_n}}{1+e^{\beta_0+\beta_1 x_1+...+\beta_n x_n}} \tag{3.2}$$

In the expression above, P(Y|X) is the likelihood of being in one class., and $\beta_0, \beta_1 x_1 + ... + \beta_n x_n$ are the estimated coefficients. By restricting the range, if multidimensional features of data vector X become more than the number of observations n, then model training becomes complex, resulting in bad predictive behavior. So, It can be rewritten in the form of the log as:

$$log\frac{P(X)}{1-P(X)} = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n \tag{3.3}$$

Where $\frac{P}{1-P}$ is the odds ratio of the likelihood of success and likelihood of failure which will always be positive ranging $(0, +\infty)$.

Cost Function: We use Mean Squared Error(MSE) in linear regression to derive a cost function but in logistic regression the logistic function is nonlinear so using MSE gives a non-convex graph and it might get stuck in local minima. To solve this issue it uses a Maximum likelihood estimator (MLE) to derive a cost function known as log loss given as,

$$L(\theta) = -(ylog(\hat{y}) + (1-y)log(1-\hat{y})) \tag{3.4}$$

θ is the parameter set that the model is attempting to learn. Therefore, $L(\theta)$ calculates the loss. The objective of MLE is to estimate model parameters for which the likelihood function is maximized. For this purpose of minimizing the cost function, gradient descent optimization is used.

Advantages:

- It is easier to implement, interpret, and efficient to train [10].
- They are accurate with huge data analysis, especially if the data set can be separated linearly [4].

Disadvantages:

- The capabilities of logistic regression are occasionally constrained by the basic premise that dependent and independent variables have a linear relationship [21].
- Overfitting is common in logistic regression, especially If there are too many dependent variables in the model. To get around this problem, regularization (L1 and L2) methods might be applied [22].
- Logistic regression cannot be used to tackle non-linear issues. However, it is

uncommon to encounter linearly separable data in real-world situations.

3.1.2 Decision Trees (DT):

Decision Trees can be used for classification and regression. In DT information flows in a tree-like structure from top to bottom, step by step, where the root nodes represent an event or an attribute as a question, a sub-tree is all the branches that represent event sequence as a decision rule with probabilities of if-else conditions and leaf nodes represent the end of the tree that gives one class as an output. For an effective DT, it must contain all the possibilities of an event happening and if one event is conditioned as true the other must be false.

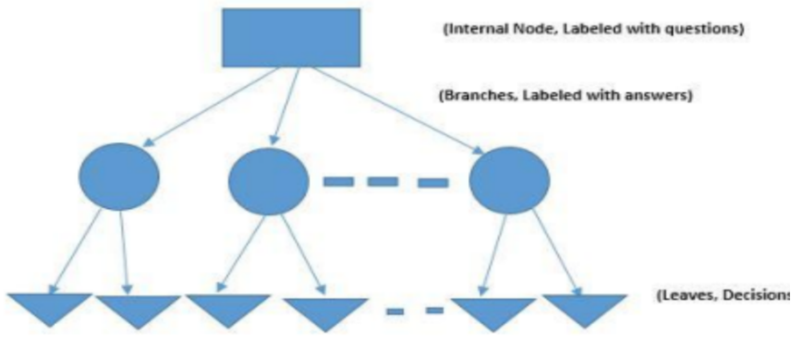

(Internal Node, Labeled with questions)
(Branches, Labeled with answers)
(Leaves, Decision

Figure 3.2: Decision Tree Visualization [4].

As seen in the above figure the data set is interpreted as a set of rules (branches) from the root (internal node) to the leaf node [1] [4]. The dataset is split recursively into smaller chunks, based on the attributes until a stop criterion is reached. It aims to find a set of data points as one group with uniquely identifiable attributes until the last step to belong to a specific predicted class [23]. Moreover, they have fast adapting and classification abilities.

Typically, decision trees are constructed via a sequential binary process. For example, d is the dataset, and for each attribute of input data x, different split points are evaluated.

Moreover, DT splits the data into groups of homogenous subsets means one group contains more data points from one class than the other. Mathematically, the points $x \in d$ that support the decision rule are in category dtrue and the points that do not support the decision rule are represented by the category dfalse.

The decision rule is typically based on a binary test of the form "does feature i belong to category dtrue?", where i is the index of the feature and $<=$ is a comparison operator for numeric features and belongs to is a membership operator for categorical features. The category is chosen to maximize some splitting criterion that measures the homogeneity or purity of the subsets with respect to the target variable y.

Different types of splitting criteria are taken into consideration while the decision tree is deciding where to split the tree and how to apply rules and the aim is to optimize them. Accuracy, the Gini index, and cross-entropy are some possible criteria. Mathematically given as [3],

$$GiniIndex = 1 - \sum_j p_j^2 \tag{3.5}$$

$$Entropy = -\sum_j p_j.log_2.p_j \tag{3.6}$$

where p_j is the proportion of data points of class dtrue.

$$GiniIndex_{min} = 1 - 1^2 = 0 \tag{3.7}$$

$$GiniIndex_{max} = 1 - (0.5^2 + 0.5^2) = 0.5 \tag{3.8}$$

$$Entropy_{min} = -1.log_2.(1) = 0 \tag{3.9}$$

$$Entropy_{max} = -0.5.log_2(0.5) - 0.5.log_2(0.5) = 1 \tag{3.10}$$

The minimum value a Gini index and entropy calculates is 0. It means that the node cannot be split anymore and all the data points belong to one unique class, thus the less the Gini index and entropy value the pure the node is considered and the optimum split is chosen on the basis of low score values of the features in DT [6].

However, when the data points represent the two classes at the same probability, the Gini index and entropy get the maximum value in such a case. Both measures tend to minimize node impurities [3] [6].

Advantages:

- The best-known features of DT are their great interpretability, fast learning abilities, and capacity for dealing with nonlinear issues [24].
- DT is better because it can handle missing data and prune unnecessary branches.
- It can handle comparatively larger datasets and interpret a variety of predictor types [25].

Disadvantages:

- Overfitting is a common occurrence in DT, particularly when the tree is deep and the number of samples are limited. [25] [24].
- It can also be sensitive to small changes in the data. [25].
- It can be sometimes biased towards features with too many levels or values [25].

3.1.3 Random Forest (RF)

Random Forest (RF(is an ensemble learning technique that aggregates multiple decision trees to enhance the accuracy and generalization of the model. It was found to be one of the outstanding algorithms in terms of classification accuracy on various datasets [26].

As seen in figure 3.3, the RF algorithm creates a collection of DT by training them on different subsets of the training data and then utilizes this ensemble of trees to make predictions.

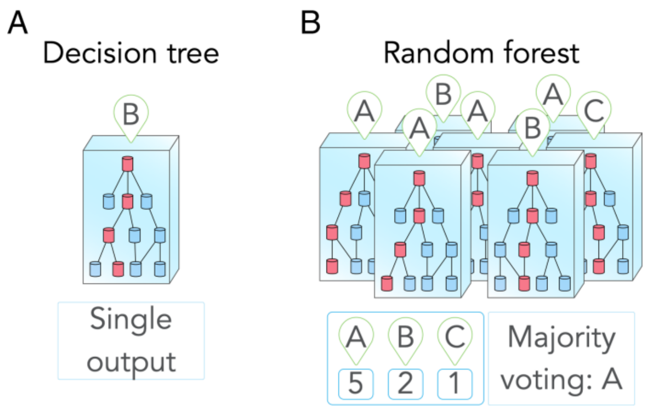


Figure 3.3: Random Forest Visualization in comparison to Decision Tree. [5].

Each DT is built on a random groups of the input samples, which helps to reduce overfitting and increase model diversity. During prediction, the algorithm aggregates the predictions of all the trees to obtain the final output.

Cost function: Random Forest minimizes the MSE between the outcome values and the actual input values for regression problems and minimizes the Gini index for classification [27]. The cost function is optimized during training by adjusting the split criteria at each branch of the DT to increase the impurity of the split.

Advantages:

- RF is a highly accurate algorithm that performs well on a wide range of problems.
- It can handle both categorical and continuous features making it a versatile algorithm.

- It is less prone to overfitting than single decision trees, as the averaging of multiple trees helps in decreasing the effectiveness of noisy data [28].
- It is computationally efficient and can be parallelized to speed up training on large datasets according to a study by Breiman et al. [27].

Disadvantages:

- RF can be challenging to interpret as its output is a compilation of several decision trees, making it arduous to comprehend the algorithm's prediction process. Nevertheless, one can overcome this challenge by utilizing feature importance measures to recognize the most significant features in the model [27] [29].
- It may not perform well on very high dimensional data or data with highly correlated features [30].
- It can be sensitive to noisy or irrelevant features, which can negatively impact performance [31].

3.1.4 Artificial Neural Network (ANN)

The working of ANN is inspired by the human brain. Biological neurons are the units here and the connections between neurons is termed as synapses. It comes under the category of deep learning models and is widely used as a supervised learning algorithm for classification and regression problems. Several nonlinear algorithms come under neural networks, unlike logistic regression such as feedforward neural networks and recurrent neural networks. It is one of the most popular algorithms in ML that is capable of modeling extremely complex relations [4].

Many units are working together and are interconnected through layers. The first layer is the input layer which consists of input variables that need to be predicted. Then the hidden layers come in between the input layer, where all the calculations are performed and atlast the output layer that shows result.

Learning of a model is performed by adaptation of the weights w also known as the strength of parameters.

$$\Delta w_{ij} = Cx_{ij}y_j$$

is the Hebb rule that represents the adaptation in weights where C is the learning rate constant, x is the input, and y represents the result. The simplest neural network is a perceptron which has a single hidden layer.

In the figure above we can see the multilayer feedforward ANN in which the first layer is the predictor layer connected to hidden layer 2. Each unit assigns a numerical value to each connection known as weight. Every Input is multiplied by respective weights and the multiplied values are added, the sum is passed to the working units present at

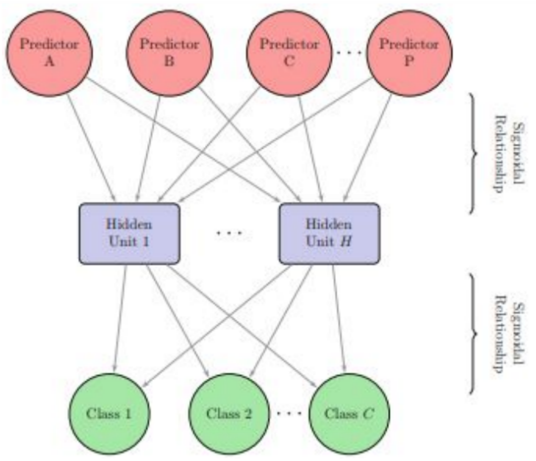


Figure 3.4: Classification Neural Network with 1 hidden layer [6].

hidden layers [32].

$$\Sigma = (x_1 w_1 + x_2 w_2 + ... + x_n w_n) \tag{3.11}$$

The row vectors of the input and weights are $x = [x_1, x_2, .., x_n]$ and $w = [w_1, w_2, .., w_n]$ respectively and their dot product is given by [32],

$$x.w = (x_1 * w_1) + (x_2 * w_2) + ... + (x_n * w_n) \tag{3.12}$$

Hence, the sum is equivalent to the dot product of the vectors x and w [32]:

$$\Sigma = x.w \tag{3.13}$$

Finally, the resulting value is sent to a nonlinear activation function which decides whether the neuron will transfer data or not. If the value is above the threshold value the neuron will fire the result to the output value otherwise the value is not passed, However, sigmoid is used as an activation function to keep the calculations between 0 and 1. The class with the greatest value then becomes the expected output class for the results.

Cost Function: The loss function L or cost function of ANN can be given as the sum of error at each layer follows where MSE is the square of the distance between the actual y_i and predicted value $\hat{y}_i$: [32]

$$L = MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3.14}$$

The objective of the algorithm is to enhance the performance of the model by optimizing the cost function using backpropagation algorithms. Derivatives are calculated at each iteration to find the optimal parameters. The process of updating the weights and bias is performed repeatedly, and the backpropagation step is iterated until convergence [6].

Advantages:

- NN works in parallel and thus is able to process large sets of information and complex datasets [33].
- NN learn from continuously adjusting weights within themselves, hence are not limited to learning through only the input features provided to them [34].
- NN stores the input data in its own network rather than a database so the loss of information has very low chances [35].
- If a neuron stops functioning NN has the ability to still produce useful output. They have the ability to provide effective and practical solutions to real-time problems [36].

Disadvantages:

- NN is sensitive to the initialization of weights [35].
- NN has nonlinear sigmoid functions which are often computationally expensive to backpropagate the error [32].
- NN might get stuck in local minima. In order to avoid this situation, it is better to train the NN model multiple times on random batches or use K-fold cross-validation techniques. They are prone to overfitting, especially if the network is trained for too many iterations or if the network becomes too complex compared to the size of the dataset [37].

