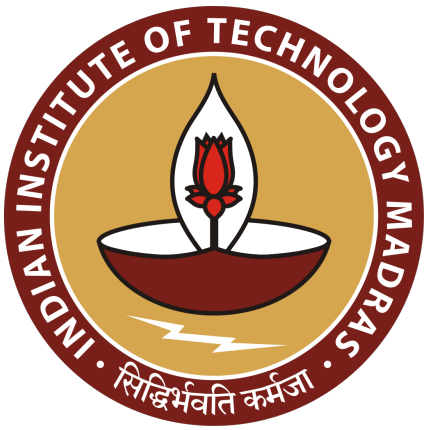
DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS
CHENNAI – 600036

A Survey on Churn Prediction Techniques and Time Series Forecasting



MTP Report

Submitted by

KARTHIKEYAN S

For the award of the degree

Of

MASTERS OF TECHNOLOGY

May 2023

THESIS CERTIFICATE

This is to undertake that the Report titled **A SURVEY ON CHURN PREDICTION TECHNIQUES AND TIME SERIES FORECASTING**, submitted by me to the Indian Institute of Technology Madras, for the award of **Masters of Technology**, is a bona fide record of the research work done by me under the supervision of **Dr. Balaraman Ravindran**. The contents of this Report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Chennai 600036 **Karthikeyan S**

Date: May 2023

Dr. Balaraman Ravindran
MTP Guide
Professor
Department of Computer Science and Engineering
IIT Madras

ACKNOWLEDGEMENTS

I want to express my sincere thanks and gratitude to my guide, Prof. Balaraman Ravindran, for assigning me this wonderful project. I am extremely grateful for his immense support and marvelous guidance throughout my whole M.Tech Project.

ABSTRACT

As every business has become more competitive, it has become essential for companies to strive towards further development and growth to survive in the industry. Businesses have become more competitive, and having a solid customer base has become more critical. Acquisition of new customers and retaining existing customers are very important to retain their business. But acquiring new customers has been saturated in many domains where every person is already a customer of some company, and thus retaining existing customers is crucial. Thus, identifying and handling potential churns in the customer base and handling them has become more significant.

Similarly, businesses need to make informed decisions to achieve their goals properly. Organizations must constantly adapt to market trends and consumer behavior changes in today's fast-paced business environment. Time series forecasting enables organizations to analyze historical data and predict future trends, allowing them to make informed decisions about resource allocation, product development, and marketing strategies.

Companies strive to create data-driven business decisions that require a variety of Algorithms and techniques to propel additional growth and progression. But numerous algorithms and techniques can apply to the above-discussed tasks; it requires lots of prerequisite knowledge to pick their way through all the sophisticated algorithms to solve their business needs. This work aims at developing reference guideline flowcharts through an extensive technical survey conducted on the above-mentioned tasks, namely Churn Prediction and Time Series Forecasting, which includes a comprehensive study of various available research and techniques on the problem of customers churning over various business domains. The flowcharts are designed to be the ideal reference for anyone with a business need to quickly sort through the algorithms and models for their demands as companies and enterprises grow and expand.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 CHURN PREDICTION

In business, the process by which customers switch from one vendor to another vendor is known as churn. Adequate knowledge about churn is essential for vendors and businesses to run their businesses and manage the resources which help in the smooth running of their services. Churners affect not just the earnings of the businesses but also the operations of the business too. Experts leaving platforms like Quora, Stack Overflow, and many forums may affect the basic functioning to a greater extent. Many businesses, like the Telecom industry, Banking industry, etc., have reached a saturated state where already every individual is a customer of some company. So in these cases, how much customer base a company can acquire from its competitor decides the winning of the race, and retaining the customers from switching to other companies remains the main goal of the companies. Likewise, the cost involved in obtaining a new customer is huge and much more than the cost involved in retaining an existing customer. Thus identifying and handling potential churners plays a pivotal role in running a successful business.

1.1.1 Significance of Churn Prediction

Various studies have concluded that the companies which retain many customers are the companies with great success and progress. The company can retain customers only if they know who potential churners are risking the business. Once these potential churners are identified, several measures, like giving them better serviceability in terms of support, cost of service, better grievance redressal, etc., can be taken to retain them. Predicting these churners alone cannot decide the success but also the effective measures taken later to solve their issues decide the business survival. Predicting churners does not just allow companies to retain their customers but also helps companies to improve their

standard of service provided as a whole. Thus the potential that churn prediction carries is very large as many essential and crucial business models like the Banking Industry, Telecom Industry (Rahman and Kumar (2020)), Gaming Industry (Liu *et al.* (2018)), E-commerce websites, Credit card companies, Online streaming platforms, Various companies' employee datasets, etc., rely on these churn prediction techniques to retain their customers with the motive of being successful in their business.

Potential churners can be identified based on various data about the customers, which may include most commonly their age, location, services subscribed, number of years, etc.

The problem of churn prediction can be solved by many approaches by considering it as a supervised, semi-super-supervised, and unsupervised problem. Also, there have been various machine learning techniques and deep learning approaches for solving the problem of churn prediction. The problem can be treated as just a classification problem where the main objective remains to just find whether a customer will be a potential churn or not. Also, this problem can be approached as a regression problem by finding within how much duration we can expect the customer to churn from the business. The problem can also be approached as an unsupervised problem by trying to find optimal clusters among the customers and finding the cluster with the most churners, and analyzing the particular cluster for the behavior or the reason that may induce more further churners in the future. There are many more ways by which we can model the problem as multi-label classification too.

This work focuses on exploring and analyzing various existing methods for solving the problem of churn prediction by approaching the problem as classification, Regression, and clustering problems with the focus of developing a generic guideline to solve problems related to churn prediction.

1.2 TIME SERIES FORECASTING

Time series forecasting involves analyzing past data to identify patterns and trends and then using that information to make predictions about the future values of a variable. Time series data is a sequence of observations taken at regular intervals over time, and these observations can be used to build a model that can forecast future values based on historical data.

Time series forecasting can be done using various techniques such as statistical models, machine learning algorithms, and deep learning models. Statistical models such as ARIMA and exponential smoothing are popular choices for time series forecasting. Machine learning algorithms such as decision trees, random forests, and support vector machines can also be used for time series forecasting. Deep learning models such as recurrent neural networks and LSTM networks are also becoming popular in time series forecasting.

Time series forecasting is an important tool for predicting future trends, sales, demand, and other key metrics. Businesses can use time series forecasting to make informed decisions about inventory management, production planning, and resource allocation. For example, a retail store can use time series forecasting to predict future sales based on historical sales data and plan its inventory accordingly. A manufacturing company can use time series forecasting to predict demand for its products and plan production schedules accordingly. Time series forecasting can help businesses to optimize their operations, reduce costs, and improve profitability.

1.2.1 Significance of Time Series Forecasting

Time series forecasting can help businesses to optimize their operations and improve their bottom line. By accurately predicting future demand, businesses can plan their production schedules, adjust their inventory levels, and allocate resources more efficiently. This can help businesses to reduce costs and improve profitability.

Overall, time series forecasting is an essential tool for businesses that want to stay competitive in today's fast-paced market. By leveraging the power of data analytics and machine learning, businesses can gain valuable insights into the future and make informed decisions that can help them to achieve their goals and succeed in their respective industries.

This work focuses on several statistical approaches like ARIMA, SARIMA, VAR/VAX, Holt winter machines, etc, along with various machine learning and Deep learning algorithms for exploring and analyzing various Time series forecasting problem domains that might be of great interest to various business perspectives like Sales, Demand, Price, and availability.

CHAPTER 2

BACKGROUND

2.1 DATA SAMPLING

The main problem with churn datasets is that the data remains mostly unbalanced and biased towards a single class. This leads to a major deterioration in the robustness of the model we are building. Data sampling techniques can be majorly classified into oversampling: adding data points within the class that has fewer data points, undersampling: removing data points from the class that has more data points, and a combination of both oversampling and undersampling. Geiler *et al.* (2022).

2.1.1 SMOTE(Synthetic minority oversampling technique)

The most widely used state-of-the-art method for the purpose of sampling the data is SMOTE(Synthetic Minority Oversampling Technique). SMOTE over-samples the class which has lesser data points by creating new refined instances along the line segment created by the KNN approach. Han *et al.* (2005) Simply we can over-sample a class by duplicating instances at random. But that won't add any useful advantage to the model and thus SMOTE considers random instances in the class with lesser data points and the k-nearest neighbors $\{d_i\}_{i\epsilon\{1,2,..k\}}$ are used to create new refined instances as follows,

$$D_i^{\text{new}} = d + \mathcal{U}([0,1])\text{x}(d_i - d)$$

Smote helps in preventing the model from overfitting by creating very refined instances. But in a few cases, if we overdo the process, the new refined instances created may be completely ambiguous and may not fit in the behavior that other data points exhibit. This section explains the various traditional ML algorithms that have been used so far in various research for churn prediction.

2.2 CLASSIFICATION ALGORITHMS

2.2.1 KNN algorithm

KNN or K-nearest neighbor, a simple and effective algorithm for classification, uses K data points available in the training dataset identified as closest to the new data point to classify the new point. It is a supervised learning technique and thus requires data points to be labeled.

KNN Algorithm involves Finding the K nearest points requires calculating the distance between all data points in the dataset and the new data point(and various distance measures like Euclidean, Manhattan, etc can be used for the same), and choosing the K nearest points among them based on the distance measure used and with the knowledge of the k points labeling the new data point.

KNN can be represented as

$$p(C_i = g|x_i) = \frac{1}{k} \sum_{j \epsilon \Omega_k} \mathbb{1}\{x_j\}$$

where x_i is the data point considered to be labeled and it assigns the respective label that is found majorly among its k near data points found represented by Ω_k. The indicator function $\mathbb{1}$ denotes that the value takes the value of one if the datapoint belongs to the positive class and zero if it belongs to the negative class. Geiler *et al.* (2022)

Advantages :

KNN, unlike other ML techniques, doesn't require a training phase and thus it can adapt well to the new data and thus it is more robust to noise present in the data. KNN ignores the missing data while distance calculation and thus handles the missing data points very well.

Disadvantages:

KNN has many disadvantages as their computational complexity is very high and they are sensitive to inapplicable features. Along with this many drawbacks are discussed in Dubey and Pudi (2013) Tan (2005), and have been concluded that KNN doesn't perform

great in Churn data.

2.2.2 Logistic Regression

Logistic regression is a binary classification method used to find the probability of the occurrence of an event using independent variables. The features can be categorical or continuous in nature. Logistic regression uses the logistic function (sigmoid curve) and predicts the outcome of the variable. The sigmoid curve is obtained as a result of the combination of predictor variables.

Logistic regression evaluates the output as the posterior probability as follows

$$Y = \frac{e^{\beta_0+\beta_1*x}}{1 + e^{\beta_0+\beta_1*x}}$$

where Y denotes the output evaluated, the intercept is denoted by β_0, and β_1 refers to the co-efficient of the input point x. These beta values must be estimated through the process of Maximum Likelihood estimation by the Newton–Raphson algorithm. Here the 2nd order derivative of the likelihood term is calculated.

Advantages:

1. It is a simple and efficient model that is easy to implement and interpret.
2. It can handle both continuous and categorical predictor variables.
3. It can provide estimates of the probability of an event occurring, which can be useful for decision-making.
4. It is robust to outliers and can still produce reliable results even when there are a few unusual or extreme observations in the data.

Disadvantages:

1. It can only be used for binary classification (predicting two classes). If you have more than two classes, you would need to use a different type of model.
2. It assumes that the relationship between the predictor variables and the outcome is linear, which may not always be the case.

3. It is sensitive to small changes in the predictor variables, which can make the model less stable and more difficult to interpret.

4. It may not perform well on highly imbalanced datasets, where one class is much more common than the other.

2.2.3 SVM (Support Vector Machines)

Vapnik and Vapnik (1998) introduced a set of supervised methods called Support Vector Machines (SVMs), that could be used for various tasks like classification, regression, etc. The algorithm works by finding the hyperplane in a high-dimensional space that segregates different classes into their utmost level. Once trained, the SVM model can then use this hyperplane, denoted below, to classify new data points.

$$x_i | \sum_{j=1}^{d} x_{ij}\beta_j + \beta_0 = x_i^T\beta + \beta_0 = 0$$

where β_j are coefficients. SVM are defined as $\min \beta, \beta_0 ||\beta||^2$ with respect to $y_i(X_i^T\beta + \beta_0 \geq 1$, for all all positive non-zero integers upto n.Geiler *et al.* (2022)

Effective handling of dimensionally huge datasets is great convenience offered by SVMs, making them a useful tool for working with complex data sets. They are also effective when the number of features (dimensions) is much greater than the number of samples, which can occur in certain applications such as text classification.

SVMs can be kernelized, which means that they can apply a nonlinear transformation to the input data and find a separating hyperplane in the transformed space. This makes them particularly well-suited for tasks involving data that can't be separated using a linear function or line in the original feature space.

Since SVM only takes into account the support vectors, i.e., the points that are close to the boundary, it is an interesting candidate for moderately imbalanced datasets, although it performs poorly when the class distribution is too skewed Tian *et al.* (2011)

Advantages:

1. Efficient: SVMs can efficiently perform both linear and nonlinear classification tasks, making them useful tools for working with complex data sets.

2. Versatile: SVMs can handle high-dimensional data effectively.

3. Robust: SVMs are relatively resistant to overfitting, because of their optimization criteria and the use of kernel functions.

4. Interpretable: SVMs can provide clear, interpretable decision boundaries, which can be useful for understanding how the model is making predictions.

Disadvantages:

1. Sensitivity to scaling: Highly sensitive to input features being scaled and thus utmost care should be taken while scaling the data before training.

2. Complexity: SVMs can be more complex to implement and tune compared to some other machine learning algorithms, which can make them more difficult to use for beginners.

3. Limited to two-class classification: SVMs are limited to binary classification tasks, so they may not be suitable for multi-class classification problems.

4. Limited to linear decision boundaries: SVMs are limited to linear decision boundaries in this transformed space. This can be a limitation for certain types of data.

5. SVM performs very badly when we implement it on datasets that have skewed class distribution. Tian *et al.* (2011)

2.2.4 Decision Tree

A decision tree is a flowchart-like tree structure used to make decisions based on certain conditions. It breaks down a complex problem into smaller and simpler decisions, making it easier to solve.

Each internal node in the tree represents a ”test” on an attribute, and each leaf node represents a class label. The branches represent the possible outcomes of the test, and the decision tree is constructed by considering each possible attribute at each step and choosing the attribute that provides the most information gain.

Mathematically, the decision tree algorithm uses the concept of entropy to determine which attribute to split on at each step. Entropy is a measure of the amount of disorder or randomness in a system. The decision tree algorithm tries to maximize the information gain at each step by minimizing the entropy.

The entropy of a system is calculated using the following formula: Entropy = $-\sum(p(i) * \log_2(p(i)))$ where p(i) is the probability of the i-th outcome.

An effective decision point Geiler *et al.* (2022) is obtained based on the proportion $\hat{p}_{mk}$ of class k over region R_m with N_m observations as follows

$$\hat{p}_{mk} = \frac{1}{N} \sum_{x_i \sigma R_m} \mathbb{1}(y_i = k)$$

Advantages:

1. They are easy to understand and interpret since they involve simple decisions based on the values of the features. This makes them useful for explaining decisions to non-technical stakeholders.

2. They can handle high-dimensional data, i.e., data with many features, without requiring any special preparation.

3. They are robust to noise and missing values in the data since the tree structure allows for a high degree of tolerance.

4. They can be used for both classification and regression tasks.

5. They can be easily updated with new data, making them useful for real-time applications.

Disadvantages:

1. They can be prone to overfitting, especially when the tree is deep and has a large number of branches. Overfitting means that the model is too closely tied to the training data, and may not generalize well to new data.

2. Highly unstable

3. They may not always provide the most accurate predictions, especially when the data is complex and has multiple non-linear relationships.

4. They may require pre-pruning, i.e., limiting the maximum depth of the tree, to prevent overfitting. This can be done manually or automatically, but it requires careful tuning to get good results.

2.2.5 Random Forest

Random forest is an ensemble machine learning method. They work by forming and uniting various decision trees and thus resulting in a forest. Thus random forests remain

immune to the decision trees' behavior of overfitting.

In simple terms, a random forest is a set of decision trees grouped together for the task of classification or regression. Prediction output from every tree is considered and the prediction that seems most likely is chosen by random forest. The trees in a random forest are trained on different parts of the training data, and they use a variety of different characteristics of the training data to make their predictions. This diversity of trees helps the random forest make better predictions than any individual tree would be able to.

Advantages:

1. They can be used for both classification and regression tasks.
2. They can handle high-dimensional data and a large number of training examples.
3. They are resistant to overfitting, due to the way they are constructed (by training multiple decision trees on different parts of the training data).

Disadvantages:

1. They can be computationally expensive to train, especially when the number of trees is large or the data is highly dimensional.
2. They may not provide the best predictions for certain types of problems, such as when the decision boundary is very irregular or when the data is heavily imbalanced.
3. They are not as interpretable as some other models, such as decision trees or linear regression, because it is difficult to understand how the individual trees are combined to make the overall prediction.

2.2.6 XGBoost

Extreme Gradient Boosting or XGBoost is one of the famous and robust machine learning algorithms for regression and classification problems, based on boosting decision trees. It optimizes a cost function by iteratively adding decision trees to the model while minimizing a regularized objective function. The algorithm uses gradient boosting to improve the performance of individual decision trees by adjusting the weights of each observation in the training set (Chen and Guestrin (2016). The formula for XGBoost can be represented as:

$$F(x) = w_0 + \sum_{j=1}^{M} T(x, \theta_j)$$

where $F(x)$ is the prediction of the model for a given input x, w_0 is the initial prediction, $T(x; \theta_j)$ is a decision tree with parameters θ_j, and the sum is taken over all M trees in the model. Advantages of XGBoost include its ability to handle missing data and its strong performance on a wide range of data types and sizes. It also has a built-in mechanism to prevent overfitting and can handle unbalanced datasets. However, it can be computationally intensive and requires careful tuning of hyperparameters to achieve optimal performance.

2.2.7 LightGBM

LightGBM is a gradient-boosting algorithm that utilizes decision trees to model the relationship between features and the target variable. The model works by iteratively adding decision trees to improve the predictions. The prediction of the model is calculated as the sum of the predictions of all the trees. The formula for the prediction of LightGBM can be expressed as:

$$\hat{y}_i = \sum_{j=1}^{M} f_j(x_i)$$

where $\hat{y}_i$ is the predicted value for the i-th observation, M is the number of trees, f_j is the prediction of the j-th tree and x_i is the feature vector for the i-th observation.

Advantages of LightGBM include its speed, as it uses a histogram-based approach to find the best-split points, and its ability to handle large datasets with high dimensionality. It also has good accuracy and can handle missing values (Ke *et al.* (2017).

Disadvantages of LightGBM include its sensitivity to overfitting, which can be addressed by tuning the hyperparameters, and its lack of interpretability, as it can be difficult to understand the importance of each feature in the model.

2.3 REGRESSION ALGORITHMS

Linear Regression:

Simple linear regression is a statistical technique that uses a linear function to represent the relationship between a dependent variable and a single independent variable. The model estimates the slope and intercept of the line of best fit that reduces the sum of the squared errors between the prediction and the ground truth. The formula for simple linear regression can be expressed as:

$$y = \beta_0 + \beta_1 x + \epsilon$$

where y, x, β_0, β_1, and ϵ are the dependent variable, the independent variable, the intercept, the slope, and the error term respectively. Advantages of simple linear regression include its simplicity and ease of interpretation. It can also be useful for making predictions and identifying the strength and direction of a relationship between two variables. Disadvantages of simple linear regression include its assumption of linearity, which may not hold for all datasets. It also assumes that the error term has constant variance and is normally distributed, which may not be the case for all datasets. Additionally, it can be sensitive to outliers and influential observations.

Multiple Linear Regression:

Multiple linear regression is a statistical method that resembles the relationship between a dependent variable and multiple independent variables using a linear function. The model estimates the coefficients for each independent variable that best fits the data and predicts the value of the dependent variable. The formula for multiple linear regression can be expressed as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$$

where y is the dependent variable, $x_1, x_2, ..., x_n$ are the independent variables, β_0 is the intercept, $\beta_1, \beta_2, ..., \beta_n$ are the coefficients, and ϵ is the error term. Advantages of multiple linear regression include its ability to model complex relationships between multiple variables, its ease of interpretation, and its ability to make predictions based on the values of the independent variables. Disadvantages of multiple linear regression include its assumption of linearity, which may not hold for all datasets. It also assumes that the error term has constant variance and is normally distributed, which may not be the case for all datasets. Additionally, it can be sensitive to outliers and influential observations, and multicollinearity can cause problems if the independent variables are highly correlated.

Polynomial regression:

Polynomial regression is a statistical method that models the correspondence between a dependent variable and an independent variable using a polynomial function. The model estimates the coefficients for each term in the polynomial equation that best fits the data and predicts the value of the dependent variable. The formula for polynomial regression can be expressed as:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + ... + \beta_n x^n + \epsilon$$

where y, x, is the dependent variable and the independent variable, is the β_0 intercept, $\beta_1, \beta_2, ..., \beta_n$ are the coefficients, $x^2, x^3, ..., x^n$ are the polynomial terms, and ϵ is the error term. Advantages of polynomial regression include its ability to model non-linear relationships between variables and capture more complex patterns in the data. It also allows for more flexibility in fitting the data compared to linear regression. Disadvantages of polynomial regression include its potential for overfitting, which can occur when the polynomial degree is too high and the model becomes too complex. It can also be sensitive to outliers and influential observations. Additionally, it may not be appropriate for datasets with sparse or unevenly distributed data points.

Lasso regression:

Lasso regression is a method that can be used as a safeguard against overfitting in linear regression models by adding a penalty term to the cost function. The penalty term is proportional to the absolute value of the regression coefficients, which encourages the model to have sparse coefficients and eliminates irrelevant features. The formula for the cost function of lasso regression can be expressed as:

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\beta(x_i) - y_i)^2 + \lambda \sum_{j=1}^{n} |\beta_j|$$

where $J(\beta)$ is the cost function, $h_\beta(x_i)$ is the predicted value for the i-th observation, y_i is the actual value, β_j is the j-th regression coefficient, λ is the regularization parameter, and n is the number of features. Advantages of lasso regression include its ability to perform feature selection by eliminating irrelevant features and reducing the complexity of the model. It also works well for datasets with a large number of features. Disadvantages of lasso regression include its sensitivity to the choice of the regularization parameter, which can be difficult to determine, and its inability to handle correlated features. Additionally, lasso regression may not work well for datasets with small sample sizes or when there are many important features with similar coefficients.

