ACADEMIC CITY UNIVERSITY COLLEGE

INTRODUCTION TO ARTIFICIAL INTELLIGENCE.

MID-SEMESTER EXAMS.

PREDICTING CUSTOMER CHURN FOR A TELECOM COMPANY.

12th March, 2024

GROUP 2

Bervelyn Sawyerr-Markwei - 10201100095

Delasie Kwadjo Bansah - 10201100123

Mohammed Coulibaly - 10201100157

TABLE OF CONTENTS

INTRODUCTION.

Imagine a telecom industry where customers constantly switch providers. This "revolving door" effect, known as churn, is a major drain on profits. Customer churn is a major problem and one of the most important concerns for large telecom companies. Due to the direct effect on companies' revenues, especially in the telecom field, companies seek to develop means to predict potential customer churn. Studies show keeping existing customers is much cheaper than acquiring new ones [1]. That's why predicting churn is crucial for telecom companies to hold onto their subscriber base.

This report dives into the world of customer churn prediction for telecom networks. We will explore what makes customers leave, analyze a dataset on customer churn, do some exploratory data analysis on the dataset, utilise statistical tests and decision trees to identify significant predictors of churn, and prepare the final dataset for modelling by selecting relevant features.
Then, we will split the dataset into training and testing sets, develop a logistic regression model to predict customer churn, and evaluate the model's performance using appropriate metrics. We will then use the decision tree model as a supplementary approach to identify the primary factors causing churn and propose actionable strategies to reduce customer churn based on the findings of the model. By proactively spotting customers about to jump ship, telecom companies can create targeted strategies to win them back, ultimately reducing churn and securing financial health.

OBJECTIVE OF THIS PROJECT.

This project aims to develop a predictive model that utilizes logistic regression and decision tree algorithms to predict the likelihood of customer churn for a telecom company (make it interpretable). By identifying customers at high risk of churning, the company can take proactive measures to address the underlying issues and improve customer retention. Additionally, understanding the key factors that contribute to churn will enable targeted interventions.

LITERATURE REVIEW.

This section dives deep into existing research to uncover the best practices for predicting customer churn in telecom companies. We will be examining various studies to understand the factors in the telecom industry that influence customer churn, how machine learning algorithms can be used to identify customers likely to churn, and how researchers assess the effectiveness of churn prediction models. By piecing together these studies, this review aims to paint a clear picture of the current landscape of customer churn prediction in telecom. This knowledge can empower both researchers and telecom companies to develop even more powerful strategies to keep their valued customers on board.

Lewlisa Saha et al. came up with a way to predict the churn percentage of customers with higher accuracy without comprising profit. In their study, various types of learning strategies were investigated to address this challenge and build a churn prediction model. The techniques that were tested and used to select the best model for building the customer churn prediction model were ensemble learning techniques (Adaboost, random forest (RF), extreme randomized tree (ERT), xgboost (XGB), gradient boosting (GBM), and bagging and stacking), traditional classification techniques (logistic regression (LR), decision tree (DT), and k-nearest neighbor (kNN), and artificial neural network (ANN)), and the deep learning convolutional neural network (CNN). The evaluation of the model was conducted using two datasets: the Southeast Asian telecom industry and American telecom market datasets [2].

They came up with a research model that would be used to determine the rate of customer churn. The amount of recharge over a specific time determined the high-value customers for whom the

churn was predicted in the first dataset. There are three distinct phases to this period: the good phase, during which the customer is content and acts normally; the action phase, during which the customer's experience becomes strained; and the churn phase, during which the customer has withdrawn. The dataset utilized in their study was four months' worth of data: July, August, September, and June. Here, June and July represented the good phase, and the high-value clients were identified by examining the amount recharged during these two months. The amount that was utilized to denote a high value in this instance was also greater than the 70th percentile of the average recharge amount. The churn tag was applied to the churn instances, as indicated in Figure 1, following the removal of high-value customers, and new feature extraction was carried out. These preprocessing stages were not utilized in the second dataset; instead, the data was cleaned by eliminating redundant information [2][3].

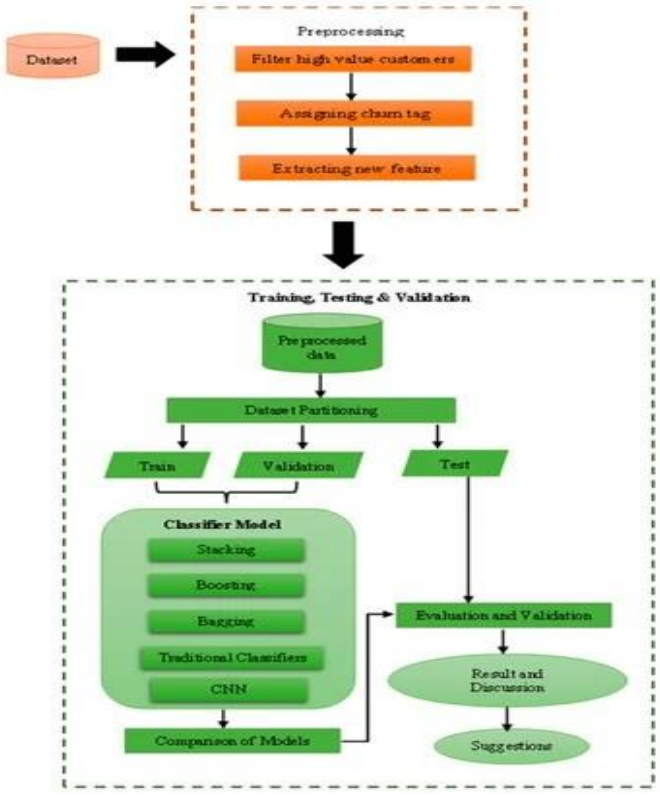


Figure 1.The Proposed Research Model by Lewlisa Saha et al [2]

After the dataset was cleaned, several machine-learning techniques were applied to create predictions. An ensemble model for stacking was one of the categorization techniques used by the predictive system. Figure 2 illustrates the two levels of the stacking model. Level 0 consisted of DT, RF, LR, kNN, and SVM, while Level 1 consisted of LR. Soft voting was used to predict the final result. Other ensemble models that were employed included RF, ERT, AdaBoost, GBM, XGB, and Bagging. CNN was the deep learning technique employed, and the conventional classifiers used, aside from ensemble learning, were DT, kNN, LR, and ANN. Performance measures such as accuracy, precision, F1 score, sensitivity, specificity and AUC-ROC were also used in the study [2][3].