3.2 Comparison of Classification Algorithms

Table 3.1 compares different classification algorithms we used in our thesis in terms of their computational complexity, interpretability, robustness, and other important factors. The variables used in this context are "N" for the sample size, "d" for the number of features and "L" for the number of layers in the neural network.

In terms of computational complexity, logistic regression has linear time complexity with respect to the number of samples N and the number of features d. Decision trees and random forests have higher complexity due to the need to construct decision trees, but the complexity is still manageable for most datasets. Neural networks have higher complexity due to the number of layers L, but can still be trained efficiently using modern hardware and software.

In terms of interpretability, logistic regression, and decision trees are relatively easy to interpret because they provide coefficients or rules that directly relate the input features to the output label. Random forests and neural networks are less interpretable because

Algorithm	**Complexity**	**Interpretability**	**Robustness**	**Other Factors**
Logistic Regression	$O(dN)$	High	Sensitive to outliers	Good for linearly separable data.
Decision Tree	$O(Ndlogd)$	High	Sensitive to outliers	Good for non-linearly separable data.
Neural Network	$O(Ndlogd)$	Low	Robust to overfitting	Good for high dimensional data.
Random Forest	$O(NLd)$	Low	Robust to non-linearities	Good for non-linear data.

Table 3.1: Comparision of Classification Algorithm

they involve combining multiple decision trees or layers into a single model.

In regards to their robustness, logistic regression is affected by outliers and may underperform in datasets with high noise levels. Decision trees are at risk of overfitting and may not be able to generalize well to new data. Random forests are more robust against overfitting and excel in handling high-dimensional data. On the other hand, neural networks are resistant to non-linearities and can effectively capture intricate non-linear relationships between the input features and output labels.

Other important factors to consider include the suitability of each algorithm Various categories of data such as those that are linearly separable, non-linearly separable, high-dimensional, complex non-linear, the ease of training and tuning each algorithm, and the availability of software packages and libraries for each algorithm.

It's worth noting that the choice of classification algorithm and the importance of different factors depending on the specific problem and the trade-offs between different algorithms.

3.3 Model Tuning

Choosing the appropriate metric for optimization is crucial for determining the appropriate hyperparameter values. The process of searching for the optimal hyperparameter configuration is a delicate task [38]. However, it is not feasible to discover a flawless model because it requires evaluating numerous values for each hyperparameter, which can become impractical in terms of time and computational expenses, particularly for large datasets. Therefore, we will explore one of the widely used techniques for model tuning, which is called Grid Search.

3.3.1 Grid Search

Grid Search is a method for hyperparameter optimization in ML, where a range of hyperparameters is specified, and a grid of all possible combinations of hyperparameters is created. Each combination is then evaluated by training the model using the specified hyperparameters and a performance metric such as accuracy, precision, recall, or F1 score is computed. The combination of hyperparameters that produces the best performance metric is then selected as the optimal set of hyperparameters [39].

Suppose we are given a set of hyperparameters $H = h_1, h_2, ..., h_n$ with each hyperparameter having a range of possible values represented as $R(h_i)$. To create a grid of hyperparameter values, we take the Cartesian product of all the hyperparameters' ranges, which is denoted as:

$$G = R(h_1) * R(h_2) * ... * R(h_n). \tag{3.15}$$

For each combination of hyperparameters in G, we train a model using a training set and evaluate its performance on a validation set. Let f(h) be the performance metric of the model trained with hyperparameters h, and let D be the validation set. We define the optimal hyperparameters h* as:

$$h* = argmaxf(h) \tag{3.16}$$

In other words, h* is the combination of hyperparameters that results in the highest performance metric on the validation set.

Example: For example, we have a dataset of 1000 samples, with 10 features, and a binary classification task. We want to train a Decision Tree model to classify this dataset, and we want to optimize the hyperparameters using Grid Search. Let's say we use the following ranges for the hyperparameters:

$$max_{depth} : [2, 4, 6, 8]$$

$$min_{samples-split} : [2, 4, 6, 8]$$

$$min_{samples-leaf} : [1, 2, 3, 4]$$

We can create a grid of all possible combinations of hyperparameters. For each combination, we can train a Decision Tree model using the specified hyperparameters and evaluate its performance on a validation set. Let's say we use accuracy as the performance metric. After training and evaluating all possible combinations of hyperparameters, we find that the combination that produces the best accuracy on the validation set is:

$$max_{depth} : 6$$

$$min_{samples-split} : 4$$

$$min_{samples-leaf} : 2$$

We can then use these hyperparameters to train a final DT model on the entire data-set and evaluate its performance on a test set.

Grid search is a brute-force method that can be computationally expensive, especially when the number of hyperparameters and the range of values for each hyperparameter is large. However, it is a simple and effective method for hyperparameter tuning that can often lead to improved model performance [40] [38] [41].

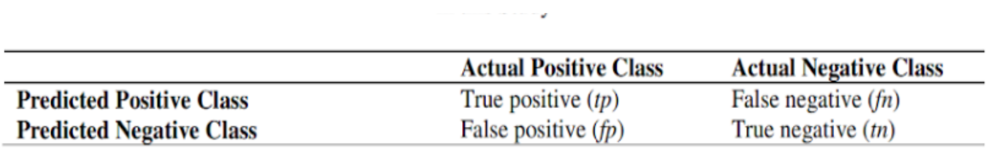
3.4 Evaluation Techniques

Once the model is trained the output is generated as a class label with a probability value of 0 to 1 for belonging to a certain class, we need to measure the quality of classification algorithms using performance evaluation measures for the classification models. It depicts how well the model performs and how effective the trained classifier will be while classifying the unknown data.

From the review [21], the evaluation criteria can be represented in multiple ways, i.e, either through a threshold value through the probability or percentage of effectiveness, or by ranking metric [42]. Each one of these is a scalar quantity which is represented by a numerical value. Thus, compare and contrast analysis of different classifier performances becomes an easy task and later assists in model selection. The evaluation criteria have a strong foundation as a discriminator in order to distinguish and select the best solution among the variants during the training phase.

3.4.1 Confusion Matrix

One common method being used by researchers to evaluate the generalization performance of a model is through a confusion matrix(CM), which can be discussed using the following figure.



	Actual Positive Class	Actual Negative Class
Predicted Positive Class	True positive (*tp*)	False negative (*fn*)
Predicted Negative Class	False positive (*fp*)	True negative (*tn*)

Figure 3.5: Confusion Matrix for Binary Classification

Figure 3.5 shows the confusion matrix table in which the columns represent the actual class labels and rows depict the predicted class labels. If we look at the cells, the classes that are predicted correctly are shown diagonally labeled as TP and TN, and the cells present on the opposite side to it denote wrongly predicted classes labeled as FP and FN. TP is an abbreviation for true positive means that the classified label is true and is positive according to the scenario, for e.g, if the patient has cancer and is classified as TP it means it is true because he does carry cancer disease, TN (true negative) means the classified label is true for the same case, i.e., a person doesn't have cancer and also he is classified as cancer free. In case of negative, the prediction made is also false, suppose, if the patient is not suffering from cancer but gets diagnosed with it then it is FP (false positive) whereas, on the other side, FN (false negative) means a person suffering from a disease doesn't get diagnosed with it respectively.

3.4.2 Accuracy

Accuracy depicts the proportion of true predictions to the sum of instances evaluated including true and false predictions. It is the basic and one of the most popular performance metrics which gives the researchers the general idea of the percentage of correctness which refers to all rightly classified instances by a classifier when tested against unseen data, but it has some drawbacks, such as, it does not scrutinize class imbalance. For instance, if a dataset contains 10 instances belonging to class 1 and 1000 instances belonging to class 2 the model could possibly predict every entry to be in class 2 and produce very high accuracy results [21]. Despite its limitations towards minority class instances it is still used widely by researchers to choose optimal solutions for either binary or multi-class problems. It is considered easy to compute with little complexity [14] [1] [18] [27]. The formula for accuracy can be given as:

$$Accuracy(acc) = \frac{(TP+TN)}{(TP+FP+TN+FN)} \tag{3.17}$$

3.4.3 Precision

The measure of correctly predicted from the total predicted in a positive class defines precision [39]. The formula for precision can be given as:

$$Precision(p) = \frac{(TP)}{(TP+FP)} \tag{3.18}$$

3.4.4 Recall or Sensitivity

The true positive rate, also called recall, measures the proportion of positive instances that are correctly identified [39]. The formula for recall/sensitivity can be given as:

$$Recall = \frac{(TP)}{(TN+FN)} \tag{3.19}$$

3.4.5 Specificity

The measure of wrongly predicted cases from the total predicted in a negative class defines a specificity or in other words Out of total real negative cases, how many were classified as negative? The formula for recall/sensitivity can be given as [39]:

$$Specificity = \frac{(TN)}{(TN+FP)} \tag{3.20}$$

$$1 - Specificity = FPR \quad \text{(False positive rate)} \tag{3.21}$$

where FPR is:

$$FPR = \frac{(FP)}{(TN+FP)} \tag{3.22}$$

3.4.6 ROC-AUC

The Receiver Operating Characteristic (ROC) curve is a graphical representation that shows the performance of a binary classifier as the classification threshold is varied. It is generated by the process that involves graphing the TPR and FPR at various threshold values The points corresponding to different thresholds are connected to create the ROC curve, with TPR on the y-axis and FPR on the x-axis. The area under the ROC curve (AUC) is a measure of the overall performance of the classifier, where a perfect classification has a value of 1, while a value of 0.5 indicates random guessing [7].

Some of the pros of ROC-AUC curve is that it provides a comprehensive view of the performance of the classifier across all possible threshold values, which can be useful for comparing classifiers and selecting the optimal operating point. Additionally, it is

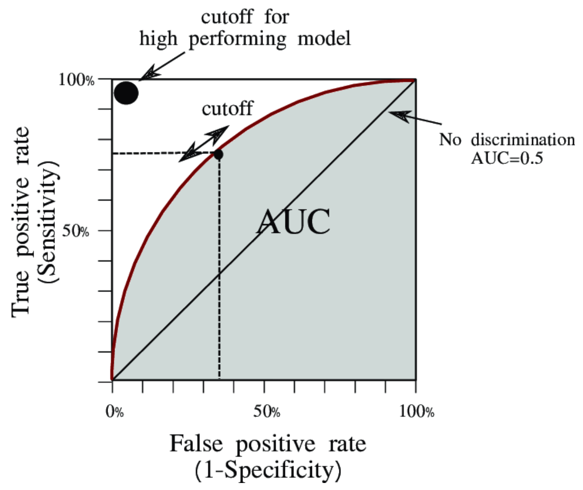


Figure 3.6: ROC Curve Illustration [7].

relatively insensitive to class imbalance, which is a common problem in binary classification [7].

Whereas some of the cons of using ROC-AUC curve is that it does not provide information about the specific operating point of the classifier, which may be important in some applications. Secondly, it is less interpretable than other metrics such as precision and recall, which may be more informative for specific applications. Also, the ROC AUC curve assumes that the cost of false positive and false negative errors is equal, which may not be true in all applications.

3.4.7 F-Measure (FM)

With accuracy, the issue is that if the dataset is highly unbalanced, the model will finish up correctly learning how to forecast the class with the majority of instances but will not learn how to identify the minority class. However, the accuracy value of the model will still remain quite high. To overcome the issues that accuracy measures have, researchers found new ways to calculate the mean values to give equal significance to both positive and negative data instances. Some examples of alternatives are geometric, harmonic, and arithmetic mean. The F-score measure is a harmonic mean of Precision and Recall values, which means FP and FN are equally significant. Compared to "Accuracy" it is reported as a very useful metric, with better-performing results in rectifying the classifier for binary class issues. F-measure can be given mathematically as:

$$F1Score = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \tag{3.23}$$

3.4.8 G-mean1

It is the geometric mean of sensitivity and precision [17]. Mathematically, represented as:

$$GSP = \sqrt{precison * sensitivity} \tag{3.24}$$

3.4.9 G-mean2

It is the geometric mean of sensitivity and specificity [17]. Mathematically, represented as:

$$GSS = \sqrt{spec * sens} \tag{3.25}$$

In order to optimize classifiers for binary classification issues, the FM and GM were also reported as good discriminators and performed better than accuracy [43]. Due to a single evaluation objective, the remaining indicators are insufficient to distinguish between and choose the best solution (either positive or negative class). As far as we know, no prior work has used the FM and GM to discriminate between and choose the optimum solution for multiclass classification issues.