Ridge Regression: Ridge regression, also known by the name L2 regularization, is a method used as a safeguard against overfitting in linear regression models by adding a penalty term to the cost function. The penalty term is directly proportional to the square of the regression coefficients, which encourages the model to have small and smooth coefficients. The formula for the cost function of ridge regression can be expressed as:

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\beta(x_i) - y_i)^2 + \lambda \sum_{j=1}^{n} \beta_j^2$$

where $J(\beta)$ is the cost function, $h_\beta(x_i)$ is the predicted value for the i-th observation, y_i

is the actual value, β_j is the j-th regression coefficient, λ is the regularization parameter, and n is the number of features. Advantages of ridge regression include its ability to model's complexity reduction and improve its generalization performance. It is also less sensitive to the choice of the regularization parameter compared to lasso regression. Disadvantages of ridge regression include its inability to perform feature selection and its tendency to shrink all the coefficients toward zero, which can lead to biased estimates. Additionally, it may not work well for datasets with a small number of features or when there are many important features with similar coefficients.

Support Vector Regression:

Support Vector Regression, a type regression algorithm, uses Support Vector Machines (SVMs) to build a regression model. It aims to find a hyperplane in a high-dimensional space that maximizes the margin between the predicted values and the actual values. SVR can use different types of kernels for mapping of input data into a higher dimensional space, including linear, polynomial, radial basis function (RBF), and sigmoid kernels. The formula for SVR with a linear kernel can be expressed as: $y = \sum_{i=1}^{n} w_i x_i + b$ where y is the predicted value, w_i is the weight associated to the i-th feature, x_i is the i-th feature value, and b is the intercept. The formula for SVR with a polynomial kernel can be expressed as: $y = \sum_{i=1}^{n} \alpha_i K(x_i, x) + b$ where α_i is the Lagrange multiplier, $K(x_i, x)$ is the polynomial kernel function, and b is the intercept. The formula for SVR with an RBF kernel can be expressed as: $y = \sum_{i=1}^{n} \alpha_i K(x_i, x) + b$ where α_i is the Lagrange multiplier, $K(x_i, x)$ is the RBF kernel function, and b is the intercept. The formula for SVR with a sigmoid kernel can be expressed as: $y = \sum_{i=1}^{n} \alpha_i K(x_i, x) + b$ where α_i is the Lagrange multiplier, $K(x_i, x)$ is the sigmoid kernel function, and b is the intercept. Advantages of SVR include its ability to handle non-linear relationships between the variables and its robustness to outliers. It also allows for fine-tuning of the hyperparameters to optimize the model's performance. Disadvantages of SVR include its sensitivity to the choice of the kernel function and its

complexity, which can lead to longer training times and higher computational requirements. Additionally, it may not perform well for datasets with a large number of features or when there is noise in the data.

ElasticNet regression:

ElasticNet regression is a hybrid model that combines L1 and L2 regularization to overcome the limitations of each method. It is used to prevent overfitting in linear regression models by adding a penalty term to the cost function that includes both the L1 and L2 norms of the regression coefficients. The formula for the cost function of ElasticNet regression can be expressed as:

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\beta(x_i) - y_i)^2 + \alpha\rho \sum_{j=1}^{n} |\beta_j| + \frac{\alpha(1-\rho)}{2} \sum_{j=1}^{n} \beta_j^2$$

where $J(\beta)$ is the cost function, $h_\beta(x_i)$ is the predicted value for the i-th observation, y_i is the actual value, β_j is the j-th regression coefficient, α is the regularization parameter, ρ controls the balance between L1 and L2 regularization, and n is the number of features. Advantages of ElasticNet regression include its ability to handle high-dimensional datasets and perform feature selection. It also works well when there are many correlated features, as it can select a group of features together. Disadvantages of ElasticNet regression include its sensitivity to the choice of the regularization parameters, which can be difficult to tune, and its tendency to shrink the coefficients towards zero, which can lead to biased estimates. Additionally, it may not work well for datasets with a small number of features or when there are many important features with similar coefficients.

SGD Regressor:

Stochastic Gradient Descent (SGD) regressor is a linear regression algorithm that updates the model parameters by minimizing the cost function on a randomly selected subset of the training data in each iteration. It is an efficient and scalable algorithm for large datasets. The formula for the cost function of SGD regressor can be expressed as:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 + \alpha R(\theta)$$

$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 + \alpha R(\theta)$ where $J(\theta)$ is the cost function, $h_\theta(x_i)$ is the predicted value for the i-th observation, y_i is the actual value, θ are the regression coefficients, α is the regularization parameter, and $R(\theta)$ is the regularization term. Advantages of SGD regressor include its efficiency and scalability for large datasets, as it only uses a random subset of the training data in each iteration. It also works well when the data is noisy, as it updates the parameters on each observation. Disadvantages of SGD regressor include its sensitivity to the choice of hyperparameters, such as the learning rate and regularization parameter, which can be difficult to tune. It may also not perform well for datasets with a small number of observations or when the features are highly correlated. Additionally, it may require more iterations to converge compared to other algorithms.

2.4 CLUSTERING ALGORITHMS

K-Means:

K-means is an unsupervised machine learning algorithm used for clustering data points into K groups based on their similarity. The algorithm works by iteratively updating the centroids of the clusters until convergence. The formula for the K-means algorithm can be expressed as follows: Initialize K cluster centroids randomly Assign each data point to the nearest centroid Recalculate the centroid for each cluster by taking the mean of all the points assigned to that cluster Repeat steps 2 and 3 until convergence, i.e., until the centroids no longer change or the maximum number of iterations is reached.

Given a set of n data points $X = x_1, x_2, ..., x_n$ and the number of clusters K, the goal of K-means clustering is to find K cluster centroids $\mu_1, \mu_2, ..., \mu_K$ that reduces the sum of squared distances between each data point and its assigned cluster centroid. This can be

expressed as: argmin $\sum_{i=1}^{n} \left|x_i - \mu_j\right|^2$ where j is the index of the nearest centroid to x_i, and |.| denotes the Euclidean distance. Each data point is assigned to the nearest centroid and updates the centroid positions until convergence. Advantages of K-means clustering include its simplicity and speed, making it suitable for large datasets. It also does not require prior knowledge of the number of clusters and can handle different types of data. Additionally, it has a clear interpretation of the results and can be used for outlier detection. Disadvantages of K-means clustering include its sensitivity to the initial random selection of centroids, which can lead to suboptimal clustering. It also assumes that clusters are spherical and equally sized, which may not always be true for real-world datasets. Finally, it can be difficult to determine the optimal number of clusters, as it is often a subjective decision based on the data and the problem domain.

MiniBatchKMeans:

MiniBatchKMeans is a variant of the K-means clustering algorithm that uses mini-batches of data instead of the full dataset to compute the cluster centroids. This reduces the computational time and memory requirements of the algorithm. The formula for MiniBatchKMeans is similar to that of K-means, but instead of updating the centroids using the entire dataset, a small random subset (or mini-batch) of the data is used at each iteration. This can be expressed mathematically as: Initialize K cluster centroids randomly Select a random subset (mini-batch) of data points Each data point is assigned to the nearest centroid Update the centroid for each cluster by taking the mean of all the points assigned to that cluster in the mini-batch Repeat steps 2-4 for a fixed number of iterations or until convergence. Advantages of MiniBatchKMeans include its faster convergence and lower memory requirements compared to the standard K-means algorithm. It can also handle larger datasets and can be used for real-time data processing. Additionally, it can produce similar results to the standard K-means algorithm with less computational time. Disadvantages of MiniBatchKMeans include its sensitivity to the size of the mini-batch and the number of iterations, which can affect the

quality of the clustering. It can also be less accurate than the standard K-means algorithm, particularly for datasets with unevenly sized clusters. Finally, it may require more careful tuning of hyperparameters to achieve optimal results.

Gaussian Mixture Models:

Gaussian Mixture Model (GMM) is a statistical model used for clustering, density estimation, and generating new data points from an underlying distribution. GMM assumes that the data points are generated from a mixture of Gaussian distributions with unknown parameters, such as the mean and covariance. The algorithm uses an expectation-maximization (EM) approach to estimate these parameters and cluster the data points. The formula for GMM involves computing the probability density function (PDF) of each Gaussian distribution and taking the weighted sum of these distributions. Mathematically, the formula for GMM can be expressed as: $p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$ where: $p(x)$ is the pdf of the mixture model K is the number of Gaussian distributions in the mixture model π_k is the weight of the k-th distribution, representing the proportion of data points that belong to that distribution $\mathcal{N}(x|\mu_k, \Sigma_k)$ is the PDF of the k-th Gaussian distribution with mean μ_k and covariance Σ_k x is a data point. Advantages of GMM include its ability to model complex data distributions and capture the underlying structure of the data. It can also handle different shapes and sizes of clusters, unlike K-means which assumes that clusters are spherical and have equal variance. Additionally, GMM can provide soft clustering results, meaning that data points can belong to multiple clusters with different probabilities. Disadvantages of GMM include its sensitivity to the choice of the number of Gaussian distributions in the mixture model, which can affect the quality of the clustering results. It can also be computationally expensive, particularly for high-dimensional data. Finally, GMM can suffer from overfitting if the number of parameters in the model is very much relative to data points

Spectral clustering:

Spectral clustering is a technique used to cluster data points by transforming the data into a lower dimensional space using eigenvectors. The goal is to group data points that are interconnected forming a cohesive group. The algorithm works by constructing an affinity matrix. This affinity matrix is then transformed into a Laplacian matrix, which is decomposed using eigenvectors. Eigenvectors formed by the smallest eigenvalues are then used to form a lower dimensional space, in which the data points are clustered using a standard clustering algorithm. The formula for Spectral Clustering in LaTeX is: Given a set of n data points $x_1, x_2, \ldots, x_n$, we construct an affinity matrix $W \in \mathbb{R}^{n \times n}$ where W_{ij} represents the similarity between points i and j. The Laplacian matrix is defined as $L = D - W$, where $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $D_{ii} = \sum_{j=1}^{n} W_{ij}$. The eigenvectors $u_1, u_2, \ldots, u_k$ corresponding to the k smallest eigenvalues of L are then used to form a new feature representation of the data, $Y \in \mathbb{R}^{n \times k}$, where $Y_{ij} = u_{ij}$. The advantages of Spectral Clustering include its ability to handle non-linearly separable data and its robustness to noise and outliers. Additionally, it can handle data sets with arbitrary shapes and sizes. However, Spectral Clustering requires the computation of an affinity matrix and the eigendecomposition of a large matrix, which can be computationally expensive for large datasets. Also, determining the optimal number of clusters can be challenging.

Affinity Propagation:

Affinity Propagation is a clustering algorithm that does not require the user to specify the number of clusters beforehand. The algorithm works by iteratively passing messages between data points, which results in the identification of ”exemplar” points that represent the different clusters. The similarity between data points is calculated using a similarity matrix that reflects the degree of similarity between the features of each point. The algorithm converges when the exemplar points no longer change. The formula for the similarity matrix is: s(i,j) $= -\left\|x_i - x_j\right\|^2$ where x_i and x_j are data points, and $s(i, j)$

is the similarity between them. Advantages of Affinity Propagation include its ability to identify the number of clusters automatically, its effectiveness on small to medium-sized datasets, and its ability to handle non-linearly separable data. However, its disadvantages include its high computational complexity, sensitivity to the initial choice of exemplars, and potential for overfitting.

Hierarchical clustering:

Hierarchical clustering is a clustering mechanism that builds a hierarchy of clusters by recursively dividing or merging them based on a similarity measure. The algorithm starts by considering each data point as its own cluster and then proceeds iteratively by merging or dividing the clusters until a stopping criterion is met. The result is a dendrogram that shows the hierarchical relationship between the clusters. Distance computation between two clusters in hierarchical clustering is typically represented by one of several distance measures, such as Euclidean distance or Manhattan distance. One advantage of hierarchical clustering is its ability to visualize the hierarchy of clusters through the dendrogram, which can provide insights into the underlying structure of the data. Additionally, hierarchical clustering does not require a pre-specified number of clusters, unlike some other clustering algorithms. However, hierarchical clustering when implemented over large datasets may be computationally expensive, as the algorithm requires computing distances between all pairs of data points. Additionally, the choice of linkage criteria and distance measures can significantly affect the resulting clusters.

DBSCAN:

DBSCAN is an algorithm under the clustering method in which nearby points are grouped together in a high-density region while marking points in low-density regions as noise. The algorithm defines a cluster as a set of data points that are within an epsilon (ϵ) neighborhood of a core point, where ϵ is a user-defined distance threshold, and the core point has at least a minimum number of neighboring points (MinPts). The

algorithm iteratively expands the cluster by adding neighboring points until no new points can be added. DBSCAN has several advantages, including the ability to handle clusters of arbitrary shapes and sizes, the ability to handle noise and outliers, and the ability to automatically determine the number of clusters. However, it can be sensitive to the choice of ϵ and MinPts parameters, and it may not perform well on datasets with varying densities. The formula for the ϵ-neighborhood of a point p_i is given by: $N_\epsilon(p_i) = p_j \in D | dist(p_i, p_j) \leq \epsilon$, where D is the dataset, $dist$ is a distance metric, and ϵ is the distance threshold. The formula for the core points is given by: $|N_\epsilon(p_i)| \geq MinPts$, where $MinPts$ is the at least number of points required to form a cluster.

K-Prototypes:

K-Prototypes is a clustering algorithm used for datasets that have both numerical and categorical features. It is an extension of the K-Means algorithm, where the distance measure is a weighted combination of the Euclidean distance for numerical features and the dissimilarity measure for categorical features. Minimization of the sum of squared errors between data points and their assigned cluster centroids, similar to K-Means is the primary goal. However, in K-Prototypes, the dissimilarity between a categorical feature of a data point and the centroid is measured using the Gower distance, which takes into account the type of the feature. The algorithm starts by randomly assigning data points to clusters and then iteratively updates the cluster centroids and the data point assignments. The K-Prototypes algorithm can be represented by the following formulas in latex: Given a dataset of N data points, where each data point has p features represented by x¡sub¿ij¡/sub¿ and c clusters, the objective is to minimize the following cost function: $J = \sum_{i=1}^{N} \sum_{j=1}^{p} w_j d(x_{ij}, c_{kj})^2 + \sum_{i=1}^{N} \sum_{j=1}^{p} (1 - w_j) \delta(x_{ij}, c_{kj})$ Advantages of K-Prototypes: Can handle datasets with both numerical and categorical features Can handle large datasets Can converge to a global optimum Disadvantages of K-Prototypes: Can be sensitive to the initialization of cluster centroids May not perform well with

high-dimensional datasets Obtaining the optimal number of clusters can be challenging.

Fuzzy C-Means :

Fuzzy C-Means assigns a degree of membership to each data point for each cluster, indicating the degree to which the data point belongs to that cluster. A fuzzy number between 0 and 1 represents the membership's degree. FCM is based on the minimization of an objective function that represents the sum of squared distances between data points and their assigned centroids weighted by the degree of membership. The objective function is defined as:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} \left(\frac{u_{ij}}{m}\right)^m d_{ij}^2$$

where N is dataset size in terms of the number of points in the dataset, C is the number of clusters, u_{ij} is the degree of membership of data point i to cluster j, m is a fuzzifier parameter that controls the degree of fuzziness, and d_{ij} is the Euclidean distance between data point i and cluster centroid j. FCM is advantageous because it can handle noisy data, and it provides a soft clustering solution that can capture the degree of membership of data points to multiple clusters. However, it can be sensitive to the initialization of cluster centers and the fuzzifier parameter, and it may not perform well when dealing with high-dimensional data or clusters with irregular shapes.

2.5 TIME SERIES FORECASTING

2.5.1 Statistical/ Machine Learning models

ARIMA:

ARIMA (Autoregressive Integrated Moving Average) are time series models used for forecasting. ARMA models are a subset of ARIMA models where the differencing term is set to zero. ARIMA models capture the autoregressive and moving average nature of time series data, while also accounting for trends and seasonality (Ho and Xie (1998). The ARIMA(p, d, q) model consists of three components: the autoregressive (AR)

component of order p, the integrated (I) component of order d, and the moving average (MA) component of order q. The AR(p) component models the relationship between the current observation and the previous p observations. The MA(q) component models the relationship between the current observation and the previous q errors. The I(d) component models the trend by differencing the time series d times. Advantages of ARIMA/ARMA models include their ability to capture trends, seasonality, and autocorrelation in time series data (Kalpakis *et al.* (2001). They are also relatively easy to interpret. But they require a stationary time series, meaning that the mean, variance, and autocorrelation must remain constant over time. Additionally, selecting the optimal values for the model parameters (p, d, q) can be difficult and time-consuming.

SARIMA:

SARIMA(Seasonal ARIMA), is an extended version of the ARIMA model that can account for seasonality in time series data. SARIMA models are used to forecast time series data that exhibit seasonal patterns or periodic fluctuations. The model includes three components: the autoregressive (AR) component, the differencing (I) component, and the moving average (MA) component, as well as a seasonal component (s) that can be added to each of these components. The SARIMA(p,d,q)(P, D, Q)s model is a generalized version of the ARIMA model with additional parameters to control for seasonal variation. The advantages of using SARIMA models include their ability to capture both trend and seasonality in time series data, their flexibility in handling a wide range of time series patterns, and their usefulness in producing reliable forecasts. However, one disadvantage of SARIMA models is that they can be more complex and computationally expensive to estimate compared to simpler time series models(Chen and Guestrin (2016). The formula for SARIMA(p,d,q)(P,D,Q)s is written as:

$$\phi_p(B)(1 - \Phi_P(B^s))\nabla^d\nabla_s^D y_t = \theta_q(B)(1 - \Theta_Q(B^s))\epsilon_t$$

where $\phi_p(B)$ and $\theta_q(B)$ are the autoregressive and moving average polynomials of the ARMA component, $\Phi_P(B^s)$ and $\Theta_Q(B^s)$ are the seasonal autoregressive and moving average polynomials of the SARMA component, d and D are the orders of differencing for the non-seasonal and seasonal components, respectively, s is the period of the seasonality, c is a constant term, y_t is the time series, and ϵ_t is the error term.

Vector Autoregression (VAR):

Vector Autoregression (VAR) is a multivariate time series model used to forecast multiple variables based on their linear relationships. In VAR, each variable is modeled as a linear combination of its own past lags and the lags of other variables in the system. VARX extends VAR to include exogenous variables. VAR models are estimated using maximum likelihood or least squares methods. The order of the VAR model is determined using AIC(Akaike Information Criterion) or BIC (Bayesian Information Criterion). VAR models are advantageous in that they can capture the dynamic interdependencies among variables and allow for the examination of the impact of shocks to one variable on the others. However, they can be computationally demanding and require a large sample size to estimate accurately. The formula for VAR(p) can be expressed as: $y_t = c + \sum_{i=1}^{p} A_i y_{t-i} + \epsilon_t$ where y_t is a vector of variables at time t, c is the intercept term, A_i is a matrix of coefficients for lags i=1,2,...,p, and ϵ_t is the error term. The formula for VARX(p) can be expressed as: $y_t = c + \sum_{i=1}^{p} A_i y_{t-i} + BX_t + \epsilon_t$ where X_t is a matrix of exogenous variables at time t and B is a matrix of coefficients for the exogenous variables. VAR models can be extended to include seasonal lags and exogenous variables, resulting in a seasonal VAR (SAR) or a VARX model, respectively. SAR models can be combined with VARX models to create a seasonal VARX (SARX) model. VAR models are commonly used in macroeconomic forecasting, finance, and social sciences. They are especially useful when the variables in the system are interdependent and influence each other over time. VAR models can also be used for impulse response analysis and forecast error variance decomposition.

Holt-Winters method:

Holt-Winters method, or triple exponential smoothing, is a time series forecasting method used to model and predict time series data with seasonal components. It extends the simple exponential smoothing and double exponential smoothing methods to include a seasonal component. The method estimates three smoothing parameters, namely, the level, trend, and seasonal components, using historical data to generate forecasts. The formula for the Holt-Winters method is expressed as: $\hat{y}t + h = l_t + hb_t + st - m + h^+$ where $\hat{y}t + h$ is the value that is forecasted during the time step $t + h$, l_t is the estimated level at time t, b_t is the estimated trend of the time series at time t, $st - m + h^+$ is the estimated seasonal component at time t, h is the forecast horizon, and m is the length of the seasonal period. Holt-Winters method has the advantage of being able to capture seasonal trends in time series data and provide accurate forecasts for longer time horizons (Kalekar *et al.* (2004). However, it requires a sufficient amount of historical data to estimate the smoothing parameters accurately and may not perform well for time series with irregular patterns or sudden changes. Additionally, the method is computationally intensive and may be slow for large datasets.

2.5.2 Packages

ForeTiS:

ForeTiS (Forecasting Time Series) is an open-source Python library for time series forecasting. It provides a wide range of forecasting methods, including ARIMA, SARIMA, Prophet, Exponential Smoothing, and Random Walk. ForeTiS allows users to easily perform forecasting tasks with minimal code, using a simple and intuitive API (Eiglsperger *et al.* (2023). The library also includes several evaluation metrics and visualizations for model selection and validation. The forecasting methods in ForeTiS are highly customizable, allowing users to define their own models and hyperparameters.

FEDOT:

FEDOT (Fully Automatic Data Scientist) is an open-source Python library for automated

machine learning (Sarafanov *et al.* (2022). It offers a variety of algorithms for regression, classification, clustering, and time series forecasting. FEDOT uses an evolutionary algorithm to search for the best model and hyperparameters for a given dataset. The library allows users to specify various constraints, such as model complexity, execution time, and accuracy requirements. FEDOT also provides a simple and intuitive API for users to interact with. Amazon Forecast is a cloud-based service for time series forecasting. It uses machine learning algorithms such as ARIMA, DeepAR, and Prophet to provide accurate and scalable forecasts. Amazon Forecast allows users to easily import their data, train models, and generate forecasts using a simple and intuitive API. The service also provides several evaluation metrics and visualizations for model selection and validation. Amazon Forecast can be easily integrated with other AWS services such as S3, AWS Lambda, and Amazon SageMaker.

Vertex AI:

Vertex AI is an ML platform powered by the cloud offered by Google Cloud. It provides a wide range of tools for model development, training, and deployment. Vertex AI supports various machine learning algorithms. The platform also includes several automated machine-learning tools such as AutoML Tables, AutoML Vision, and AutoML Natural Language. Vertex AI allows users to easily deploy their models to production using a simple and intuitive API.

Prophet:

Prophet is a forecasting library developed by Facebook (Zhao and Zhang (2020). It is designed for time series forecasting with seasonal patterns. Prophet uses a Bayesian framework to model the trend, seasonality, and holiday effects in the data. The library also includes several advanced features such as trend changepoints, custom seasonalities, and uncertainty estimation. Prophet provides a simple and intuitive API for users to perform forecasting tasks with minimal code.

Advantages:

ForeTiS: highly customizable, intuitive API, supports multiple forecasting methods

FEDOT: automated machine learning, customizable constraints, intuitive API

Amazon Forecast: cloud-based, scalable, integrates with other AWS services

Vertex AI: cloud-based, supports various machine learning algorithms, automated machine learning tools

Prophet: designed for seasonal patterns, Bayesian framework, advanced features

Disadvantages:

ForeTiS: limited documentation, relatively new library

FEDOT: requires knowledge of machine learning concepts, may require significant computation resources

Amazon Forecast: requires a subscription to AWS, limited customization options

Vertex AI: requires a subscription to Google Cloud, limited customization options

Prophet: designed for seasonal patterns, may not perform well on non-seasonal data.

