A Internship Project Report

on

TELECOM CHURN PREDICTION

INFOSYS SPRINGBOARD INTERNSHIP4.0

By

Tadiboina Varshini

varshiniyadav99@gmail.com

Under the Esteemed Guidance of

Mr. K. Bhaskar

Abstract

The increasing number of telecom service providers has intensified the issue of customer churn. This project proposes a method to predict customer churn in the telecom sector using advanced machine learning techniques.

This project focuses on predicting customer churn using machine learning techniques. By analyzing customer data, the project aims to identify patterns and indicators that signal the likelihood of a customer discontinuing their service.

The performance of these algorithms was compared to find the most accurate model for predicting churn like machine learning models such as Logistic Regression, Decision Tree, Random Forest. This model is used to build a model that identifies whether a customer is at risk of churning or not.

The accuracy of my model after building Logistic regression is 99.7% where as the accuracy of Decision Tree algorithm is 100%.

The importance of this model is to identify the Churners and to take necessary steps. The dataset used in this project is extracted from Kaggle (Where the datasets are stored in a repository).

Index

ACKNOWLEDGEMENT

Subject: Thank You for the Internship Opportunity.

I am writing to express my sincere thanks for offering me the internship opportunity at Infosys. I am thrilled to join the team and to contribute to Churn Prediction project.

I am particularly excited about the chance to learn from experts and to develop my skills in this project. I am committed to working hard and contributing positively to the team. I look forward to gaining valuable experience and making meaningful contributions during my time at Infosys.

Thank you once again for this incredible opportunity. I am eager to start and be a part of such a prestigious organization.

Sincerely,

Tadiboina Varshini.

Overview

The telecom industry is highly competitive, and customer retention is a critical concern for telecom operators. Churn prediction involves identifying customers who are likely to discontinue their service subscription. By predicting churn, telecom companies can take proactive measures to retain these customers, thereby reducing revenue loss and increasing customer satisfaction.

Implementation

Data Collection: Gather relevant customer data from various sources.

Data Preprocessing: Clean the data, handle missing values, encode categorical variables, and normalize features.

Feature Selection: Identify and select relevant features that influence churn.

Model Training: Train the machine learning model using the preprocessed data.

Model Evaluation: Evaluate the model's performance using metrics such as accuracy, precision, recall, and F1-score.

Prediction and Action: Use the trained model to predict churn and implement targeted retention strategies.

Contents

This Project contains 4 Milestones and each Milestone has 2 Weeks of duration. Totally this is a internship project of 8 weeks duration.

Milestone 1: Data Analyzing and Preparation.

Step-1**:** Gathering Data.

Step-2**:** Analyzing Data.

Step-3**:** Understanding Data.

Step-4**:** Installing Required Tools.

Milestone 2: Data Preprocessing

Step-1**:** Converting data types of variables which are misclassified**.**

Step-2**:** Removing Duplicate records.

Step-3**:** Removing Unique value variables.

Step-4**:** Removing Zero variance variables.

Step-5**:** Outlier Treatment

- Using Boxplot: Q3+(1.5 IQR) & Q1-(1.5*IQR)
- Standardization: +/-3 Sigma approach
- Capping & Flooring

Step-6**:** Missing Value Treatment

Remove records if NA's are less than 5%

Remove if NA's are 50% in any variable

Impute with Mean/Median, if variable is numeric and with Mode if variable is categorical.

Step-7**:** Removing highly correlated variables

Step-8**:** Multicollinearity (VIF > 5).

Milestone 3: Model Selection, Training, and Evaluation

Step-1: Model Selection

--- Selecting the model.

Step-2: Model Training

--- Splitting Data into Training and Test Data.

---- Making Predictions

Step-3: Model Evaluation

- Performance Metrics
- Confusion Matrix
- Classification Report
- Accuracy.