3.5 Comparison of Classification Evaluation Techniques

Comparing the performance and complexities of these metrics is not straightforward since they have different strengths and weaknesses depending on the problem and the application. Generally, accuracy is a simple and intuitive metric but can be misleading in imbalanced datasets. Precision and recall are more informative for imbalanced datasets but can be difficult to optimize simultaneously. The F-measure combines precision and recall and is useful when there is no clear trade-off between them. Specificity is a useful metric for problems where false positives are particularly costly, such as medical diagnosis. The ROC AUC metric is useful for comparing classifiers and selecting the optimal operating point, but it does not provide information about the specific operating point of the classifier [44] [44].

Table 3.2 is comparing different classification evaluation techniques in terms of their computational complexity, interpretability, robustness, and other important factors. N represents the number of samples.

In terms of computational complexity, most classification evaluation techniques have linear time complexity with respect to the number of samples N. The ROC AUC evaluation technique has a higher computational complexity of O(N log N) due to the need

Metric	**Complexity**	**Interpretability**	**Robustness**
Confusion Matrix	$O(N)$	High	Sensitive to class imbalance
Accuracy	$O(dN)$	High	Sensitive to class imbalance
Precision	$O(dN)$	High	Sensitive to class imbalance
Recall	$O(dN)$	High	Sensitive to class imbalance
Specificity	$O(dN)$	High	Sensitive to class imbalance
F1-Score	$O(dN)$	High	Sensitive to class imbalance
Confusion Matrix	$O(dN)$	High	Sensitive to class imbalance
G-mean	$O(dN)$	High	Less sensitive to class imbalance
ROC-AUC	$O(NlogN)$	Low	Robust to class imbalance

Table 3.2: Comparision of Evaluation Metric

for sorting and calculating the area under the curve [44] [45].

In terms of interpretability, some evaluation techniques such as the confusion matrix, accuracy, precision, and recall are relatively easy to interpret because they directly report the number of correct and incorrect predictions for each class. Other techniques such as the G-Mean and ROC AUC are less interpretable because they combine different metrics into a single score.

In terms of robustness, some evaluation techniques such as the confusion matrix, accuracy, and F1-score are sensitive to class imbalance because they treat each class equally. Other techniques such as the G-Mean and ROC AUC are less sensitive to class imbalance because they balance the metrics based on the prevalence of each class.

Other important factors to consider include the dependence of some metrics on the positive or negative class (such as precision and specificity), and the ease of calculation for some techniques (such as the confusion matrix and accuracy).

It's worth noting that the choice of evaluation technique and the importance of different factors depends on the specific problem and the trade-offs between different metrics [44] [45].

4 Data Preprocessing and Analysis

Data preprocessing is the preliminary step in data analysis, which involves the cleaning and transformation of raw data into a more suitable format for further processing. This may include activities such as removing missing values, dealing with outliers, standardizing variables, and reducing noise in the data, among others. The objective of data preprocessing is to improve the quality of data and facilitate more accurate and efficient analysis in various data science tasks, such as data mining, machine learning, and statistical analysis.

For preparing data, a variety of tools and techniques are utilized. In this thesis, we have utilized the techniques of missing value imputation, feature selection, feature removal, outlier removal, correlation, and data sampling.

4.1 Handling Missing Data

Missing data refers to the absence of values in a dataset. Missing values in datasets are a big problem when acquiring real-time data. It can affect forecasting tasks in classification and regression problems if not handled timely and lead to incorrect results. The data preparation step in ML to deal with missing values is the preprocessing step done right before modeling classification algorithms. It helps in dealing with the problems related to data quality and usually, a sizable amount of time is required. Defining errors and handling missing values are examples of cleaning processes. Estimation methods should be employed to substitute reasonable values for missing data points in datasets [45]. Several straightforward and widely recognized techniques exist to address data missingness, commonly referred to as traditional methods. On the other hand, there are more intricate methods referred to as imputation methods that are slightly more complex than traditional approaches. This doesn't let the dataset compromise on its quality by imputing values through a certain mechanism where the data gets missed, but to fill the gaps with meaningful values we certainly need to work on understanding the type of data that is inappropriate for our data and not classified as typical data. Low-quality data are those that do not conform to the underlying expectations based on the degree of fit [43]. A few reasons for poor data quality in ML include Noise, Outliers, and Relevance Class imbalance.

Noise: The noise in the data is the meaningless information that is received in such a way that it becomes unreadable for a program or invalid. It may take unnecessary storage space and lead to bad analysis results so dealing with it prior to modeling a classifier is always a good idea [46].

Outliers: Values that are out of range for other objects in the data collection due to being either too large or small to handle, thus these values may disturb the model behavior and are unwell for data characteristics are outliers. Outlier detection and handling are important in preprocessing steps for better prediction results [46].

Relevance: The Correlation of the data with the purpose for which it is gathered must be there to fit the needs of the classification system. The features that seem irrelevant to the task must be dealt by either totally eliminating them or by transforming them into meaningful features as discussed later in this chapter. There are also some feature selection methods in MI that help decide which features are important for DM.

Class Imbalance: If the data from one class is relatively higher in proportion than the other class that is known as the majority class. The model will eventually get trained on the majority class so well that unseen data is classified as a majority class label by default. To deal with class imbalance, overfitting, and underfitting are the possible solutions.

The way missing data is handled in machine learning models depends on the underlying mechanism causing the missingness, which can be categorized into three types:

- **Missing Completely at Random (MCAR):** This phenomenon occurs when the likelihood of a value being absent is uniform across all observations and has no association with the values of other variables in the dataset. In other words, the missingness is completely random. In the context of predicting lead scores, an example of MCAR could be when a lead's phone number is missing due to a technical error.
- **Missing at Random (MAR):** This phenomenon arises when the likelihood of a value being absent is associated with other observable variables in the dataset. In other words, the missingness is not completely random but can be predicted from other variables in the dataset. For example, if females are less likely to fill out a particular question on a lead form, the missingness is related to the gender variable. In this case, the missingness is not random, but it can be predicted from other variables in the dataset.
- **Missing Not at Random (MNAR):** This phenomenon occurs when the likelihood of a value being absent is dependent on the value itself. In other words, the missingness is related to the value of the variable that is missing. In the context of lead scoring, an example of MNAR could be when a lead chooses not to provide their income information because they are uncomfortable sharing it.

Handling missing data in machine learning models can be challenging, and different techniques can be used depending on the type of missingness present in the dataset. Techniques such as mean imputation, median imputation, and model-based imputation can be used to handle missing values. It is important to carefully consider the missing data mechanism and choose an appropriate method for handling missing data in the context of predicting lead scores.

4.1.1 Traditional Method

Mean or Median Imputation: The technique involves substituting missing values with the mean or median value of the available data. This approach is frequently utilized when the missingness is MCAR.

Forward or Backward Filling: In this method, missing values are replaced by the last observed value or the next observed value in the series. This method is commonly used when the missingness is MAR.

Listwise or Complete Case Analysis: In this method, observations with missing values are removed from the dataset. This method is commonly used when the amount of missingness is very small and the model can still be trained with the available data.

4.1.2 Algorithm-based Method

K-Nearest Neighbor (KNN) Imputation: This method replaces missing values with the mean or median value of the k-nearest neighbors. It is often employed when the missingness is MAR.

Expectation-Maximization (EM) Algorithm: In this method, the missing values are imputed using an iterative approach that estimates the missing values based on the available data. This method is commonly used when the missingness is MCAR or MAR. There are many factors that can cause missingness in a dataset, including data collection errors, data entry errors, data processing errors, and non-response from study participants. Other factors that can contribute to missingness include data that is deliberately withheld, such as sensitive personal information.

In summary, handling missing data is an important step in machine learning, and there are various methods available to handle missingness. The choice of method depends on the type of missingness in the dataset, the amount of missingness, and the specific requirements of the analysis or modeling task. Regenerate response

4.2 Data Splitting and Resampling

There are a certain number of instances or data points available while working with a data set. It is significant to remember that while assessing a model's performance, one should consider cases that were not used in the model's development. It is done, in order to get a fair performance assessment that is unbiased.

Before training the model the data need to be split into training and testing or validation datasets. Training datasets contain a random number of examples from the data present but the same dataset that is used to construct a model cannot be used for evaluating the model performance to avoid overfitting.

When a model learns the behavioral data and the inaccurate values present in the training dataset to an extent that it affects prediction negatively in many cases, then overfitting occurs. As the model was created specifically to forecast the examples that were not used in the training phase, it gets worse at predicting new and unforeseen situations.

Several resampling strategies are used to address the overfitting and underfitting problems. When a supervised learning model is being trained, it assists in overcoming data biases and offers a useful performance evaluation. They are usually applied to accurately estimate the performance of a model. We have used random undersampling and oversampling with SMOTE in our thesis.

4.2.1 Random Undersampling

Random Undersampling is a method for handling imbalanced datasets in machine learning, where the majority class is heavily overrepresented and the minority class is underrepresented. Random Undersampling involves removing instances from the majority class randomly until the class distribution is balanced. This method can lead to the loss of important information and may not be effective if the minority class is too small [47].

Given a dataset d with n samples, where the majority class is represented by M samples and the minority class is represented by N samples, we want to balance the class distribution [48].

- Randomly select M - N samples from the majority class to remove.
- Train a machine learning model on the balanced dataset.

Figure 4.1 illustrates random undersampling where the peach color represents majority class M and the blue color is minority class N, by randomly removing the values from the majority class the dataset is balanced.

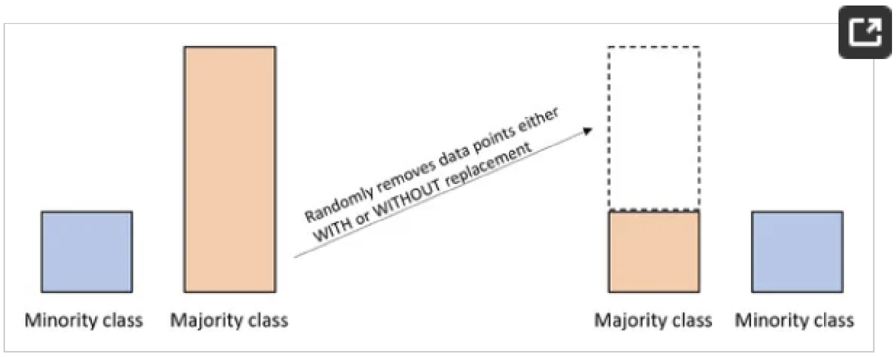

Randomly removes data points either WITH or WITHOUT replacement
Minority class
Majority class
Majority class
Minority class

Figure 4.1: Random Undersampling approach [8].

4.2.2 Random Oversampling with SMOTE

The basic idea is to randomly duplicate samples from the data-set until the class distribution is balanced. This process is termed as oversampling in ML. The figure 4.2 illustrates that how minority class N duplicated data randomly to increase the data samples.

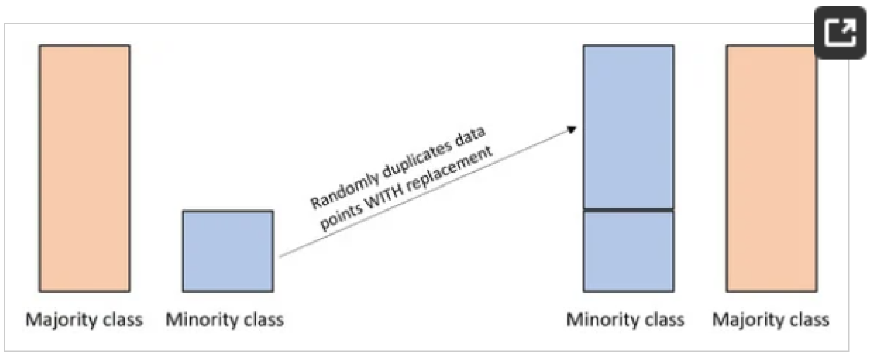


Figure 4.2: Random Oversampling approach [8].

One alternative to random oversampling is Synthetic Minority Over-sampling Technique (SMOTE), where the minority class is underrepresented. SMOTE creates synthetic samples for the minority class by interpolating between existing samples, which can improve the performance of the machine learning model in terms of adaptability to unseen data. [9].

SMOTE is slightly more complex, as it involves generating synthetic examples by interpolating between existing examples. Given a dataset X with n samples, where the majority class is represented by M samples and the minority class is represented by N samples, we want to balance the class distribution. The work for generating synthetic examples using SMOTE is as follows:

- Select a minority class with example N.

- Select k nearest neighbors of N from the minority class
- Randomly select one of the k neighbors, call it $N^{'}$
- Generate a new example N_{new} by linearly interpolating between N and $N^{'}$.

Mathematically. it is represented as:

$$N_{new} = N + rand(0,1) * (N^{'} - N) \tag{4.1}$$

Here, rand(0,1) is a random number generator that produces a value between 0 and 1, and N_{new} is the newly generated example. This process of generating new example involves the repetition of interpolation process until the desired number of synthetic examples have been generated.

Figure 4.2 shows an example of random oversampling and SMOTE applied to a two-dimensional dataset with two classes.