2.5.3 Deep learning models

N-BEATS (ElementAI):

N-BEATS is a deep-learning model designed for time series forecasting. A stack of Fully connected layers is employed to extract features from the input time series, and a gating mechanism to control the flow of information within the network. The output layer predicts future values of the time series. The formula for N-BEATS is not straightforward as it involves multiple fully connected layers with non-linear activations.

Spacetimeformer:

Spacetimeformer is a deep learning model for time series forecasting that uses a transformer-based architecture. It combines spatial and temporal information to model the dependencies between different time series. The model includes a spatiotemporal attention mechanism that enables it to focus its attention on several parts of the input time series. The formula for Spacetimeformer involves multiple attention layers with different types of attention mechanisms.

DeepAR (Amazon):

DeepAR is a deep learning model for time series forecasting that uses a recurrent neural

network (RNN) architecture. It models the time series as a sequence of vectors and uses an autoregressive approach to predict future values (Lim and Zohren (2021). The model includes a prediction network and an encoder network that captures the temporal dependencies of the input time series. The formula for DeepAR involves multiple recurrent layers with different types of activation functions.

Temporal Fusion Transformer or TFT (Google):

TFT is a deep learning model for time series forecasting that uses a transformer-based architecture. It combines multiple input modalities, including time series data and categorical features, to model the dependencies between different inputs. The model includes multiple gating mechanisms and residual connections to improve its performance. The formula for TFT involves multiple attention layers with different types of attention mechanisms.

Time Transformer:

Time Transformer is a deep learning model for time series forecasting that uses a transformer-based architecture. It includes a self-attention mechanism that allows it to attend to different parts of the input time series. The model also includes a causal convolutional layer that enforces the causality of the predictions. The formula for Time Transformer involves multiple attention layers with different types of attention mechanisms.

TCN (Temporal Convolutional Network):

TCN is a deep learning model for time series forecasting that uses a convolutional neural network (CNN) architecture. It includes dilated convolutional layers that capture long-term dependencies in the input time series. The model also includes residual connections and skip connections to improve its performance. The formula for TCN involves multiple convolutional layers with different dilation rates.

WaveNet:

WaveNet is a deep learning model for time series forecasting that uses a dilated causal convolutional neural network (CNN) architecture. It includes dilated convolutional

layers that capture long-term dependencies in the input time series. The model also includes gated activation functions and skip connections to improve its performance. The formula for WaveNet involves multiple convolutional layers with different dilation rates.

Advantages and disadvantages:

N-BEATS: Advantage - efficient and scalable for long time series. Disadvantage - not suitable for short time series or high-frequency data.

Spacetimeformer: Advantage - can handle multiple time series with different temporal resolutions. Disadvantage - requires large amounts of data and computational resources.

DeepAR: Advantage - can handle missing values and irregular time series. Disadvantage - requires careful tuning of hyperparameters and may overfit on small datasets.

TFT: Advantage - can handle multiple input modalities and temporal dependencies. Disadvantage - requires careful preprocessing of the input data and may be sensitive to hyperparameter tuning.

Time Transformer: Advantage - can handle irregular time series and can attend to different parts of the input series. Disadvantage - may not perform well on long-time series.

TCN: Advantage - can handle long time series and is computationally efficient. Disadvantage - may be sensitive to the choice of hyperparameters

2.6 EVALUATION METRICS

There are several metrics available to evaluate the machine learning model employed for the prediction task. This work emphasizes prevailing methods of evaluation metrics to be used for the problem of Churn prediction.

we will look into predominant evaluation metrics based on the confusion matrix.

A confusion matrix is a table where the actual classes are represented by rows and predicted classes are represented by columns. The confusion matrix plays a major role in evaluating models when there is an imbalance in the classes. Since data imbalance is

found most commonly in churn-like data, using confusion matrix-based evaluation metrics is the right thing to be done.

- True Positive (TP): Predicted label: Churners and True label: Churners.
- False Positive (FP): Predicted label: Churners and True label: Non-Churners.
- True Negative (TN): Predicted label: Non-Churners and True label: Churners.
- False Negative (FN): Predicted label: Non-Churners and True label: Non-Churners.

The evaluation metrics based on the confusion matrix are as follows:

- **Accuracy**:
  - Accuracy is the percentage of correct predictions made by the model.
  - Accuracy = (Num of correct predictions) / (Total num of predictions)
  - $\implies$ Accuracy = $\frac{\text{TP+TN}}{\text{TP+FP+TN+FN}}$
  - Range: 0 to 1.
  - Even though accuracy reflects the overall performance, accuracy may tend to behave towards the majority class and thus cannot be taken as the optimum measure for churn-like datasets.
- **Precision**:
  - Precision is the percentage of positive predictions that are actually correct made by the model.
  - Precision = (Num of correct positive predictions) / (Total num of positive predictions)
  - $\implies$ Precision = $\frac{\text{TP}}{\text{TP+FP}}$
  - Precision reflects the positive predictions made by the model and thus it is considered to be a very useful evaluation metric.
  - Range: 0 to 1.

- High precision value signifies that the model is able to predict the positive classes very well and low precision value signifies that the model is susceptible to making false positive predictions more.

- **Recall**:
  - Recall is the percentage of correct positive predictions made by the model.
  - Recall = (Num of correct positive predictions) / (Total num of actual positives)
  - $\implies$ Recall = $\frac{\text{TP}}{\text{TP+FN}}$
  - Recall reflects how much the model is able to correctly identify the positives out of all actual positives available.
  - Range: 0 to 1.
  - High Recall value signifies how much the model is able to predict the positives very well and low recall value signifies that the model is susceptible to making false negative predictions more.

- **F1 Score**:
  - F1 score is a single measure that consists of both precision and recall
  - F1 score is the harmonic mean of precision and recall.
  - $\implies$ F1 Score = $2 * \frac{\text{Precision * Recall}}{\text{Precision + Recall}}$
  - Since the F1 score is sensitive to imbalanced data, places where measures like accuracy are not able to evaluate the model appropriately, the F1 score can give us more information on the model's performance.
  - Range: 0 to 1.
  - High F1 score value signifies that the model is really performing well and low F1 score value signifies that the model is performing poorly.

- **Matthews correlation coefficient (MCC)**:
  - Matthews correlation coefficient (MCC) is a metric that can be used to identify the quality of a binary classification and evaluate the model.
  - It considers true positives, true negatives, false positives, and false negatives and produces a score between -1 and 1.

- An MCC of 1 represents a perfect prediction, 0 represents a random prediction, and -1 represents a perfectly incorrect prediction. MCC is regarded as a fair and balanced measure, meaning it can be used even if the classes are of different sizes.
- $\implies \text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$
- Range: 0 to 1.
- Advantages of using MCC include its ability to handle imbalanced classes and its ability to be used with binary classification problems. It can also be a helpful metric when assessing a model's performance in circumstances when the costs of false positives and false negatives vary (Chicco *et al.* (2021).
- Disadvantages of using MCC include its sensitivity to the prevalence of the class in the data and its inability to handle multi-class classification problems.

- **AUC (Area Under the ROC Curve)**:
  - AUC is an estimate of the likelihood of classifying the positive point more appropriately it could have been done on a negative point when a positive and a negative point chosen randomly are considered. AUC is an estimate of the likelihood by which the model classifies a randomly chosen positive point more than the randomly chosen negative point.
  - The AUC measure is calculated on the TPR vs FPR plot. TPR and FPR are True positive rates or sensitivity of the model and False positive rates (number of predictions predicted as positive classed wrongly by the class over total actual negative points) respectively.
  - The curve obtained curve is called the receiver operating characteristic (ROC) curve. The area constituted under the ROC curve is called AUC.
  - If the ROC curve is near the top right position it means the model is robust and the AUC value will be high. Similarly, when the ROC curve is flat to the diagonal, it indicates that the model is not good and results in a low AUC value.
  - Range: 0 to 1
  - High AUC score means the model is robust and performing good ans a low AUC score refers to the fact that the model is not performing that much good.

The churn prediction task is approached from various perspectives posing the problem as a supervised problem (classification and Regression) and an unsupervised

problem(Clustering). So there are many evaluation metrics that are used to evaluate the regression models and Clustering Algorithms.

Mean Absolute Error (MAE):

Mean Absolute Error (MAE) is a frequently used evaluation metric for regression tasks. It measures the average magnitude of errors between the predicted values and the true values. Specifically, MAE is calculated as the mean of the absolute differences between the ground truth and predicted values.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

where n corresponds to the number of data samples, y_i is the true value, and $\hat{y}_i$ is the predicted value. The advantages of using MAE are that it is easy to understand and interpret since it represents the average error in the same units as the target variable. Additionally, it is less sensitive to outliers compared to other evaluation metrics such as Mean Squared Error (MSE). The disadvantages of using MAE are that it doesn't differentiate between overestimation and underestimation of errors, and it can be heavily influenced by the presence of extreme values in the dataset.

Mean Absolute Percentage Error (MAPE):

Mean Absolute Percentage Error (MAPE) is a frequently used evaluation metric for forecasting models, particularly in the domain of economics and finance. It measures the percentage difference between actual and predicted values. MAPE is calculated as the mean of the absolute percentage errors, which are the absolute differences between actual and predicted values divided by actual values, multiplied by 100. The formula for MAPE can be written as:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{A_i - F_i}{A_i}\right| \times 100$$

where n signifies the total number of data points, A_i is the actual value, F_i is the forecasted value. The advantages of using MAPE include its simplicity and interpretability. It is

also easy to understand for non-technical stakeholders, as it is expressed in percentage terms. However, one disadvantage of using MAPE is that it can be sensitive to extreme values, leading to a larger impact on the overall score. Additionally, it can produce misleading results when the actual values are close to zero or very small. Finally, it does not account for the direction of the error, which can be a limitation in some applications.

Mean Squared Error (MSE):

Mean Squared Error (MSE) is a widely utilized metric for the evaluation of a regression model's performance. It is the mean of the squared values of the differences between the ground truth values and the predicted values. The formula for MSE is given as:

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

where n is the number of observations, y_i is the actual value, and $\hat{y}_i$ is the predicted value. MSE has the advantage of penalizing larger errors more heavily than smaller errors. It is also useful when outliers are present in the data as it is less sensitive to outliers than other metrics like Mean Absolute Error (MAE). However, MSE is sensitive to the scale of the data and can produce large errors when the predicted and actual values are far apart. In summary, MSE is a popular metric for evaluating regression models, but it must be utilized along with several other metrics for a complete picture of model performance.

Root Mean Squared Error (RMSE):

Root Mean Squared Error (RMSE) is another commonly used evaluation metric for regression models, which measures the square root value of the average squared difference between the predicted and actual values. The RMSE penalizes large errors more than the Mean Absolute Error, as the errors are squared before being averaged. RMSE is expressed in the same units as the predicted and actual values and its values range between 0 and infinity. The formula for RMSE is:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

where the number of data points is n, y_i is the actual value of the i-th sample, and $\hat{y}_i$ is the predicted value of the i-th sample. Advantages of using RMSE as an evaluation metric include its sensitivity to large errors and its ability to highlight the variance of errors. Disadvantages of using RMSE include the fact that it may be significantly impacted by outliers and that it may not be the best metric for models where the errors follow a non-normal distribution.

Akaike information criterion:

The AIC or Akaike information criterion is a metric of the quality of a statistical model in terms of its ability to explain the data while penalizing for the number of parameters used. It is commonly used in model selection to choose between competing models. AIC is based on the maximum likelihood estimation of the parameters in the model and adjusts the likelihood for the number of parameters used, thereby preventing overfitting. AIC being in a lower value signifies a robust fit of the model. The formula for AIC is: AIC = -2 ln(L) + 2k where L is the likelihood function's maximum value of the model, and k is the number of parameters in the model.

Advantages of AIC include its ability to compare models with different numbers of parameters and its bias towards parsimony. Disadvantages include its reliance on maximum likelihood estimation and the fact that it may not perform well in small sample sizes. Additionally, AIC assumes that the model is correctly specified, and its use in non-linear models can be complicated.

R-Squared:

The coefficient of determination, or R-Squared, is a statistical indicator that shows how much of the variance in the dependent variable can be accounted for by the independent variable or variables. To assess a model's goodness of fit, regression analysis frequently uses it. The formula for R-squared is given as: $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$ where the sum of squared residuals is the SS_{res}, which measures the difference between the actual ground truth and the dependent variable's predicted values, and the total sum of squares is SS_{tot}, which

measures the difference between the actual values of the dependent variable and their mean. R-squared ranges from 0 to 1, with higher values indicating a better fit of the model. An R-squared value of 1 means that all the variance in the dependent variable is explained by the independent variable(s). Advantages of R-squared include its simplicity and ease of interpretation. It provides a standardized measure of the goodness of fit, making it easy to compare different models. However, its main disadvantage is that it does not indicate whether the model is correctly specified, and it can be sensitive to outliers and influential data points. It is therefore important to use other diagnostic measures to assess the validity of the model.

Normalized Root Mean Squared Error (NRMSE):

Normalized Root Mean Squared Error (NRMSE) is a widely used evaluation metric in time series forecasting. It is a variation of the Root Mean Squared Error (RMSE) metric that normalizes the error by dividing it by the range of the target variable. This makes it possible to compare the accuracy of models that are trained on different datasets. The NRMSE formula is:

$$NRMSE = \frac{RMSE}{y_{max} - y_{min}}$$

where Root Mean Squared Error is $RMSE$ and y_{max} and y_{min} are the maximum and minimum values of the target variable, respectively. The ability to compare the performance of models trained on various datasets is one of NRMSE's advantages. Disadvantages include its sensitivity to outliers and not taking the magnitude of the error into consideration, which may be more important in some applications.

WAPE and WMAPE:

Weighted Absolute Percentage Error (WAPE) and Weighted Mean Absolute Percentage Error (WMAPE) are two additional evaluation metrics that are used in forecasting. WAPE is similar to MAPE, but takes into account the weight of each observation, while WMAPE calculates the average of the absolute percentage errors, weighted by the actual

values. The WAPE formula is:

$$WAPE = \frac{\sum_{t=1}^{T} w_t |y_t - \hat{y}_t|}{\sum_{t=1}^{T} w_t y_t}$$

where w_t is the weight of the tth observation. The WMAPE formula is:

$$WMAPE = \frac{\sum_{t=1}^{T} w_t |\frac{y_t - \hat{y}_t}{y_t}|}{\sum_{t=1}^{T} w_t}$$

where w_t is the weight of the tth observation. Advantages of WAPE and WMAPE include their ability to consider the weight of each observation, which may be important in some applications. Disadvantages include their sensitivity to outliers and the fact that they may be difficult to interpret, especially in cases where the weights are not readily available.

Silhouette Score:

The Silhouette Score is a metric used to evaluate the quality of a clustering algorithm's output. It measures the degree of similarity between a data point and its own cluster compared to other clusters. The score ranges from -1 to 1, with values closer to 1 indicating well-clustered data and values closer to -1 indicating poorly-clustered data. The formula for the Silhouette Score of a data point i is:

$$s(i) = \frac{b(i) - a(i)}{\max a(i), b(i)}$$

where $a(i)$ represents the average distance between point i and all other data points in the same cluster as i and $b(i)$ is the smallest average distance between point i and any other cluster that i is not a member of. The overall Silhouette Score for clustering is the mean of the scores for all data points. Advantages of the Silhouette Score include its ability to handle arbitrary-shaped clusters and its interpretation as a measure of cluster cohesion and separation. However, the curse of dimensionality and its dependence on the choice of distance metric may reduce its effectiveness.

Calinski-Harabasz index:

The Calinski-Harabasz index is a clustering evaluation metric that assesses the quality

of clusters based on their variance and separation. The index measures the ratio of the between-cluster variance to the within-cluster variance. A higher Calinski-Harabasz score indicates better-defined clusters. The formula for the Calinski-Harabasz index is given:

$$CH(k) = \frac{Tr(B_k)}{Tr(W_k)} \frac{n-k}{k-1}$$

where the number of clusters is k, the total number of data points is n, Tr(B_k) is the trace of the between-cluster scatter matrix, and Tr(W_k) is the trace of the within-cluster scatter matrix. The advantage of the Calinski-Harabasz index is that it is easy to calculate and computationally efficient. It also does not require knowledge of the true labels or ground truth. However, it can be sensitive to the scale of the data and can produce misleading results for non-globular clusters.

Davies-Bouldin index:

The Davies-Bouldin index is an evaluation metric used in clustering that measures the average similarity between clusters and their separation. It calculates the ratio between the sum of the within-cluster distances and the distance between cluster centers. The number of clusters that minimizes this index is the optimal number. The formula for the Davies-Bouldin index is:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right)$$

where k is the number of clusters, σ_i is the average distance between each point in cluster i and its centroid, and the distance between centroids of clusters i and j is d(c_i,c_j). The smaller the Davies-Bouldin index value, the better the clustering. The advantage of the Davies-Bouldin index is that it takes into account both the compactness and separation of clusters. It is also relatively easy to compute and can handle non-convex clusters. However, it may not work well when the clusters have different densities, and it can be sensitive to the number of clusters and the initial seeds.

CHAPTER 3

RESULTS AND ANALYSIS

3.1 RESEARCH WORK

As a part of this work, multiple research papers and journals on the topic of churn prediction and Time Series Forecasting were read and reviewed. About 80 research papers from various journals on the topic of Churn prediction and Time Series Forecasting were compiled and analyzed comprehensively for this work. Along with that, multiple sources of publicly available datasets of different domains, Notebooks, code repositories, relevant articles, Tools, and APIs are also collected. All these pieces of information are documented as an Excel sheet file for future reference for both tasks. Various datasets from the compiled list representing various domains are selected, numerous supervised Machine learning models have been implemented over them, and the observations of the experimentation are documented for further analysis. These algorithms are compared based on various decision points such as the type of data they can handle, training time, the computational complexity of the model, interpretability, robustness, etc. Based on the results obtained, conclusions have been proposed regarding the usage of various algorithms based on various situations and business needs. Flowcharts are constructed for both tasks which can act as the referential guide for someone to choose the optimal algorithm for their needs and situation.

3.2 EXPERIMENTATION

3.2.1 Training Pipeline

A robust pipeline for the tasks of Churn prediction and Time series Forecating has been discussed in this section of the work. The pipeline is as follows:

Dataset Preprocessing and feature engineering :

- Load the required dataset (Churn prediction or Time series dataset) into a data frame

- Handling missing and NULL values

  - Preparing the dataset properly is the most crucial segment in training a machine learning model. Cleaning the data plays a major role as many missing values and NULL values may be found in the dataset which may prevent the model from achieving great results.

- Discarding irrelevant features

  - Since the dataset may contain many features that are irrelevant or not so important and have no effect on the final prediction, those features can be discarded.

  - This can be achieved by finding the correlation between the feature in consideration and the target feature. When a feature is really important and has an impact on the final target feature, the correlation will be very high(1.0), whereas a feature that has no impact on the target variable and any change in these features doesn't change the target variable and doesn't contribute anything to the final target feature. These features will have very low correlation values (near 0) and thus using these methods we can discard the irrelevant features from the dataset.

- Encoding categorical features

  - The features need not always be numerical in nature. It may be categorical too which cannot be used by the algorithm for prediction as it is. In such cases, we need to convert the categorical data into one hot encoding so that it can be used by the algorithm for prediction.

- Converting non-numerical data into numerical data

  - Many features may have string values and they need to be handled before feeding them into the model.

  - Eg: Gender(Male, Female, Others), Churn (True, False), Senior citizen(True, False) these features are non-numeric in nature and we need to convert them into numerical data before proceeding with the next steps

Train-test split:

Train test split is a method of judging the performance of a machine learning model. This gives an estimate of the performance of the algorithm without any bias. This helps in the prevention of overfitting and helps in the appropriate evaluation of the algorithm.

Various types of research have been conducted to conclude what proportion of train test split is ideal for datasets. Even Though this may vary based on situations, mostly the 80-20 train-test split performs very well over a majority of applications, and thus, we will use the 80-20 train-split in this pipeline.

Selecting Algorithm and Training the model:

Various machine learning algorithms which could be used for churn prediction have been discussed in the background. During the experimentation, most of the algorithms have been implemented for the datasets taken into consideration and have been trained upon them.

All these algorithms contain various parameters that can be tweaked to achieve better results and thus, selecting good values for these parameters results in better learning of the algorithm. Grid search is one of the ways of selecting the best parameters for the algorithms. Grid search works by trying out different values for the parameters and choosing the value among them that results in the best results.

Evaluation of the model and Prediction:

Once the training stage has been completed, the estimation of the performance of the model is done over various evaluation metrics such as Accuracy, precision, recall, F1 score, MCC and Area under the ROC curve for the Classification tasks (Churn prediction posed as classification), MSE, RMSE, and R-squared for the Regression tasks (Churn prediction posed as Regression), Silhouette Score, Calinski-Harabasz index and Davies-Bouldin index for Clustering methods (Churn Prediction posed as Unsupervised problem), Mean Absolute Error, Mean Absolute Percentage Error, Mean Squared Error, Root Mean Squared Error, Akaike information criterion, R-Squared, Normalized Root Mean Squared Error, Weighted Absolute Percentage Error, and Weighted Mean Absolute Percentage Error for Time series Forecasting. The models are saved and then can be used to do predictions on never seen data and this process will yield a good evaluation of the algorithm.

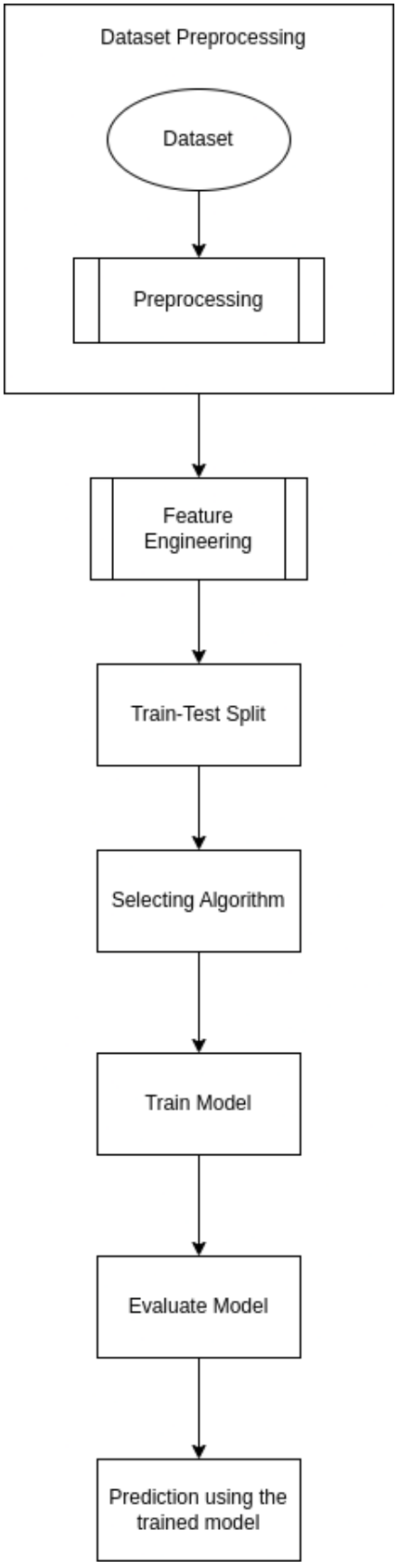


Figure 3.1: Training Pipeline followed

3.2.2 Datasets

Churn Prediction

Many datasets were collected as part of this work, and the links to the dataset are listed in the documentation sheet. Among all the Churn prediction datasets collected as part of this work, the following datasets have been selected and various machine learning algorithms have been implemented to understand and analyze the working of various machine learning techniques for the problem of Churn prediction.

