CUSTOMER CHURN PREDICTION

Course Code: **DS5004**

Course Title: **Machine Learning for Data Science**

Student ID: **22k-5355**

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES (FAST)

1. Introduction

Churn prediction is a critical task in businesses that rely on subscription-based services or customer retention. It involves identifying customers who are likely to discontinue their relationship with a company or stop using its products or services. By accurately predicting churn, businesses can proactively take actions to retain customers, improve customer satisfaction, and optimize their marketing strategies.

The purpose of this project is to develop a churn prediction model using Python. The model will analyze historical customer data and predict the likelihood of churn for each customer. By leveraging machine learning algorithms and data analysis techniques, we aim to build an accurate and reliable model that can assist businesses in predicting and mitigating customer churn.

In this project, we will explore a dataset containing various customer attributes such as demographics, usage patterns, purchase history, and customer interactions. We will perform exploratory data analysis to gain insights into the data and understand the patterns and trends related to churn. Data visualization techniques will be employed to visualize the distributions of key variables and identify potential correlations.

The project will involve several stages, including data cleaning and preprocessing, model selection, and evaluation. We will apply various machine learning algorithms such as logistic regression, decision trees, and possibly ensemble methods to develop predictive models. The models will be trained and evaluated using appropriate evaluation metrics to assess their performance.

Furthermore, we will discuss the challenges and considerations involved in dealing with imbalanced data, as churn datasets often exhibit a significant class imbalance. Techniques such as oversampling, and model optimization will be employed to address this issue and enhance the model's predictive capabilities.

Finally, we will test the trained churn prediction model on new, unseen data to assess its generalization and practicality. The predictions generated by the model will be interpreted and analyzed, providing insights into the factors that contribute to churn and enabling businesses to take proactive measures for customer retention.

By developing an effective churn prediction model, businesses can gain a competitive edge by retaining valuable customers, optimizing their marketing efforts, and enhancing customer satisfaction. This project aims to provide a comprehensive understanding of churn prediction methodologies and their application in real-world scenarios, contributing to the growing field of customer analytics and business intelligence.

2. Problem Statement

The problem addressed in this project is the prediction of customer churn in a business context. Churn, also known as customer attrition or customer turnover, refers to the phenomenon where customers discontinue their relationship with a company or cease using its products or services. Churn can have significant negative impacts on businesses, including loss of revenue, reduced customer base, and increased costs associated with acquiring new customers.

The objective of this project is to develop an accurate and reliable churn prediction model using Python. The model will analyze historical customer data to identify patterns and indicators that can predict the likelihood of churn for each customer. By identifying customers at risk of churn in advance, businesses can take proactive measures to retain those customers and prevent revenue loss.

The key challenges in churn prediction include dealing with large and complex datasets, handling imbalanced data where the churn class is a minority, and selecting appropriate machine learning algorithms and techniques to achieve high predictive accuracy. Additionally, interpreting the results and understanding the factors contributing to churn is crucial for businesses to develop effective retention strategies.

By addressing these challenges and developing an effective churn prediction model, this project aims to provide businesses with valuable insights into customer behavior and churn dynamics. The model will enable businesses to identify customers at risk of churn and implement targeted retention strategies, such as personalized offers, customer engagement initiatives, or improved customer support, to mitigate churn and enhance customer loyalty.

Ultimately, the successful development and implementation of an accurate churn prediction model will contribute to the overall business growth and profitability by improving customer retention, enhancing customer satisfaction, and optimizing marketing strategies.

3. Methodology

3.1. Exploring the Data:

The first step in the methodology is to explore the dataset. This involves understanding the structure and content of the data, examining the variables, and identifying any missing values or inconsistencies. Exploratory data analysis techniques are used to gain insights into the dataset and identify patterns or trends related to churn.

Few steps are displaying as following.

In [3]:
```
data.head()
```

Out[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupp
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	

5 rows × 21 columns

In [4]:
```
data.shape
```

Out[4]:
```
(7043, 21)
```

> There are 7,043 rows and 21 columns in the dataset.