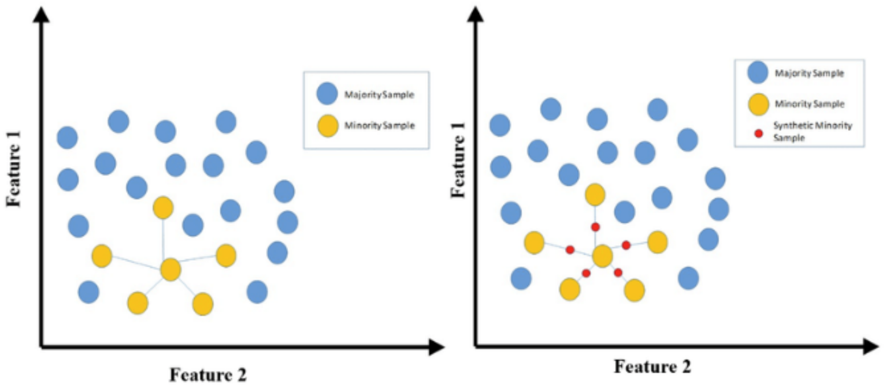


Figure 4.3: SMOTE Oversampling approach [9].

The blue circles represent the majority class, and the yellow ones represent the minority class. The left-hand plot is without oversampling and the right-hand side is a plot when the synthetic examples in red are generated to balance the minority class.

For example In a study by He and Garcia (2009) in [49], Random Oversampling with SMOTE was used to enhance the performance of a machine learning model for credit card fraud detection. The study found that Random Oversampling with SMOTE was an effective method for handling class imbalance and improving the performance of the machine learning model.

4.3 Feature Selection

As transformation is applied for better decisions, similarly feature selection methods(FSM) helps to make data more understandable and effective. It works to reduce the dimensionality problem in ML task as it aims to remove the data that is not relevant and is redundant. It can effectively lessen the computing time and promote model accuracy. [18] [20]. The steps of assessment that are typically used for choosing characteristics are covered in this section. The most popular ML feature selection techniques are as follows:

4.3.1 Correlation-based Feature Selection (CFS)

The technique called correlation-based feature selection is a type of method that screens features based on how much they are associated with the target variable. Pearson's correlation coefficient is a statistical measure that defines a linear relationship between two variables, x and y given as;

$$r = \frac{\sum(x_i - \mu x)(y_i - \mu y)}{\sqrt{(\sum(x_i - \mu x)^2)} * \sqrt{(\sum(y_i - \mu y)^2)}} \tag{4.2}$$

where x_i and y_i are the values of variables x and y, μx and μy are their means, and $\sum$ represents the sum over all data points. The value of correlation coefficient lie in between -1 and 1. Here, -1 depicts a perfect negative correlation, 0 show no correlation, and 1 vlaue represent a highly correlated variable. A simple example of CFS is discussed below:

Example: Suppose we have a dataset with five features (x1, x2, x3, x4, x5) and a target variable (y). We calculate the correlation coefficient between each feature and the target variable and select the two features (x1 and x3) with the highest correlation.

It is used widely in image recognition, text classification, and gene expression analysis. It has been shown to be effective in reducing the dimensionality of datasets and improving the performance of ML models [50] [51].

4.3.2 Recursive Feature Elimination (RFE)

Recursive Feature Elimination (RFE) is a popular feature selection method that selects the most important features by recursively removing the least important features based on a specified model. RFE can be used with any supervised learning algorithm that provides a feature importance score or coefficient for each feature, such as linear regression and decision trees.

The RFE algorithm starts with all the features and trains a model using the specified learning algorithm. It then ranks the features based on their importance scores and eliminates the least important feature, until a desired set of features results. A simple example of RFE is given as:

Example: Suppose we have a dataset with four features (x1, x2, x3, x4) and a target variable (y). First, we train a model using all the features and obtain feature importance scores as: F1=0.6, F2=0.4, F3=0.2, F4=0.1, and eliminate the least important feature x4 with the least score. Then again train a model using the remaining features x1, x2, x3 and obtain feature importance scores as: F1=0.6, F2=0.4, F3=0.2. This time x3 gets eliminated. Further train a model using the remaining features x1, x2 and obtain feature importance scores. The algorithm stops since the desired number of features (in this case, two) is reached.

RFE has proven to be effective in dimensionality reduction of high-dimensional data. It has been used in various domains, including bioinformatics, image processing, and finance. It aims to simplify the training dataset, decrease learning time and enhance accuracy [52] [53] [54].

4.3.3 Variance Inflation Factor (VIF)

Variance Inflation Factor (VIF) is a technique used to detect and quantify multicollinearity among a set of features in a linear regression model. Multicollinearity refers to a statistical phenomenon where two or more variables which are independent are strongly and linearly correlated with each other generating problems in the ML model, such as inflated standard errors and unstable coefficients.

The mathematical formula for VIF can be written as follows: Given P as a set of features in linear regression model which are independent variables represented as $x_1, x_2, ..., x_p$. The VIF for variable xi is calculated as:

$$VIF(xi) = \frac{1}{1 - R2(x_i)} \tag{4.3}$$

where $R2(x_i)$ is the coefficient of determination of a regression model where x_i is the dependent variable and all the other independent variables are used as predictors. A VIF value greater than 1 represents highly correlated variables. [51].

Example: In a study by Kumar et al. [55] VIF was used to detect multicollinearity for predicting air quality. The study found that VIF was an effective method for detecting multicollinearity and removing highly correlated independent variables from the model.

5 Experiment and Results

This chapter is divided into three parts, in section 5.1 we describe the experiment and provide Exploratory Data Analysis (EDA) to examine and summarize characteristics of a dataset. However, due to company data policy the visualization presented is only on small part of data. In section 5.2 we present the results of each classifier with and without the application of sampling techniques on the data. In section 5.3, we describe the findings from the results on the provided data.

5.1 Experiment

5.1.1 Data Description

Data for this thesis was provided by a company that offers DataVault and Salesforce solutions. The company stores its Sales and Marketing data in Salesforce. Table 5.1 shows the type of data used in this thesis.

Name	**Type**	**Description**
Id	Alphanumeric	Unique identifier
Company	Categorical	Company of a particular lead.
Country	Categorical	Country of a particular lead.
City	Categorical	City of a particular lead.
Email	Categorical	Email of a particular lead.
Language	Categorical	Language of a particular lead.
Lead Source	Categorical	Source of a particular lead, e,g, newspaper, website, etc
Lead Priority	Categorical	How much priority is given to particular.
Pitch Level	Categorical	Defines that if a lead is allowed for communication.
Do not Call	Binary	True if a lead is agreed for future calls else false.
Do not Email	Binary	True if a lead is agree for future emails else false.
Privacy	Binary	True if lead asked for data privacy else false.
Status	Categorical	Defines status of a lead e.g. qualified, unqualified, etc.
Email Bounced	Binary	True if emails sent are bounced else false
Is Prioritized	Binary	True if the lead has priority else false
Lead Account	Categorical	Account or Department for a lead.
Qualified Date	Date	Date of qualification
Number of Employees	Integer	Total employees in lead's company.
Industry	Categorical	Industry for a lead company.
Annual Revenue	Float	Total revenue of a lead.
Is Converted	Binary	True if the lead is converted else false.

Table 5.1: Characterstics of the data

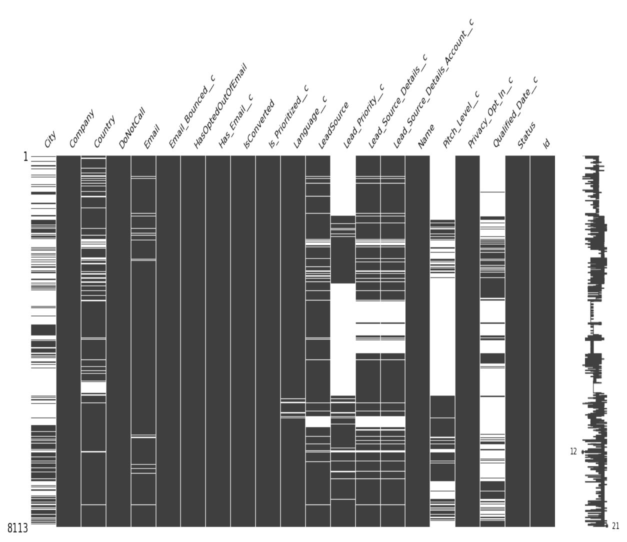
The data was collected from the company's Internal Salesforce Org. The data was collected by integrating the data mining tool called Rapid Miner with Salesforce. The data is taken from the year 2018 to 2019. The data consists of over 50,000 observations (rows). Every observation contains 34 features (columns). The data is collected from the lead, contact, and activity object (table). The class label, isConverted, is a value of 0 or 1. When a client is interested and wants to know more about the product then the Converted is set to 1. When the client is not interested in the product then the Converted is set to 0. The data have almost a 10.08% Converted rate. We split the 70% data into training sets and 30% into the testing set.

5.1.2 Data Preprocessing

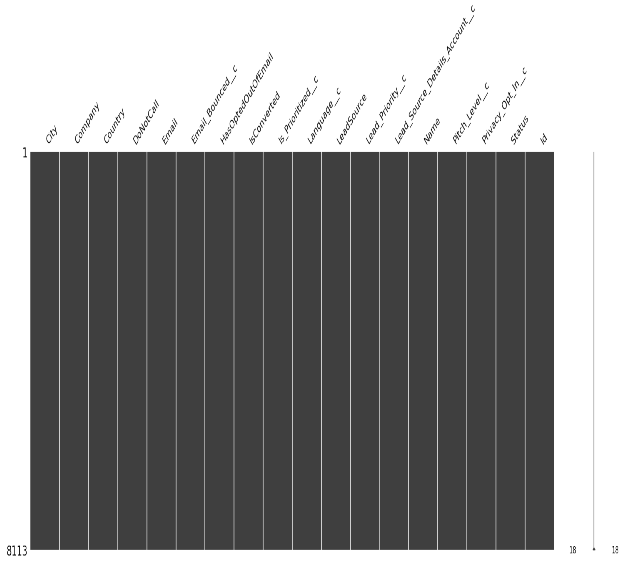
The following are some common data preprocessing steps that is applied to prepare data for lead scoring experiments.

Missing Value Imputation

We take the missing value percentage of each column(feature). We drop the columns having more than 70% of missing values. We handle missing values in remaining some by replacing them with the value that occurs the most. For example, We take the 'country' column, around 20% of the data available is Germany so we impute Germany in the missing values of the country column. We use python code and libraries to remove missing values from the data. Figure 5.1 and 5.2 illustrate the missingnness in the data before and after imputation. The white area represent the missinngness in each feature whereas, the black area represent that the data is not missing.



(a) Data before removing missing values.



(b) Data after removing missing values.

Figure 5.1: Comparision of data with and without missingness.

Feature Transformation

To improve the accuracy of classifiers we transform few features. The data in the country column looks imbalance and can impact the accuracy of the classifier. So we decided to convert the country into their respective Continent. So we make a group of the countries that are in the same continent and represent the country with their respective continent as shown in figure 5.2.

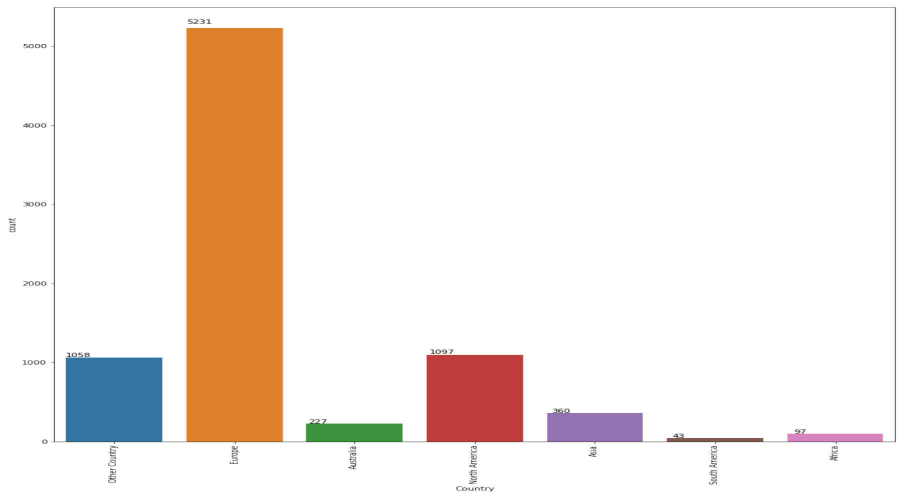
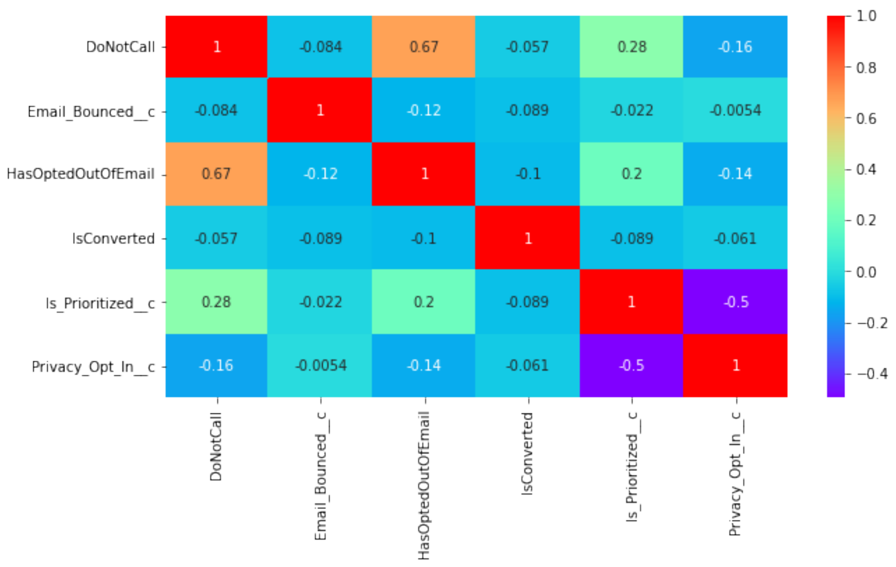


Figure 5.2: Country Transform into continent.