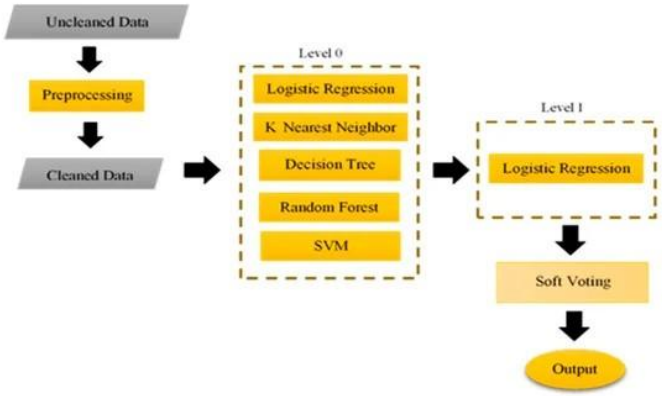


Figure 2. The Proposed Stacking Model by Lewlisa Saha et al [2]

Upon testing, CNN and ANN had the best accuracy, at 99% and 98% on the first dataset, and 98 % and 99% on the second dataset. They both produced an AUC score of 0.99 on the first dataset and 0.98 and 0.96 on the second dataset, respectively. From their results, they concluded that AI-driven customer churn prediction, as well as machine learning and smart data, could help telecom companies know their customers and understand the demands of those customers.

A highly effective hybridized algorithmic technique for churn prediction was presented by Ammar et al. [4]. The accuracy attained by the suggested algorithm and the standard Firefly algorithm was similar and comparable. However, in terms of very low latency, the hybrid firefly algorithm outperformed the standard firefly algorithm. These algorithms' effects on F-Mean, Accuracy, Time, PR, and ROC were investigated. Telecommunications companies should have prediction models that are both accurate and efficient [5]. Reducing the size and dimensions of the model to its optimal can improve its efficiency. Accurate prediction and efficient feature extraction are facilitated by the employed feature selection and extraction techniques.

The study thus demonstrates that despite having a smaller size (232 KB), the prediction model was still able to complete tasks of a similar nature and produce results that were 92% more accurate than the original model (303 KB). A churn classification model was proposed using the multilayer perceptron neural network concept, which demonstrated the utility of both logistic regression and logistic boost in the construction of a churn prediction model. Research was done on efficient classifiers like random forests, decision tree classifiers, and KNNs. The four phases of classifiers were data access, data wrangling, training, and insight acquisition. Every element was significant in the analysis of the churn prediction data. Here, the suggested model received input from data access. Distributed data collection involves the use of data wrangling [6]. The model built the training procedure, and the test data set was used to process and gain insights that helped predict the final output and examine the churn prediction's outcome. Following the experiment, it was observed that higher monthly rates were associated with higher churn. There were significant churns, even with low overall charges. Higher monthly charges at shorter tenure

led to lower total charges when all three parameters—Total Charges, Monthly Cost, and Tenure—were combined, suggesting that these traits were all linked to higher churn [6].

When doing the gap analysis for this report, we realized that in the model proposed by Lewlisa Saha et al., the bagging, DT, kNN, LR, XGB, GBM, ERT, AdaBoost, and RF. Except for AdaBoost, DT, kNN, and LR techniques, which underperformed on the second dataset despite producing a good result on the first performed admirably on both datasets, with accuracies exceeding 90% for each. Given that the dataset they used had fewer churning customers than non-churning customers, it is clear that the churn distribution was wildly unbalanced and the performance of the applied model was severely disrupted by this data imbalance. The only way to solve this imbalance in our model is to use a dataset with a higher churn percentage or a higher number of churn customers in training the model.

From the model implemented by Ammar et al., we realized that after comparing the normal Firefly algorithm to the proposed model, the proposed model performed very well with high metric scores (accuracy, precision, F1 score). The only downside was that the rate of false positives was high, and also the total model runtime was slow. To solve the imbalance in our model, we need to introduce epochs in our model so as to track the runtime.

METHODOLOGY.

This section of the report shows how we developed a customer churn prediction model by examining a dataset on customer churn, performing exploratory data analysis, finding significant churn predictors using statistical tests and decision trees, and preparing the final dataset for modeling by choosing pertinent features. After dividing the dataset into training and testing sets, we created a logistic regression model to forecast customer attrition and assess the model's effectiveness with the help of relevant metrics. The decision tree model was then used as an additional method to determine the main reasons behind customer churn. All this was done using a jupyter notebook (.ipynb file).

Data Preprocessing Phase.

The process of creating unprocessed raw data for machine learning models is known as data preprocessing. The process of developing a machine-learning model starts with this. The most difficult and time-consuming part of data science is this. To simplify machine learning algorithms, data preprocessing is necessary [7]. It has 6 steps;

- Importing Libraries: This is the stage where the necessary libraries that would be used in the machine learning model are imported. A library is a set of functions that can be called into and used in the algorithms [7]. Some of the libraries that would be used here include numpy, pandas, seaborn, sklearn, matplotlib.
- Importing Datasets: This is the stage where the datasets are imported into the model for usage[7]. In his project, we would be using a dataset on Telco Customer Churn from Kaggle.

- Check for missing values: This is the stage whereby the columns in the dataset are checked to see if there are missing values. If there are, those values are estimated by taking the mean, median or mode of the column. Or in large datasets, the entire row can be removed [7].
- Arrange the Data: At this stage, the data is arranged in numerical form to prevent any error in the latter stages of the model. At this stage, all text values are converted to numerical form [7].
- Scaling: At this stage, scaling is done to convert the data values into shorter ranges through rescaling and standardization [7].
- Distribution of Data into Training, Evaluation and Validation Sets

In our work, we first imported the libraries.

Figure 3. Import Libraries

Then we imported the dataset. Since the dataset contained 7043 rows, we could not display all, so we displayed just the first 5 rows, as seen in the diagram below.

a. Import the dataset.

```
[2]: churn_data = pd.read_csv('WA_Fn-UseC_-Telco-Customer-Churn.csv')

[3]: # Display the first few rows of the dataframe.
     print("First few rows of the dataset:")
     print(churn_data.head())
```

```
First few rows of the dataset:
   customerID  gender  SeniorCitizen Partner Dependents  tenure PhoneService  \
0  7590-VHVEG  Female              0     Yes         No       1           No

                                                                             1
```

```
1  5575-GNVDE    Male              0      No         No      34          Yes
2  3668-QPYBK    Male              0      No         No       2          Yes
3  7795-CFOCW    Male              0      No         No      45           No
4  9237-HQITU  Female              0      No         No       2          Yes

      MultipleLines InternetService OnlineSecurity  … DeviceProtection  \
0  No phone service             DSL             No  …               No
1                No             DSL            Yes  …              Yes
2                No             DSL            Yes  …               No
3  No phone service             DSL            Yes  …              Yes
4                No     Fiber optic             No  …               No

  TechSupport StreamingTV StreamingMovies        Contract PaperlessBilling  \
0          No          No              No  Month-to-month              Yes
1          No          No              No        One year               No
2          No          No              No  Month-to-month              Yes
3         Yes          No              No        One year               No
4          No          No              No  Month-to-month              Yes

              PaymentMethod MonthlyCharges  TotalCharges Churn
0          Electronic check          29.85         29.85    No
1              Mailed check          56.95        1889.5    No
2              Mailed check          53.85        108.15   Yes
3  Bank transfer (automatic)          42.30       1840.75    No
4          Electronic check          70.70        151.65   Yes

[5 rows x 21 columns]
```

Figure 4. First 5 rows.

