JSS MAHAVIDYAPEETHA

JSS Science and Technology University

"CUSTOMER CHURN ANALYSIS"

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

By

Taneeshka Naganath Reddy - 01JCE21CS116

Madhusudhan - 01JCE21CS060

Harsha N C - 01JST22UCS410

Sanket- 01JST22UCS430

Under the guidance of

Bindiya AR

Assistant Professor

Department of CSE

JSS STU, Mysuru-06

2023-2024

Department of COMPUTER SCIENCE AND ENGINEERING

TABLE OF CONTENTS

Chapter 1: Introduction…….3

Chapter 2: Literature Review……4-6

Chapter 3: Present Work -Design and Implementation……7-27

Chapter 4: Results and Discussion……28-32

Chapter 5: Conclusion and Future Work……33-34

Chapter 6: References……35

ABSTRACT

In today's competitive business landscape, customer churn poses a significant challenge to companies across various industries. To address this issue, our project focuses on developing a robust machine learning model to predict customer churn. Leveraging the power of Python, we employ advanced machine learning techniques to analyse historical customer data and identify patterns indicative of potential churn behavior.We are using Flask, which is a popular web framework Flask's flexibility and scalability enable us to integrate ML models into a web application.

Furthermore, to enhance usability and accessibility, we integrate our machine learning module seamlessly into a frontend application. By leveraging Flask's compatibility with frontend frameworks, such as HTML and CSS we create an intuitive user interface that empowers stakeholders to interact with the predictive model effortlessly. This integration facilitates real-time predictions and enables businesses to proactively address customer churn, ultimately fostering customer retention and maximising profitability.

Our project not only delivers a predictive solution for customer churn but also showcases the seamless integration of machine learning into frontend applications, demonstrating its practical application in real-world scenarios. Through this endeavour, we aim to equip businesses with actionable insights to mitigate churn and foster long-term customer relationships.

CHAPTER 1 :INTRODUCTION

In the dynamic landscape of the television industry, where viewership preferences constantly evolve and competitors vie for audience attention, retaining customers is paramount for sustained success. Customer churn, the phenomenon where subscribers discontinue their services with a telecom company, presents a significant challenge that directly impacts revenue streams and market competitiveness. Recognizing the critical importance of retaining subscribers, telecom companies are increasingly turning to data-driven approaches to predict and prevent churn.This project is specifically tailored to address the unique challenges of customer churn within the telecom industry. By leveraging advanced machine learning techniques, specifically Python programming and Flask framework, we aim to develop a sophisticated predictive model capable of identifying subscribers at risk of churn.

The primary objective of this project is twofold: firstly, to utilize historical subscriber data, including viewing patterns, subscription plans, and customer interactions, to train and validate an accurate churn prediction model. Secondly, to integrate this model seamlessly into a frontend application tailored for telecom company stakeholders, enabling real-time insights and proactive churn mitigation strategies.

Through this targeted approach, we seek to empower telecom companies with actionable insights to better understand subscriber behaviour, anticipate churn triggers, and implement targeted retention efforts. By unraveling the complexities of customer churn specific to the telecom industry, our project endeavour to foster customer loyalty, maximise subscriber lifetime value, and ultimately drive sustainable growth in an increasingly competitive market.

This introduction provides a focused overview of our project, highlighting the unique challenges and objectives specific to customer churn analysis within the telecom industry. As we delve deeper into the methodology and implementation, we aim to shed light on the intricacies of subscriber behaviour and illuminate effective strategies for churn mitigation in this dynamic and rapidly evolving landscape.

CHAPTER 2 : LITERATURE REVIEW

Customer churn, the phenomenon where customers cease doing business with a company, is a critical concern for businesses across various industries. Predicting churn accurately allows companies to take proactive measures to retain customers and mitigate revenue loss. In recent years, the advent of advanced analytics and machine learning techniques has significantly enhanced the accuracy of churn prediction models.

2.1]Techniques for Customer Churn Prediction

1. **Machine Learning:** Machine learning algorithms like logistic regression, survival analysis, and decision trees are popular for churn prediction. These algorithms analyze historical customer data to identify factors that correlate with churn and then use these factors to predict churn likelihood for new customers [4].
2. **Statistical Modelling:** Statistical techniques can be used to identify churn risk factors and quantify their impact. This can be helpful for understanding the relative importance of different factors.
3. **Data Mining Techniques:** Techniques like clustering and association rule learning can be used to identify groups of customers with similar churn patterns and uncover relationships between customer attributes and churn behavior.

2.2] Ning Lu, Hua Lin, Jie Lu, Guangquan Zhang (May 2014.)"A Customer Churn Prediction Model in Telecom Industry Using Boosting", IEEE Transactions on Industrial Informatics, vol. 10, no.2

Ning Lu suggests using boosting algorithms to improve customer churn prediction models in which customers are clustered based on the boosting algorithm's weight. A high-risk customer cluster was found. Logistic regression (LR) is used as a learner, and each cluster has a churn prediction model. The results showed that boosting algorithm separates churn data better than a single logistic regression model.[1]

2.3] Farquad, H. & Vadlamani, Ravi &Surampudi, Bapi. (2014). "Churn Prediction using Comprehensible Support Vector Machine: an Analytical CRM Application." Applied Soft Computing. 19. 10.1016/j.asoc.2014.01.031.

M.A.H. Farquad present an avenue to overwhelm the fault of general SVM that creates a black box model. The author constructed an approach that divides into three phases. In the 1st phase, SVM Recursive Feature Elimination (SVM-RFE) is hired to decrease the feature set. During 2nd phase, dataset with minimal features are extracted and then apply the SVM techniques to do the classification. In the last phase, rules are extracted manually. After extracting the rules, Naive Bayes is combined with Decision tree and produce the results.[2]

2.4] Nabahirwa Edwine, Wenjuan Wang, Wei Song, Denis Ssebuggwawo, (2022) "Detecting the risk of customer churn in the telecom sector: a comparative study, Math. Probl Eng. 2022". Article ID 8534739, 16 pages.

Edwine et al. , have done a comparative analysis of customer churn prediction models in the telecom industry. They have used three best-fit algorithms, namely KNN, RF, and SVM, along with an optimization algorithm for hyperparameter tuning. They have concluded that the basic versions of these algorithms perform lesser than the amalgamation (RF with grid search optimization algorithm) with a low-ratio undersampling strategy.[3]

2.5] Edvaldo Domingos, Blessing Ojeme, Olawande Daramola,(2021) "Experimental analysis of hyperparameters for deep learning-based churn prediction in the banking sector", Computation 9 (3) 34.