Datasets		
Dataset Name	Num of data points	Num of features
Telecom customer churn dataset	7043	21
Bank Customer Churn Prediction	10000	14
IBM HR Analytics Employee Attrition Performance	1470	35
Orange Telecom Prevention and Predicting Churn	2666	20
Ecommerce Customer Churn Analysis and Prediction	5630	20

Telecom customer churn dataset:

- 7043 unique data points and 21 features
- This is a standard sample dataset provided by IBM
- It consists of information about customers who left within the last month and it is denoted by the column churn
- Information about the services subscribed by the customers is also found in this dataset
- Account information like what method of payment they prefer, the amount paid every month, etc are also found here
- Personal data like age, gender, etc are also found in this dataset.
- Data is preprocessed as follows
  - "CustomerID" features are ignored.

– Categorizing the Senior Citizen column for encoding as they have boolean values (true and false) Converting them into 1 and 0 is necessary.

Bank Customer Churn Prediction:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	I
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No	No	Month-to-month	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No	No	One year	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	No	Month-to-month	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No	No	One year	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No	No	Month-to-month	

5 rows × 21 columns

Figure 3.2: Dataset: Telecom customer churn

- 10000 unique data points and 14 features
- This dataset consists of details about the bank customer details over the region of France, Spain, and Germany
- Data is preprocessed as follows
  - "Row Number" and "CustomerID" features are ignored.
  - The categorical features "Geography", "Gender" are converted into one hot encoding.
  - Min-max scaling the categorical variables.

IBM HR Analytics Employee Attrition Performance:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0

Figure 3.3: Dataset: Bank Customer Churn Prediction

- 1470 unique data points and 35 features
- This imaginary dataset fabricated by the scientists at IBM consists of data about education details, job involvement, environment, job satisfaction, performance, etc of employees and the goal is to find whether an employee will churn or not.

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	2
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	3
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	4

Figure 3.4: Dataset: IBM HR Analytics Employee Attrition & Performance

Orange Telecom Prevention and Predicting Churn:

- 2666 unique data points and 20 features
- Orange telecom dataset consists of data that are already cleaned. This customer activity dataset
- Since the dataset has been already cleaned and made available we need not perform any data cleaning process or data preprocessing steps

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	Total intl minutes	Total intl calls	Total intl charge	Customer service calls	Churn
0	LA	117	408	No	No	0	184.5	97	31.37	351.6	80	29.89	215.8	90	9.71	8.7	4	2.35	1	False
1	IN	65	415	No	No	0	129.1	137	21.95	228.5	83	19.42	208.8	111	9.40	12.7	6	3.43	4	True
2	NY	161	415	No	No	0	332.9	67	56.59	317.8	97	27.01	160.6	128	7.23	5.4	9	1.46	4	True
3	SC	111	415	No	No	0	110.4	103	18.77	137.3	102	11.67	189.6	105	8.53	7.7	6	2.08	2	False
4	HI	49	510	No	No	0	119.3	117	20.28	215.1	109	18.28	178.7	90	8.04	11.1	1	3.00	1	False

Figure 3.5: Dataset: Orange Telecom Prevention and Predicting Churn

Ecommerce Customer Churn Analysis and Prediction:

- 5630 unique data points and 20 features
- This dataset contains details of customers of an online retail e-commerce company

	CustomerID	Churn	Tenure	PreferredLoginDevice	CityTier	WarehouseToHome	PreferredPaymentMode	Gender	HourS
0	50001	1	4.0	Mobile Phone	3	6.0	Debit Card	Female	3.0
1	50002	1	NaN	Phone	1	8.0	UPI	Male	3.0
2	50003	1	NaN	Phone	1	30.0	Debit Card	Male	2.0
3	50004	1	0.0	Phone	3	15.0	Debit Card	Male	2.0
4	50005	1	0.0	Phone	1	12.0	CC	Male	NaN

Figure 3.6: Dataset: Ecommerce Customer Churn Analysis and Prediction

Time Series Forecasting

Several datasets were collected as part of this work, and the links to the dataset are listed in the documentation sheet, and also the dataset is downloaded and made available in the **GitHub**. Three categories of Time series data have been collected for analysis namely Sales forecasting datasets, Demand forecasting datasets, and Price forecasting datasets Among all the Time series datasets collected few of them are selected and several algorithms have been run upon them.

Datasets		
Dataset Name	Num of data points	Num of features
SBIN (State Bank of India) Stock Price	252552	6
Rossmann Store Sales	1058209	9
Walmart Recruiting - Store Sales	536634	5
Petrol Price	830	2
Gold Price	10788	2
Aviation / Flight Fare	452088	13

SBIN (State Bank of India) Stock Price Dataset:

- For this work, the stock prices of State Bank of India from NSE India have been downloaded from 1^{st} January 2000 to 10^{th} April 2023.
- 252552 data points and 6 features
- The closing price of the stock is being forecasted
- Various models have been implemented for this dataset, and it has been evaluated based on many evaluation metrics.

Date	Open	High	Low	Close	Adj Close	Volume
2000-01-03	22.267091751098633	22.9888858795166	22.101974487304688	22.9888858795166	16.4235782623291	25152894
2000-01-04	22.9888858795166	24.720245361328125	22.535995483398438	24.446624755859375	17.465002059936523	47648560
2000-01-05	23.493667602539062	24.97499656677246	23.116260528564453	23.441774368286133	16.747129440307617	36396207
2000-01-06	23.776725769042997	25.286357879638672	23.776725769042997	24.625892639160156	17.593080520629883	70573968
2000-01-07	24.673070907592773	26.409147262573242	24.05978012084961	25.786424636840820	18.422178268432617	83453217

Figure 3.7: Dataset: SBIN (State Bank of India) Stock Price Dataset

Rossmann Store Sales Dataset:

- The data consists of sales information about 1115 Rossmann stores.
- 10,17,209 training data points and 41,000 test data points and 1115 store information, and 9 different features.

Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHolid...	SchoolHoli...
1	5	2015-07-31	5263	555	1	1	0	1
2	5	2015-07-31	6064	625	1	1	0	1
3	5	2015-07-31	8314	821	1	1	0	1
4	5	2015-07-31	13995	1498	1	1	0	1
5	5	2015-07-31	4822	559	1	1	0	1

Figure 3.8: Dataset: Rossmann Store Sales Dataset

Walmart Recruiting - Store Sales Dataset:

- Historical sales data of 45 Walmart stores for 5^{th} Feb 2010 to 1^{st} Nov 2012.
- 4,21,570 test data points, 1,15,064 test points and 5 features

	Store	Dept	Date	Weekly_Sales	IsHoliday	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
0	1	1	2010-02-05	24924.50	False	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
1	1	2	2010-02-05	50605.27	False	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
2	1	3	2010-02-05	13740.12	False	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
3	1	4	2010-02-05	39954.04	False	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
4	1	5	2010-02-05	32229.38	False	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106

Figure 3.9: Dataset: Walmart Recruiting - Store Sales Dataset

Petrol Price Dataset:

- Dataset consists of Petrol price data in USD.
- 814 train data points and 16 test data points.
- Univariate

Date	Petrol (USD)
6/9/2003	74.59
6/16/2003	74.47
6/23/2003	74.42
6/30/2003	74.35
7/7/2003	74.28

Figure 3.10: Dataset: Petrol price Dataset

Gold Price Dataset:

- Publicly available dataset available on Quandl, where datasets regarding financial and economic nature are available.
- Dataset consists of Gold price rates from January 1970 to March 2020.
- Consists of 10788 data points.
- Univariate

Date	Value
1970-01-01	35.2
1970-04-01	35.1
1970-07-01	35.4
1970-10-01	36.2
1971-01-01	37.4

Figure 3.11: Dataset: Gold price Dataset

Aviation / Flight Fare Dataset:

- Scraped data collected from EaseMyTrip.com
- 452088 datapoints and 13 features.

Date_of_jo...	Journey_day	Airline	Flight_code	Class	Source	Departure	Total_stops	Arrival	Destination	Duration_i...
2023-01-16	Monday	SpiceJet	SG-8169	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.0833
2023-01-16	Monday	Indigo	6E-2519	Economy	Delhi	After 6 PM	non-stop	Before 6 AM	Mumbai	2.3333
2023-01-16	Monday	GO FIRST	G8-354	Economy	Delhi	After 6 PM	non-stop	Before 6 AM	Mumbai	2.1667
2023-01-16	Monday	SpiceJet	SG-8709	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.0833
2023-01-16	Monday	Air India	AI-805	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.1667

Figure 3.12: Dataset: Flight fare Dataset

3.2.3 Performance of the models on the datasets

3.2.4 Churn Prediction as Supervised Problem (Classification)

Table 3.1: Performance on Telecom customer churn dataset

Algorithms	Metrics				
	Accuracy	Precision	Recall	F1 Score	MCC
Support Vector Classifier	0.81	0.94	0.83	0.88	0.48
Random Forest	0.823	0.9376	0.8415	0.8870	0.51
Logistic Regression	0.82	0.912	0.854	0.88	0.50
Decision Tree	0.73	0.81	0.82	0.81	0.17
KNN	0.76	0.85	0.83	0.84	0.37
LightGBM	0.81	0.90	0.84	0.87	0.47
XGBoost	0.80	0.89	0.84	0.87	0.46

Table 3.2: Performance on Bank Customer Churn Prediction

Algorithms	Metrics				
	Accuracy	Precision	Recall	F1 Score	MCC
Logistic regression with degree 2 pol kernel	0.8604	0.9652	0.8725	0.91658	0.521
SVM with RBF Kernel	0.8619	0.9817	0.8631	0.9186	0.522
SVM with pol kernel	0.8569	0.9813	0.8587	0.9159	0.50
Random forest classifier	0.8864	0.9827	0.8865	0.9321	0.61
XGBoost	0.8799	0.9712	0.8880	0.9277	0.59
KNN	0.86	0.96	0.87	0.92	0.54
LightGBM	0.92	0.98	0.92	0.95	0.74

Table 3.3: Performance on IBM HR Analytics Employee Attrition & Performance

Algorithms	Metrics				
	Accuracy	Precision	Recall	F1 Score	MCC
Logistic regression	0.8435	0.9788	0.8493	0.9094	0.31
Random forest	0.8197	0.9873	0.8233	0.8979	0.37
Gradient Boosting Classifier	0.8333	0.9619	0.8502	0.9026	0.41
XGboost	0.82	0.96	0.84	0.90	0.35
LightGBM	0.83	0.97	0.84	0.90	0.36
KNN	0.80	0.98	0.81	0.89	0.17
SVC	0.82	0.98	0.82	0.89	0.28

Table 3.4: Performance on Orange Telecom Prevention and Predicting Churn

Algorithms	Metrics				
	Accuracy	Precision	Recall	F1 Score	MCC
Logistic regression(Baseline_model)	0.8503	0.8924	0.9463	0.9186	0.02
Logistic Regression(SMOTE)	0.7605	0.9431	0.7785	0.8529	0.27
Decision Tree	0.8024	0.9328	0.8389	0.8834	0.26
Random forest classifier	0.9162	0.9245	0.9866	0.9545	0.46
SVM classifier linear	0.7605	0.9504	0.7718	0.8519	0.30
XGBoost classifier	0.9281	0.9536	0.9664	0.9600	0.60
KNN	0.68	0.92	0.69	0.79	0.16
LightGBM	0.93	0.95	0.97	0.96	0.63

Table 3.5: Performance on Ecommerce Customer Churn Analysis and Prediction

Algorithms	Metrics				
	Accuracy	Precision	Recall	F1 Score	MCC
Random Forest	0.9742	0.9989	0.9711	0.9848	0.904
Logistic Regression	0.8996	0.9724	0.9132	0.9418	0.589
SVM	0.83	1.0000	0.8357	0.9105	
KNN	0.87	0.95	0.90	0.92	0.49
Decision tree	0.96	0.97	0.98	0.97	0.86
XGboost	0.97	0.99	0.98	0.98	0.91
LightGBM	0.96	0.99	0.97	0.98	0.88

3.2.5 Churn Prediction as Supervised Problem (Regression)

As we discussed, Churn prediction is the task of identifying whether a customer will churn or not. Even though it sounds more like just a binary classification problem where we can only fit a data point into either one of the two possible outcomes, Churn prediction can also be posed as a Regression problem. Here an attempt has been made to pose this problem as a Regression problem as there are many advantages involved with solving this as a regression problem as follows. Regression models can handle huge amounts of data and features. Usually, Churn Prediction datasets are huge with lots and lots of features. The training and prediction pipeline we discussed earlier applies to this problem too. Instead of having the values of the target variable as just 1 (Churner) or 0 (Non-Churner), here we will have some real values and try to predict them.

The data is prepared by using a highly robust classification algorithm (XGBoost) to predict the probability values of whether the customer is churner or not and not thresholding them as 0 or 1. Thus, now we have a churn Prediction dataset with real target values. This dataset is fed into several Regression algorithms, and the results are evaluated. Here we try to correlate the probability values with the likelihood that a customer will churn. So in this way, we have a probability value representing the churning likelihood of customers, which allows businesses to focus more on retaining strategies on highly risky customers. The results are tabulated in table 3.6.

Table 3.6: Performance on Telecom customer Churn Regression dataset

Algorithms	Metrics		
	Mean Squared Error	Root Mean Squared Error	R-squared
Linear Regression	0.13	0.37	0.29
Multiple Linear Regression	0.137	0.3704	0.295
Ridge Regression	0.137	0.375	0.295
Lasso Regression	0.168	0.41	0.136
Elastic Regression	0.1389	0.372	0.286
SVR with linear kernel	0.138	0.372	0.289
SVR with RBF kernel	0.1388	0.3726	0.286
SGD Regression	0.1375	0.37	0.2933
Polynomial Regression	0.131	0.362	0.323
XGBoost Regressor	0.128	0.358	0.339
Random Forest Regressor	0.1299	0.3605	0.3322
Gradient Boosting Regression	0.127	0.356	0.345

3.2.6 Churn Prediction as Unsupervised Problem (Clustering)

Churn prediction is typically considered a supervised learning problem, as it involves predicting whether a customer is likely to leave a company or not based on historical data. However, it is possible to approach churn prediction as an unsupervised learning problem as well. In an unsupervised learning approach, the goal is to find out any patterns and anomalies in the data without using any labeled examples. This can be done using techniques such as clustering, anomaly detection, and dimensionality reduction. Anomaly detection algorithms can also be used to identify customers who exhibit unusual behavior or patterns that are not typical for the rest of the customer base. Dimensionality reduction techniques, such as principal component analysis (PCA) or t-SNE, can be used to reduce the dimensionality of the data and identify the most important features that contribute to churn. This can help to identify the key factors that are driving customer churn and inform strategies for reducing it. While unsupervised learning approaches can be useful for identifying customer data patterns, they have limitations. For instance, they lack the ability to predict individual customer churn accurately and may not be able to account for external factors that could influence churn (such as changes in the competitive landscape).

The pipeline followed is as follows:

1) **Data preparation:** The data must be prepared for clustering, an unsupervised approach. So the target variable column is removed from the dataset, and now we have an unlabelled dataset. Now the standard cleaning procedures like handling missing values and NaN values, removing outliers, encoding categorical features, and removing features that are not relevant.

2) **Feature Scaling:** The dataset features are scaled to ensure that they are similar in scale by normalization.

3) **Cluster Analysis:** As the dataset is prepared for the models, the clustering algorithms are now used on the dataset to group similar customers into clusters based on customer information.

4) **Cluster Evaluation:** Once clustering is done, the quality of the clusters is evaluated using the silhouette score, Davies-Bouldin index, and Calinski-Harabasz index.

5) **Cluster Interpretation:** The clusters with the most churners are found once the clusters are formed. Then the customers in those clusters are analyzed for the features among the customers that act as the main reason behind the churning.

6) **Insights:** Once the clusters with more churners are analyzed for any distinguishable features or reasons, the retention actions are to be strategized.

Table 3.7: Performance on Telecom customer Churn Clustering dataset

Algorithms	Metrics		
	Silhouette Score	Calinski-Harabasz Index	Davies-Bouldin Index
K-Means	0.1494	1255.8	2.13
MiniBatch K-means	0.1482	1254.6	2.1309
Gaussian Mixture Model	0.1245	998.88	2.5309
DBSCAN	-0.1792	43.6	1.292
Fuzzy C-Means	0.115	760.08	2.322
Spectral Clustering	0.1015	291.2	2.06

3.2.7 Performance of Time Series Forecasting models

SBIN (State Bank of India) Stock Price Forecasting:

Table 3.8: Time series forecasting - Performance on SBIN Dataset

Algorithms	Metrics					
	MAE	MSE	RMSE	AIC	R-Squared	NRMSE
ARIMA	222.094	56621.85	237.95	-39132.89	-6.7588	0.6285
SARIMA	220.89	56084.069	236.82	30020.96	-6.6851	0.6255
Holt-Winters Method	209.668	50230.350	224.121	15228.676	-5.883	0.592
VAR	114.45	18192.861	134.88	6.1352	-1.4795	0.3563
Prophet	124.51	22493.169	149.977	39233.951	-0.7279	0.3339

Rossmann Store Sales Forecasting:

Table 3.9: Time series forecasting - Performance on Rossmann Store Sales Dataset

Algorithms	Metrics					
	MAE	MSE	RMSE	AIC	R-Squared	NRMSE
ARIMA	1479.69	3062042.93	1749.86	1847.50	-44.416	0.245498
SARIMA	598.59	546211.40	739.061	1806.29	0.413	0.124
Random forest	624.023	950343.46	974.855	2324584	0.900	0.025
XGBoost	576.669	708830.51	841.920	2275067	0.925	0.021
Prophet	5927522	48566444724282	6968962	949.418	0.129	0.238

Walmart Recruiting - Store Sales Forecasting:

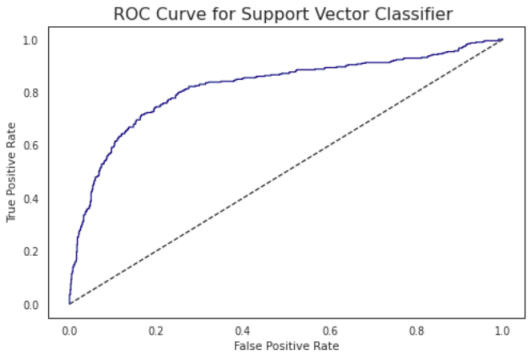
Table 3.10: Time series forecasting - Performance on Walmart Recruiting - Store Sales Dataset

Algorithms	Metrics					
	MAE	MSE	RMSE	AIC	R-Squared	NRMSE
SARIMA	66979.98	6941994737.5	83318.63	1301.23	0.474	0.052
Random forest	55337.90	5593031820	74786.57	1301.23	0.567	0.047
Decision tree	870.39	1079177.55	1038.8347	156.917	-0.406	1.186
KNN	553.35	668393.64	817.553	168.126	0.128	0.223
Linear	7132.51	93476390	9668.31	217.53	-120.859	2.646

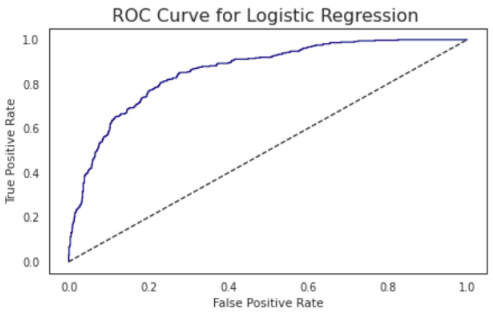
Gold price Forecasting:

Table 3.11: Time series forecasting - Performance on gold price Dataset

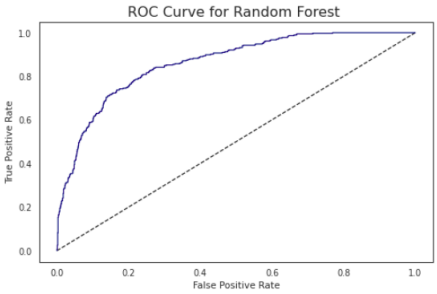
Algorithms	Metrics					
	MAE	MSE	RMSE	AIC	R-Squared	NRMSE
ARIMA	75.012	9238.05	96.114	98503.026	-1.2122	0.4192
Auto ARIMA	1542.380	2383123.91	68.0573	75000	-569.66	0.47
Prophet	359.75	137141.68	370.326	414.863	-83.398	2.841
Holt winter	28.2950	1083.9621	32.9236	47402.5832	0.3329	0.50