Correlation

To achieve high accuracy we eliminate the highly correlated data. We use the python seaborn library and plot the correlation matrix. Based on our plot in figure 5.3 we figure out that "Do not call" "Has Opted out of email" are highly correlated with correlation .67.



	DoNotCall	Email_Bounced__c	HasOptedOutOfEmail	IsConverted	Is_Prioritized__c	Privacy_Opt_In__c
DoNotCall	1	-0.084	0.67	-0.057	0.28	-0.16
Email_Bounced__c	-0.084	1	-0.12	-0.089	-0.022	-0.0054
HasOptedOutOfEmail	0.67	-0.12	1	-0.1	0.2	-0.14
IsConverted	-0.057	-0.089	-0.1	1	-0.089	-0.061
Is_Prioritized__c	0.28	-0.022	0.2	-0.089	1	-0.5
Privacy_Opt_In__c	-0.16	-0.0054	-0.14	-0.061	-0.5	1

Figure 5.3: Correlation Matrix

Feature Selection

We use the Recursive Feature Elimination (RFE) and Variance Inflation Factor(VIF) technique which progressively removes the least important feature(s) from a model until the desired number of features is achieved. We implement these techniques by using the python sklearn library.

5.1.3 Exploratory Data Analysis

Exploratory data analysis (EDA) is a crucial first step in any data analysis project. It involves examining and visualizing the data to gain insights into its underlying structure, patterns, and relationships. In this section, we will explore a few features through visual representation.

Figure 5.4, represent the ratio of lead conversion on the lead that has chosen do not call and has opted out of email. We see that the lead that allow communication from sales representative on call and email has more chances to convert than the lead who do not allow communication on call and email.

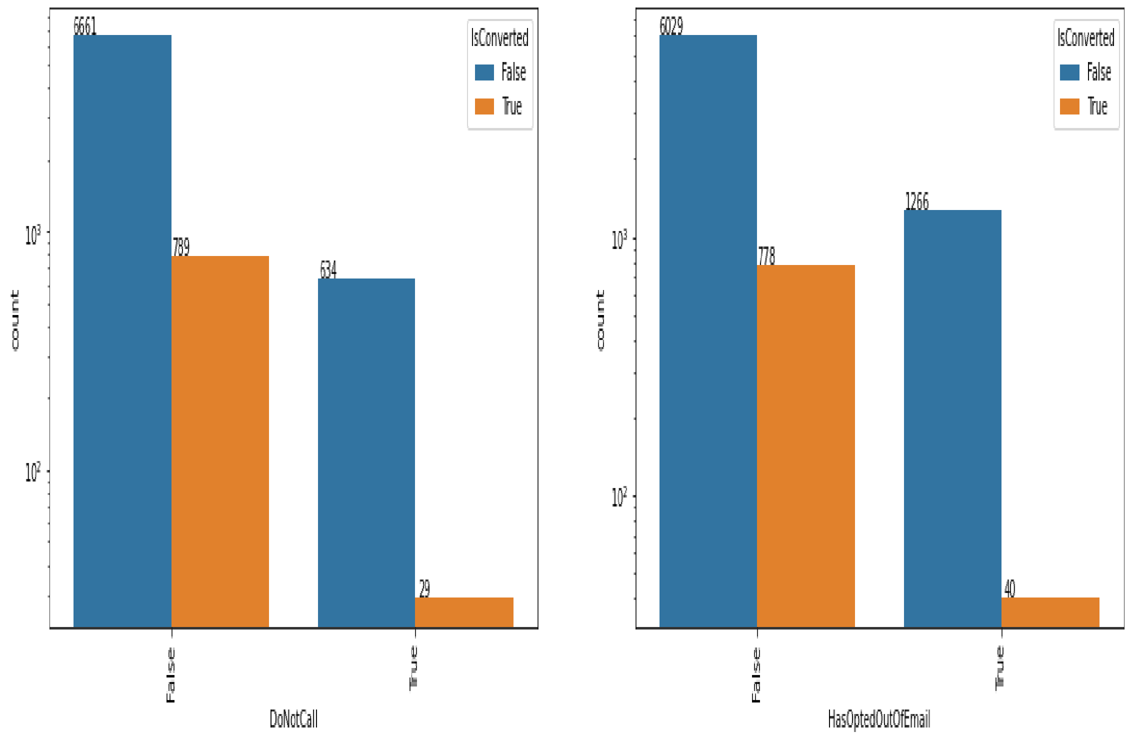


Figure 5.4: Lead Conversion ratio for column "Do Not Call" and "Has opted out of Email". .

Figure 5.5, represent the ratio of lead conversion on the lead with their communication language. We see that the lead that communicate in German and English have more conversion ratio compare to other language. It is also seen that the leads that speak Russian, Greek, Portuguese Romanian, Hungarian and Estonian languages do not convert.

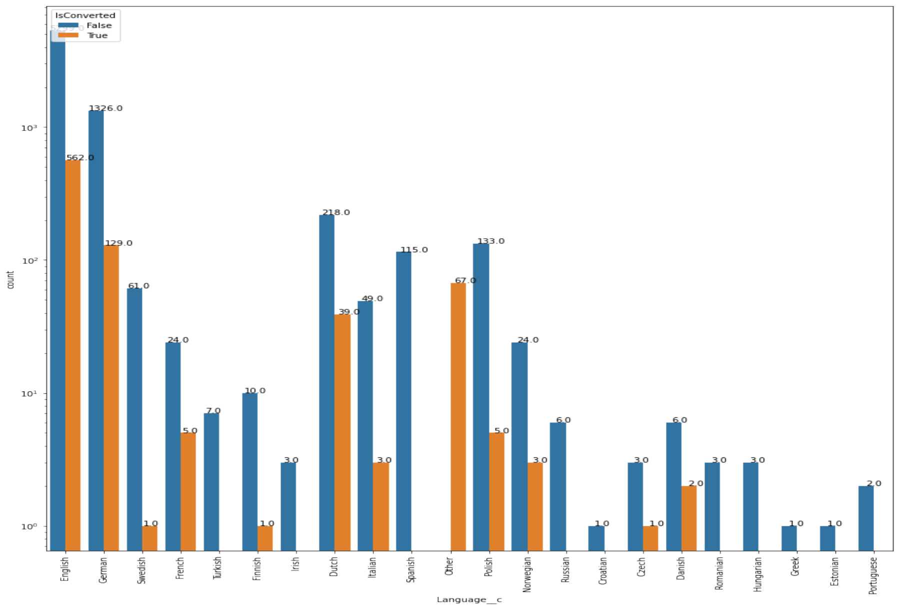


Figure 5.5: Lead Conversion ratio as per Language.

Figure 5.6, represent the ratio of lead conversion with respect to level of pitch. We see that the lead that ask the sales reps to Asked for Permission has greater chance of conversion. On the other hand lead that allowed communication with any medium has also good chance of converting.

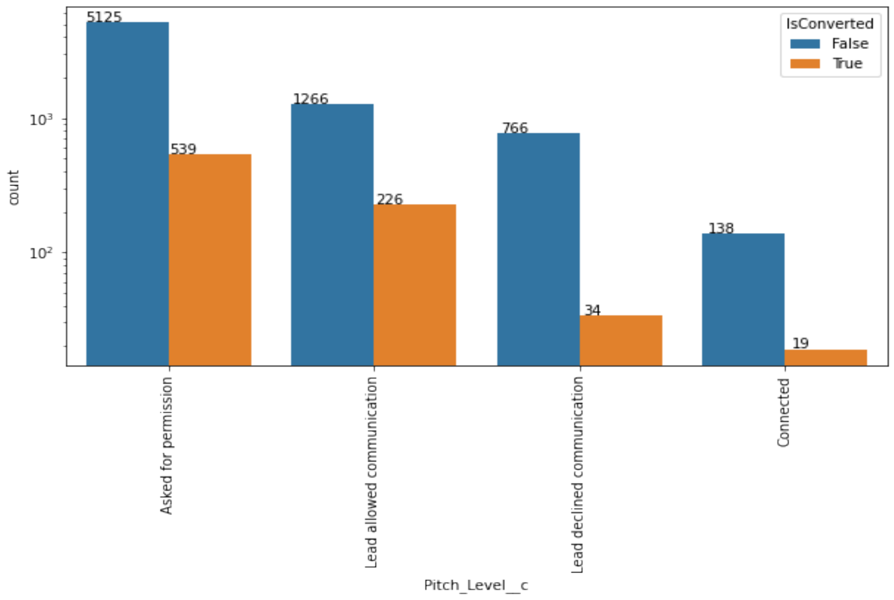


Figure 5.6: Lead Conversion ratio as per Pitch Level.

5.2 Results and Analysis

5.2.1 Results without Sampling

Performance Evaluation of Classifiers without Sampling				
Evaluation Metric	**Linear Regression**	**Decision Tree**	**Neural Network**	**Random Forest**
Accuracy	0.92	0.90	0.94	0.89
ROC-AUC	0.86	0.67	0.94	0.85
Precision	0.88	0.91	0.93	1
Sensitivity	0.34	0.13	0.52	0.08
Specificity	0.99	0.99	0.99	1
F-Measure	0.49	0.23	0.66	0.15
G-mean 1	0.54	0.34	0.69	0.28
G-mean 2	0.58	0.36	0.72	0.28

Table 5.2: Evaluation metric results for Classifier without Sampling.

Performance Evaluation of Classifiers with Grid Search		
Evaluation Metric	**Decision Tree with Grid Search**	**Random Forest with Grid Search**
Accuracy	0.94	0.92
ROC-AUC	0.94	0.91
Precision	0.90	0.97
Sensitivity	0.49	0.30
Specificity	0.99	0.99
F-Measure	0.63	0.46
G-mean 1	0.66	0.54
G-mean 2	0.70	0.55

Table 5.3: Evaluation metric results for Classifier with model tuning technique Grid Search on without Sampling data.

Table 5.2 present the results of the evaluation metrics for four different classifiers, namely Linear Regression, Decision Tree, Neural Network, and Random Forest, on the dataset with under-sampling. The evaluation metrics used in the analysis include Accuracy, ROC-AUC, Precision, Sensitivity, Specificity, F-Measure, G-mean 1, and G-mean 2.

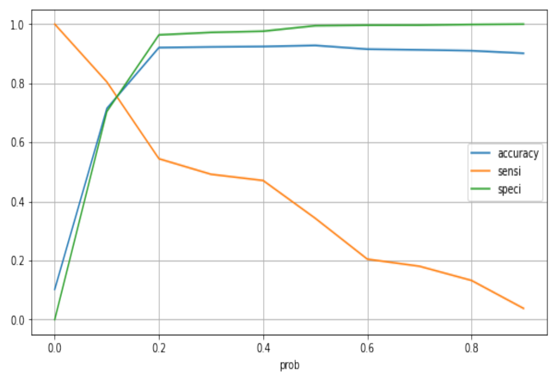
After examining table 5.2, evaluation metrics on data without sampling, the neural network classifier appears to perform the best on most metrics, with the highest accuracy, ROC-AUC, precision, sensitivity, F-measure, and G-mean 1 and G-mean 2 scores. The linear regression classifier performs well, with the second-highest accuracy and precision scores and the second-highest G-mean 1 and 2 scores, but performs relatively poorly on sensitivity and ROC-AUC. The random forest classifier has the lowest accuracy score among the four classifiers, but it has the highest specificity and precision scores. The decision tree classifier performs the worst on most metrics, with the lowest scores on the accuracy, ROC-AUC, precision, sensitivity, F-measure, and G-mean 1 and 2.