Edvaldo and Olawande have proposed that as of not long ago, conventional AI strategies (for example, MLP and SVM were effectively utilised for pivot forecast, however with impressive exertion in the design of the preparation boundaries. They were foreseeing even information for client relationships with the executives in finance utilising Deep Neural Networks (DNN). [4]

The results of the previous studies lead to useful insights into the industry and help them predict customer churn at an early stage and retain customers.
Logistic regression better interprets the data.
K-Nearest Neighbors offers accurate predictions.
Random Forest is preferred as it splits the trees based on a subset of features to improve the bagging.
SVM is particularly well-suited for binary classification tasks

We will be using a combination of **Random Forest, KNN (K-Nearest Neighbors), Decision Tree, Support Vector Classifier (SVC), Logistic Regression with XGBoost** which provides more accuracy.

CHAPTER 3: PRESENT WORK CARRIED OUT

DESIGN AND IMPLEMENTATION

3.1]DESIGN

3.1.1] Dataset - Telco Customer Churn Dataset

The data set includes information about:

- Customers who left within the last month – the column is called Churn
- Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies
- Customer account information – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges
- Demographic info about customers – gender, age range, and if they have partners and dependents

Link : https://www.kaggle.com/datasets/blastchar/telco-customer-churn

Size : 7043 rows , 21 columns

Features:

FEATURES	DESCRIPTION	FEATURES	DESCRIPTION
CustomerID	Customer ID	DeviceProtection	Whether the customer has device protection or not (Yes, No, No internet service)
gender	Whether the customer is a male or a female	TechSupport	Whether the customer has tech support or not (Yes, No, No internet service)

FEATURES	DESCRIPTION	FEATURES	DESCRIPTION
SeniorCitizen	Whether the customer is a senior citizen or not (1, 0)	StreamingTV	Whether the customer has streaming TV or not (Yes, No, No internet service)
Partner	Whether the customer has a partner or not (Yes, No)	StreamingMovies	Whether the customer has streaming movies or not (Yes, No, No internet service)
Dependents	Whether the customer has dependents or not (Yes, No)	Contract	The contract term of the customer (Month-to-month, One year, Two year)
tenure	Number of months the customer has stayed with the company	PaperlessBilling	Whether the customer has paperless billing or not (Yes, No)
PhoneService	Whether the customer has a phone service or not (Yes, No)	PaymentMethod	The customer's payment method (Electronic check, Mailed check, Bank transfer (automatic), Credit card
MultipleLines	Whether the customer has multiple lines or not (Yes, No, No phone service)	MonthlyCharges	The amount charged to the customer monthly
InternetService	Customer's internet service provider (DSL, Fiber optic, No)	TotalCharges	The total amount charged to the customer
OnlineSecurity	Whether the customer has online security or not (Yes, No, No internet service)	Churn.	Whether the customer churned or not (Yes or No)

OnlineBackup	Whether the customer has online backup or not (Yes, No, No internet service)		

3.1.2] Data Preprocessing

1] Data Cleaning:

- **Null Values :**In data cleaning , we check if any null values are present. If any null values present ,it is be handled by dropping them or replacing them by the mean. It is done with the help of **SimpleImputer** library.
- Converting object data type to float.

2] Feature Engineering:

We plot distributions for numerical and categorical features to check for outliers and compare feature distributions with target variable

- **Numerical features distribution:** Numeric summarising techniques (mean, standard deviation, etc.) don't show us spikes, shapes of distributions and it is hard to observe outliers with it. That is the reason we use histograms.
- **Categorical feature distribution** : To analyze categorical features, we use bar charts

3] One Hot Encoding:

One hot encoding is a technique that we use to represent categorical variables as numerical values in a machine learning model. We are using this technique to convert all the categorical features in our dataset , for example: gender', 'Partner', 'Dependents', 'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport ,etc into numerical values so that it will be understood by our machine learning models.

4] Data Standardization:

The StandardScaler from sklearn.preprocessing is a tool used to standardize features by removing the mean and scaling to unit variance. Here's what it does in detail:

- Removing the Mean: It subtracts the mean value of each feature from the data. This centers the data around zero.

- Scaling to Unit Variance: It divides the data by the standard deviation of each feature. This scales the data to have a standard deviation of one.
- Standardizing the data can be important for machine learning algorithms, especially those that rely on distance metrics (like k-nearest neighbors) or assume normally distributed data (like linear regression).

3.1.3] Handling Imbalanced Data

Target variable distribution shows that we are dealing with an imbalanced problem as there are many more non-churned as compare to churned users. The model would achieve high accuracy as it would mostly predict majority class - users who didn't churn in our example. So to minimize the influence of imbalance classes:

Use SMOTE (Synthetic Minority Oversampling Technique) :

- SMOTE helps address this by creating new, **artificial customers** who churned (Churn_yes), specifically focusing on the minority class.
- **Find the "Churn Club":** SMOTE first identifies the customers who churned (the minority class). Think of them as a special club.
- **Look for Similarities:** For each churning customer, SMOTE finds other churning customers who are similar based on their features (like purchase history, demographics, etc.). Imagine these similar customers as close friends in the "Churn Club."
- **Create New "Friends":** SMOTE then creates new, synthetic customers by taking a churning customer and its closest friend. It blends their features together, like creating a new friend who shares some characteristics with both. This new friend represents a **synthetic churn case**.
- **More "Friends" for the Club:** SMOTE repeats steps 2 and 3 multiple times for each churning customer, creating a group of new synthetic churners. This essentially expands the "Churn Club" with more "friends."
- SMOTE doesn't create perfect copies of existing customers. It creates new data points that are similar but not identical.

3.1.4] Model Selection:

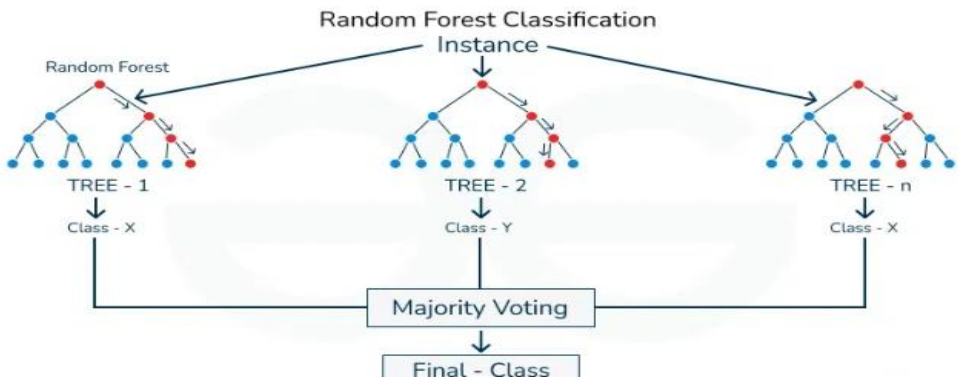
3.1.4.1] Random Forest Classification

Random Forest Classification is an ensemble learning technique designed to enhance the accuracy and robustness of classification tasks. The algorithm builds a multitude of decision trees during training and outputs the class that is the mode of the classification classes. Each decision tree in the random forest is constructed using a subset of the training data and a random subset of features introducing diversity among the trees, making the model more robust and less prone to overfitting.