After doing that, we checked the dataset information. Upon doing that, we realized the dataset contains the following information:

1. Customer ID: A unique ID that identifies each customer
2. Gender: The customer's gender
3. Senior Citizen: Indicates if the customer is 65 or older
4. Partner: Indicates if the customers have a sharing account
5. Dependent: Indicates if the customer lives with any dependents. Dependents could be children, parents, grandparents, etc.
6. Tenure: How long the customer has been a customer

7. Phone Service: Indicates if the customer subscribes to home phone service with the company
8. Multiple Lines: Indicates if the customer subscribes to multiple telephone lines with the company
9. Internet Service: Indicates if the customer subscribes to the Internet service
10. Online Security: Indicates if the customer subscribes to an additional online security service
11. Online Backup: Indicates if the customer subscribes to an additional online backup service
12. Device Protection: Indicates if the customer subscribes to additional device protection for their Internet equipment
13. Tech Support: Indicates if the customer subscribes to a technical support service
14. Streaming TV: Indicates if the customer uses their Internet service to stream television programming from a third-party provider
15. Streaming Movies: Indicates if the customer uses their Internet service to stream movies from a third-party provider
16. Contract: Indicates the customer's current contract type
17. Paperless Billing: Indicates if the customer has chosen paperless billing
18. Payment Method: Indicates how the customer pays their bill
19. Monthly Charges: Indicates the customer's current total monthly charge for all their services from the company
20. Total Charges: Indicates the customer's total charges
21. Churn: Indicates whether the customer has left

We also checked the data types found in each column.

```
[4]: #Checking data info.
     churn_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customerID        7043 non-null   object
 1   gender            7043 non-null   object
 2   SeniorCitizen     7043 non-null   int64
 3   Partner           7043 non-null   object
 4   Dependents        7043 non-null   object
 5   tenure            7043 non-null   int64
 6   PhoneService      7043 non-null   object
 7   MultipleLines     7043 non-null   object
 8   InternetService   7043 non-null   object
 9   OnlineSecurity    7043 non-null   object
 10  OnlineBackup      7043 non-null   object
 11  DeviceProtection  7043 non-null   object
```



```
 12  TechSupport       7043 non-null   object
 13  StreamingTV       7043 non-null   object
 14  StreamingMovies   7043 non-null   object
 15  Contract          7043 non-null   object
 16  PaperlessBilling  7043 non-null   object
 17  PaymentMethod     7043 non-null   object
 18  MonthlyCharges    7043 non-null   float64
 19  TotalCharges      7043 non-null   object
 20  Churn             7043 non-null   object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

b. Conduct preliminary data exploration.

```
[5]: # Data exploration.
     print("\nData exploration:")
     print("Dataset shape:", churn_data.shape)
     print("\nData types:")
     print(churn_data.dtypes)

Data exploration:
Dataset shape: (7043, 21)

Data types:
customerID           object
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges         object
Churn                object
dtype: object
```

Figure 5. Dataset Info and Data types

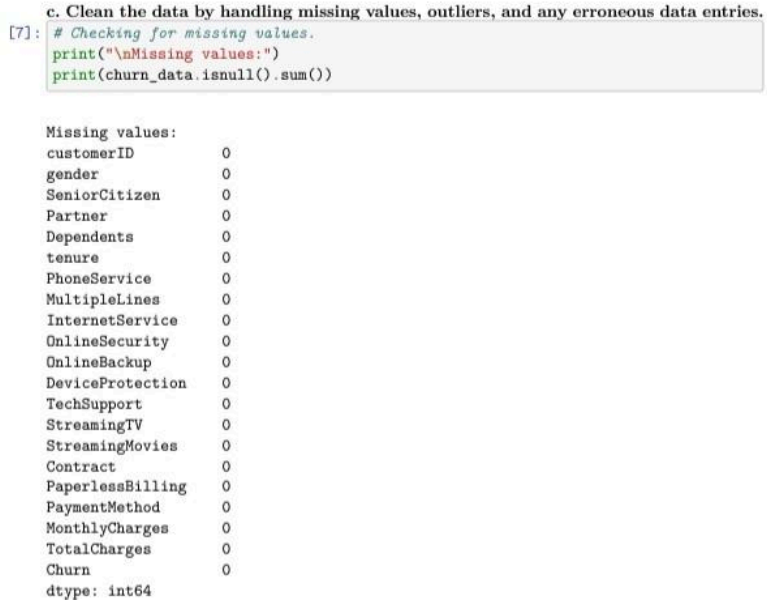
Then we used the .describe() function to check the statistical features of the numerical columns (columns whose type were int and float)

```
[6]: # Summary statistics.
     print("\nSummary statistics:")
     print(churn_data.describe())

Summary statistics:
       SeniorCitizen       tenure  MonthlyCharges
count    7043.000000  7043.000000     7043.000000
mean        0.162147    32.371149       64.761692
std         0.368612    24.559481       30.090047
min         0.000000     0.000000       18.250000
25%         0.000000     9.000000       35.500000
50%         0.000000    29.000000       70.350000
75%         0.000000    55.000000       89.850000
max         1.000000    72.000000      118.750000
```

Figure 6. Statistics of numerical columns

We then checked for missing values using the .isnull().sum() function but found none.



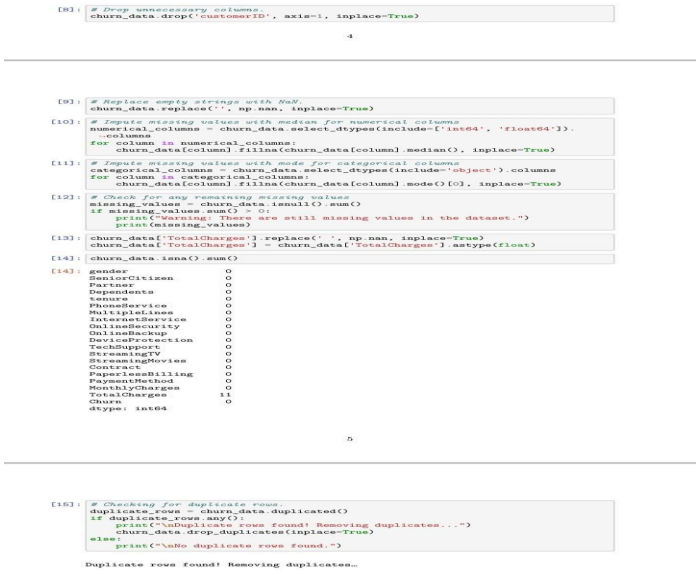
c. Clean the data by handling missing values, outliers, and any erroneous data entries.