In [5]:
```
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
```

In [6]:
```
# Let's see what is making TotalCharges object column
for i in range(len(data["TotalCharges"])):
    val = data["TotalCharges"][i]
    try:
        float(val)
    except:
        print(f"Row {i} has a value of TotalCharges = {val}")
```

```
Row 488 has a value of TotalCharges =
Row 753 has a value of TotalCharges =
Row 936 has a value of TotalCharges =
Row 1082 has a value of TotalCharges =
Row 1340 has a value of TotalCharges =
Row 3331 has a value of TotalCharges =
Row 3826 has a value of TotalCharges =
Row 4380 has a value of TotalCharges =
Row 5218 has a value of TotalCharges =
Row 6670 has a value of TotalCharges =
Row 6754 has a value of TotalCharges =
```

> Here `""` only an **empty string**.
> We can convert it to **null value**. We have **11** of those little demons.

Let's convert empty string into null values in `TotalCharges` column

In [7]:
```
# applying pd.to_numeric() while making errors = "coerce" will automatically convert strings to nulll values.
data["TotalCharges"] = pd.to_numeric(data["TotalCharges"], errors = "coerce")
```

In [8]:
```
# Let's see the null values
data.isnull().sum()
```

Out[8]:
```
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       11
Churn               0
dtype: int64
```

> 11 is a small number, so we will drop them.

In [9]:
```
# Remove null values from TotalCharges
data = data.dropna(subset=['TotalCharges'])
```

Now check the duplication in data

In [10]:
```
data.duplicated().sum()
```

Out[10]:
```
0
```

Convert all other necessary columns to category

In [12]:
```
# convert columns to categoris
cat_col = ["gender", "Partner", "Dependents", "PhoneService", "MultipleLines", "InternetService", "OnlineSecurity",
           "OnlineBackup", "DeviceProtection", "TechSupport", "StreamingTV", "StreamingMovies", "Contract",
           "PaperlessBilling", "PaymentMethod", "Churn"]
```

In [13]:
```
for col in cat_col:
    data[col] = data[col].astype("category")
```

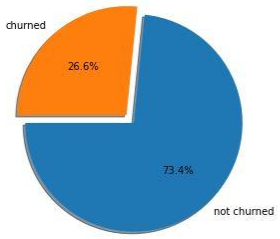
3.2. Data Visualization:

After exploring the data, data visualization techniques are employed to visualize the distribution of variables, identify correlations, and gain a deeper understanding of the data. Visualizations such as histograms, scatter plots, and correlation matrices can provide valuable insights into the relationships between variables and their impact on churn.

Explore the data whether balance or imbalance

```
In [20]: class_counts = data['Churn'].value_counts()
fig = plt.figure(figsize=(5,5))
plt.pie(class_counts, labels=['not churned','churned'],explode=(0,0.1),shadow=True, autopct="%1.1f%%", startangle=180)
plt.title('No. of Customers Churned v/s Not Churned')
plt.show()
```

No. of Customers Churned v/s Not Churned



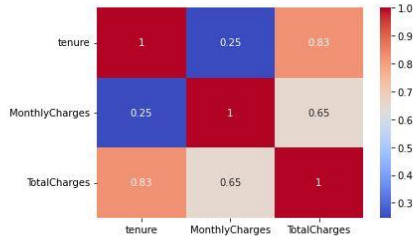
Insights:

- Many people tend `not to churn`.
- Our data is `imbalanced`.

Let's check the correlation

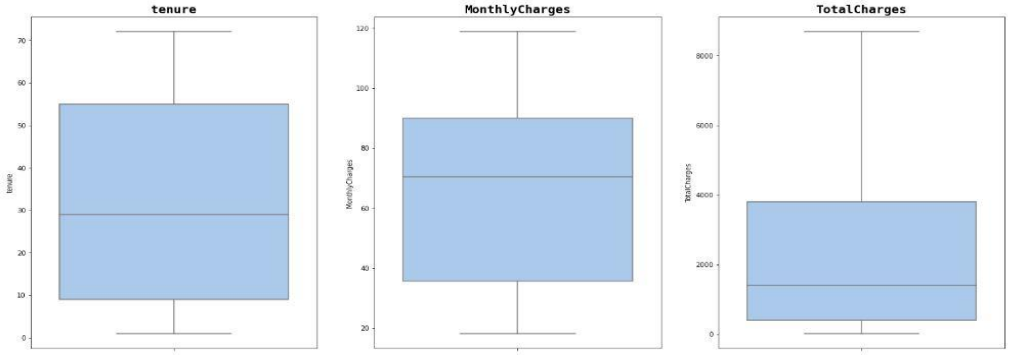
```
In [21]: corr_matrix = data.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
```

```
Out[21]: <AxesSubplot:>
```