The random forest algorithm employs a technique called bagging (Bootstrap Aggregating) to create these diverse subsets.

During the training phase, each tree is built by recursively partitioning the data based on the features. At each split, the algorithm selects the best feature from the random subset, optimizing for information gain or Gini impurity. The process continues until a predefined stopping criterion is met, such as reaching a maximum depth or having a minimum number of samples in each leaf node.

Once the random forest is trained, it can make predictions, using each tree "votes" for a class, and the class with the most votes becomes the predicted class for the input data.



Feature selection in Random Forests is inherently embedded in the construction of individual decision trees and the aggregation process.

During the training phase, each decision tree is built using a random subset of features, contributing to diversity among the trees. The process is, known as feature bagging, helps prevent the dominance of any single feature and promotes a more robust model.

The algorithm evaluates various subsets of features at each split point, selecting the best feature for node splitting based on criteria such as information gain or Gini impurity. Consequently, Random Forests naturally incorporate a form of feature selection, ensuring that the ensemble benefits from a diverse set of features to enhance generalization and reduce overfitting.

ADVANTAGES:

Handling Overfitting: By averaging the results of multiple trees, it reduces the risk of overfitting, which is common in decision trees.

Versatility: It works well with both numerical and categorical data, making it suitable for diverse datasets.

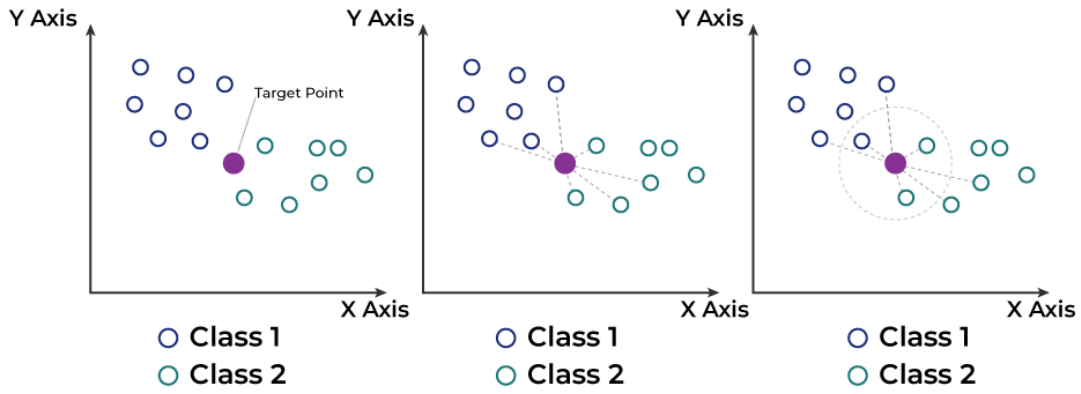
Feature Importance: It provides insights into feature importance, helping to understand which factors are most influential in predicting churn.

DISADVANTAGES:

More trees slow down the model.

3.1.4.2] KNN (K-Nearest Neighbors)

The K-Nearest Neighbors (KNN) algorithm operates on the principle of similarity, where it predicts the label or value of a new data point by considering the labels or values of its K nearest neighbors in the training dataset.



Step 1: Selecting the optimal value of K

- K represents the number of nearest neighbors that needs to be considered while making prediction.

Step 2: Calculating distance

- To measure the similarity between target and training data points, Euclidean distance is used. Distance is calculated between each of the data points in the dataset and target point.

Step 3: Finding Nearest Neighbors

- The k data points with the smallest distances to the target point are the nearest neighbors.

Step 4: Voting for Classification or Taking Average for Regression

- In the classification problem, the class labels of are determined by performing majority voting. The class with the most occurrences among the neighbors becomes the predicted class for the target data point.

- In the regression problem, the class label is calculated by taking average of the target values of K nearest neighbors. The calculated average value becomes the predicted output for the target data point.

Let X be the training dataset with n data points, where each data point is represented by a d-dimensional feature vector Xi and Y be the corresponding labels or values for each data point in X. Given a new data point x, the algorithm calculates the distance between x and each data point Xi in X using a distance metric, such as Euclidean distance: $\text{distance}(x, X_i) = \sqrt{\sum_{j=1}^{d}(x_j - X_{i_j})^2]}$

The algorithm selects the K data points from X that have the shortest distances to x. For classification tasks, the algorithm assigns the label y that is most frequent among the K nearest neighbors to x. For regression tasks, the algorithm calculates the average or weighted average of the values y of the K nearest neighbors and assigns it as the predicted value for x.

ADVANTAGES:

Simplicity: KNN is a simple and intuitive algorithm that makes predictions based on the closest training examples in the feature space.

Non-Parametric: It does not assume any underlying distribution for the data, making it flexible and useful for various types of data distributions.

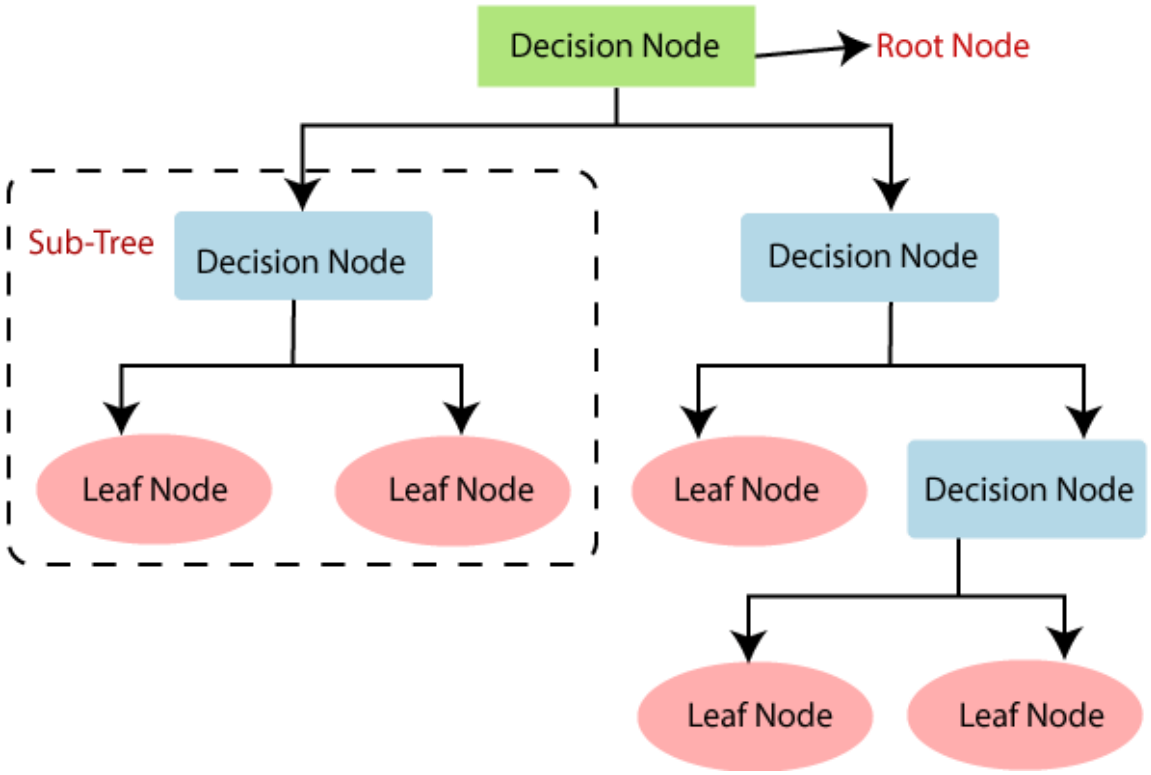
Interpretability: The predictions are easy to interpret as they are based on the majority vote among the nearest neighbors.