```
[7]: # Checking for missing values.
     print("\nMissing values:")
     print(churn_data.isnull().sum())
```

```
Missing values:
customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

Figure 7. Check for missing values.

We proceeded to drop unnecessary columns if any were present, replace empty strings with NaN, and input missing values with the median for numerical columns and with mode for categorical columns. We then did a check on missing values again and checked for duplicates. There were duplicates, so we removed them.



```
[8]: # Drop unnecessary columns.
     churn_data.drop('customerID', axis=1, inplace=True)
```

4

```
[9]: # Replace empty strings with NaN.
     churn_data.replace('', np.nan, inplace=True)

[10]: # Impute missing values with median for numerical columns
      numerical_columns = churn_data.select_dtypes(include=['int64', 'float64']).
        .columns
      for column in numerical_columns:
          churn_data[column].fillna(churn_data[column].median(), inplace=True)

[11]: # Impute missing values with mode for categorical columns
      categorical_columns = churn_data.select_dtypes(include='object').columns
      for column in categorical_columns:
          churn_data[column].fillna(churn_data[column].mode()[0], inplace=True)

[12]: # Check for any remaining missing values
      missing_values = churn_data.isnull().sum()
      if missing_values.sum() > 0:
          print("Warning: There are still missing values in the dataset.")
          print(missing_values)

[13]: churn_data['TotalCharges'].replace(' ', np.nan, inplace=True)
      churn_data['TotalCharges'] = churn_data['TotalCharges'].astype(float)

[14]: churn_data.isna().sum()

[14]: gender             0
      SeniorCitizen      0
      Partner            0
      Dependents         0
      tenure             0
      PhoneService       0
      MultipleLines      0
      InternetService    0
      OnlineSecurity     0
      OnlineBackup       0
      DeviceProtection   0
      TechSupport        0
      StreamingTV        0
      StreamingMovies    0
      Contract           0
      PaperlessBilling   0
      PaymentMethod      0
      MonthlyCharges     0
      TotalCharges      11
      Churn              0
      dtype: int64
```

5

```
[15]: # Checking for duplicate rows.
      duplicate_rows = churn_data.duplicated()
      if duplicate_rows.any():
          print("\nDuplicate rows found! Removing duplicates...")
          churn_data.drop_duplicates(inplace=True)
      else:
          print("\nNo duplicate rows found.")
```

```
Duplicate rows found! Removing duplicates...
```

Figure 8. Check for duplicates.

For the final part of this phase, we encoded categorical variables to make the application of the logistic regression model easier. We did that using the LabelEncoder() function.

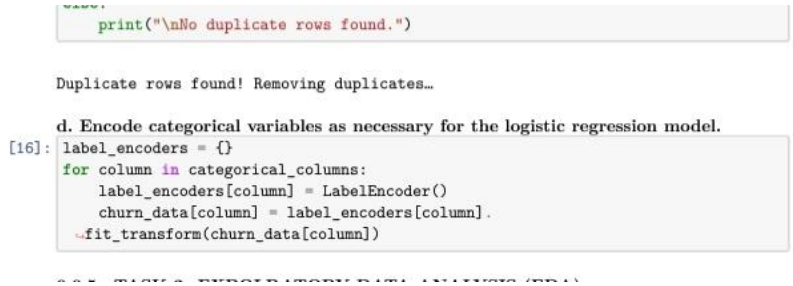


Figure 9. Encoding Categorical variables

Exploratory Data Analysis (EDA).

At this stage, an in-depth analysis of the dataset characteristics was done to compare some parameters against others. We did this by first creating a distribution for the numerical variables and plotting the results in the form of a histogram.