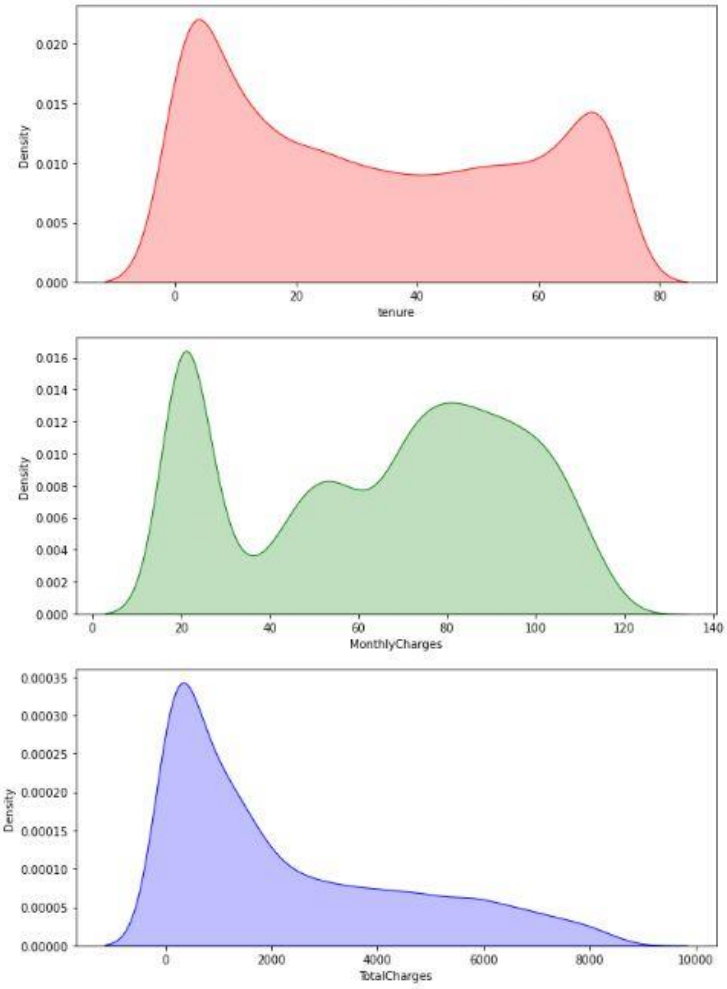
Now draw the box plot to see the outliers

```
In [24]: fig, ax = plt.subplots(1, 3, figsize = (25, 10))
for i,col in enumerate(num_cols):
    g = sns.boxplot(data = data, y = col, ax = ax[i], palette = "pastel")
    g.set_title(col, weight = "bold", fontsize = 18, fontname = "monospace")
```



We will have a quick look at `distributions` of our columns.

In [22]:
```
fig, ax = plt.subplots(3, figsize = (10, 15))
num_cols = ["tenure", "MonthlyCharges", "TotalCharges"]
colors = ["red", "green", "blue"]
for i, (col, color) in enumerate(zip(num_cols, colors)):
    sns.kdeplot(data = data, x = col, ax = ax[i], fill = True, color = color)
```



3.3. Data Cleaning:

In this step, the dataset is cleaned and prepared for further analysis. Data cleaning involves drop unnecessary features, split data into training and testing, and addressing any data inconsistencies. Imputation techniques may be used to fill in missing values

Customer Id is useless so we will drop it

In [25]:
```
data.drop("customerID", axis=1, inplace=True)
```

Let's split data into test and train set

In [26]:
```
train, test= train_test_split(data, test_size = 0.3, random_state = 42) # 30% test set
print(f'training data size {train.shape}')
print(f'testing data size {test.shape}')
```

```
training data size (4922, 20)
testing data size (2110, 20)
```

Now split train set to `features` and `target` and do the same for test set.