DISADVANTAGES:

Sensitive to feature scaling, high dimensionality

3.1.4.3]Decision Tree:

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.



For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Attribute Selection Measures(ASM):

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)
- Gini Index= 1- $\sum_j P_j^2$

ADVANTAGES:

Feature Importance: Like Random Forest, decision trees provide insights into feature importance, which can be valuable for understanding the factors influencing churn.

Handling Non-Linear Data: Decision trees can capture non-linear relationships between features and the target variable.

DISADVANTAGES:

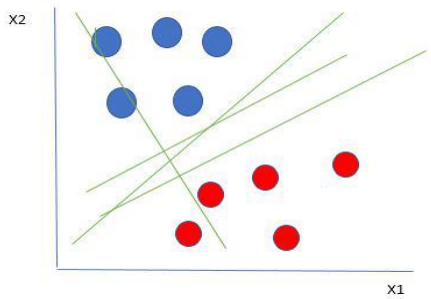
Prone to overfitting, may not capture complex relationships

3.1.4.4] Support Vector Classifier (SVC):

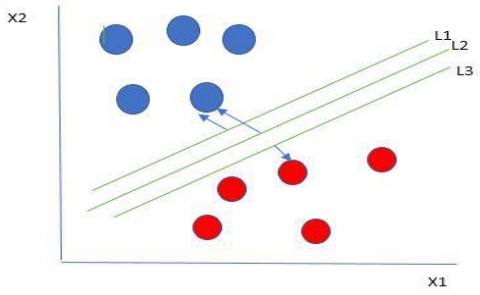
Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression.

The main objective of the SVM algorithm is to find the optimal hyperplane(the decision boundary that is used to separate the data points of different classes in a feature space) in an N-dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

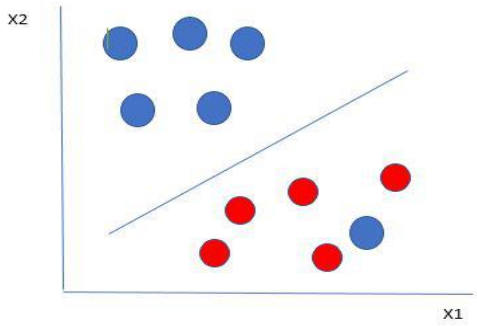
Let's consider two independent variables x1, x2, and one dependent variable which is either a blue circle or a red circle.



From the figure above it's very clear that there are multiple lines (our hyperplane here is a line because we are considering only two input features x1, x2) that segregate our data points or do a classification between red and blue circles. So how do we choose the best line or in general the best hyperplane that segregates our data points? One reasonable choice as the best hyperplane is the one that represents the largest separation or margin between the two classes.

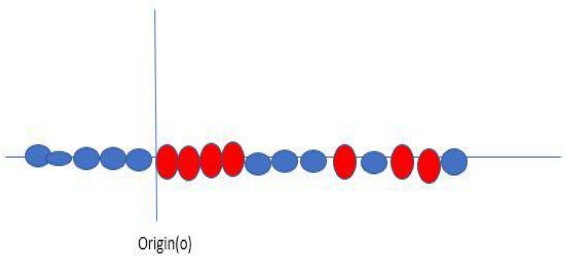


So we choose the hyperplane whose distance from it to the nearest data point on each side is maximized. If such a hyperplane exists it is known as the **maximum-margin hyperplane/hard margin**. So from the above figure, we choose L2. Let's consider a scenario like shown below

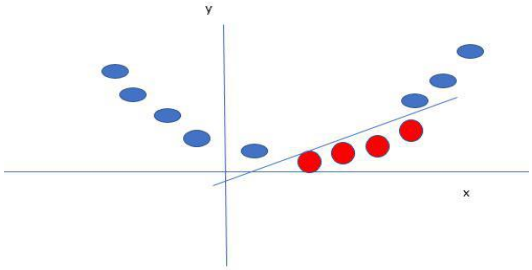


Here we have one blue ball in the boundary of the red ball. So how does SVM classify the data? It's simple! The blue ball in the boundary of red ones is an outlier of blue balls. The SVM algorithm has the characteristics to ignore the outlier and finds the best hyperplane that maximizes the margin. SVM is robust to outliers.

In case of data that cannot be linearly seperable:



Say, our data is shown in the figure above. SVM solves this by creating a new variable using a **kernel**. We call a point xi on the line and we create a new variable yi as a function of distance from origin o.so if we plot this we get something like as shown below



In this case, the new variable y is created as a function of distance from the origin. A non-linear function that creates a new variable is referred to as a kernel.

ADVANTAGES:

Margin Optimization: SVC works by finding the hyperplane that best separates the classes with the maximum margin, which can lead to better generalization on unseen data.

Kernel Trick: It allows the use of the kernel trick to handle non-linear data by transforming it into a higher-dimensional space where it becomes linearly separable.

DISADVANTAGES:

Requires careful parameter tuning, computationally expensive

3.1.4.5] Logistic Regression

This is a well-established approach for binary classification tasks like churn prediction. It's interpretable (coefficients indicate feature influence) and computationally efficient, but assumes a linear relationship between features and the target variable.