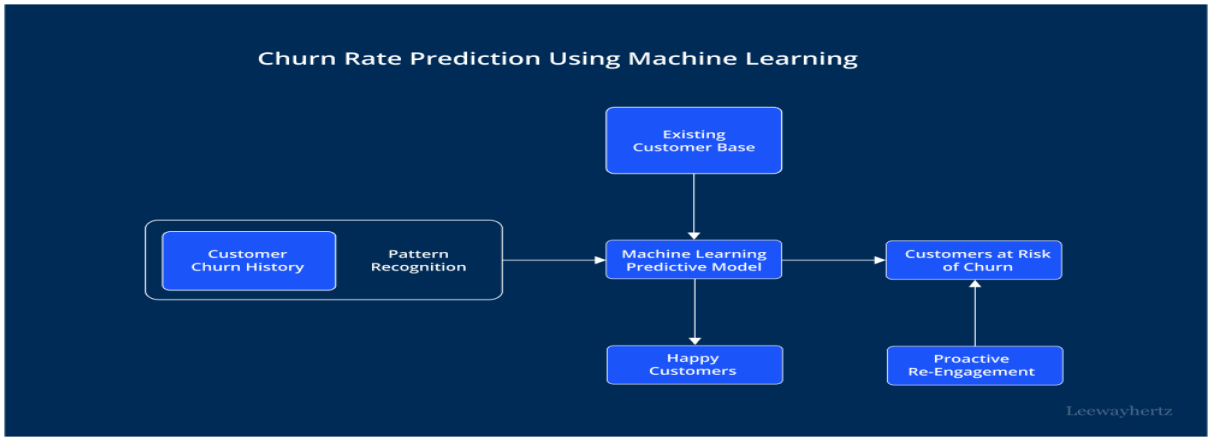
Step-4: Hyper Parameter Tuning

Milestone 4: Documentation

We are evaluating the performance of other models such as Decision Tree and Random Forest.

Introduction

This document provides a step-by-step guide for preprocessing a dataset and using the preprocessed data to build a logistic regression model. The process includes data cleaning, feature selection, normalization, and model building using Python and the **scikit-learn** library.



The preprocessing steps ensure that the data is clean, well-structured, and suitable for analysis. The logistic regression model is chosen for its simplicity and effectiveness in binary classification tasks, such as churn prediction.

Data preprocessing is an important step in building the model. It refers to the cleaning, and transforming. The goal of data preprocessing is to improve the quality of the data.

Prerequisites

Ensure you have the following libraries installed:

numpy, pandas, matplotlib, scikitlearn

Milestone 1: Data Analyzing and Preparation.

Step-1: Data Gathering

The data gathering step involves collecting raw data from various sources that are relevant to the problem you are trying to solve.
This step is crucial because the quality and quantity of the data you gather will directly impact the effectiveness of your model.

Step-2: Data Analysis

The Data Analysis step is crucial for understanding the underlying patterns and relationships in your dataset before proceeding with model building. This step involves both exploratory data analysis (EDA) and statistical analysis to gain insights into the data.

Performing initial data transformations if necessary: This may include normalization, aggregation, or feature engineering.

Step 3: Understanding the Data

In this step we are going to understand about the data by including steps like Feature engineering, by identifying the patterns, by performing statistics and visualizations on data to understand the data.

Step 4: Importing and Installing Required Tools

To preprocess the data and build any model, we need to import several Python libraries. These libraries provide essential functions for data manipulation, preprocessing, and machine learning.

Installing Anaconda Navigator:

Open chrome Browser and search for downloading Anaconda Navigator for Windows.

Download the software by accepting all the terms and conditions which are available.

Now open the Navigator and launch the Jupyter Notebook.

After launching the Notebook you are directed to the jupyter browser.

In the browser open a new Notebook and then you can write the code and execute the code.