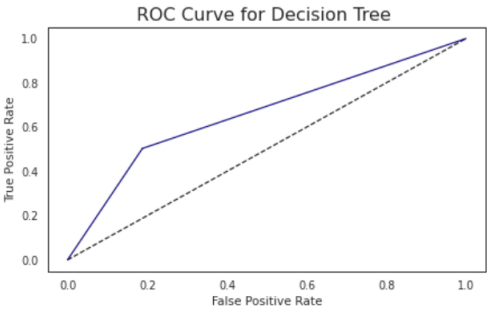
(a) SVC



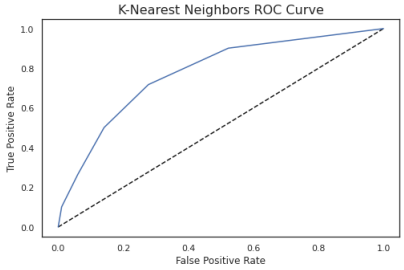
(b) Logistic Regression



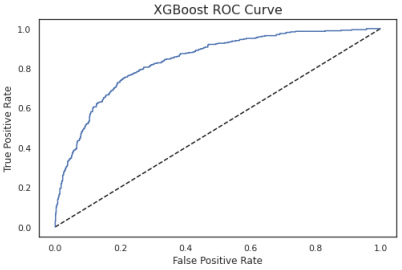
(c) Random Forest



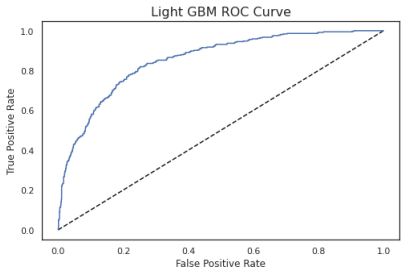
(d) Decision Tree



(e) KNN

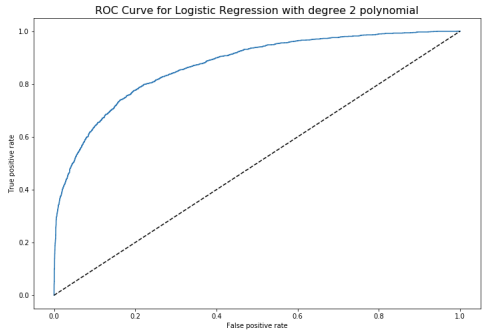


(f) XGBoost

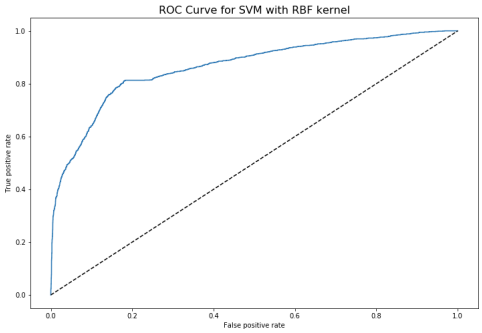


(g) LightGBM

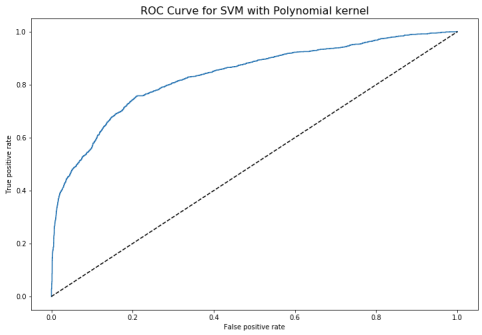
Figure 3.13: ROC graphs for Telecom customer churn dataset



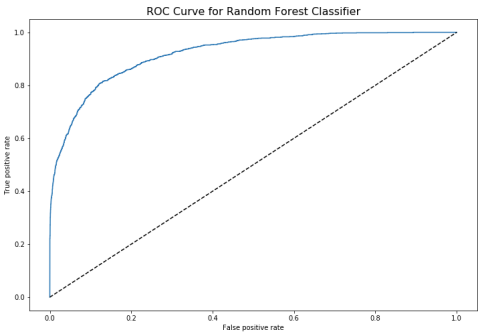
(a) Logistic Regression



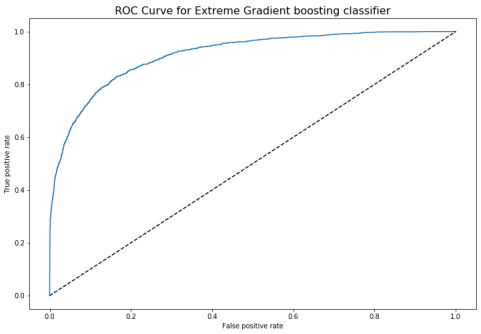
(b) SVM with RBF



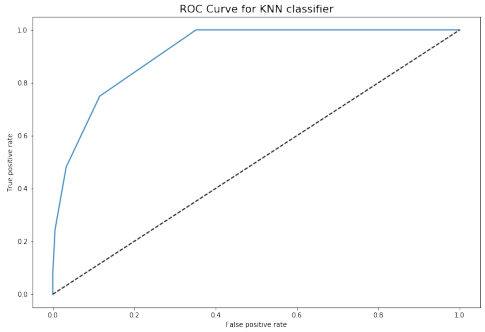
(c) SVM with polynomial



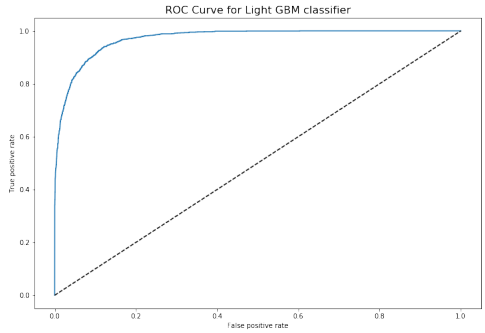
(d) Random Forest



(e) Extreme Gradient Boosting

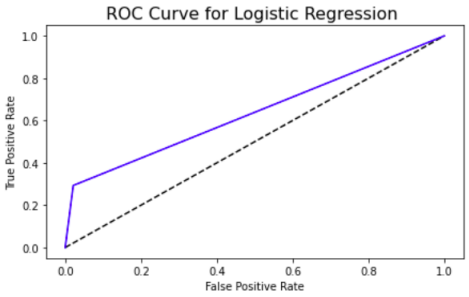


(f) KNN

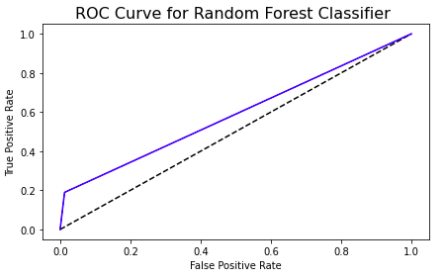


(g)LightGBM

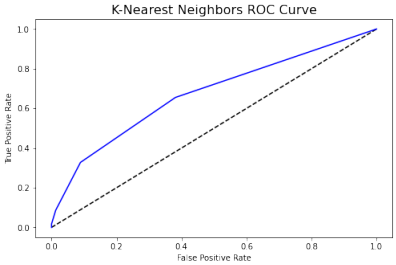
Figure 3.14: ROC graphs for Bank Customer Churn Prediction



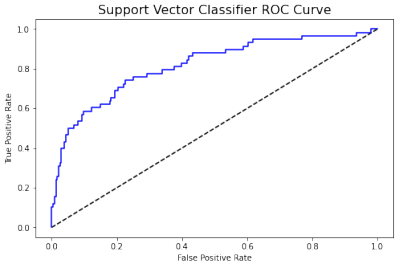
(a) Logistic Regression



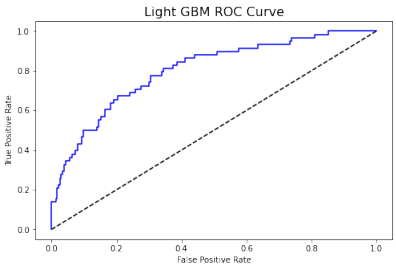
(b) Random Forest



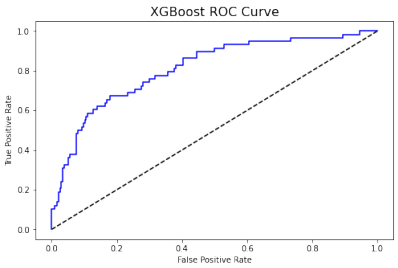
(c) KNN



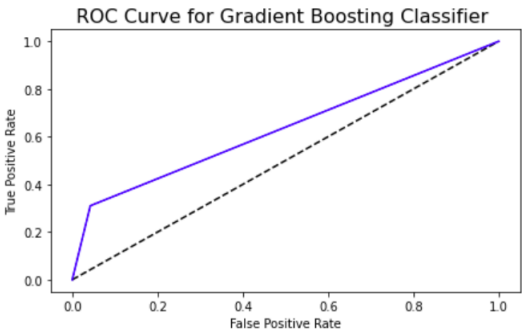
(d) SVC



(e) LightGBM

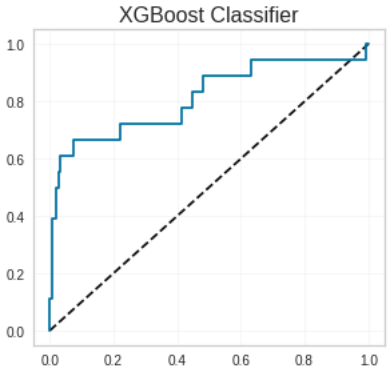


(f) XGboost

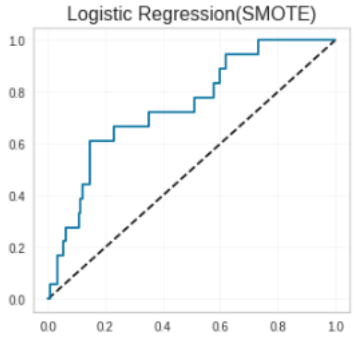


(g) Gradient Boosting

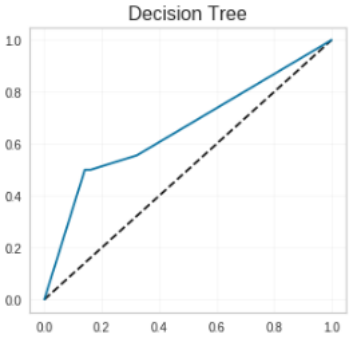
Figure 3.15: ROC graphs for IBM HR Analytics Employee Attrition & Performance



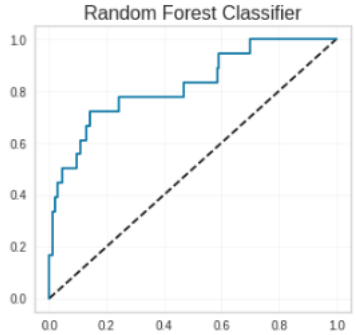
(a) XGboost



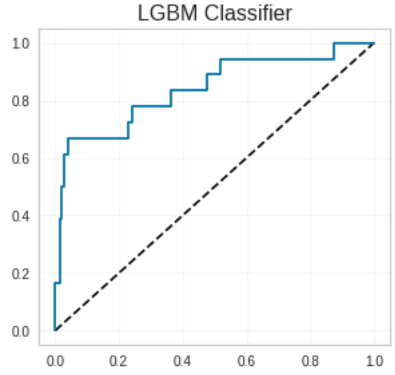
(b) Logistic Regression(SMOTE)



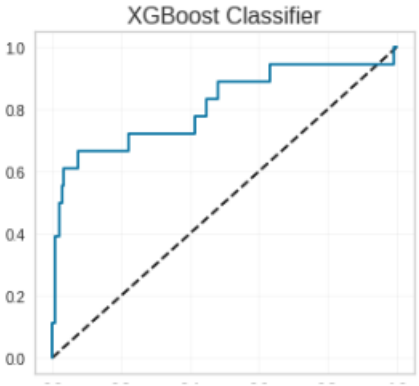
(c) Decision Tree



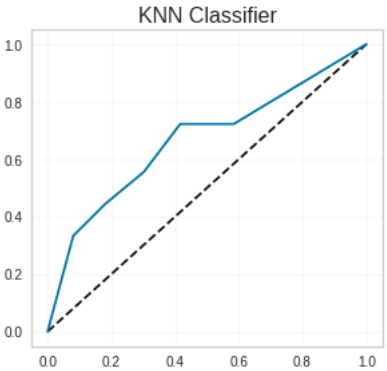
(d) Random Forest



(e)LightGBM

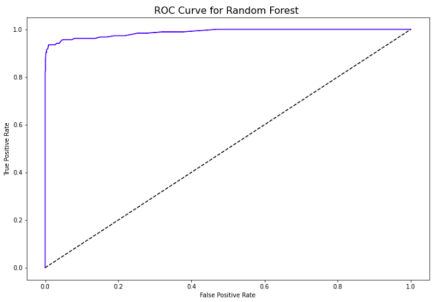


(f)SVM with RBF kernel

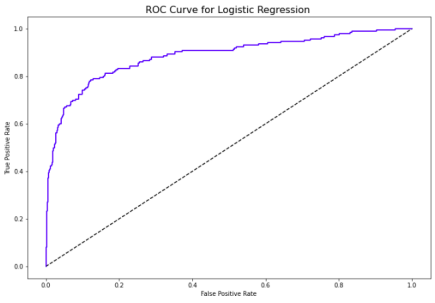


(g) KNN

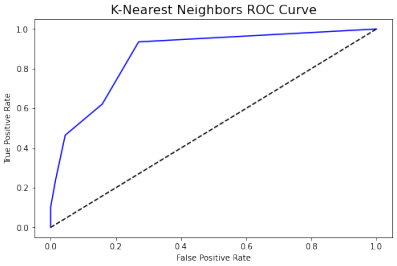
Figure 3.16: ROC graphs for Orange Telecom Prevention and Predicting Churn



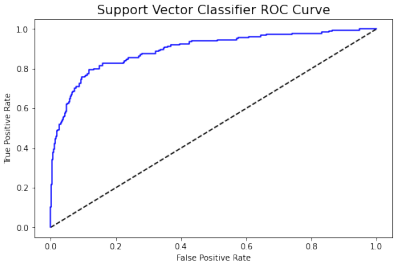
(a) Random Forest



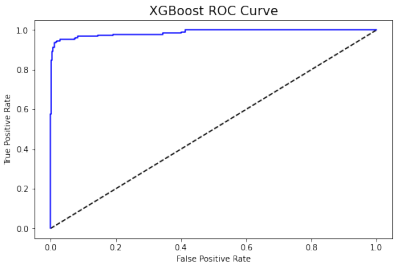
(b) Logistic Regression



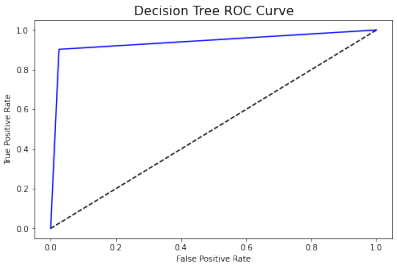
(c) KNN



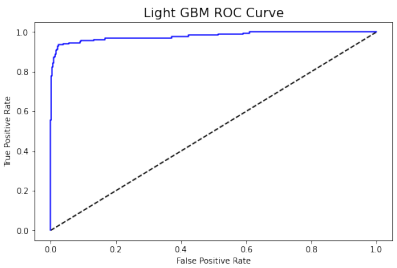
(d) SVM



(e) XGboost



(f) Decision tree



(g) LightGBM

Figure 3.17: ROC graphs for E-commerce Customer Churn Analysis and Prediction

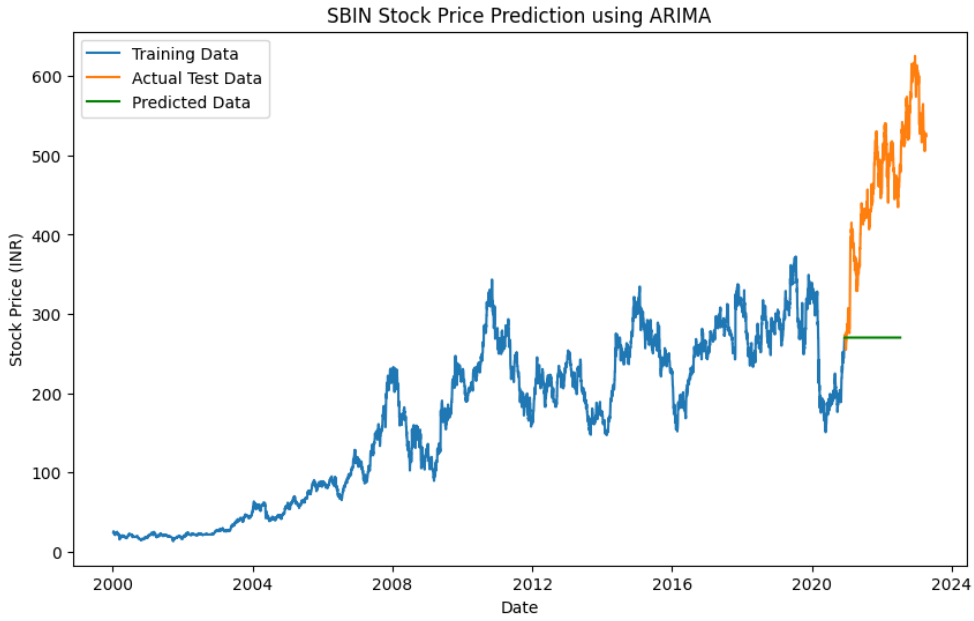


Figure 3.18: ARIMA Actual vs Predicted Graph for SBIN stock price dataset

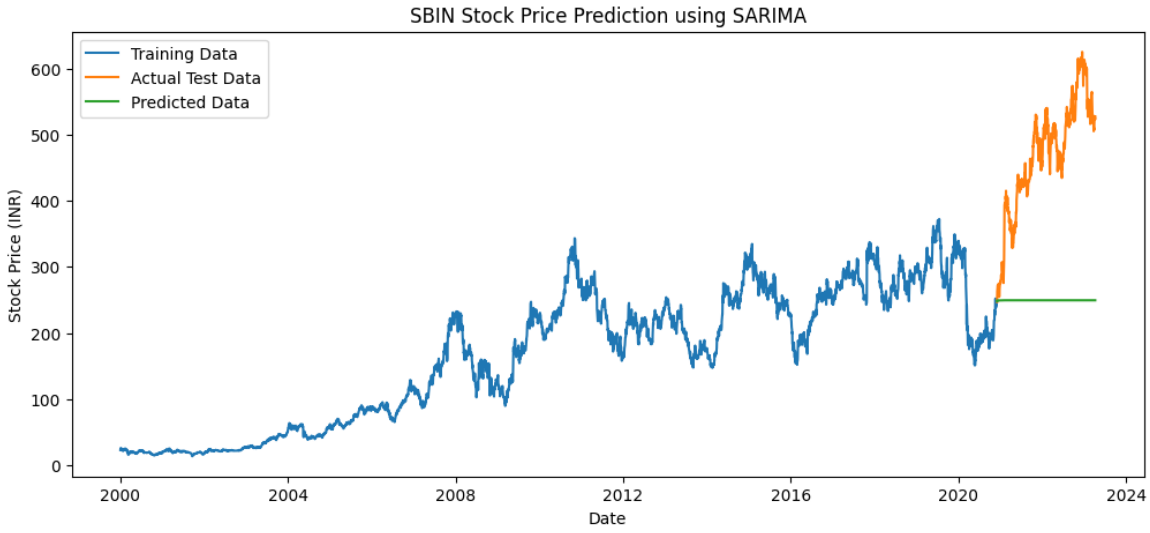


Figure 3.19: SARIMA Actual vs Predicted Graph for SBIN stock price dataset

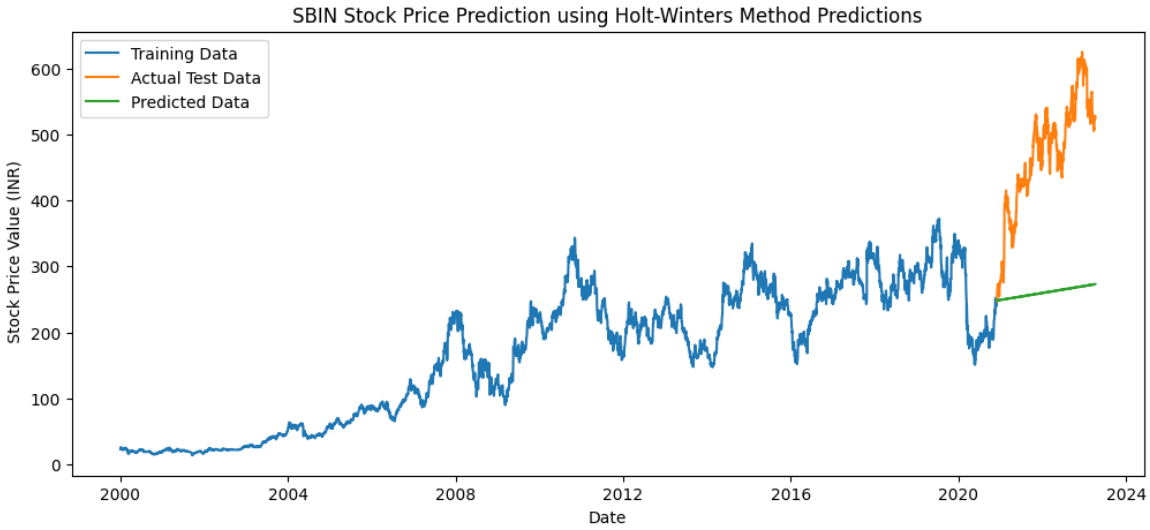


Figure 3.20: Holt Winter Method Actual vs Predicted Graph for SBIN stock price dataset