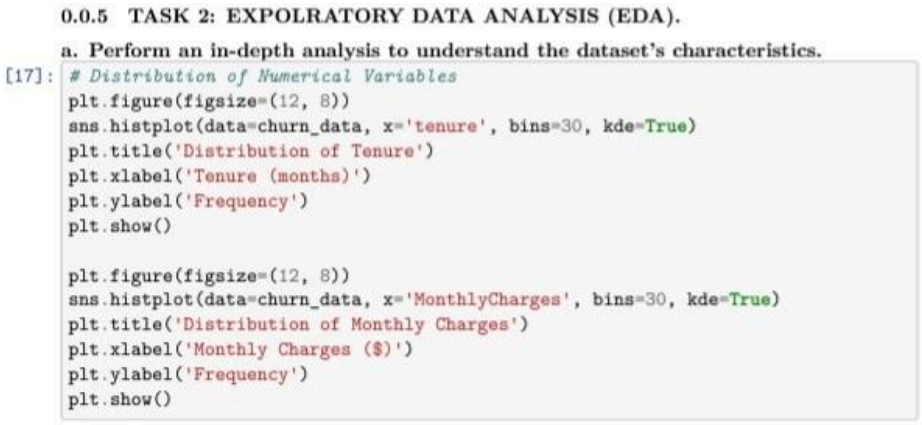


Figure 10. Distribution of Numerical Variables

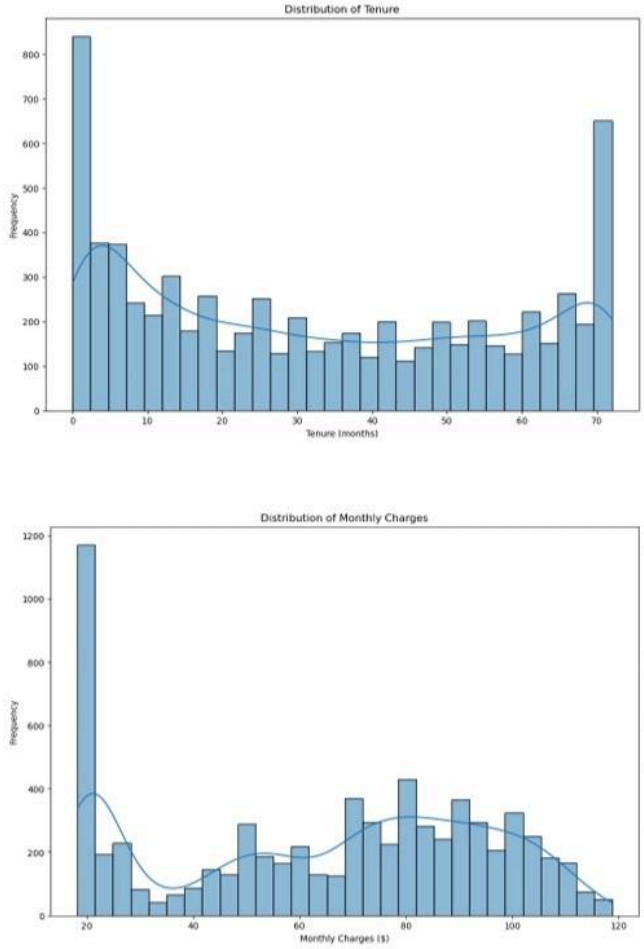


Figure 11. Results from Distribution for Tenure and Monthly Charges

The same was done for categorical variables such as contract and payment methods as well. But the results were represented with a bar chart this time.

```
[18]: # Distribution of Categorical Variables.
      plt.figure(figsize=(10, 6))
      sns.countplot(data=churn_data, x='Contract')
      plt.title('Distribution of Contract Types')
      plt.xlabel('Contract Type')
      plt.ylabel('Count')
      plt.show()

      plt.figure(figsize=(10, 6))
      sns.countplot(data=churn_data, x='PaymentMethod')
      plt.title('Distribution of Payment Methods')
      plt.xlabel('Payment Method')
      plt.ylabel('Count')
      plt.xticks(rotation=45)
      plt.show()
```

Figure 12. Distribution of Categorical Variables.

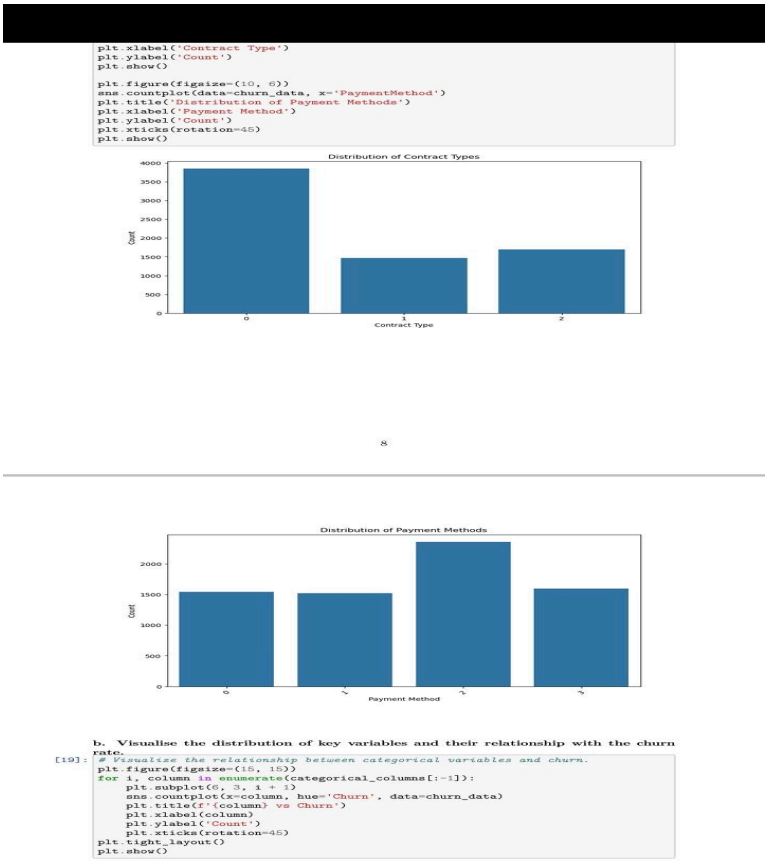


Figure 13. Results from Distribution of Contract Types and Payment Methods

After this step, the next step was to visualize the distribution of the key variables and compare it to the churn rate. We used subplot and countplot to make the interpretation easier.

b. Visualise the distribution of key variables and their relationship with the churn rate.

```
[19]: # Visualize the relationship between categorical variables and churn.
plt.figure(figsize=(15, 15))
for i, column in enumerate(categorical_columns[:-1]):
    plt.subplot(6, 3, i + 1)
    sns.countplot(x=column, hue='Churn', data=churn_data)
    plt.title(f'{column} vs Churn')
    plt.xlabel(column)
    plt.ylabel('Count')
    plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Figure 14. Visualizing the Relationship between Key Variables and Churn.

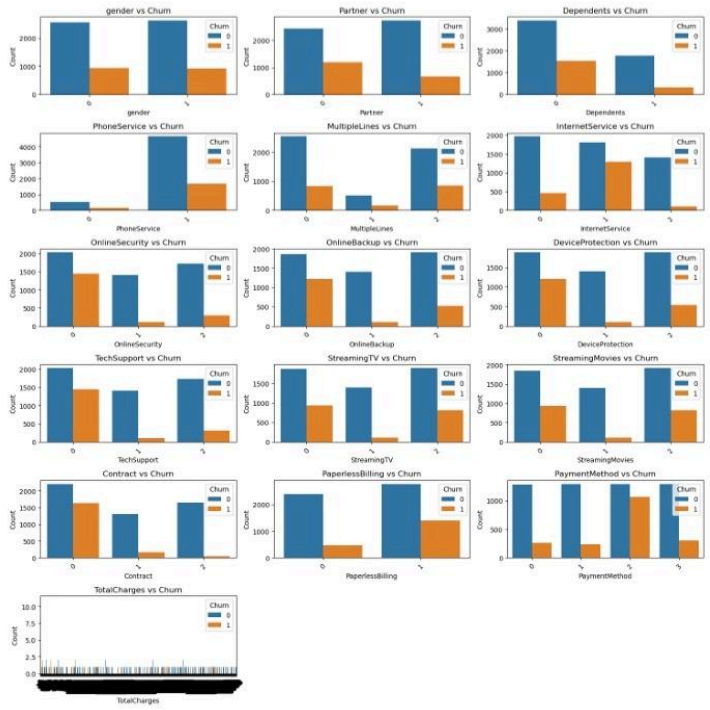


Figure 14. Results from Visualization.

Feature Selection.

In this phase, we had to use statistical tests and decision trees to identify significant churn patterns and predictions. We did that by first separating the features and target variable, splitting the dataset into training and test sets, performing chi-square tests for categorical variables to obtain p-values and then displaying that along with the feature names as seen in Figure 15.

After that, we performed an ANOVA test for the numerical variables to get their p-values as well and then displayed that along with the feature names as predictors in Figure 16.

0.0.6 TASK 3: FEATURE SELECTION.

a. Utilise statistical tests and decision trees to identify significant predictors of churn.

[20]:
```
# Separate features and target variable.
X = churn_data.drop('Churn', axis=1)
y = churn_data['Churn']
```

[21]:
```
# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[22]:
```
# Perform chi-square test for categorical variables.
chi2_selector = SelectKBest(chi2, k='all')
```



```
chi2_selector.fit(X_train, y_train)
```