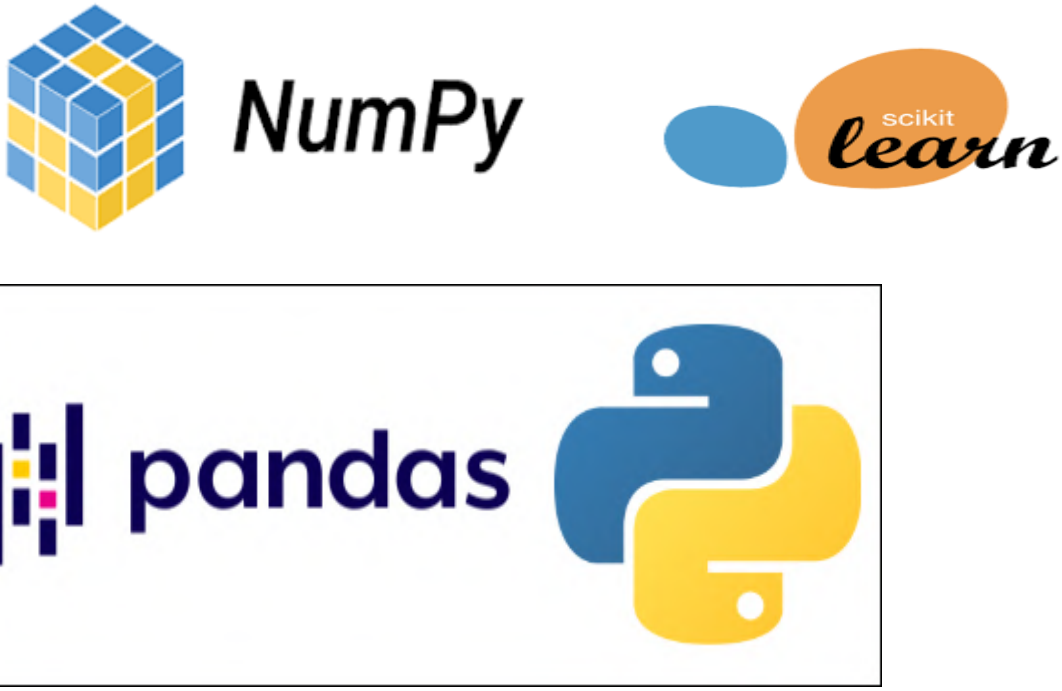
Required Libraries

pandas: For data manipulation and analysis.

numpy: For numerical operations.

scikit-learn: For machine learning tools, including data preprocessing and model building.

matplotlib: For plotting the data.



Milestone-2: Data Preprocessing

There are some basic steps to be followed to perform any kind of operations on the data.

Importing Datasets

Before starting the preprocessing, you need to import the necessary Python libraries. These libraries provide functions and tools for handling data, performing statistical analysis, and building machine learning models.

Loading the Dataset

Load the dataset into a pandas DataFrame. This makes it easier to manipulate and analyze the data.

Data Exploration

Exploring the data helps you understand its structure and content. This step includes displaying the first few rows of the dataset, summary statistics, and checking for missing values.

Here, we are following a step-by-step procedure for processing the data.

1.Converting Datatype of Misclassified Variables

In some datasets, variables might be misclassified, meaning their data types do not match the nature of the data they hold. Converting these misclassified variables to their appropriate data types is essential for accurate data analysis and modeling.

2.Removing Duplicates Records

Removing duplicate records is an important step in data preprocessing to ensure the quality and integrity of the dataset. Duplicate records can lead to biases in the analysis and model training, causing inaccurate results.

3.Removing Unique Value Variables

Unique value variables are features in a dataset where each instance has a unique value. Removing such variables helps in reducing the complexity of the model and can improve its performance.