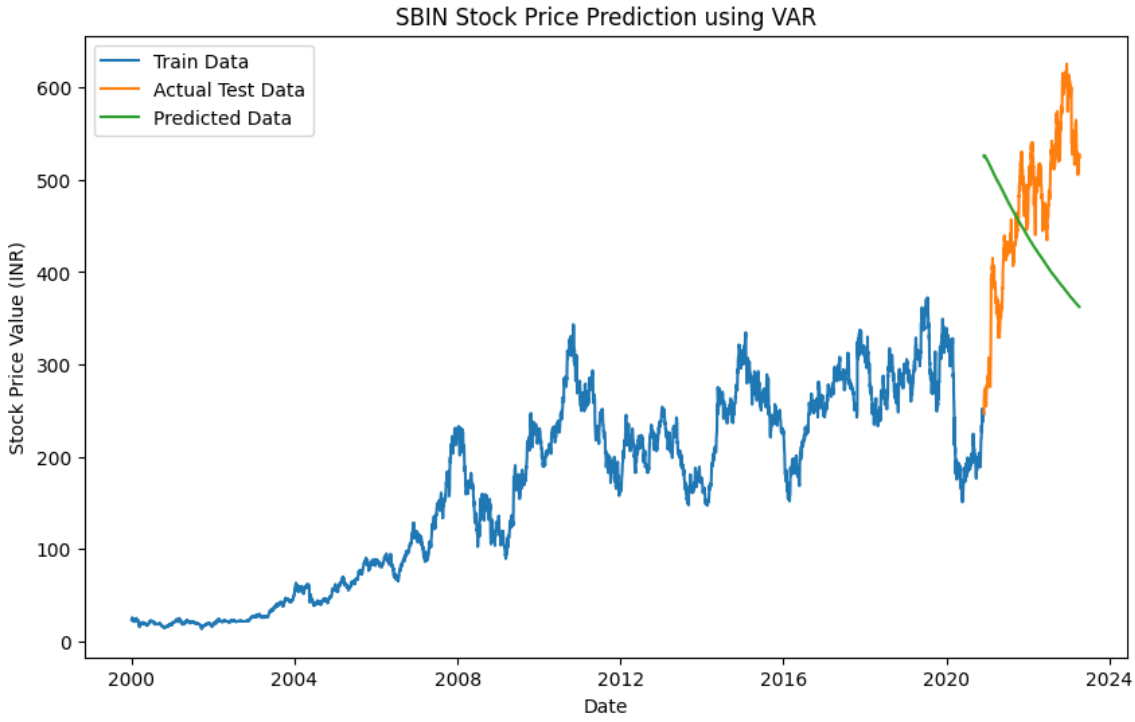


Figure 3.21: VAR Actual vs Predicted Graph for SBIN stock price dataset

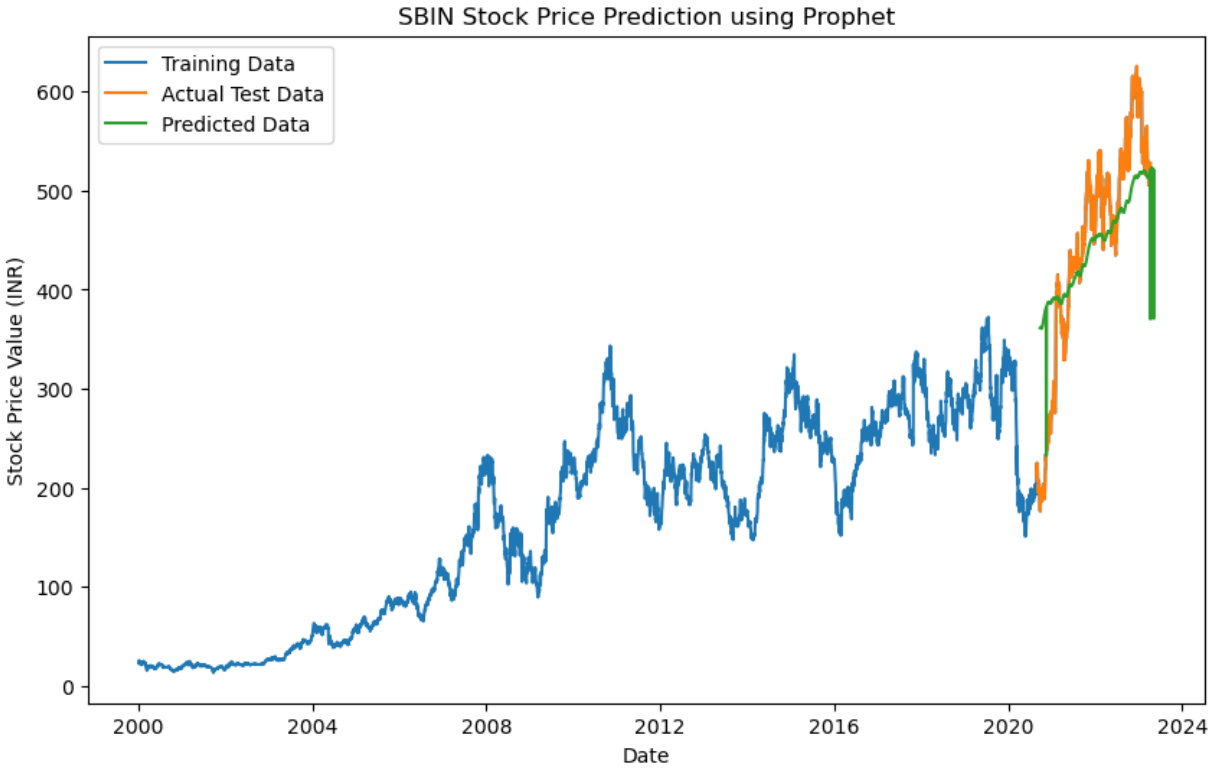


Figure 3.22: Prophet Actual vs Predicted Graph for SBIN stock price dataset

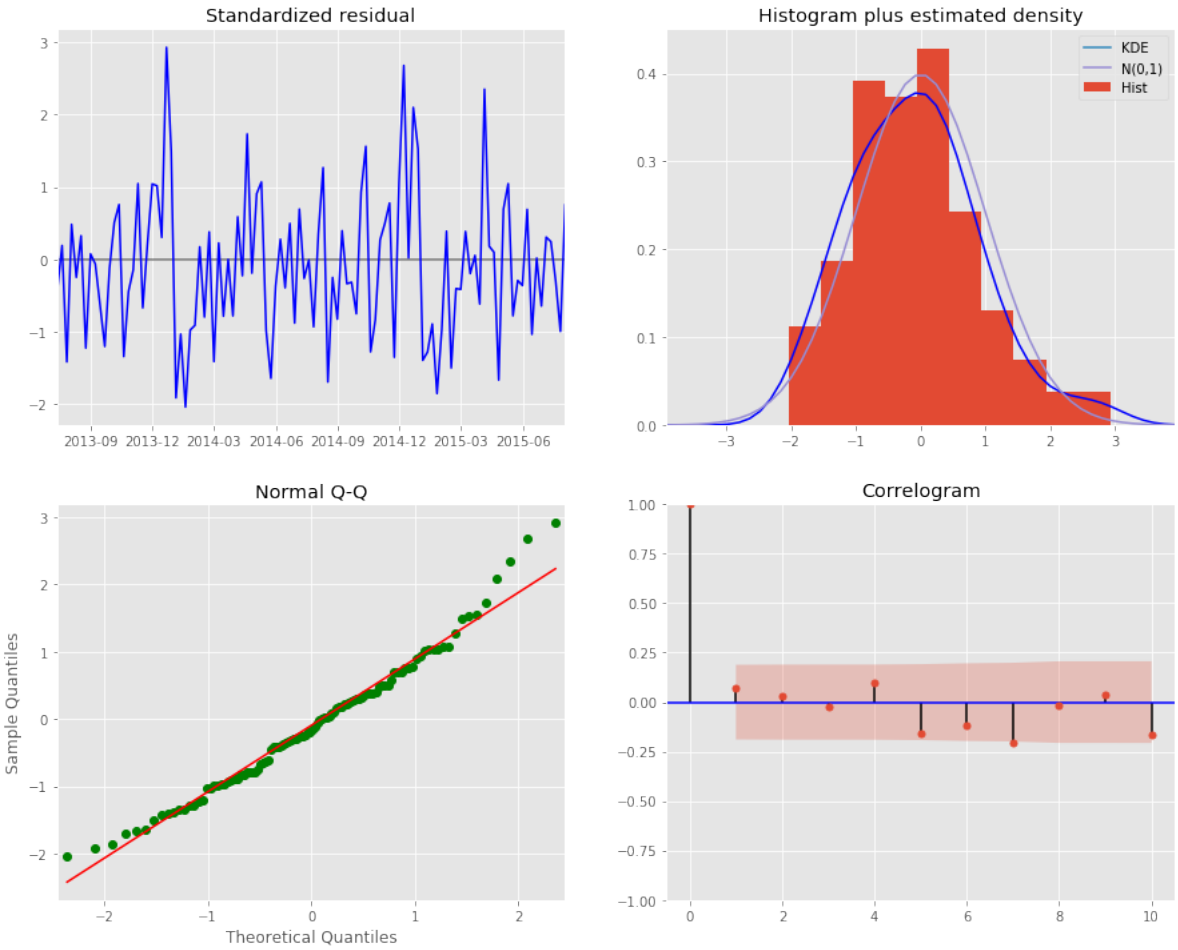


Figure 3.23: SARIMA diagnostics for Rossmann store sales dataset

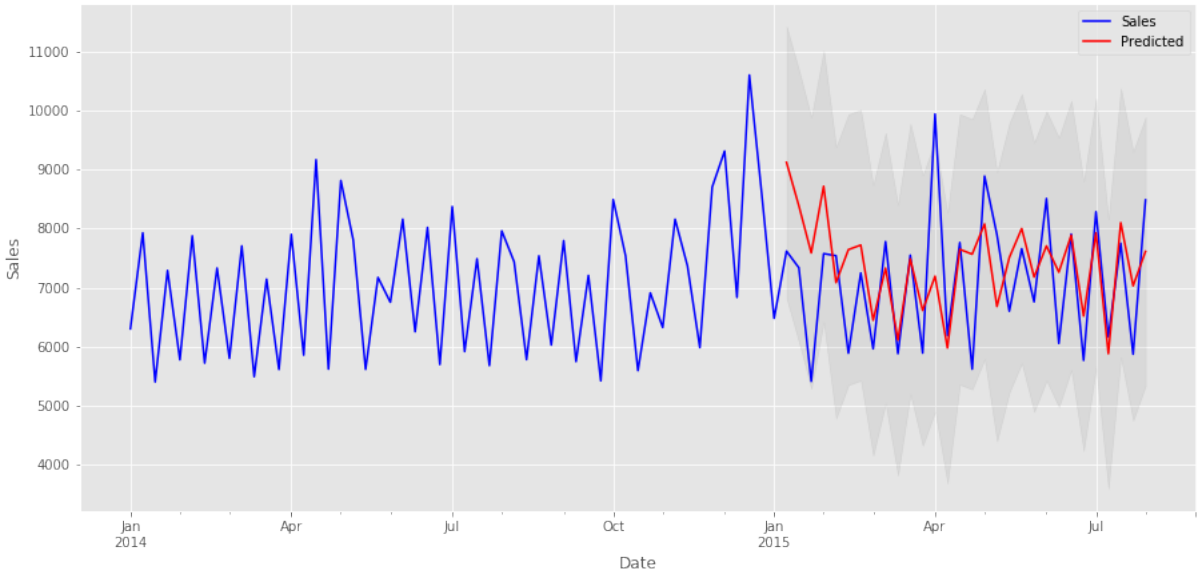


Figure 3.24: SARIMA Actual vs Predicted Graph for Rossmann store sales dataset

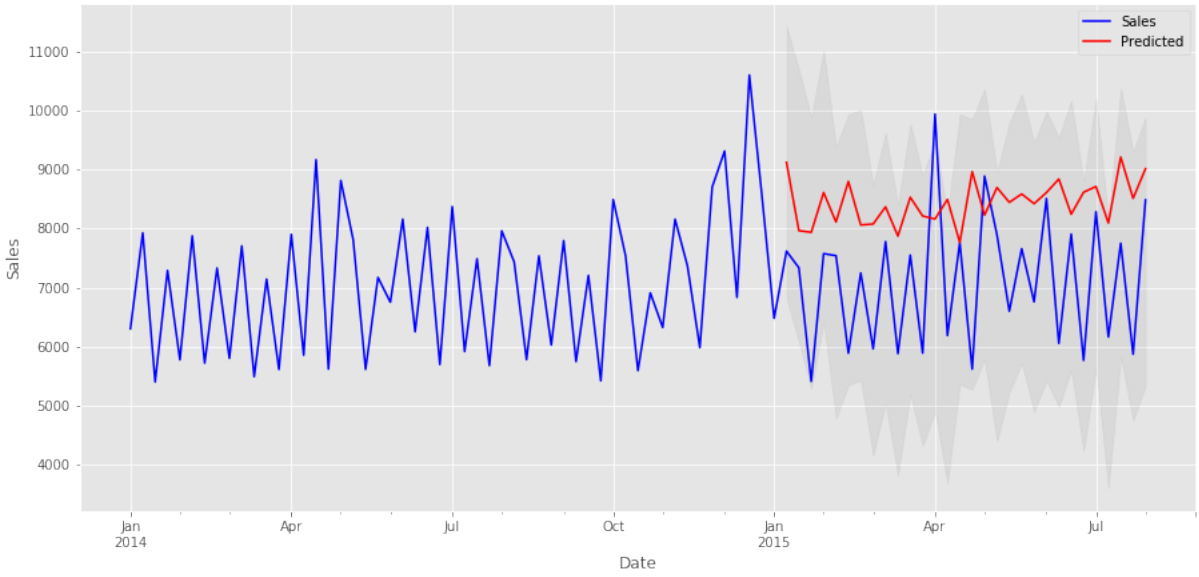


Figure 3.25: ARIMA Actual vs Predicted Graph for Rossmann store sales dataset

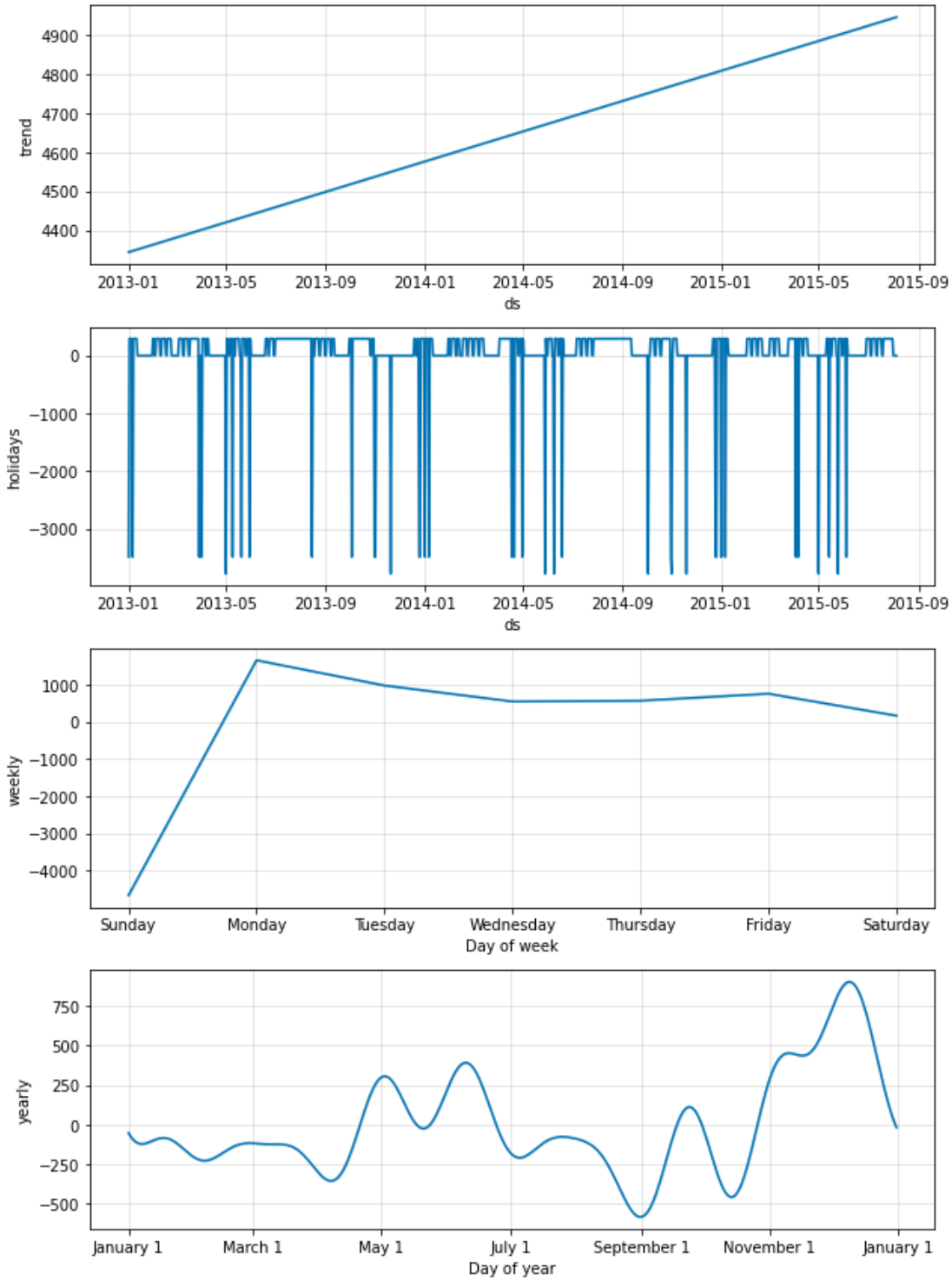


Figure 3.26: Prophet analytics for Rossmann store sales dataset

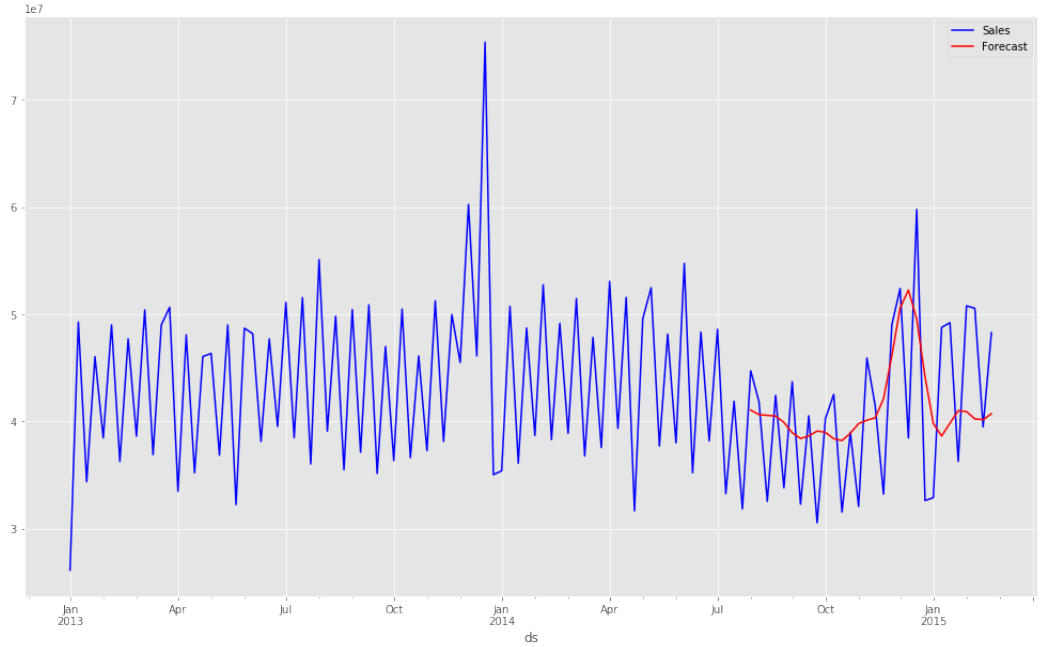


Figure 3.27: Prophet Actual vs Predicted Graph for Rossmann store sales dataset

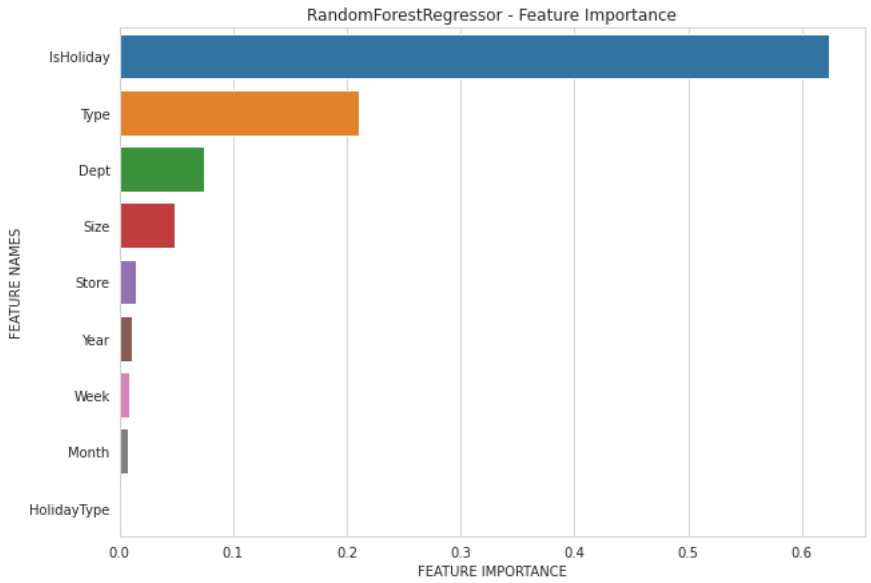


Figure 3.28: Feature importance learnt by Random forest Graph for Walmart store sales dataset

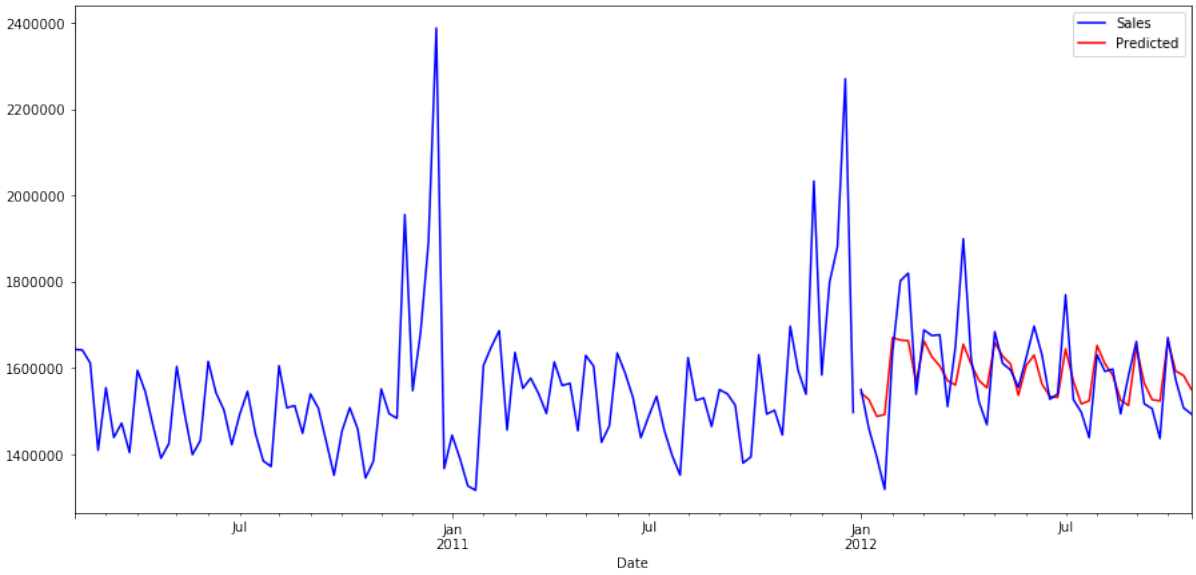


Figure 3.29: Random forest Actual vs Predicted Graph for Walmart store sales dataset