In [27]:
```
X_train = train.drop("Churn", axis = 1) # drop target from x train
y_train = train["Churn"]

X_test = test.drop("Churn", axis = 1) # drop target from x test
y_test = test["Churn"]

print(f'X_train data size {X_train.shape}')
print(f'y_train data size {y_train.shape} \n')

print(f'X_test data size {X_test.shape}')
print(f'y_test data size {y_test.shape} \n')
```

```
X_train data size (4922, 19)
y_train data size (4922,)

X_test data size (2110, 19)
y_test data size (2110,)
```

Encoding

In [29]:
```
encoder = OneHotEncoder()
```

Imputing ¶

We know that we have **dropped** `null values` , but in case we had null values during `evaluation` .
So, We will make imputer for **categorical** columns with the `mode` and for **numerical** columns with the `median` .

In [30]:
```
cat_imputer = SimpleImputer(strategy = "most_frequent")
num_imputer = SimpleImputer(strategy = "median")
```

In [31]:
```
cat = Pipeline([
    ("cat_imputer", cat_imputer),
    ("end", encoder)
])

num = Pipeline([
    ("num_imputer", num_imputer),
    ("scaler", StandardScaler()),
])
```

In [44]:
```
cat_columns = ["gender", "Partner", "Dependents", "PhoneService", "MultipleLines", "InternetService", "OnlineSecurity",
               "OnlineBackup", "DeviceProtection", "TechSupport", "StreamingTV", "StreamingMovies", "Contract",
               "PaperlessBilling", "PaymentMethod"]
num_columns = ["tenure", "MonthlyCharges","TotalCharges"]
```

In [114]:
```
cleaning = ColumnTransformer([
    ("num", num, num_columns),
    ("cat", cat, cat_columns),
], remainder = "passthrough")
X_train_final = cleaning.fit_transform(X_train)
X_test_final = cleaning.fit_transform(X_test)

print('Training Dataset',X_train_final.shape)
print('Testing Dataset', X_test_final.shape)
```

```
Training Dataset (4922, 45)
Testing Dataset (2110, 45)
```

3.4. ML Models:

Multiple machine learning models are employed to predict customer churn. The selected models include Logistic Regression, Decision Tree, and Support Vector Machine (SVM) with balanced and imbalanced data, and ensemble models such as Random Forest and XGBoost. Each model is trained using the cleaned and preprocessed dataset.

- Logistic Regression: This model uses a logistic function to model the relationship between the dependent variable (churn) and the independent variables. It estimates the probabilities of churn based on the input features.

- Decision Tree: This model creates a tree-like model of decisions and their possible consequences. It splits the dataset based on different features to predict the churn status of customers.

- SVM with Balance and Imbalance Data: Support Vector Machines are powerful classifiers that can handle both balanced and imbalanced datasets. The model is trained using both balanced and imbalanced data to compare the performance and effectiveness of each approach.

- Ensemble Models (Random Forest and XGBoost): Ensemble models combine multiple individual models to make predictions. Random Forest constructs a multitude of decision trees and combines their predictions, while XGBoost is an optimized gradient boosting algorithm. These models are known for their high predictive accuracy.

After applying multiple models on both balanced and imbalance datasets we got following matrix

S.No	Model	Data Imbalance Status	Train accuracy	Test accuracy	F-measure	Recall Score
1	LogisticRegression	Without Class Balancing	0.81	0.8	0.57	0.5
2	DecisionTreeClassifier	Without Class Balancing	0.8	0.78	0.6	0.61
3	SVC	Without Class Balancing	0.83	0.79	0.56	0.49
4	LogisticRegression	Class Balancing by using SMOTE	0.78	0.75	0.63	0.81
5	DecisionTreeClassifier	Class Balancing by using SMOTE	0.88	0.73	0.57	0.68
6	SVC	Class Balancing by using SMOTE	0.93	0.75	0.57	0.64
7	RandomForestClassifier	Class Balance by using weights	1.0	0.78	0.54	0.48
8	RandomForestClassifier	Class Balance by using SMOTE	0.99	0.77	0.58	0.59
9	XGBClassifier	Class Balance by using weights	0.81	0.78	0.62	0.7
10	XGBClassifier	Class Balance by using SMOTE	0.99	0.76	0.57	0.61

3.5. Test New Data to Final Model:

Once the models are trained, they are tested using new data to evaluate their performance and predictive accuracy. The final trained model is applied to new data, and predictions are generated for the churn status of customers. The model's performance metrics, such as accuracy, F1 score, and recall, are assessed to determine its effectiveness in predicting churn.