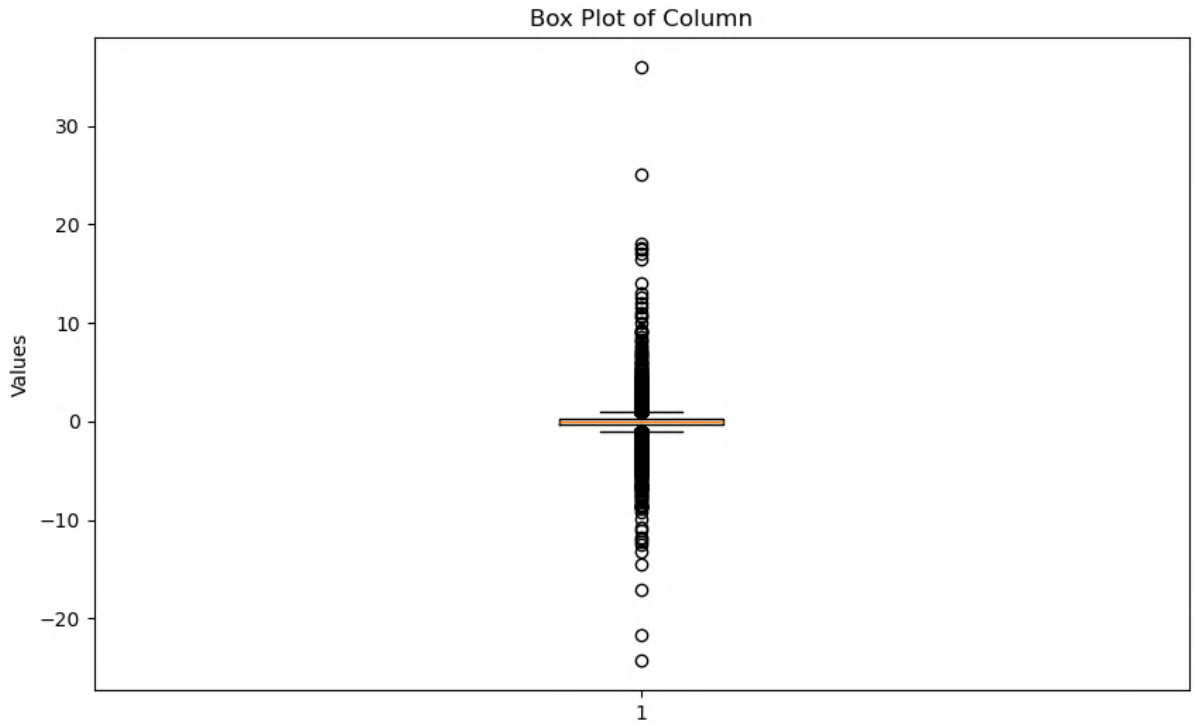
4. Removing Zero Variance Variables

Removing zero variance variables is an important step in data preprocessing. Zero variance variables are those that have the same value across all observations. Including such variables in your dataset can lead to unnecessary complexity and potential issues with model performance.

5.Outlier Treatment

Box Plot

Outliers are data points that differ significantly from other observations in the dataset. one common method to detect and visualize outliers is using a boxplot. We can remove outliers using Boxplot: Q3+(1.5 IQR) & Q1-(1.5*IQR), Standardization: +/-3 Sigma approach, Capping & Flooring.



6. Missing Value Treatment

Missing value treatment is a crucial part of data preprocessing.

Ignoring missing values can lead to inaccurate analysis and poor model performance.

we need to remove variables, records if NA's are less than 5%, Remove if NA's are 50% in any variable. Impute with Mean/Median, if variable is numeric and with Mode if variable is categorical.

We need to consider threshold as 3 to prevent overfitting, and to ensure meaning splits.

In our dataset we are not having any missing values.

7. Removing Highly Correlated Variables

Highly correlated variables can introduce multicollinearity in a dataset, which can negatively impact the performance of certain machine learning models, especially linear models like logistic regression. Multicollinearity occurs when two or more predictor variables are highly correlated, meaning they provide redundant information about the target variable.

8. Multicollinearity

Multicollinearity occurs when two or more predictor variables in a regression model are highly correlated. One common way to detect multicollinearity is by calculating the Variance Inflation Factor (VIF). A VIF value greater than 5 indicates significant multicollinearity among the variables.

Milestone-3: Model Selection, Training and Building

Once the data is preprocessed, the next step is to build and evaluate a logistic regression model. This section describes the steps involved in creating a logistic regression model using the preprocessed data.

Step 1: Model Selection

Here, We are considering Logistic Regression for building the model.

Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome.

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is binary. Like all regression analyses, logistic regression is a predictive analysis.

Initializing the Logistic Regression Model

The first step is to import the necessary library and initialize the logistic regression model. The Logistic Regression class from the scikit-learn library is used for this purpose**.**

Step 2: Training the Model

Train the logistic regression model using the training data. The model learns the relationship between the input features and the target variable.

Logistic regression has two phases:

Training: We train the system using stochastic gradient descent and the cross-entropy loss.