Linear Combination of Inputs:

- Logistic regression starts by taking a linear combination of the input features (like in linear regression).
- Suppose you have two features, x1 and x2. Logistic regression creates a formula like this: $z=b0 + b1 \cdot x1 + b2 \cdot x2$, where b0 is the intercept, and b1 and b2 are the coefficients.

Apply the Sigmoid Function:

- The linear combination z is then passed through a special function called the sigmoid function, which squeezes the output to be between 0 and 1.
- The sigmoid function is: Sigmoid(z)=1/(1+e^-z)

Interpreting the Output:

- The output of the sigmoid function is a probability value between 0 and 1.
- If the output is closer to 1, it means the model predicts a high probability that the input belongs to the positive class (e.g., 1 or yes).
- If the output is closer to 0, it means the model predicts a high probability that the input belongs to the negative class (e.g., 0 or no).

ADVANTAGES:

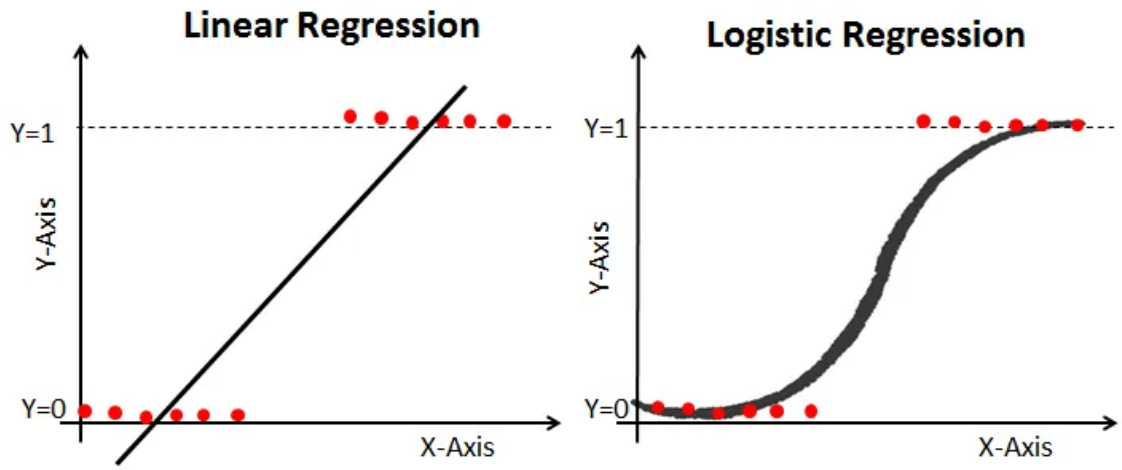
Simplicity and Speed: Logistic regression is a simple yet powerful model that is quick to train and can be very effective for binary classification problems like churn prediction.

Probabilistic Interpretation: It provides probabilities for class membership, offering a measure of confidence in the predictions, which can be useful for decision-making processes.

Baseline Model: Logistic regression often serves as a good baseline model to compare the performance of more complex models.

DISADVANTAGES:

Assumes linear relationships, may not be ideal for complex data



3.1.4.6] XGBoost:

XGBoost, short for e**X**treme **G**radient **Boost**ing, is a powerful and efficient open-source implementation of the gradient boosting algorithm. It's widely used for supervised learning tasks, particularly for regression and classification problems, due to its robustness, accuracy, and performance.XGBoost is an ensemble learning method that builds a strong predictive model by combining the outputs of several weaker models, typically decision trees. It improves the model by iteratively adding trees, focusing on correcting the errors made by the previous trees.

How Does XGBoost Work?

XGBoost works by building an ensemble of trees in a sequential manner, where each new tree aims to correct the errors of the existing ensemble. Here's a step-by-step explanation of how it works:

1. Initialize the Model

- **Start with Initial Prediction**: Begin with an initial prediction for all instances, typically using a uniform probability for classification.

2. Compute Residuals

- **Calculate Residuals**: For each instance, compute the residual (error) between the actual value and the predicted value. These residuals indicate how much the current model is off.

3. Fit a Decision Tree

- **Train a Decision Tree on Residuals**: Fit a new decision tree to the residuals. The goal of this tree is to predict the residuals (errors) from the previous step, effectively learning the patterns that were missed.

4. Update Predictions

- **Update Model**: Update the model's predictions by adding the predictions of the new tree, scaled by a learning rate (shrinkage parameter) to control the contribution of each tree.

5. Iterate

- **Repeat**: Repeat steps 2-4 for a specified number of iterations (trees) or until the model performance stops improving.

6. Regularization and Pruning

- **Apply Regularization**: Regularization terms are applied to the objective function to avoid overfitting.

  Regularization: XGBoost includes L1 (Lasso) and L2 (Ridge) regularization terms, which help prevent overfitting .

- **Tree Pruning**: Prune the trees to remove unnecessary branches, ensuring a simpler and more generalized model.

3.1.5] Model Evaluation Metrics:

1] .**Classification accuracy** is the accuracy we generally mean, whenever we use the term accuracy. We calculate this by calculating the ratio of correct predictions to the total number of input Samples.

2] Precision : There is another metric named Precision. Precision is a measure of a model's performance that tells you how many of the positive predictions made by the model are actually correct. It is calculated as the number of true positive predictions divided by the number of true positive and false positive predictions.

P= TP/(TP+FP)

3] Recall : Lower recall and higher precision give you great accuracy but then it misses a large number of instances. The more the F1 score better will be performance.

R =TP/(TP+FN)

4] **F1 Score** It is a harmonic mean between recall and precision. Its range is [0,1]. This metric usually tells us how precise (It correctly classifies how many instances) and robust (does not miss any significant number of instances) our classifier is.

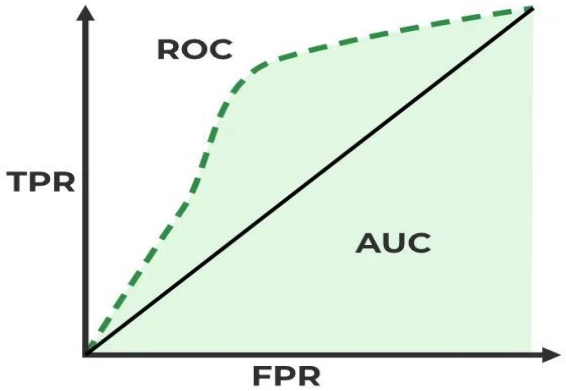
F1=(2*P*R)/(P+R)

5]**The AUC-ROC curve/ Area Under the Receiver Operating Characteristic curve**, is a graphical representation of the performance of a binary classification model at various classification thresholds. It is commonly used in machine learning to assess the ability of a model to distinguish between two classes, typically the positive class (e.g., presence of a disease) and the negative class (e.g., absence of a disease).