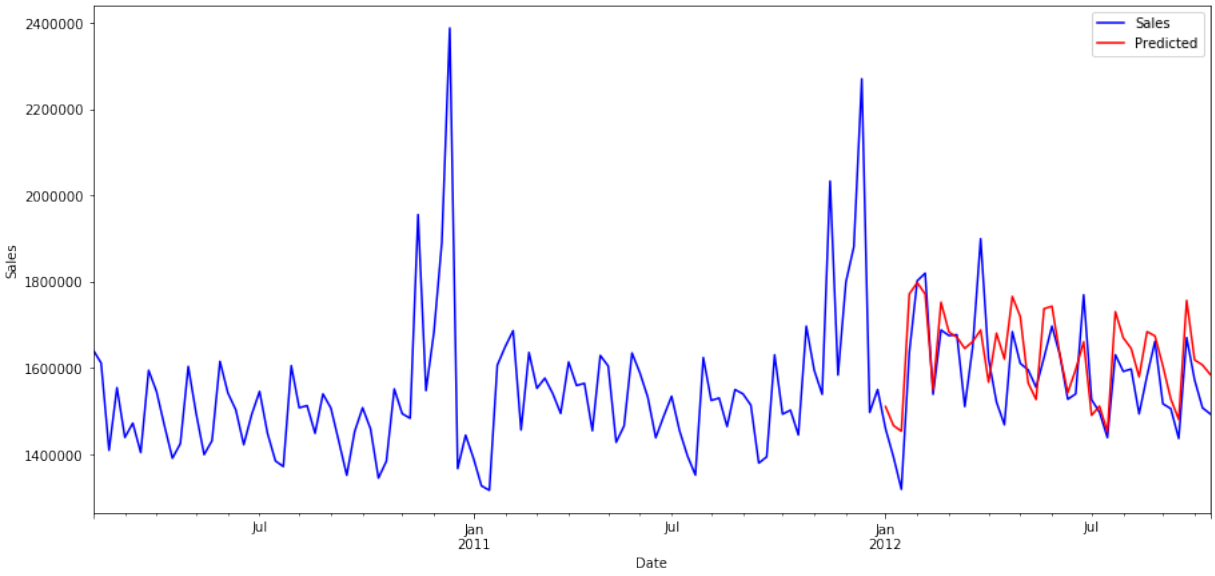


Figure 3.30: SARIMA Actual vs Predicted Graph for Walmart store sales dataset

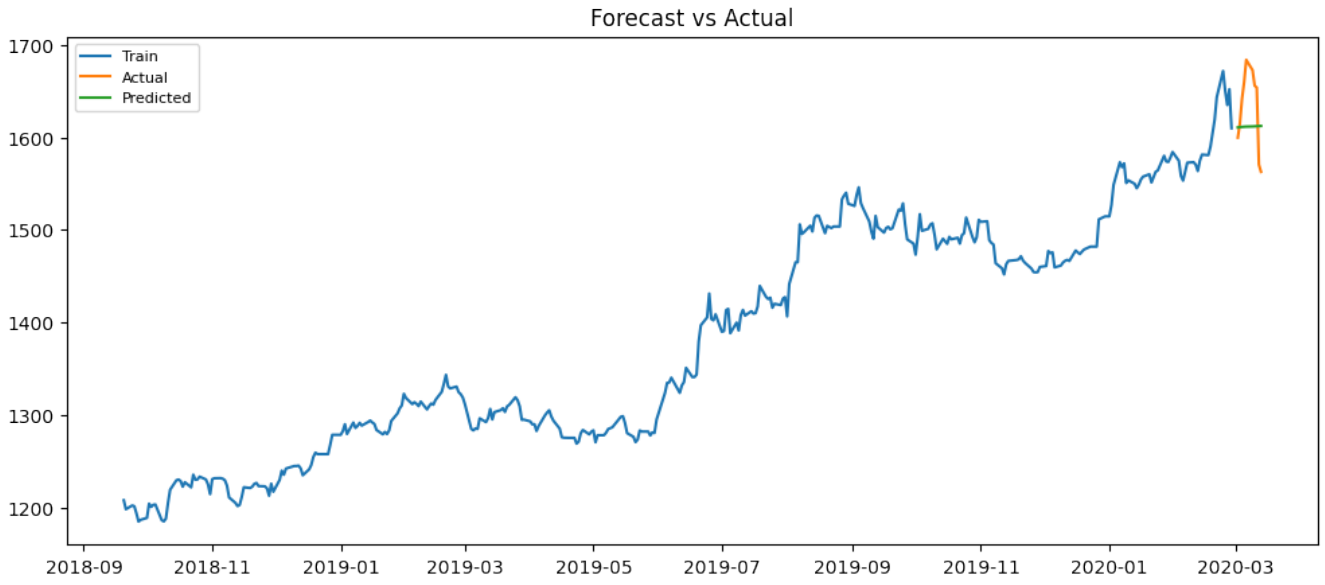


Figure 3.31: ARIMA Actual vs. Predicted Graph for Gold price dataset

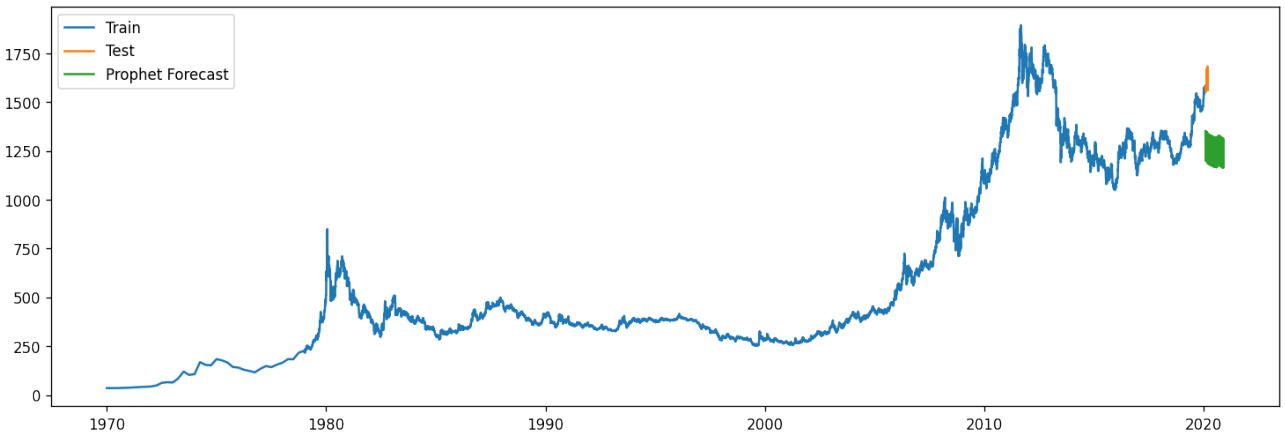


Figure 3.32: Prophet Actual vs Predicted Graph for Gold price dataset

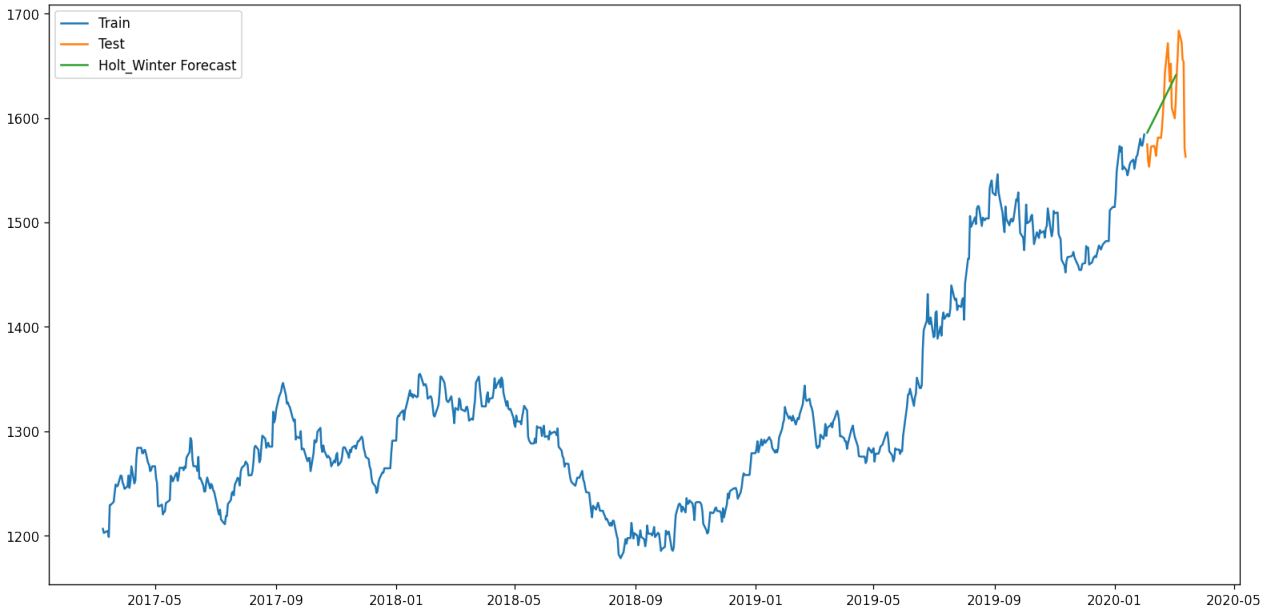


Figure 3.33: Holt winter Actual vs. Predicted Graph for Gold price dataset

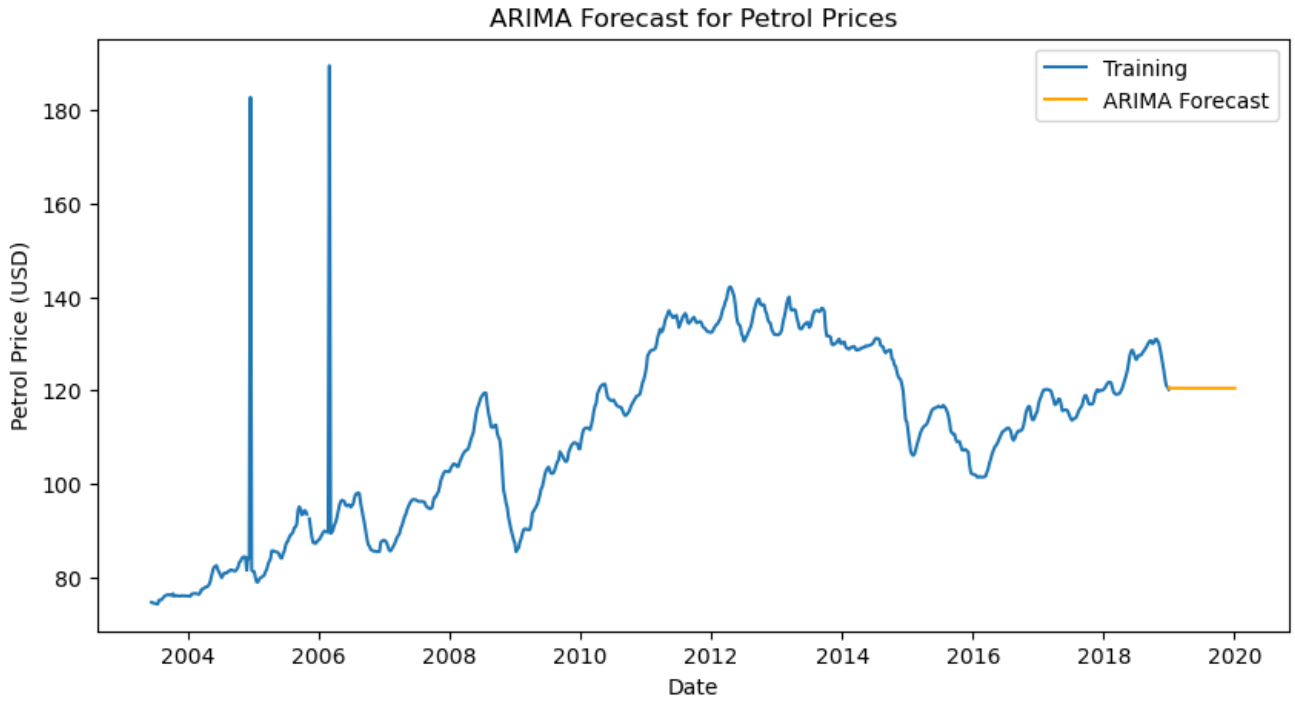


Figure 3.34: ARIMA Actual vs. Predicted Graph for Petrol price dataset

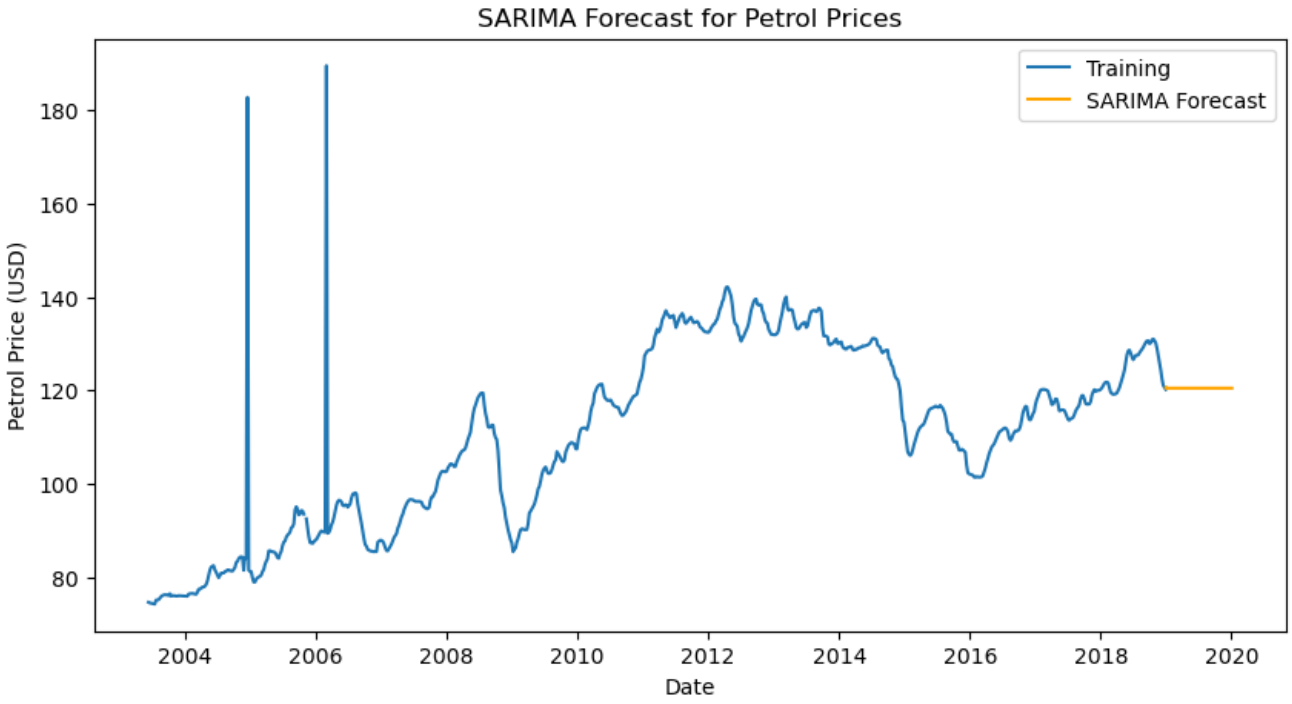


Figure 3.35: SARIMA Actual vs. Predicted Graph for Petrol price dataset

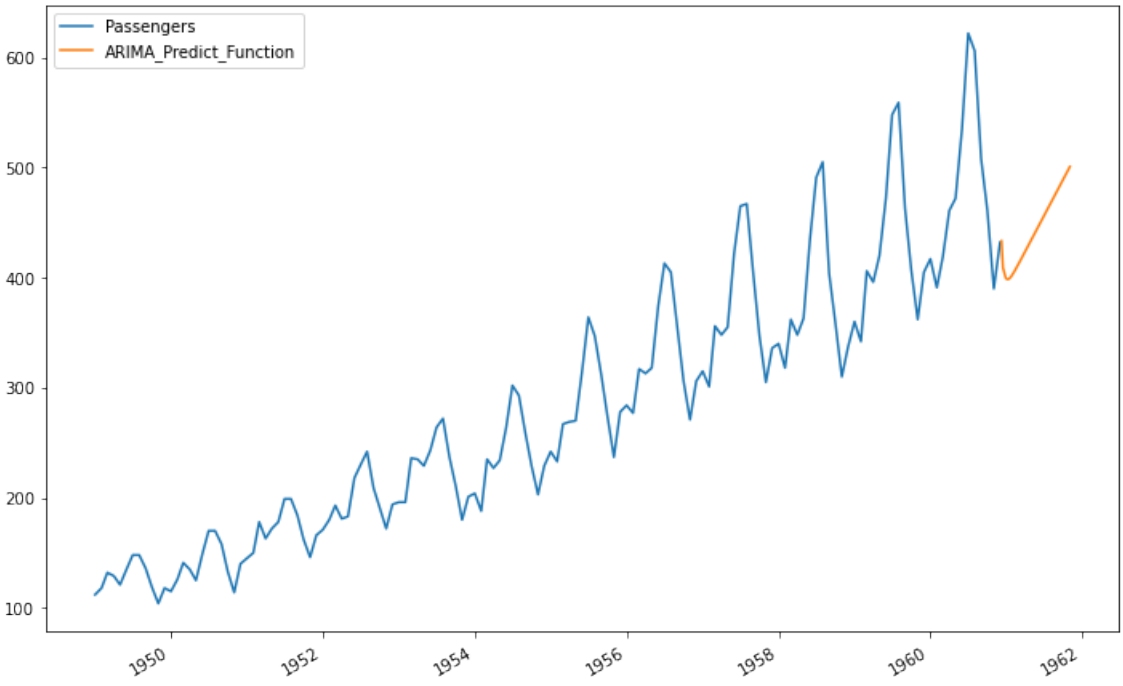


Figure 3.36: ARIMA Actual vs. Predicted Graph for Flight fare dataset

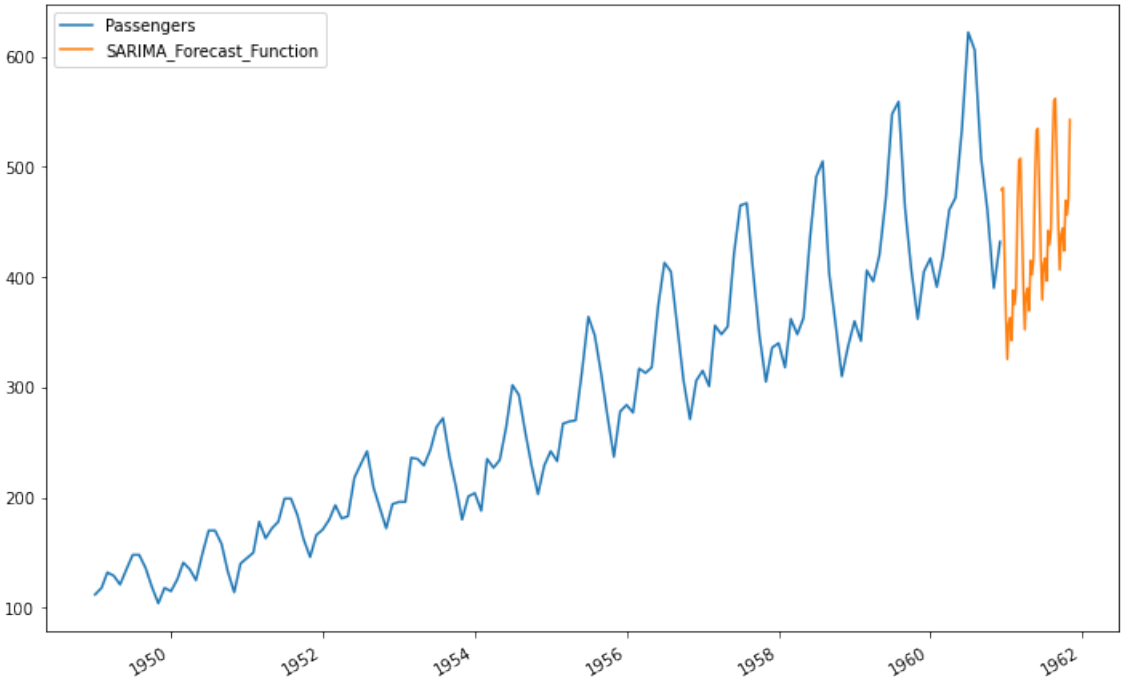


Figure 3.37: SARIMA Actual vs. Predicted Graph for Flight fare dataset

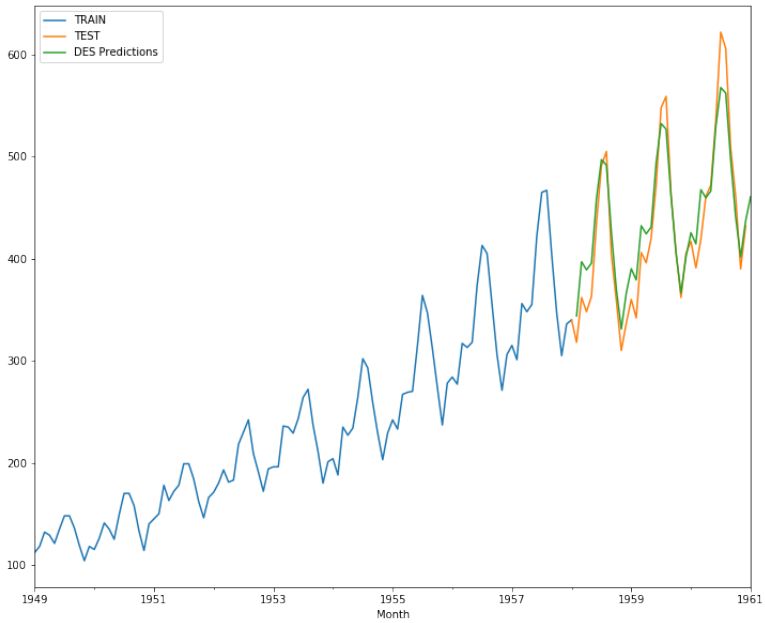


Figure 3.38: DES Actual vs. Predicted Graph for Flight fare dataset

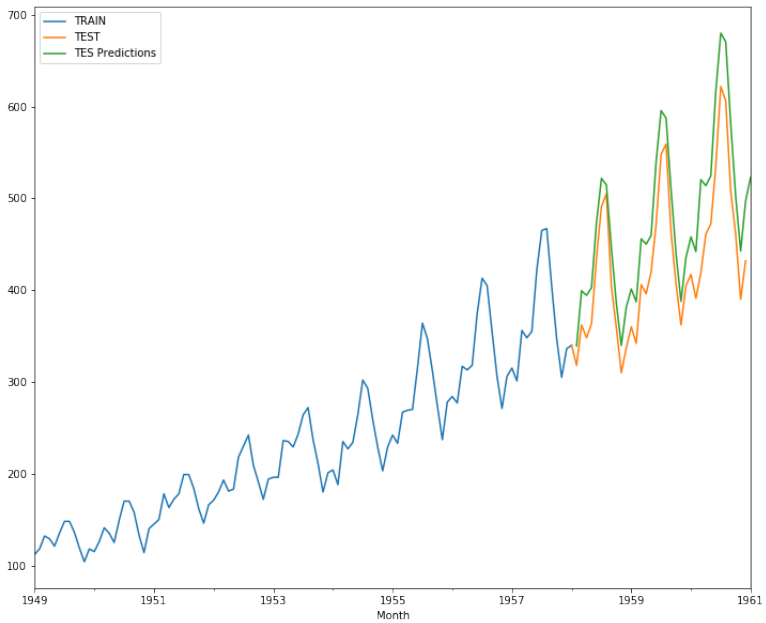


Figure 3.39: TES Actual vs. Predicted Graph for Flight fare dataset

CHAPTER 4

CONCLUSION

4.1 OBSERVATIONS

4.1.1 Churn Prediction as Classification, Regression (Supervised) and Clustering Problem (Unsupervised)

Based on the experiments conducted on various datasets, we can arrive at the following conclusions for the Churn prediction,

- Ensemble approaches perform better than all other available traditional approaches. Especially, LightGBM and Extreme Gradient boosting-based algorithms exhibit robust performance almost over all datasets.

- Mostly, all of the Churn datasets are class imbalanced. Thus, we cannot rely just on accuracy to evaluate our model. The F1 score, along with the AUC score, can be used as a good evaluation metric based on these churn-like datasets.