However, it is essential to note that the choice of the best classifier ultimately depends on the specific problem and the available data. While the neural network classifier appears to be the best-performing classifier in this evaluation, other classifiers may be more suitable for different data types and problems. Additionally, model tuning techniques such as Grid Search can also improve the performance of the classifiers.

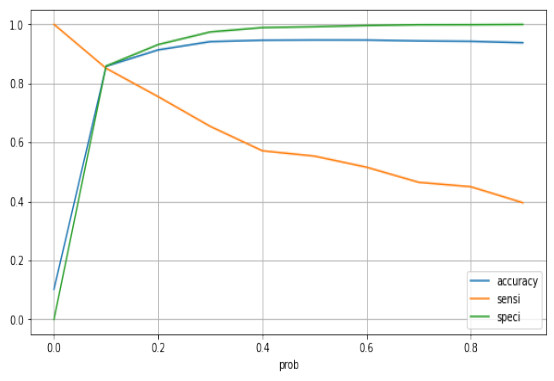
As shown in Table 5.3, after applying Grid Search to the Decision Tree and Random Forest classifiers, both classifiers show improvements in their performance metrics. In particular, the Decision Tree classifier with Grid Search achieves the highest accuracy and ROC-AUC scores among all classifiers and also shows improvements in precision, F-measure, and G-mean 1 and 2 scores. The Random Forest classifier with Grid Search also shows improvements in precision, F-measure, and G-mean 1 and 2 scores, but still has lower accuracy and ROC-AUC scores than the other classifiers.

Comparing the Decision Tree classifier with Grid Search to the other classifiers, it now performs better than the neural network classifier on accuracy and ROC-AUC metrics, but still has lower precision and sensitivity scores. On the other hand, the Random Forest classifier with Grid Search shows improved precision and F-measure scores, but still has lower accuracy and ROC-AUC scores than the neural network and linear regression classifiers.

Overall, the performance improvements after applying Grid Search demonstrate the effectiveness of this model-tuning technique in improving the performance of classifiers. However, it is still important to carefully evaluate and compare different classifiers and model-tuning techniques to select the best approach for a specific problem and dataset. Figure 5.7 shows the accuracy, sensitivity, and specificity cut-off between the two best models. Whereas Fig 5.8, shows the ROC curve for all the classifiers.

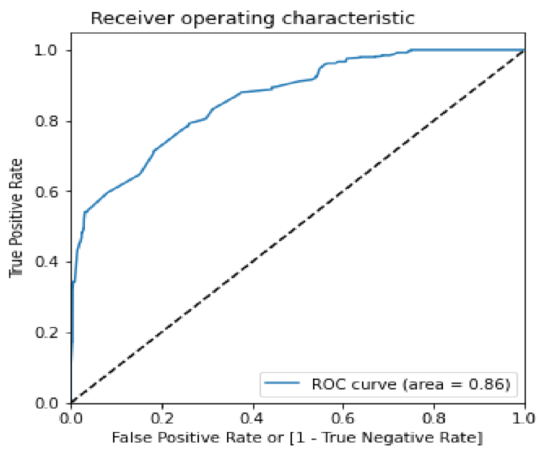


(a) Cut-off: Accuracy, sensi, speci for LR

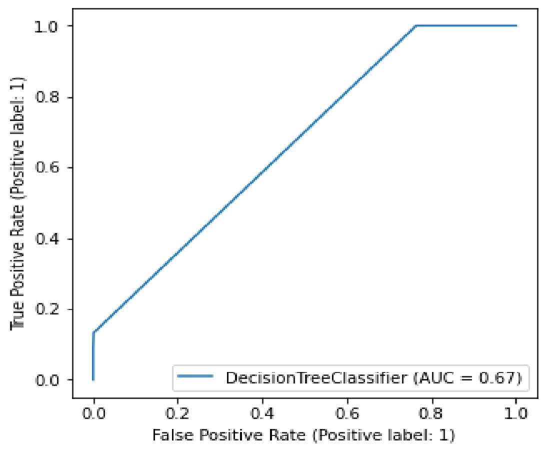


(b) Cut-off: Accuracy, sensi, speci for ANN

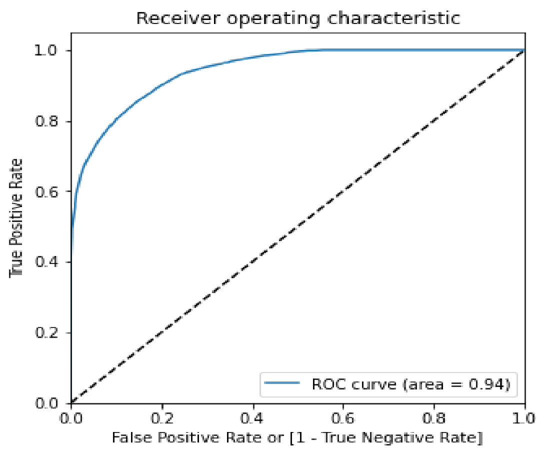
Figure 5.7: Cut-off for two best models i.e. Neural Network and Linear Regression Classifier on data without sampling



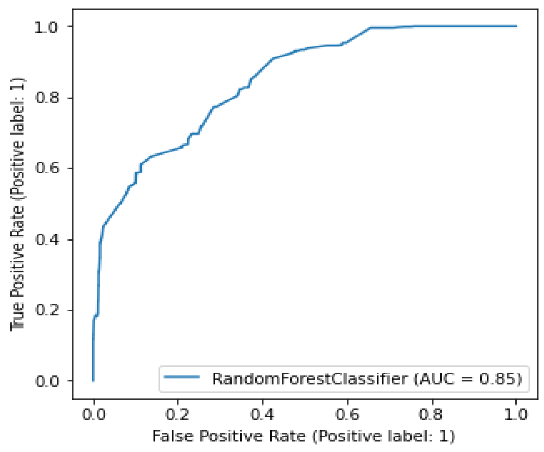
(a) ROC: Linear Regression without sampling



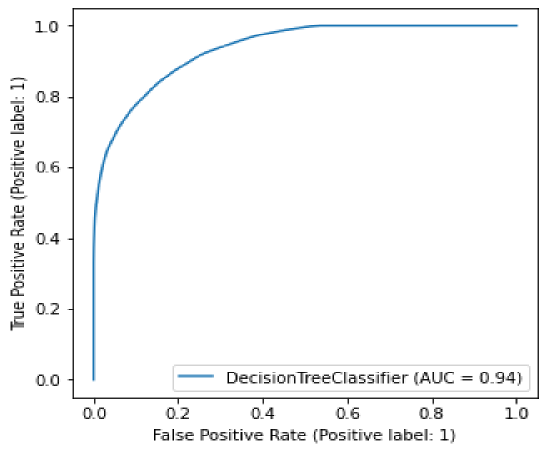
(b) ROC: Decision Tree without sampling



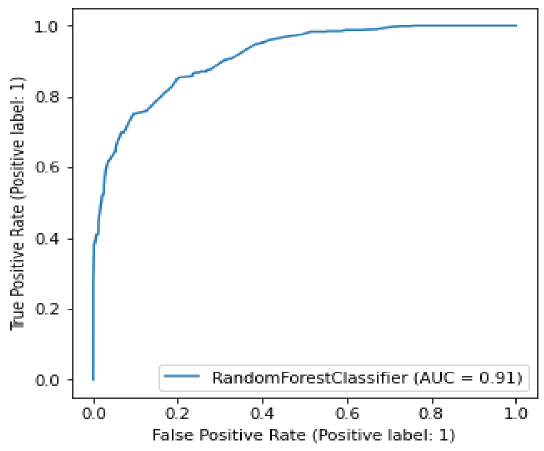
(c) ROC: Neural Network without sampling



(d) ROC: Random Forest without sampling



(e) ROC: Decision Tree with Grid Search without sampling



(f) ROC: Random Forest with Grid Search

Figure 5.8: ROC for Classifier without sampling data.

5.2.2 Results without Random Under Sampling

Performance Evaluation of Classifiers with Under Sampling				
Evaluation Metric	**Linear Regression**	**Decision Tree**	**Neural Network**	**Random Forest**
Accuracy	0.76	0.69	0.85	0.76
ROC-AUC	0.86	0.67	0.95	0.85
Precision	0.68	0.65	0.92	0.73
Sensitivity	0.34	0.79	0.77	0.84
Specificity	0.84	0.58	0.93	69
F-Measure	0.78	0.72	0.84	0.78
G-mean 1	0.78	0.72	0.84	0.78
G-mean 2	0.76	0.68	0.85	0.76

Table 5.4: Evaluation metric results for Classifier with Under Sampling data.

Performance Evaluation of Classifiers with Grid Search		
Evaluation Metric	**Decision Tree with Grid Search**	**Random Forest with Grid Search**
Accuracy	0.83	0.79
ROC-AUC	0.94	0.91
Precision	0.94	0.76
Sensitivity	0.79	0.87
Specificity	0.58	0.72
F-Measure	0.72	0.81
G-mean 1	0.72	0.81
G-mean 2	0.68	0.79

Table 5.5: Evaluation metric results for Classifier with model tuning technique Grid Search on with Under Sampling data.

The given results present the evaluation metrics for four different classifiers, namely Linear Regression, Decision Tree, Neural Network, and Random Forest, on the dataset with under-sampling. The evaluation metrics used in the analysis include Accuracy, ROC-AUC, Precision, Sensitivity, Specificity, F-Measure, G-mean 1, and G-mean 2.

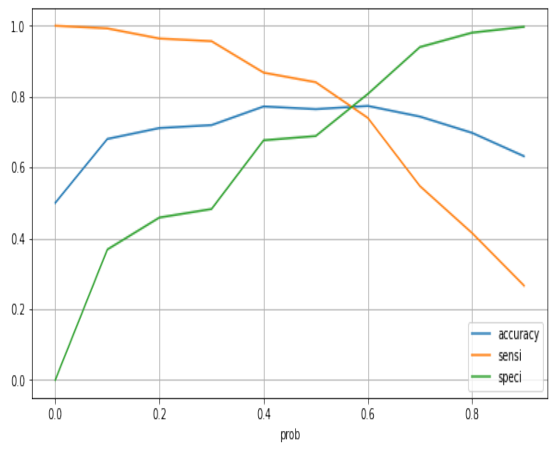
Based on the results, the Neural Network classifier performed the best with the highest accuracy of 0.85, highest ROC-AUC of 0.95, highest precision of 0.92, and highest sensitivity of 0.77. The Decision Tree classifier had the highest sensitivity of 0.79, but it had a lower accuracy of 0.69 and ROC-AUC of 0.67 compared to other classifiers.

The Random Forest and Linear Regression classifiers had the same accuracy of 0.76, while the Random Forest had a higher ROC-AUC of 0.85 compared to Linear Regression's ROC-AUC of 0.86. The Precision, F-measure, and G-mean values of the Random Forest classifier were higher than those of Linear Regression.

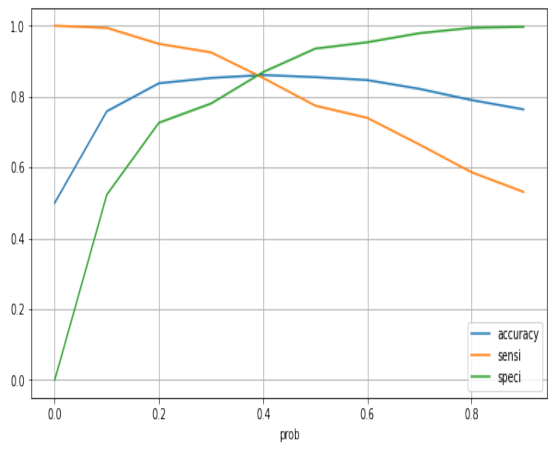
In Table 5.5, the evaluation metrics for Decision Tree and Random Forest classifiers with Grid Search model tuning technique on under-sampled data are presented. The results show that the Decision Tree classifier with Grid Search had a higher accuracy of 0.83 and ROC-AUC of 0.94, while the Random Forest classifier with Grid Search had a higher Precision of 0.76 and Sensitivity of 0.87. The G-mean 1 and G-mean 2 values were also higher for both classifiers with Grid Search than those without Grid Search.

In summary, the Neural Network classifier performed the best in the initial analysis, while the Decision Tree and Random Forest classifiers with Grid Search had improved performance in the second analysis. It is important to note that the choice of evaluation metric depends on the problem's specific requirements, and different metrics can provide different insights into the classifier's performance.

Figure 5.9 shows the accuracy, sensitivity, and specificity cut-off between the two best models. Whereas Fig 5.10, shows the ROC curve for all the classifiers on under-sampling data.