[22]:
```
SelectKBest(k='all', score_func=<function chi2 at 0x000001B3669E9160>)
```

[23]:
```
# Get p-values and feature names
chi2_p_values = pd.DataFrame({'Feature': X_train.columns, 'P-value':␣
↪chi2_selector.pvalues_})
```

[24]:
```
# Display significant predictors (features with low p-values)
print("Significant predictors from chi-square test:")
print(chi2_p_values[chi2_p_values['P-value'] < 0.05])
```

```
Significant predictors from chi-square test:
              Feature        P-value
1       SeniorCitizen   1.745672e-26
2             Partner   1.961850e-13
3          Dependents   3.869433e-24
4              tenure   0.000000e+00
6       MultipleLines   1.118400e-04
7     InternetService   1.865738e-03
8      OnlineSecurity  1.504921e-102
9        OnlineBackup   5.085717e-41
10  DeviceProtection   1.726240e-31
11       TechSupport   5.169126e-99
12       StreamingTV   3.978609e-02
14          Contract  1.207369e-189
15  PaperlessBilling   5.102443e-19
16     PaymentMethod   3.070344e-10
17    MonthlyCharges   0.000000e+00
18      TotalCharges   0.000000e+00
```

Figure 15. Chi-Square Test and Displaying Predictors for Categorical Variables

[25]:
```
# Perform ANOVA for numerical variables
f_selector = SelectKBest(f_classif, k='all')
f_selector.fit(X_train, y_train)
```

[25]:
```
SelectKBest(k='all')
```

[26]:
```
# Get p-values and feature names
f_p_values = pd.DataFrame({'Feature': X_train.columns, 'P-value': f_selector.
↪pvalues_})
```

[27]:
```
# Display significant predictors (features with low p-values)
print("\nSignificant predictors from ANOVA:")
print(f_p_values[f_p_values['P-value'] < 0.05])
```

```
Significant predictors from ANOVA:
              Feature        P-value
1       SeniorCitizen   1.004514e-31
```



```
2             Partner   7.227092e-25
3          Dependents   2.961863e-34
4              tenure  1.265991e-156
6       MultipleLines   7.318155e-05
7     InternetService   7.781412e-05
8      OnlineSecurity  6.148803e-114
9        OnlineBackup   9.418919e-49
10  DeviceProtection   4.559455e-37
11       TechSupport  7.042114e-110
12       StreamingTV   2.140546e-02
14          Contract  3.851491e-205
15  PaperlessBilling   6.323818e-45
16     PaymentMethod   1.320043e-13
17    MonthlyCharges   7.151651e-51
18      TotalCharges   3.964310e-66
```

Figure 16. ANOVA Test and Displaying Predictors for Numerical Variables

The next step was to determine feature importance using decision trees. We used DecisionTreeClassifier to get the tree, and then we derived the feature importance and sorted out the features by experience. Then we displayed the significant predictors from the Decision Tree.

```
[28]: # Feature importance using Decision Tree
      dt = DecisionTreeClassifier(random_state=42)
      dt.fit(X_train, y_train)

[28]: DecisionTreeClassifier(random_state=42)

[29]: # Get feature importance
      feature_importance = pd.DataFrame({'Feature': X_train.columns, 'Importance': dt.
       ↪feature_importances_})

[30]: # Sort features by importance
      feature_importance.sort_values(by='Importance', ascending=False, inplace=True)

[31]: # Display top significant predictors based on feature importance
      print("\nSignificant predictors from Decision Tree:")
      print(feature_importance.head(10))

      Significant predictors from Decision Tree:
                  Feature  Importance
      18     TotalCharges    0.207254
      17   MonthlyCharges    0.201098
      14         Contract    0.164833
      4            tenure    0.118304
      8    OnlineSecurity    0.048938
      16    PaymentMethod    0.041787
      7   InternetService    0.029864
      1     SeniorCitizen    0.026787
      0            gender    0.023332
      2           Partner    0.022082
```

Figure 17. Feature Importance Using Decision Tree

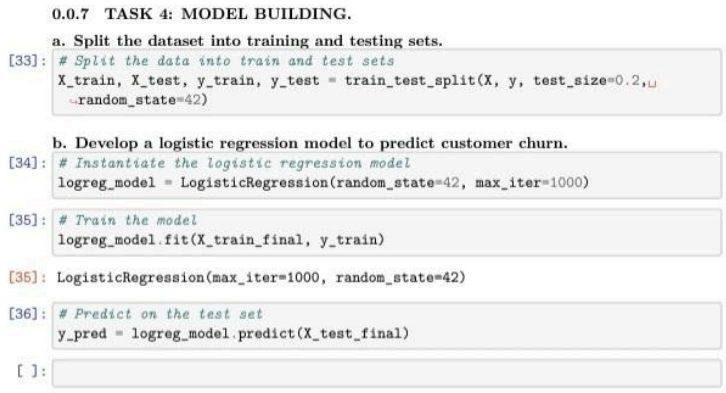
The last step under feature selection was to prepare the final dataset for modeling by selecting relevant features. This was done to pave way for the building of the model.

```
[32]: # Prepare the final dataset for modeling by selecting relevant features
      selected_features = ['Contract', 'tenure', 'MonthlyCharges', 'PaymentMethod',␣
       ↪'OnlineSecurity', 'TechSupport', 'InternetService', 'TotalCharges',␣
       ↪'StreamingTV']
      X_train_final = X_train[selected_features]
      X_test_final = X_test[selected_features]
```

Figure 18. Preparing Final Dataset

Model Building.

This is the stage where the churn prediction model was built and tested. After splitting the datasets into train and test sets, we developed a logistic regression model to predict customer churn and then trained the model.