Test: Given a test example x we compute p(y|x) and return the higher probability label y = 1 or y = 0

Making Predictions

After training the model, use it to make predictions on the test data. This involves using the predict method to generate predicted labels for the test set.

Step 3: Evaluating the Model

Evaluate the model's performance using various metrics such as accuracy, confusion matrix, and classification report. These metrics help assess how well the model performs on unseen data.

- **Accuracy**

Accuracy is the proportion of correctly predicted instances out of the total instances. It gives a quick overview of the model's performance.

- **Confusion Matrix**

The confusion matrix provides a detailed breakdown of the model's predictions, showing the true positives, true negatives, false positives, and false negatives.

- **Classification Report**

The classification report includes precision, recall, f1-score, and support for each class. These metrics offer a more comprehensive evaluation of the model's performance.

Performing Hyperparameter tuning on the model

Hyperparameter tuning is the process of optimizing the hyperparameters of a machine learning model to improve its performance. For logistic regression, hyperparameters include the regularization strength (C), the type of regularization (penalty), and the solver used for optimization. The goal of hyperparameter tuning is to find the best combination of these parameters to maximize the model's accuracy and generalizability.

Hyperparameter tuning is a vital step in building a robust logistic regression model.

For my model after performing hyperparameter tuning the accuracy is same as before not performing the hyperparameter tuning only (accuracy of model is 99.7%).

By following these steps, you can effectively tune the hyperparameters of your logistic regression model, resulting in better predictive performance and more reliable outcomes.

Model name	Accuracy
Logistic Regression	99.97%
Decision Tree	100%
Random Forest	100%

Before Performing Hyperparameter Tuning

Confusion Matrix

```
[[3242    0]
 [   1 1490]]
```

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	3242
1	1.00	1.00	1.00	1491
accuracy			1.00	4733
macro avg	1.00	1.00	1.00	4733
weighted avg	1.00	1.00	1.00	4733

After Performing Hyperparameter Tuning

Classification Report:

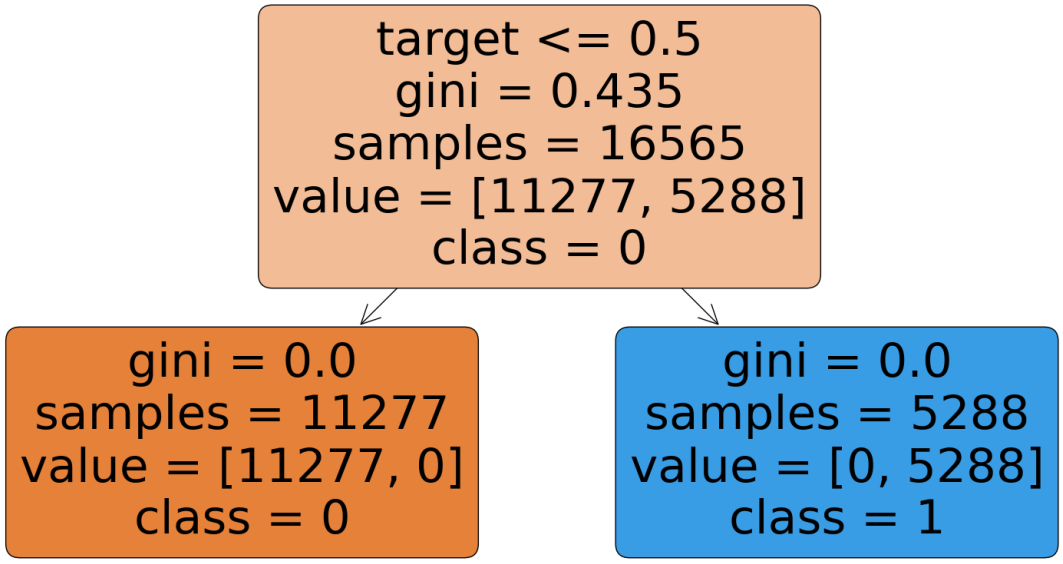
	precision	recall	f1-score	support
0	1.00	1.00	1.00	12892
1	1.00	1.00	1.00	6040
accuracy			1.00	18932
macro avg	1.00	1.00	1.00	18932
weighted avg	1.00	1.00	1.00	18932

Milestone-4: Documentation

Decision Tree

A Decision Tree is a versatile machine learning algorithm that can perform both classification and regression tasks. It works by splitting the data into subsets based on the value of input features, creating a tree-like model of decisions.