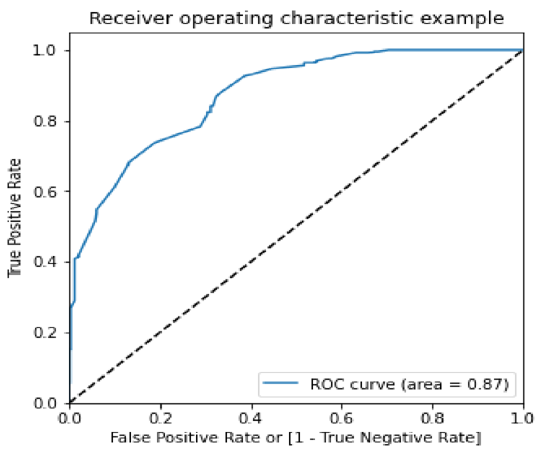


(a) Cut-off: Accuracy, sensi, speci for LR

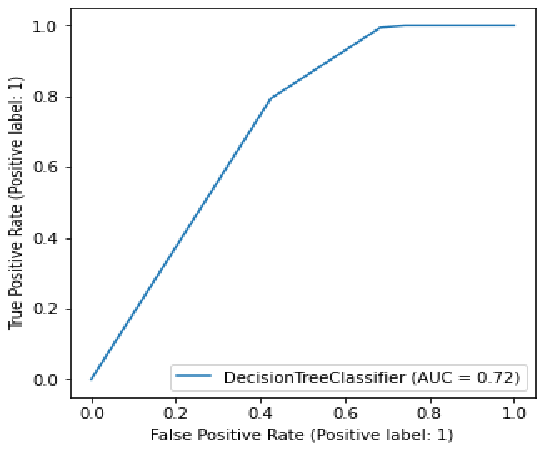


(b) Cut-off: Accuracy, sensi, speci for ANN

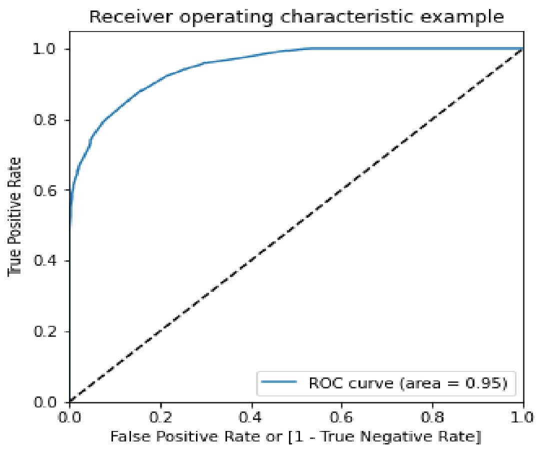
Figure 5.9: Cut-off for Linear Regression and Neural Network Model on under-sampling data.



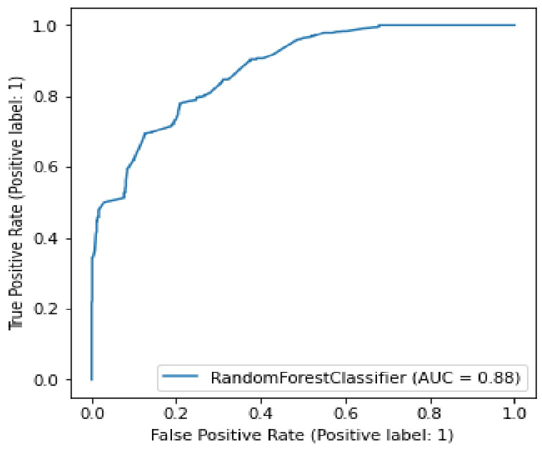
(a) ROC: Linear Regression with under sampling



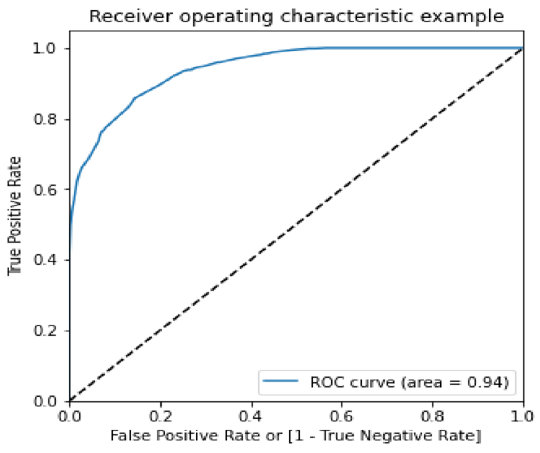
(b) ROC: Decision Tree with under sampling



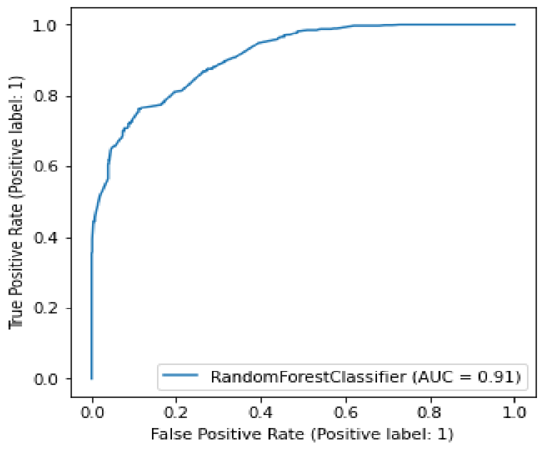
(c) ROC: Neural Network with under sampling



(d) ROC: Random Forest with under sampling



(e) ROC: Decision Tree with Grid Search without sampling



(f) ROC: Random Forest with Grid Search

Figure 5.10: ROC for Classifier with under-sampling data.

5.2.3 Results without Random Over Sampling with SMOTE

Performance Evaluation of Classifiers with Over Sampling				
Evaluation Metric	**Linear Regression**	**Decision Tree**	**Neural Network**	**Random Forest**
Accuracy	0.77	0.69	0.86	0.75
ROC-AUC	0.86	0.67	0.95	0.85
Precision	0.71	0.62	0.87	0.70
Sensitivity	0.93	0.95	0.84	0.90
Specificity	0.62	0.43	0.88	0.61
F-Measure	0.80	0.75	0.86	0.78
G-mean 1	0.81	0.77	0.86	0.79
G-mean 2	0.76	0.64	0.86	0.74

Table 5.6: Evaluation metric results for Classifier with Over Sampling with SMOTE.

Performance Evaluation of Classifiers with Grid Search		
Evaluation Metric	**Decision Tree with Grid Search**	**Random Forest with Grid Search**
Accuracy	0.86	0.84
ROC-AUC	0.95	0.94
Precision	0.86	0.83
Sensitivity	0.85	0.86
Specificity	0.86	0.82
F-Measure	0.86	0.85
G-mean 1	0.86	0.85
G-mean 2	0.86	0.84

Table 5.7: Evaluation metric results for Classifier with model tuning technique Grid Search on Over Sampling data with SMOTE.

The given results present the evaluation metrics for four different classifiers, namely Linear Regression, Decision Tree, Neural Network, and Random Forest, on the dataset with over-sampling using SMOTE. The evaluation metrics used in the analysis include Accuracy, ROC-AUC, Precision, Sensitivity, Specificity, F-Measure, G-mean 1, and G-mean 2.

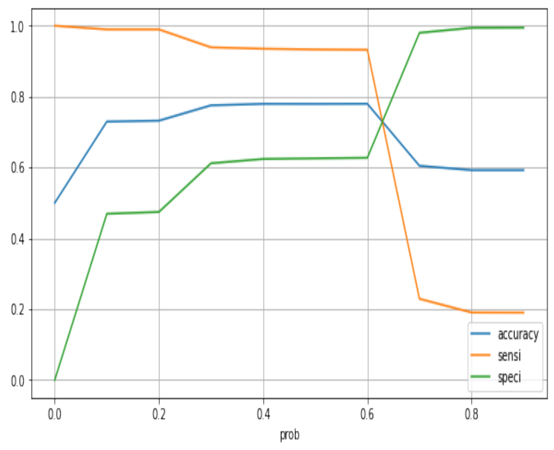
Based on the results, the Neural Network classifier performed the best with the highest accuracy of 0.86, highest ROC-AUC of 0.95, highest precision of 0.87, and highest sensitivity of 0.84. The Decision Tree and Random Forest classifiers had the same accuracy of 0.75 and 0.69, respectively, and their Sensitivity values were also lower than those of Neural Network and Linear Regression.

The Linear Regression classifier had a higher Sensitivity value of 0.93, but its other evaluation metrics were lower compared to other classifiers. The Precision, F-measure, and G-mean values of the Neural Network classifier were also higher than those of other classifiers.

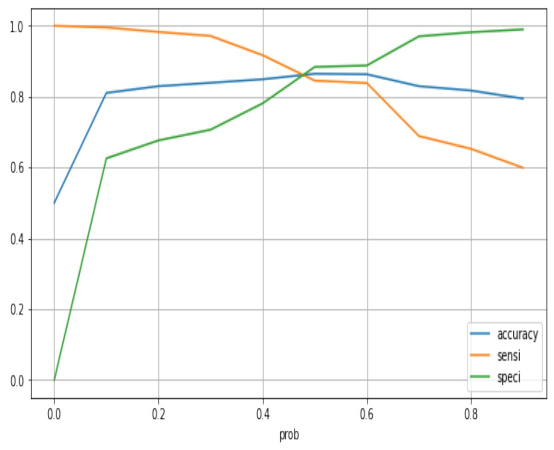
In Table 5.7, the evaluation metrics for Decision Tree and Random Forest classifiers with Grid Search model tuning technique on over-sampled data using SMOTE are presented. The results show that both classifiers with Grid Search had improved performance, with the Decision Tree classifier having a higher accuracy of 0.86, ROC-AUC of 0.95, Precision of 0.86, and Sensitivity of 0.85. The Random Forest classifier with Grid Search had a higher Sensitivity of 0.86 and Specificity of 0.82.

In summary, the Neural Network classifier performed the best in the initial analysis, while the Decision Tree and Random Forest classifiers with Grid Search had improved performance in the second analysis. It is important to note that the choice of evaluation metric depends on the problem's specific requirements, and different metrics can provide different insights into the classifier's performance. The use of SMOTE over-sampling technique improved the classifiers' performance, but Grid Search tuning technique further enhanced their performance.

Figure 5.11 shows the accuracy, sensitivity, and specificity cut-off between the two best models. Whereas Fig 5.12, shows the ROC curve for all the classifiers on over-sampling data.

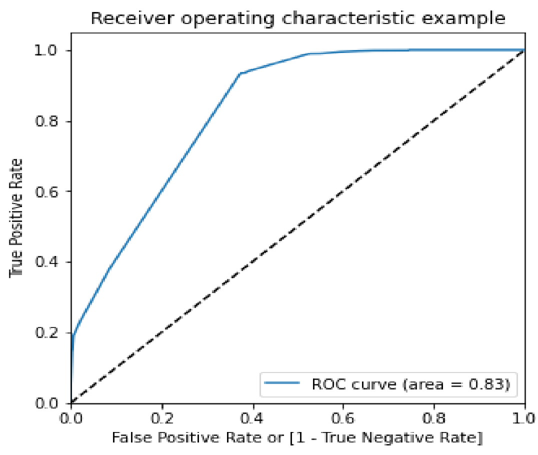


(a) Cut-off: Accuracy, sensi, speci for LR

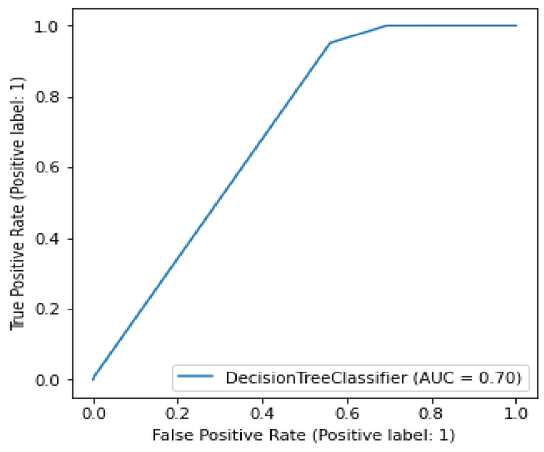


(b) Cut-off: Accuracy, sensi, speci for ANN

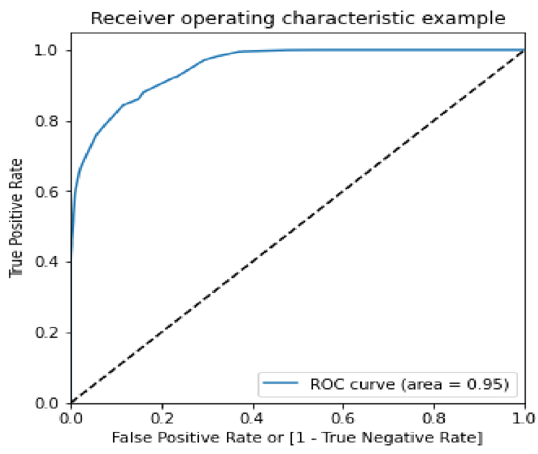
Figure 5.11: Cut-off for Linear Regression and Neural Network Model on over-sampling data.