0.0.7 TASK 4: MODEL BUILDING.

a. Split the dataset into training and testing sets.

```
[33]: # Split the data into train and test sets
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
        ↪random_state=42)
```

b. Develop a logistic regression model to predict customer churn.

```
[34]: # Instantiate the logistic regression model
      logreg_model = LogisticRegression(random_state=42, max_iter=1000)

[35]: # Train the model
      logreg_model.fit(X_train_final, y_train)

[35]: LogisticRegression(max_iter=1000, random_state=42)

[36]: # Predict on the test set
      y_pred = logreg_model.predict(X_test_final)

[ ]:
```

Figure 19. Developing Logistic Regression Model

Then we evaluated the performance of the model using metrics such as accuracy, F1 score, Precision and Recall and displayed the score for each. From that, we realized that the model has higher accuracy than precision.

c. Evaluate the model's performance using appropriate metrics (e.g., accuracy, recall, precision, F1 score).

```
[37]: # Calculate evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

[38]: # Confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

[39]: # Display evaluation metrics
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)

Accuracy: 0.7843416370106762
Precision: 0.5808580858085809
```



```
Recall: 0.5
F1 Score: 0.5374045801526718
```

Figure 20. Evaluation of Model Performance using Metrics.

The results from the evaluation metrics were plotted on a bar chart. The confusion matrix was also created and displayed showing the True Positive, True Negative, False Positive and False Negative fields.

```
[40]: # Plot evaluation metrics
metrics = ['Accuracy', 'Precision', 'Recall', 'F1 Score']
values = [accuracy, precision, recall, f1]

plt.figure(figsize=(8, 6))
plt.bar(metrics, values, color=['blue', 'green', 'red', 'purple'])
plt.title('Evaluation Metrics')
plt.xlabel('Metric')
plt.ylabel('Score')
plt.ylim(0, 1)
plt.show()
```

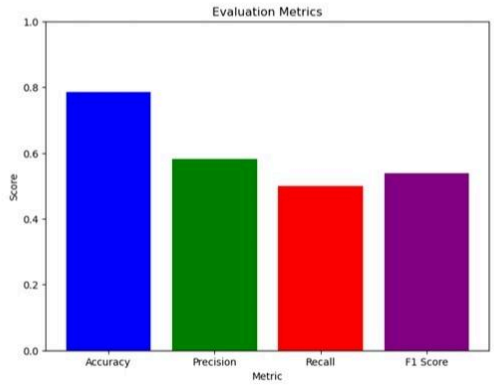


Figure 21. Plotting of Evaluation Metrics

```
[41]: # Display confusion matrix
      print("\nConfusion Matrix:")
      print(conf_matrix)
```

14

```
Confusion Matrix:
[[926 127]
 [176 176]]
```

```
[42]: # Plot confusion matrix
      plt.figure(figsize=(8, 6))
      sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=False)
      plt.xlabel('Predicted labels')
      plt.ylabel('True labels')
      plt.title('Confusion Matrix')
      plt.show()
```

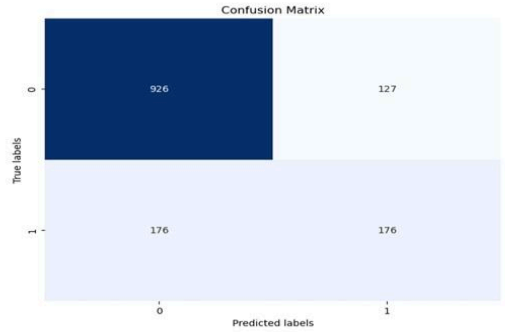


Figure 22. Confusion Matrix

Model Evaluation and Validation.

At this stage, we are using just the decision tree as a supplementary approach to predict customer churn and the factors causing it. We did that by training the decision tree classifier, getting feature importance from the decision tree model, ranking the features and then plotting the feature importances.

Then we trained the regression model as well and plotted it in the same graph as that of the decision tree model. After doing that, we found the correlation between the decision tree model results and the logistic regression model results using np.corrcoef (....) function.

0.0.8 TASK 5: MODEL EVALUATION AND VALIDATION.

a. Use the decision tree model as a supplementary approach to identify the primary factors causing churn.

[43]:
```
# Train decision tree classifier
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X_train, y_train)
```

15

[43]:
```
DecisionTreeClassifier(random_state=42)
```

[44]:
```
# Get feature importances from decision tree model
dt_feature_importances = dt_classifier.feature_importances_
```

[45]:
```
# Rank Features
feature_importance_indices = dt_feature_importances.argsort()[::-1]
ranked_features_decision_tree = X.columns[feature_importance_indices]
```

[46]:
```
# Plot feature importances
plt.figure(figsize=(10, 6))
plt.barh(X.columns, dt_feature_importances)
plt.xlabel('Feature Importance')
plt.ylabel('Features')
plt.title('Decision Tree Feature Importances')
plt.show()
```

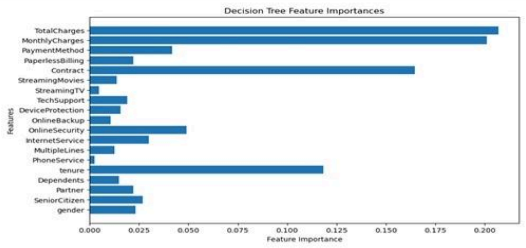


Figure 23. Decision Tree Model

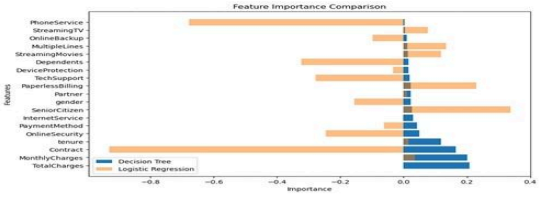
b. Compare the findings from both models to ensure consistency and reliability in the identified churn drivers.