ROC stands for **Receiver Operating Characteristics**, and the ROC curve is the graphical representation of the effectiveness of the binary classification model. It plots the true positive rate (TPR) vs the false positive rate (FPR) at different classification thresholds.

AUC stands for the Area Under the Curve, and the AUC curve represents the area under the ROC curve. It measures the overall performance of the binary classification model. As both TPR and FPR range between 0 to 1, So, the area will always lie between 0 and 1, and A greater value of AUC denotes better model performance. Our main goal is to maximize this area in order to have the highest TPR and lowest FPR at the given threshold. The AUC measures the probability that the model will assign a randomly chosen positive instance a higher predicted probability compared to a randomly chosen negative instance.

It represents the probability with which our model can distinguish between the two classes present in our target.



- **TPR** – True Positive Rate
- **FPR** – False Positive Rate

3.2 IMPLEMENTATION

3.2.1] SAVING THE MODEL USING PICKLE

‘pickle’ is a Python module that provides a way to serialize and deserialize Python objects. Serialization is the process of converting a Python object into a byte stream, which can be saved to a file or sent over a network. Deserialization is the process of converting the byte stream back into a Python object.

Using pickle, you can save complex data structures such as lists, dictionaries, and even custom objects. This is useful for tasks like saving the state of a program, caching objects, or transferring data between different Python programs.

So here we save the model with the highest accuracy (logistic regression) using pickle and extension .sav

```
import pickle

# Serialize an object to a file

data = {'name': 'Alice', 'age': 30}

with open('data.pkl', 'wb') as file:

    pickle.dump(data, file)


# Deserialize the object from the file

with open('data.pkl', 'rb') as file:

    loaded_data = pickle.load(file)


print(loaded_data)
```

3.2.2] FRONTEND USING HTML,CSS AND FLASK

Flask is a micro-framework that provides a solid foundation for building web applications in Python. Its lightweight and flexible design allows developers to choose and integrate only the components they need, making it an excellent choice for a wide range of web development projects.

Lightweight Framework: Flask is a micro-framework, meaning it's lightweight and doesn't impose a lot of overhead. This makes it ideal for small to medium-sized projects like mine, where I need to quickly set up a web server and integrate my machine learning model.

Minimal Setup: Flask allows for easy and straightforward setup, letting me focus on developing the core functionalities of the project without being bogged down by complex configurations.

Seamless Integration with Machine Learning Models: Flask is excellent for integrating machine learning models. I was able to load my pre-trained model and set up routes to handle prediction requests efficiently.

Support for RESTful APIs: Flask provides strong support for building RESTful APIs, which is essential for serving my machine learning predictions as a web service.

Flask applications can be easily deployed on various platforms, including cloud services like Heroku, AWS, and Google Cloud Platform. This flexibility ensured that I could deploy my application with minimal hassle.

The Flask application**(app.js)** is designed to predict customer churn based on the input provided by the user through an HTML form**(index.html).**

Here's an overview of how it works:

1. **Setup:** Flask application (app.py) is set up with necessary libraries and dependencies, including loading the trained machine learning model (lr_xg_model.sav).

2. **Home Route (/):** When the user accesses the root URL, the home() function is triggered. This function renders the index.html template, which contains a form for inputting customer data.
3. **Prediction Route (/predict):** When the user submits the form, the predict() function is triggered. This function extracts the input values from the form and processes them to match the format expected by the model.
4. **Data Processing:** The input values are converted to the appropriate data types (e.g., integer, float, boolean) based on the model's requirements.
5. **Model Prediction:** The processed data is then fed into the loaded machine learning model (model) for prediction. The model predicts whether the customer will churn or not.
6. **Result Display:** Depending on the prediction result (False for not churn, True for churn), an appropriate message is displayed on the web page.
7. **HTML Templates:** The HTML templates (index.html) are used to create the user interface for entering data and displaying the prediction result.
8. **Styling:** CSS styles are applied to the HTML elements to improve the visual appeal and user experience.
9. **Running the Application:** Finally, the Flask application is run with the python app.py on the terminal , which starts the web server and makes the application accessible through a web browser.

CHAPTER 4 : RESULTS AND DISCUSSION

Model Performance:

Model	Accuracy	Precision (False Class)	Recall (True Class)	F1- Score (False Class)	ROC-AUC
Logistic Regression	0.78	0.88	0.69	0.85	0.75
Support Vector Classifier (SVC)	0.74	0.84	0.56	0.83	0.68
Decision Tree	0.72	0.83	0.57	0.80	0.67
KNN	0.772	0.83	0.51	0.85	0.689
Random Forest	0.79	0.85	0.59	0.86	0.72
XGBoost	0.77	0.88	0.61	0.85	0.72

Considering the F1-score, here are some possibilities:

- High Accuracy & Prioritise Non-Churners: Logistic Regression or Random Forest could be good choices if minimising false alarms (incorrectly identifying non-churners as churners) is crucial.
- Balance & Avoiding False Alarms: KNN offers a balance between identifying churners and avoiding false alarms, making it a good all-around option.
- Focus on Identifying Churners: If capturing as many churners as possible is essential, Random Forest might be suitable (but be aware of potential false positives).

Choosing the Best Model

Based on these metrics, here's an analysis to determine the best model:

1. **Accuracy**: Logistic Regression and Random Forest both have the highest accuracy (0.78).

2. **Precision**: Both Logistic Regression and XGBoost have higher precision for the False class and similar precision for the True class.
3. **Recall**: Logistic Regression both has high recall for the True class.
4. **F1-Score**: Logistic Regression and XGBOOST and Random Forest have the highest F1-scores for both classes.
5. **ROC-AUC Score**: Logistic Regression has the highest ROC-AUC score (0.75).

Based on the performance metrics provided:

Logistic Regression appears to be the best overall model due to its high accuracy, balanced precision and recall, high F1-scores, and the highest ROC-AUC score. This indicates it has a good balance between sensitivity (recall) and specificity, and it provides a reliable measure of overall performance.

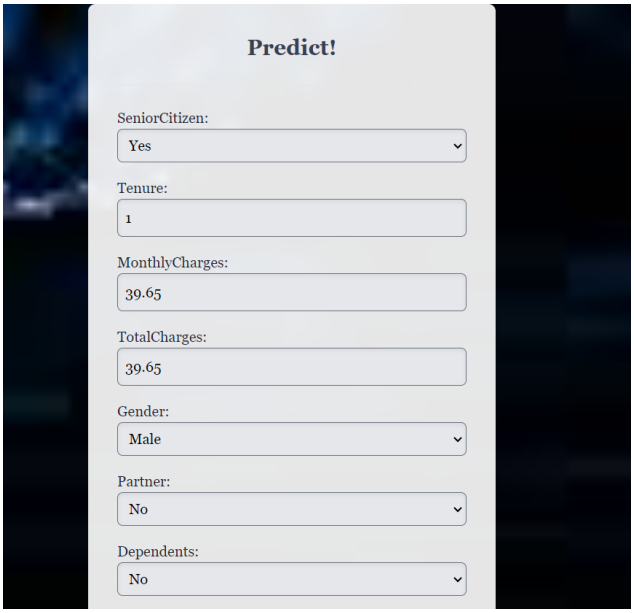
Why Choose Logistic Regression?