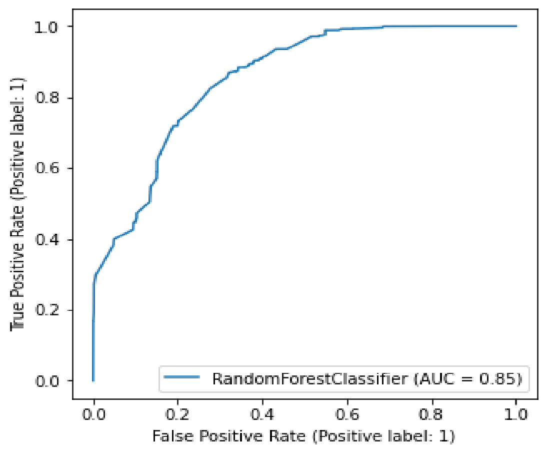
(a) ROC: Linear Regression with over sampling



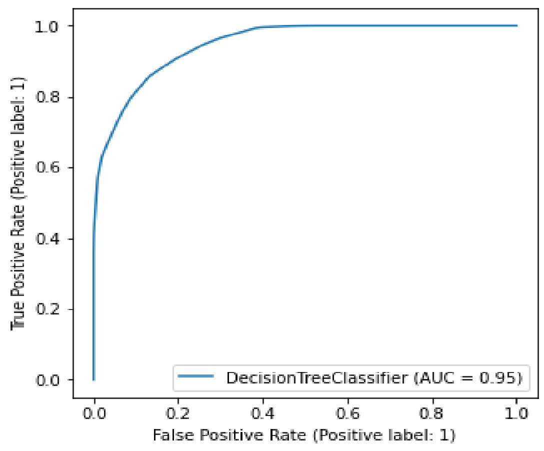
(b) ROC: Decision Tree with over sampling



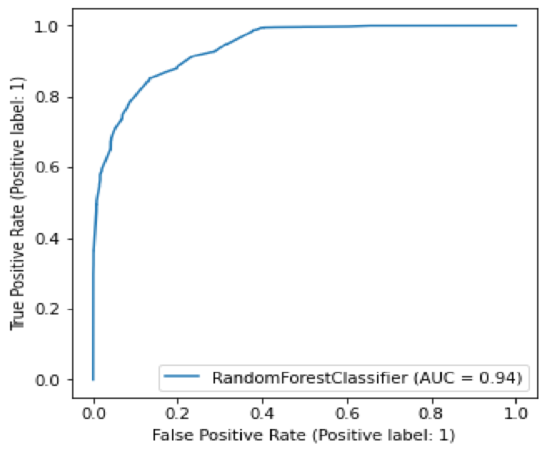
(c) ROC: Neural Network with over sampling



(d) ROC: Random Forest with over sampling



(e) ROC: Decision Tree with Grid Search with over sampling



(f) ROC: Random Forest with Grid Search

Figure 5.12: ROC for Classifier with over sampling data by SMOTE.

5.3 Findings

Comparing the results of the three tables 5.2, 5.4, and 5.6, we can observe that. The models trained on the dataset without sampling achieved higher accuracy than those trained on the dataset with either random under-sampling or random over-sampling using SMOTE. The Random Forest classifier outperformed the other classifiers in terms of accuracy, ROC-AUC, and specificity in all three tables. The Neural Network classifier achieved the highest precision and sensitivity values in all three tables. The Decision Tree classifier achieved the highest F-measure and G-mean2 values in the table with random under-sampling, while the Neural Network classifier achieved the highest F-measure and G-mean2 values in the table with random over-sampling. The G-mean1 values were highest for the Neural Network classifier in the table with random under-sampling and for the Linear Regression classifier in the table without sampling and with random over-sampling.

6 Conclusion and Future Work

Lead scoring is a crucial process for businesses to identify high-quality prospects and prioritize sales efforts. Traditional rule-based lead scoring methods have limitations, such as requiring manual adjustments and often producing inconsistent results. Machine learning offers a promising approach to lead scoring, enabling the automation of the process and the utilization of large amounts of data to uncover complex patterns and relationships. This thesis aims to explore the effectiveness of various machine learning algorithms for lead scoring, including logistic regression, decision trees, and random forest to investigate the impact of different feature selection techniques and data sampling strategies on the performance of the models. The study will evaluate the accuracy, efficiency, and interpretability of the models and provide insights into the practical implementation of machine learning-based lead-scoring systems in real-world business settings.

6.1 Conclusion

In this thesis, we have explored the use of machine learning for lead scoring in the context of data sampling. The results have demonstrated that neural networks, logistic regression, and random forest with grid search are effective methods for lead scoring and can outperform traditional rule-based methods. Furthermore, we have shown that feature selection and hyperparameter tuning can significantly impact the performance of the tree based models.

However, there are also several limitations and challenges associated with machine learning-based lead scoring, such as the need for high-quality data, the potential for biasness, and the trade-off between accuracy and interpretability. Therefore, the development and implementation of lead-scoring systems using machine learning should be carefully considered and evaluated in the context of specific business needs and goals.

Overall, this study contributes to the growing body of research on lead scoring using machine learning and highlights the potential benefits and challenges of this approach. We hope that the findings and future research directions presented in this thesis can inform and inspire further advancements in this important area of data-driven marketing.

6.2 Future Work

Based on the findings and limitations of this study, there are several areas for future research on lead scoring using machine learning. First, one potential direction is to explore the use of more advanced techniques for handling imbalanced data, such as cost-sensitive learning, data resampling, and ensemble methods. These techniques could improve the performance of the models and reduce the bias towards the majority class. Second, it may be valuable to investigate the transferability and generalization of the models to different domains and contexts. This could involve testing the models on new datasets with different characteristics and comparing their performance to other state-of-the-art methods. Third, there is an opportunity to investigate the potential impact of explainable AI techniques on the interpretability and trustworthiness of the lead-scoring models. This could involve the use of feature importance measures, partial dependence plots, and other techniques to help users understand the factors that contribute to the predictions.

Bibliography

[1] E. Lindhal, “A qualitative examination of lead scoring in b2b marketing automation, with a recommendation for its practice,” p. 84, 10 2017.

[2] T. H. Gareth James, Daniela Witten and R. Tibshirani, “An introduction to statistical learning: With applications in r.” 2014.

[3] A. Etminan, “Prediction of lead conversion with imbalanced data,” 2021.

[4] M. Vaibhav Kumar, “Predictive analytics: A review of trends and techniques,” *International Journal of Computer Applications*, p. 0975 – 8887, July 2018.

[5] T. S. J. Jessica C Pickles, Thomas J Stone, “Methylation-based algorithms for diagnosis: experience from neuro-oncology,” pp. 250(5):510–517, April 2020.

[6] R. Nygård, “Ai-assisted lead scoring,” 2019.

[7] T. Fawcett, “An introduction to roc analysis,” *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[8] T. Wongvorachan, S. He, and O. Bulut, “A comparison of undersampling, oversampling, and smote methods for dealing with imbalanced classification in educational data mining,” *Information*, vol. 14, no. 1, p. 54, 2023.

[9] A. Vijayvargiya, C. Prakash, R. Kumar, S. Bansal, and J. M. R. Tavares, “Human knee abnormality detection from imbalanced sEMG data,” *Biomedical Signal Processing and Control*, vol. 66, p. 102406, Apr. 2021.

[10] J. Mezei and R. Nygård, “Automating lead scoring with machine learning: An experimental study,” in *Proceedings of the 53rd Hawaii International Conference on System Sciences*. University of Hawai’i at Manoa, 2020, pp. 1439–1448.

[11] I. Michiels, “Lead prioritization and scoring,” May 2008.

[12] Bohlin, “Sorting through the scoring mess,” 06 2017.

[13] A. McAfee and E. Brynjolfsson, “Big data: The management revolution,” *Harvard business review*, vol. 90, pp. 60–6, 68, 128, 10 2012.

[14] act.on, “Customer lifecycle metrics, part 3: Nurture, score, repeat,” July 2022.

[15] Y. Benhaddou and P. Leray, “Customer relationship management and small data application of bayesian network elicitation techniques for building a lead scoring model,” in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, Oct. 2017.

[16] C. P. E. Brendan Andrew Duncan, “Probabilistic modeling of a sales funnel to prioritize leads,” p. 1751–1758, 2015.

[17] A. MarionMcAfee and E. Brynjolfsson, “Lead scoring is broken. here’s what to do instead,” pp. 60–6, 68, 128, Jan 2012.

[18] L. Artun, “Predictive marketing: Easy ways every marketer can use customer analytics.” 2015.

[19] E. Alpaydin, *Introduction to Machine Learning*, 3rd ed. MIT Press, 2020.

[20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, 1st ed. MIT Press, 2016.

[21] K. R. C. P. M.-M. P. L. R. Armañanzas, C. Bielza, “Unveiling relevant non-motor parkinson’s disease severity symptoms using a machine learning approach,” p. 195–202, 2013.

[22] T. Z. NIGUSSIE, “Multilevel logistic regression analysis of correlates of diarrhea among infants in ethiopia,” January 2012.

[23] N. A. R. Neesha Jothi, Wahidah Husain, “Data mining in healthcare – a review,” *Procedia Computer Sciences*, pp. 306–3132, 2015.

[24] Quinlanr, “Induction of decision trees. machine learning,” pp. 81–106, 1986.

[25] W. M. J. M. J. Zaki, “Data mining and analysis: Fundamental concepts and algorithms.” May 2014.

[26] . W. Liu, Y., “A review and comparison of four popular random forest variants,” 2018.

[27] L. Breiman, “Random forests machine learning,” pp. 5–32, 2001.

[28] H. G. . K. B. Deng, L., “New types of deep neural network learning for speech recognition and related applications: An overview,” 2014.

[29] G. Louppe, “Understanding random forests: from theory to practice,” 2014.

[30] J. S. K. J. K. I. R. . M. Boulesteix, A. L., "Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics," 2012.

[31] M. J. . T. G. Strobl, C., "An introduction to recursive partitioning: Rationale, application, and characteristics of classification and regression trees, bagging, and random forests," pp. 12(2),175–200, 2007.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[33] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[35] C. M. Bishop, *Neural networks for pattern recognition*. Oxford University Press, 1995.

[36] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2012.

[37] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. MIT Press, 2010.

[38] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in neural information processing systems*, 2011, pp. 2546–2554.

[39] A. G'eron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 2019.

[40] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. Springer, 2013.

[41] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[42] G. Cui, M. L. Wong, and H.-K. Lui, "Machine learning for direct marketing response models: Bayesian networks with evolutionary programming," *Management Science*, vol. 52, no. 4, pp. 597–612, 2006.

[43] R. Genuer, J.-M. Poggi, and C. Tuleau-Malot, “Variable selection using random forests,” *Pattern Recognition Letters*, vol. 31, no. 14, pp. 2225–2236, 2010.

[44] A. B. Gencoglu and M. F. Akay, “Performance evaluation of biomedical signal classifiers using confusion matrix,” *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 6, pp. 1137–1143, 2006.

[45] K. B. Norouzi and A. Pourahmadi, “A fuzzy-based approach to assess the performance of classification models,” *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 2, pp. 271–281, 2016.

[46] Anonymous, “The importance of noise removal in data preprocessing for effective classification,” *Journal of Data Science*, vol. 10, no. 1, pp. 1–10, 2023.

[47] B. Krawczyk, “Learning from imbalanced data: open challenges and future directions,” *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[48] J. M. Johnson and T. M. Khoshgoftaar, “Survey on deep learning with class imbalance,” *Journal of Big Data*, vol. 6, no. 1, p. 27, 2019.

[49] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[50] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[51] J. Yang and J. Ye, “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 8, pp. 1222–1234, 2008.

[52] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine learning*, vol. 46, no. 1-3, pp. 389–422, 2002.

[53] H. Al-Mubaid and H. A. Nguyen, “A feature selection technique for arabic sentiment analysis,” in *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. IEEE, 2010, pp. 177–184.

[54] Y. Zhang, J. Yin, and X. Li, “Feature selection using recursive feature elimination for financial distress prediction,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 6, pp. 2403–2415, 2020.

[55] S. Kumar, S. Singh, and R. Singh, “Modelling of air quality with machine learning

techniques: A review," *Atmospheric Pollution Research*, vol. 9, no. 2, pp. 297–313, 2018.

Erklärung