[47]:
```
# Train Logistic regression model
lr_model = LogisticRegression(max_iter=1000)
lr_model.fit(X_train, y_train)
```

[47]:
```
LogisticRegression(max_iter=1000)
```

16

[48]:
```
plt.figure(figsize=(10, 6))
plt.barh(ranked_features_decision_tree,␣
↪dt_feature_importances[feature_importance_indices], label='Decision Tree')
plt.barh(X.columns, lr_model.coef_[0], alpha=0.5, label='Logistic Regression')
plt.xlabel('Importance')
plt.ylabel('Features')
plt.legend()
plt.title('Feature Importance Comparison')
plt.show()
```



[49]:
```
# Correlation Analysis
correlation_matrix = np.corrcoef(dt_feature_importances, np.abs(lr_model.
↪coef_[0]))
correlation = correlation_matrix[0, 1]
print(f"Correlation between Decision Tree and Logistic Regression:␣
↪{correlation}")
```

```
Correlation between Decision Tree and Logistic Regression: 0.04824471242607804
```

Figure 24. Logistic Regression Model(LRM) and the Correlation between Decision Tree and LRM

Finally, we evaluated the performance of decision tree model and then compared the evaluation metrics of the logistic regression model and that of the decision tree model in a graph.

```
[50]: # Evaluate decision tree model performance
y_pred_decision_tree = dt_classifier.predict(X_test)
accuracy_decision_tree = accuracy_score(y_test, y_pred_decision_tree)
precision_decision_tree = precision_score(y_test, y_pred_decision_tree)
recall_decision_tree = recall_score(y_test, y_pred_decision_tree)
f1_score_decision_tree = f1_score(y_test, y_pred_decision_tree)
```

17

```
print("Decision Tree Model Performance:")
print(f"Accuracy: {accuracy_decision_tree}")
print(f"Precision: {precision_decision_tree}")
print(f"Recall: {recall_decision_tree}")
print(f"F1 Score: {f1_score_decision_tree}")
```

```
Decision Tree Model Performance:
Accuracy: 0.7345195729537367
Precision: 0.47229551451187335
Recall: 0.5085227272727273
F1 Score: 0.4897400820793434
```

```
[51]: # Define evaluation metrics and their scores
metrics = ['Accuracy', 'Precision', 'Recall', 'F1 Score']
values_lr = [accuracy, precision, recall, f1]
values_dt = [accuracy_decision_tree, precision_decision_tree,␣
  ↪recall_decision_tree, f1_score_decision_tree]

# Plotting
plt.figure(figsize=(10, 6))
plt.bar(metrics, values_lr, width=0.4, label='Logistic Regression')
plt.bar(metrics, values_dt, width=0.4, label='Decision Tree')
plt.xlabel('Metrics')
plt.ylabel('Score')
plt.title('Model Evaluation Metrics')
plt.legend()
plt.show()
```


18

Figure 25. Comparing the Evaluation Metrics of Logistic Regression Model to that of Decision Tree Model

INTERPRETATION OF MODEL OUTCOMES

From Figure 25, we realized that the evaluation metrics for the logistic regression model were a bit higher than the decision tree model. This means that, compared to the two models, the logistic regression model has a higher chance of predicting customer churn as compared to the decision tree model. Also, in Figure 17, we see some predictor values and their respective features. The magnitude of the coefficients determines the strength of the relationship between the predictor variable and churn, meaning higher predictor values indicate a higher likelihood of churn. We realized that Total Charges, Monthly Charges, Contract and Tenure have higher predictor values, so these are the major causes of customer churn.

Based on the findings of the logistic regression model and the identified factors that influence customer churn, here are some actionable strategies to reduce churn:

1. Improve Service Quality: The telecom companies should focus on improving the quality of products or services offered to customers. They should try their best to address any issues or complaints promptly to ensure customer satisfaction.
2. Personalized Communication: They should be willing to implement personalized communication strategies to engage with customers effectively. They should use customer data to tailor marketing messages, offers, and promotions to match their preferences and needs.
3. Retention Incentives: One thing they can do to keep their customers is to offer incentives or rewards to loyal customers. This encourages them to continue using the products or services. This could include discounts, loyalty programs, or exclusive deals.

4. Enhance Customer Support: The telcos should be willing to invest in enhancing customer support channels, such as live chat, email support, or dedicated helplines. They should also provide timely assistance and resolution to customer queries or concerns.
5. Flexible Pricing Plans: Telecom companies should always try their best to offer flexible pricing plans or packages that cater to different customer segments. They can do that by providing options for customers to customize their plans based on their usage patterns and budget.
6. Focus on Customer Experience and Monitor Customer Feedback: They should prioritize delivering a positive customer experience at every touchpoint and also streamline processes, reduce friction points, and ensure a seamless experience from initial contact to post-purchase support. They should also regularly solicit feedback from customers through surveys, reviews, or feedback forms and use this feedback to identify areas for improvement and make necessary changes to enhance the overall customer experience.

CONCLUSION.

The constant risk of losing subscribers, known as customer churn, can seriously impair a telecom company's profitability. The significance of customer churn prediction in this extremely competitive industry has been examined in this report. With the help of the Kaggle dataset, we have examined the variables that affect customer attrition and some potential mitigation techniques. Telecom companies can create focused strategies to regain the loyalty of customers by proactively identifying those who are about to leave. These tactics could be tailored discounts, enhanced service options, or focusing on particular issues that customers are having. In the end, telecom businesses can secure long-term financial success by retaining valuable customers through the use of a well-developed churn prediction model.

REFERENCES

[1]. CorebrandAI, “How a customer churn prediction model transforms service and protects revenue,” May 01, 2023. https://www.linkedin.com/pulse/how-customer-churn-prediction-model-transforms-service-protects

[2]. L. Saha, H. K. Tripathy, T. Gaber, H. El-Gohary, and E.-S. M. El-Kenawy, “Deep churn prediction method for telecommunication industry,” Sustainability, vol. 15, no. 5, p. 4543, Mar. 2023, https://www.mdpi.com/2071-1050/15/5/4543 doi: 10.3390/su15054543

[3]. Mishra, A.; Reddy, U.S. A comparative study of customer churn prediction in telecom industry using ensemble-based classifiers. In Proceedings of the 2017 International Conference on Inventive Computing and Informatics (ICICI), Coimbatore, India, 23–24 November 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 721–725. [**Google Scholar**]

[4]. A.A.Q. Ahmed, D. Maheswari; Churn prediction on huge telecom data using hybrid firefly based classification. Egypt Inform J, 18 (3) (2017), pp. 215-220, 10.1016/j.eij.2017.02.002

[5]. V. E, P. Ravikumar, C. S, S.K. M, An efficient technique for feature selection to predict customer churn in telecom industry: Proceedings of the 1st International Conference on Advances in Information Technology (ICAIT) (2019), pp. 174-179, 10.1109/ICAIT47043.2019.8987317

[6]. H. Jain, A. Khunteta, S. Srivastava, Churn prediction in telecommunication using logistic regression and logit boost: Procedia Comput Sci, 167 (2020), pp. 101-112, 10.1016/j.procs.2020.03.187

[7]. Simplilearn, "Data Preprocessing in Machine Learning: A Beginner's Guide," *Simplilearn.com*, Sep. 28, 2023. https://www.simplilearn.com/data-preprocessing-in-machine-learning-article#:~:text=Data%20preprocessing%20is%20the%20process,algorithms%20to%20reduce%20its%20complexities