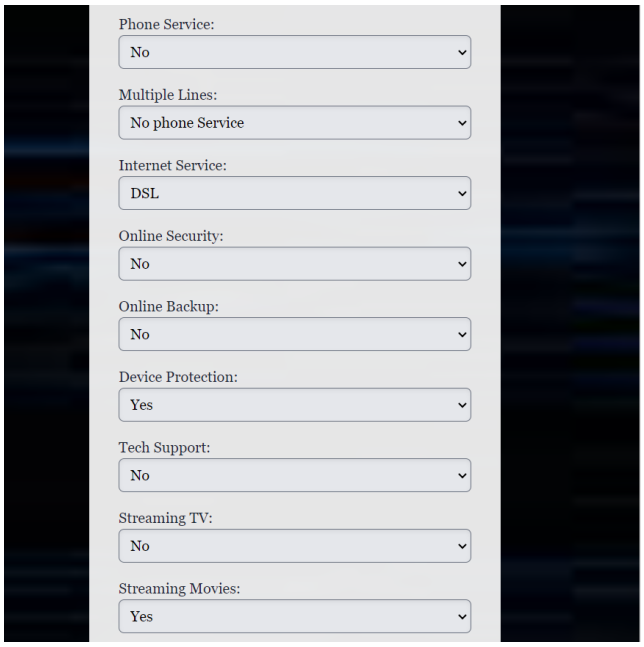
- **High Accuracy**: It has one of the highest accuracy rates among the models tested.
- **Balanced Performance**: It has balanced precision and recall, ensuring that both classes are well-represented.
- **ROC-AUC Score**: It has the highest ROC-AUC score, which means it performs well in distinguishing between the churn and non-churn classes.

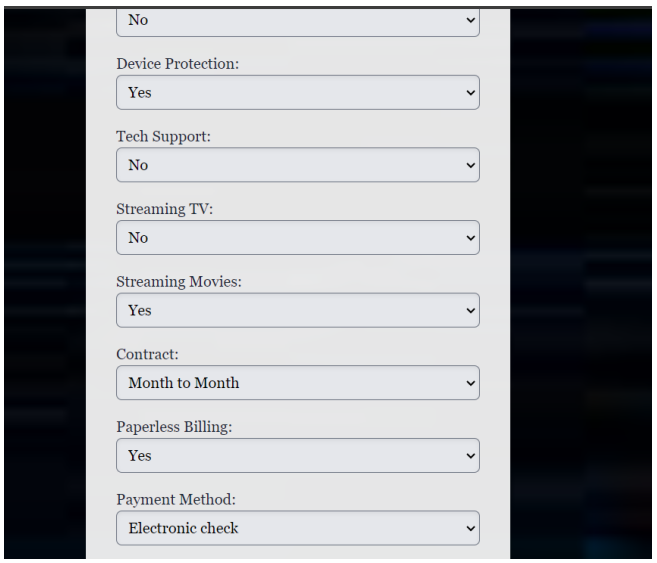
Additional Considerations:

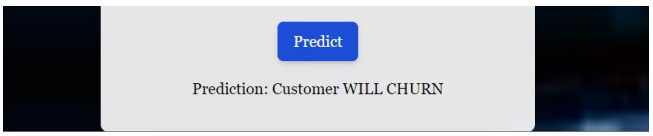
- Interpretability: If understanding the factors driving churn is important, Logistic Regression or Decision Tree might be preferable.
- Business Costs: Analyse the cost of misidentifying a non-churner versus missing a true churner. This can help you decide how much weight to give precision vs. recall.
- Model Complexity: Consider the trade-off between model complexity and interpretability. Simpler models like Logistic Regression or Decision Tree might be easier to interpret, while complex models like Random Forest might offer better performance.

TEST CASE 1:

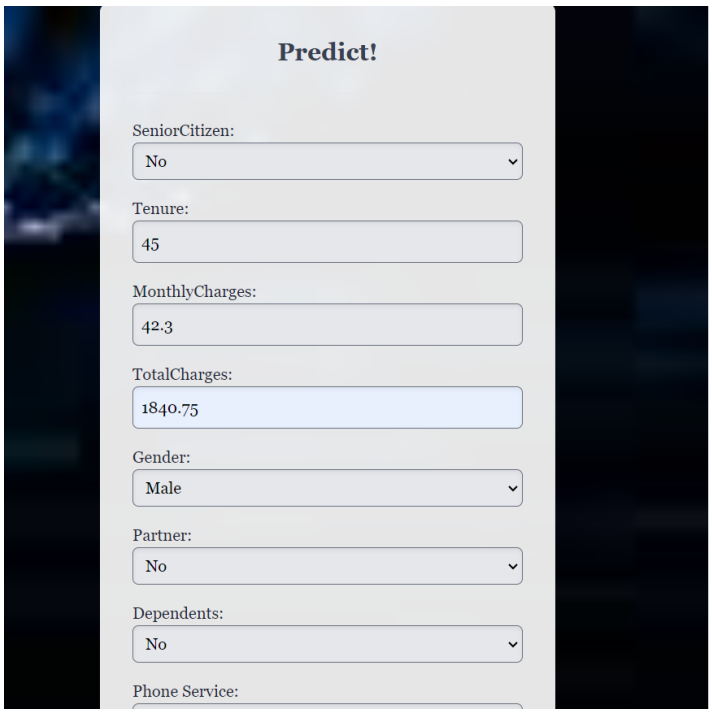

Predict!
SeniorCitizen:
Yes
Tenure:
1
MonthlyCharges:
39.65
TotalCharges:
39.65
Gender:
Male
Partner:
No
Dependents:
No


Phone Service:
No
Multiple Lines:
No phone Service
Internet Service:
DSL
Online Security:
No
Online Backup:
No
Device Protection:
Yes
Tech Support:
No
Streaming TV:
No
Streaming Movies:
Yes