- Data sampling techniques come in handy for the task of Churn prediction in a great way as it paves the way for handling the class imbalance nature of the dataset. Especially the application of SMOTE on churn prediction datasets yields enhanced results. An increase in performance is noticed when we introduce sampling techniques to handle the imbalance in the data.

- Models performed well when various feature engineering methods were introduced.

- Posing Churn prediction as a regression problem or unsupervised problem makes it better for strategizing the retaining mechanisms.

- As when posed as a regression problem, the specific probability by which a customer may churn is identified, through which the company can focus more on the customers who are risky for the business.

- As When posed as an unsupervised problem (clustering problem), clusters with more churners can be studied to infer the behavior which induces the customers to churn and thus acts as a great tool for companies to evaluate and improve their product service.

- The whole work has been done as a survey listed in an Excel sheet with detailed descriptions and links to all pieces of work and resources. The code for all the

works implemented as part of this project regarding churn prediction is available on this **GitHub**. The results and evaluations, flowcharts, and every other resource related to the work are documented in the GitHub repository.

4.1.2 Time Series Forecasting

From the experimentations, the following inferences arrive.

- Prophet works very well in almost all situations/ kinds of data.
- ARIMA and SARIMA don't work well when the data has multiple dependent variables. (Multivariate). In those cases, Holt winter methods like Triple Exponential smoothening and VAR work great.
- Machine learning regressors like Random Forest, XGBoost, and LightGBM show significant results than the traditional statistical models in datasets with huge complexity.
- Auto-ARIMA works fabulously as we don't need to manually handle the autoregressive or moving average part or the differencing part. This can come in handy for people who don't want to dig up more on finding the best model and extra work.
- The whole work has been done as a survey listed in an Excel sheet with detailed descriptions and links to all pieces of work and resources. The code for all the results implemented as part of this project regarding Time Series Forecasting is available on this **GitHub**. The results and evaluations, flowcharts, and every other resource related to the work are documented in the GitHub repository.

4.2 DISCUSSION

This technical survey focused on exploring, reviewing, analyzing, and documenting various machine learning approaches on various datasets available that are present for the problem of Churn prediction and Time series forecasting for building a referential flowchart that may help anyone with no prerequisite technical knowledge about the algorithms or the model but just with a business need or a problem.

This work consists of various research compilations on churn prediction and Time series forecasting. This proposal comprises a comprehensive study on background work on churn prediction and Time series forecasting that are available, data sampling and

preprocessing approaches, and various algorithms and evaluation metrics.

Also, various datasets available for the Churn prediction problem and Time series forecasting on various domains were discussed, and a thorough pipeline that can be followed for training the model effectively using imbalanced churn-like data with many data preprocessing techniques and feature engineering techniques are also discussed. Along with that, various existing machine learning approaches were discussed in terms of Churn prediction and Time series forecasting; they were evaluated on the diversely available datasets with various robust evaluation metrics. Hence, this technical survey achieved its intention of creating reliable referential material in the form of flowcharts and references.

4.3 FLOWCHARTS

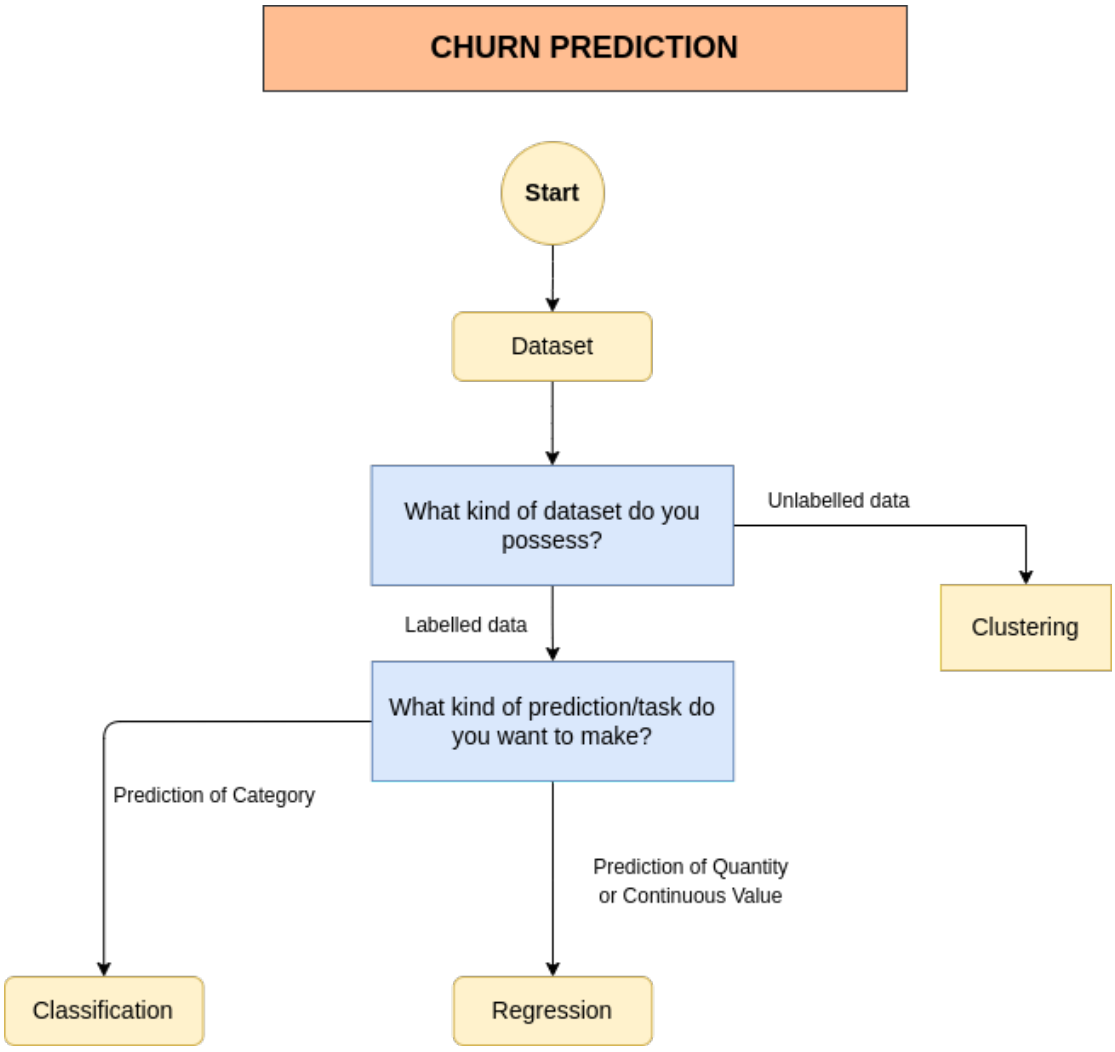


Figure 4.1: Churn Prediction- Flowchart - Problem Selection

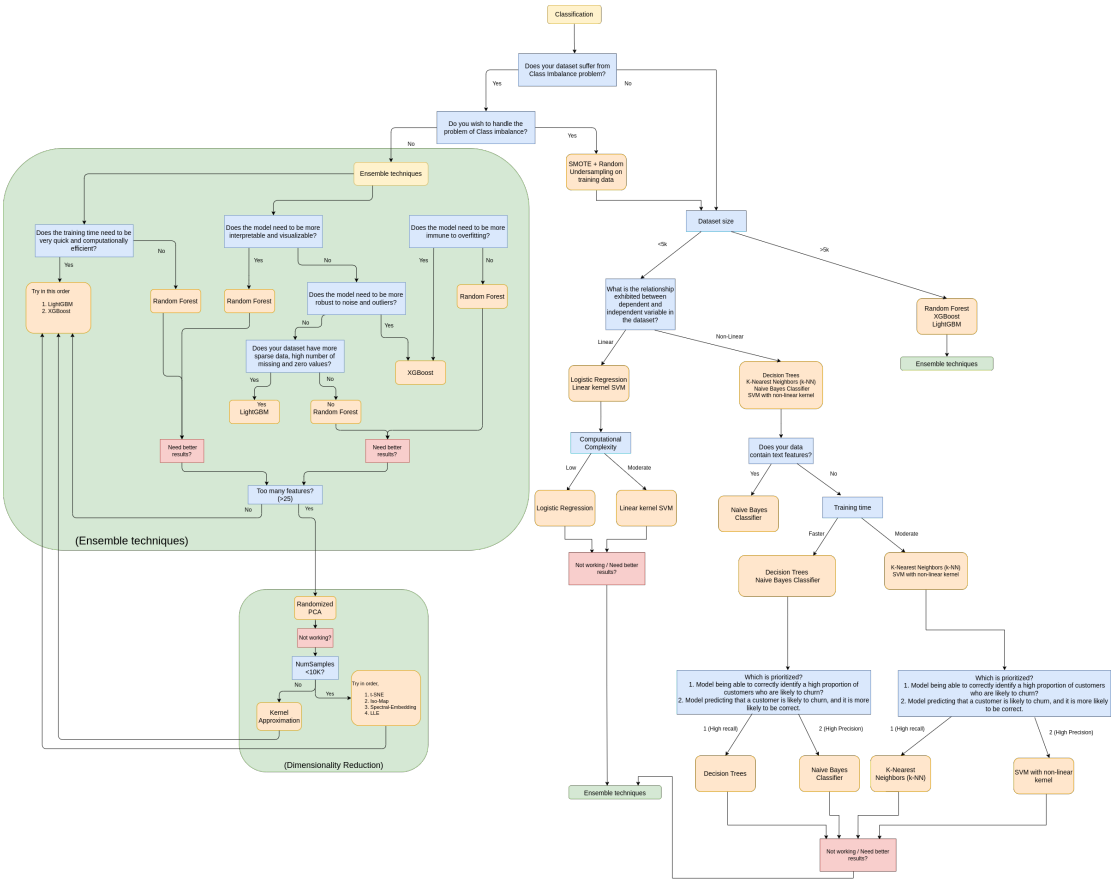


Figure 4.2: Churn Prediction- Flowchart - Classification

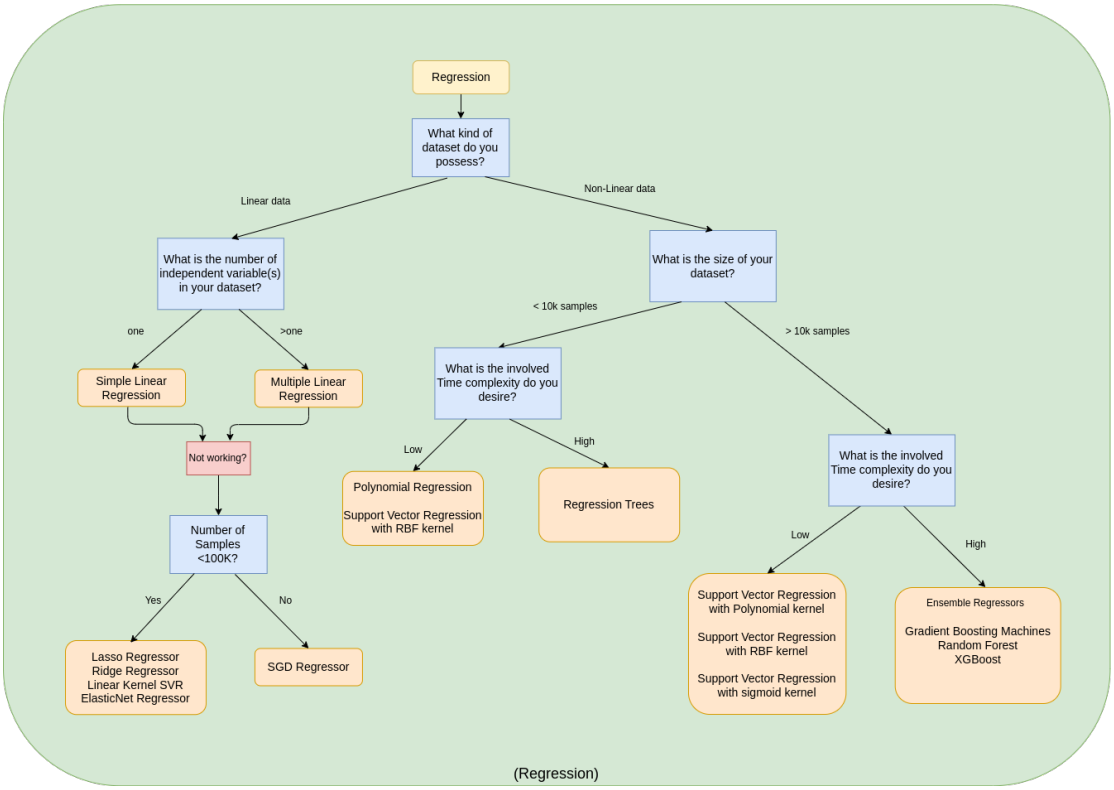


Figure 4.3: Churn Prediction- Flowchart - Regression

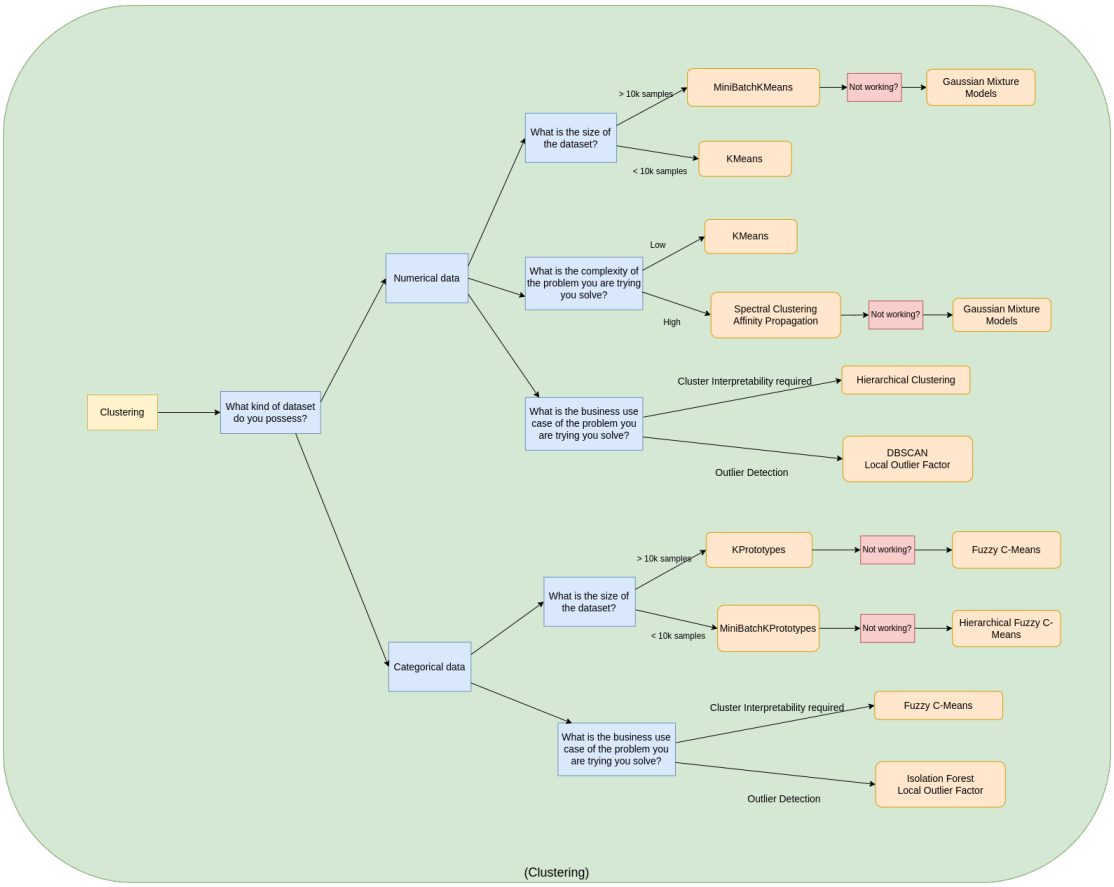


Figure 4.4: Churn Prediction- Flowchart - Unsupervised (Clustering)

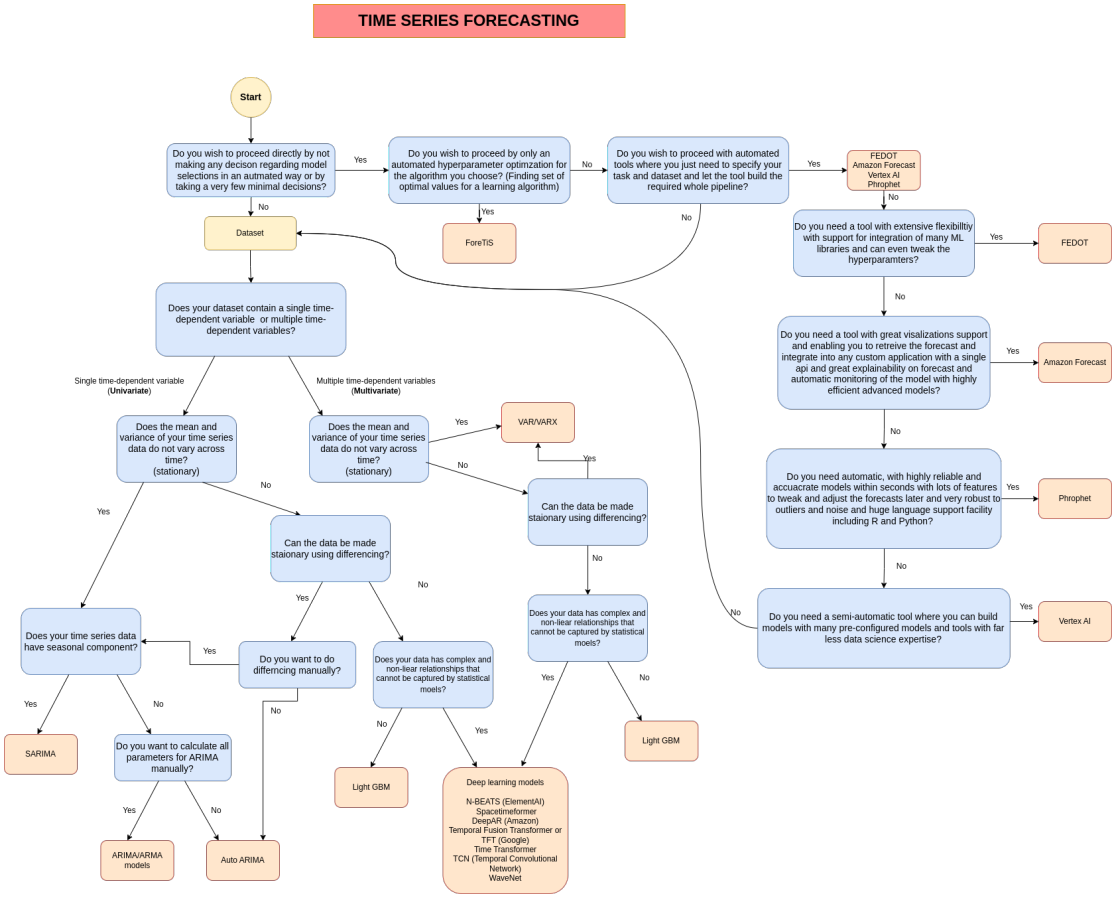


Figure 4.5: Time Series Forecasting- Flowchart

4.4 FUTURE WORK

In this work, various traditional and robust machine learning techniques, impressive deep learning methods, and ready-to-use powerful tools were reviewed and analyzed for churn prediction and Time series forecasting with the help of various datasets. The work is well maintained and documented, supporting the further continuation of work without any hassle. This work can be extended and enhanced by considering more datasets in churn prediction and Time series forecasting domains. Flowcharts that have been made as part of this work have been just made out of rigorous evaluations and analyses of various datasets and algorithms. They have not been evaluated, and thus, the flowcharts can be evaluated by either implementing them in a few real-time business models and validating

with the response we get from the businesses or by applying the flowchart merely on many online competitions like Kaggle and the evaluations and position that is obtained can be used as a way to evaluate the flowcharts.

BIBLIOGRAPHY

1. **Chen, T.** and **C. Guestrin**, Xgboost: A scalable tree boosting system. *In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016.

2. **Chicco, D.**, **M. J. Warrens**, and **G. Jurman** (2021). The matthews correlation coefficient (mcc) is more informative than cohen's kappa and brier score in binary classification assessment. *IEEE Access*, **9**, 78368–78381.

3. **Dubey, H.** and **V. Pudi**, Class based weighted k-nearest neighbor over imbalance dataset. *In Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013.

4. **Eiglsperger, J.**, **F. Haselbeck**, and **D. G. Grimm** (2023). Foretis: A comprehensive time series forecasting framework in python. *Machine Learning with Applications*, 100467.

5. **Geiler, L.**, **S. Affeldt**, and **M. Nadif** (2022). A survey on machine learning methods for churn prediction. *International Journal of Data Science and Analytics*, 1–26.

6. **Han, H.**, **W.-Y. Wang**, and **B.-H. Mao**, Borderline-smote: a new over-sampling method in imbalanced data sets learning. *In International conference on intelligent computing*. Springer, 2005.

7. **Ho, S. L.** and **M. Xie** (1998). The use of arima models for reliability forecasting and analysis. *Computers & industrial engineering*, **35**(1-2), 213–216.

8. **Kalekar, P. S.** *et al.* (2004). Time series forecasting using holt-winters exponential smoothing. *Kanwal Rekhi school of information Technology*, **4329008**(13), 1–13.

9. **Kalpakis, K.**, **D. Gada**, and **V. Puttagunta**, Distance measures for effective clustering of arima time-series. *In Proceedings 2001 IEEE international conference on data mining*. IEEE, 2001.

10. **Ke, G.**, **Q. Meng**, **T. Finley**, **T. Wang**, **W. Chen**, **W. Ma**, **Q. Ye**, and **T.-Y. Liu** (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, **30**.

11. **Lim, B.** and **S. Zohren** (2021). Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, **379**(2194), 20200209.

12. **Liu, X.**, **M. Xie**, **X. Wen**, **R. Chen**, **Y. Ge**, **N. Duffield**, and **N. Wang**, A semi-supervised and inductive embedding model for churn prediction of large-scale mobile games. *In 2018 ieee international conference on data mining (icdm)*. IEEE, 2018.

13. **Rahman, M.** and **V. Kumar**, Machine learning based customer churn prediction in

banking. *In 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. IEEE, 2020.

14. **Sarafanov, M.**, **N. O. Nikitin**, and **A. V. Kalyuzhnaya**, Automated data-driven approach for gap filling in the time series using evolutionary learning. *In 16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021)*. Springer, 2022.

15. **Tan, S.** (2005). Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, **28**(4), 667–671.

16. **Tian, J.**, **H. Gu**, and **W. Liu** (2011). Imbalanced classification using support vector machine ensemble. *Neural computing and applications*, **20**(2), 203–209.

17. **Vapnik, V.** and **V. Vapnik** (1998). Statistical learning theory wiley. *New York*, **1**(624), 2.

18. **Zhao, J.** and **C. Zhang**, Research on sales forecast based on prophet-sarima combination model. *In Journal of Physics: Conference Series*, volume 1616. IOP Publishing, 2020.