The Accuracy of my model is 100%.

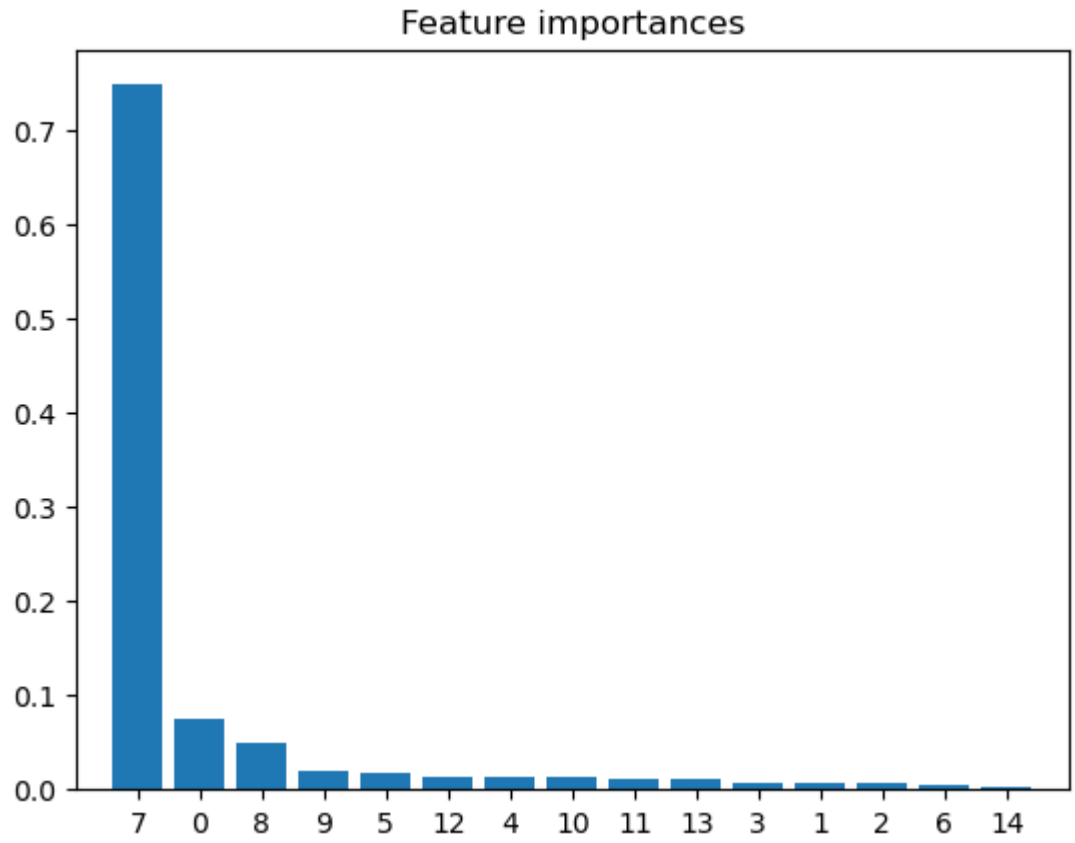


Random Forest

Random Forest is an ensemble learning method that operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

The accuracy of my model is 100%.

While both Decision Trees and Random Forests are powerful tools, they have their own strengths and weaknesses. Decision Trees are easy to interpret and require less computation, making them suitable for smaller datasets or when interpretability is crucial.



Conclusion

Preprocessing the data involves several critical steps that transform raw data into a format suitable for analysis and modeling. By carefully handling missing values, encoding categorical variables, selecting relevant features, and normalizing the data, you set the stage for building a robust and effective machine learning model. Following these preprocessing steps ensures that your data is clean, consistent, and ready for the next phase of the data science workflow.

*** *Thanking you* ***