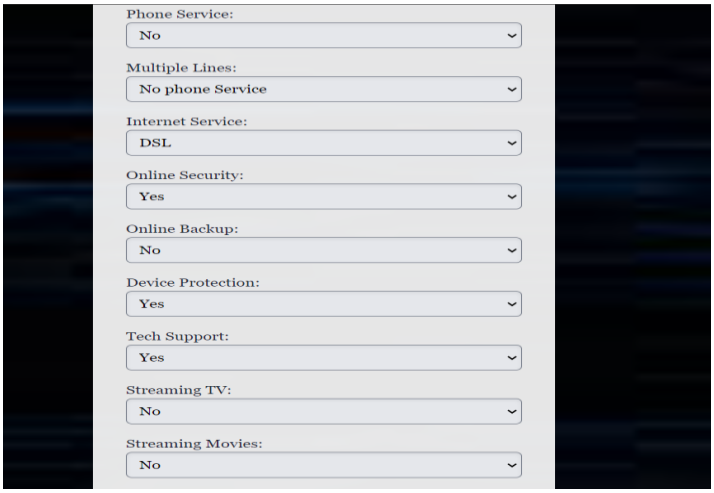

No
Device Protection:
Yes
Tech Support:
No
Streaming TV:
No
Streaming Movies:
Yes
Contract:
Month to Month
Paperless Billing:
Yes
Payment Method:
Electronic check

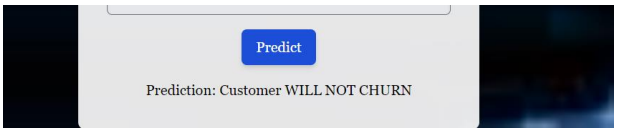

Predict
Prediction: Customer WILL CHURN

TEST CASE 2


Predict!
SeniorCitizen:
No
Tenure:
45
MonthlyCharges:
42.3
TotalCharges:
1840.75
Gender:
Male
Partner:
No
Dependents:
No
Phone Service:


Contract:
One year
Paperless Billing:
No
Payment Method:
Bank Transfer


Phone Service:
No
Multiple Lines:
No phone Service
Internet Service:
DSL
Online Security:
Yes
Online Backup:
No
Device Protection:
Yes
Tech Support:
Yes
Streaming TV:
No
Streaming Movies:
No


Predict
Prediction: Customer WILL NOT CHURN

LIMITATIONS OF OUR WORK

1.Limited Predictive Power: Binary output models simplify predictions to a yes/no decision, potentially overlooking nuanced patterns in customer behavior that could affect churn.

2.Misclassification of Customers with Security Deposits: Customers who made security deposits might be misclassified as not churning, leading to false negatives.

3.Lack of Contextual Information: Binary output models may not capture the full context of customer interactions and preferences, leading to simplistic predictions that overlook important factors influencing churn.

CHAPTER 5 : CONCLUSION & FUTURE WORK

Conclusion:-

The implementation of various data preprocessing techniques, including data cleaning, feature engineering, and standardisation, laid the foundation for effective churn prediction using the Telco Customer Churn Dataset. By addressing the challenge of imbalanced data through the SMOTE technique, we mitigated the risk of model bias towards the majority class and improved the overall performance of the predictive models.

Five different models, namely Random Forest, K-Nearest Neighbors, Decision Tree, Support Vector Classifier, and Logistic Regression, were evaluated alongside XGBoost, a powerful gradient boosting algorithm. Performance evaluation metrics, including accuracy, F1 score, precision, recall, and the AUC-ROC curve, provided insights into each model's ability to predict customer churn accurately.

Based on the comprehensive analysis of model performance, XGBoost emerged as the most suitable candidate for churn prediction in this context. Its ability to handle complex relationships, effectively manage imbalanced datasets, and achieve high accuracy makes it well-suited for real-world applications.

The integration of the selected model with Flask enables the deployment of a user-friendly web application, allowing stakeholders to leverage churn predictions for informed decision-making and proactive customer retention strategies.

Future Work:-

While this study lays a solid foundation for customer churn prediction, there are several avenues for future research and improvement:

1. **Feature Engineering:** Exploring additional features or engineered variables could enhance model performance and capture more nuanced patterns in customer behaviour.
2. **Model Tuning:** Fine-tuning hyperparameters and optimising model architectures can further improve predictive accuracy and robustness.

3. **Ensemble Methods:** Investigating ensemble techniques, such as stacking or blending multiple models, may yield superior performance by leveraging the strengths of individual algorithms.
4. **Temporal Analysis:** Incorporating time-series analysis and considering the temporal dynamics of customer churn could provide deeper insights into churn prediction and enable proactive intervention strategies.
5. **Interpretability:** Enhancing the interpretability of the predictive models can facilitate the understanding of key drivers influencing churn and aid in decision-making processes.
6. **Deployment and Monitoring:** Continuous monitoring of model performance post-deployment is crucial for ensuring its effectiveness in real-world scenarios. Implementing mechanisms for model updating and retraining based on incoming data can help maintain predictive accuracy over time.
7. **Domain-Specific Considerations:** Tailoring the churn prediction model to specific industry verticals or customer segments may yield more targeted and actionable insights.
8. **Predicting Exact Churn Time:** Extend the current model to predict not only whether a customer will churn but also the exact time or the estimated time frame when the churn is likely to happen. This could involve using time-to-event analysis techniques such as survival analysis, which is commonly used in medical research to predict the time until an event occurs. Implementing this would require additional historical data about customer behavior over time.

   In future work, enhancing the predictive capabilities of the current model to provide not only binary churn predictions but also quantifying the likelihood of churn as a percentage could offer stakeholders more nuanced insights for informed decision-making and proactive customer retention strategies

CHAPTER 6 : REFERENCES

References

[1] - Ning Lu, Hua Lin, Jie Lu, Guangquan Zhang (May 2014.)"A Customer Churn Prediction Model in Telecom Industry Using Boosting", IEEE Transactions on Industrial Informatics, vol. 10, no.2
https://ieeexplore.ieee.org/abstract/document/6329952

[2] - Farquad, H. &Vadlamani, Ravi &Surampudi, Bapi. (2014). "Churn Prediction using Comprehensible Support Vector Machine: an Analytical CRM Application." Applied Soft Computing. 19. 10.1016/j.asoc.2014.01.031
https://www.sciencedirect.com/science/article/abs/pii/S1568494614000507

[3] - Nabahirwa Edwine, Wenjuan Wang, Wei Song, Denis Ssebuggwawo, (2022) "Detecting the risk of customer churn in the telecom sector: a comparative study, Math. Probl Eng". 2022. Article ID 8534739, 16 pages.
https://onlinelibrary.wiley.com/doi/10.1155/2022/8534739

[4] - Edvaldo Domingos, Blessing Ojeme, Olawande Daramola,(2021) "Experimental analysis of hyperparameters for deep learning-based churn prediction in the banking sector", Computation 9 (3) 34.
https://www.mdpi.com/2079-3197/9/3/34