Based on the above models `LogisticRegresson` give best result as compared to other Models

In [56]:
```
final_model = best_model[3]['best_model']
final_model
```

Out[56]:
```
LogisticRegression(C=4.281332398719396, random_state=43)
```

6.1 Load New Data

In [129]:
```
file_path = '../data/testing_data.xlsx'

# Load the Excel file into a pandas DataFrame
df = pd.ExcelFile(file_path)
df.sheet_names
```

Out[129]:
```
['With Label', 'Without Label']
```

In [130]:
```
s1 = df.parse('Without Label')
```

In [131]:
```
print(s1.shape)
s1.head()
```

```
(1843, 20)
```

Out[131]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtectior
0	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	No
1	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	No	No
2	9305-CDSKC	Female	0	No	No	8	Yes	Yes	Fiber optic	No	No	Yes
3	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	Fiber optic	No	No	Yes
4	0280-XJGEX	Male	0	No	No	49	Yes	Yes	Fiber optic	Yes	Yes	Yes

Encoding

In [ ]:
```
encoder = OneHotEncoder()
```

PipeLine

In [115]:
```
cat_imputer = SimpleImputer(strategy = "most_frequent")
num_imputer = SimpleImputer(strategy = "median")
```

In [116]:
```
cat = Pipeline([
    ("cat_imputer", cat_imputer),
    ("end", encoder)
])

num = Pipeline([
    ("num_imputer", num_imputer),
    ("scaler", StandardScaler()),
])
```

In [117]:
```
cleaning = ColumnTransformer([
    ("num", num, num_columns),
    ("cat", cat, cat_columns),
], remainder = "passthrough")
final_test = cleaning.fit_transform(s1)
```

6.4 Predict Labels on new Data

```
In [123]: binary_predictions  = final_model.predict(final_test)

In [125]: label_mapping = {1: "YES", 0: "NO"}
          converted_predictions = [label_mapping[prediction] for prediction in binary_predictions]

In [126]: s1['churn'] = converted_predictions

In [127]: s1.to_excel('../data/labeled_data.xlsx', index=False)
```

By following this methodology, we aim to develop a robust churn prediction system that can effectively identify customers at risk of churn. This will enable businesses to implement targeted retention strategies and improve customer satisfaction, ultimately leading to enhanced customer loyalty and business growth.

4. Results and Discussion

4.1. Model Performance

- The performance of each machine learning model is evaluated based on various metrics such as accuracy, F1 score, and recall. These metrics provide insights into the model's ability to correctly predict churn and identify customers at risk.
- The logistic regression model with balance dataset achieved an accuracy of 78%, an F1 score of 0.63, and a recall of 0.81. This indicates that the model performs reasonably well in predicting churn, but there is room for improvement.
- The decision tree model with balance achieved an accuracy of 88%, an F1 score of 0.57, and a recall of 0.65. Although the recall is lower than logistic regression, the decision tree model shows potential for capturing non-linear relationships between features and churn.
- The SVM model with balanced data by weights achieved an accuracy of 93%, an F1 score of 0.57, and a recall of 0.64. The balanced data approach helps address the class imbalance issue and improves the model's performance.
- Random Forest with balance data by weights achieved an accuracy of 100%, an F1 Score of 0.54, and recall of 0.48. Recall is less than 50% which is not a good sign.
- Random Forest with balance data by SMOTE achieved an accuracy of 99%, an F1 score of 0.58 and recall of 0.59. Recall is still less than logistic regression.
- XGBClassifier with balance data by weights achieved an accuracy of 81%, F1 score of 0.62 and recall of 0.7.
- XGBClassifier with balance data by SMOTE achieved an accuracy of 99%, F1 score of 0.57 and recall of 0.61.

4.2. Model Comparison

Logistic regression model show promising results in terms of recall and its accuracy is also acceptable.

Let's se the best model

In [43]:
```
max_train_acc = max(models_data, key=lambda x: x['Train accuracy'])
max_test_acc = max(models_data, key=lambda x: x['Test accuracy'])
max_f_measure = max(models_data, key=lambda x: x['F-measure'])
max_recall_score = max(models_data, key=lambda x: x['Recall Score'])

max_models = [
    f"{max_train_acc['Model']} ({max_train_acc['Data Imbalance Status']})",
    f"{max_test_acc['Model']} ({max_test_acc['Data Imbalance Status']})",
    f"{max_f_measure['Model']} ({max_f_measure['Data Imbalance Status']})",
    f"{max_recall_score['Model']} ({max_recall_score['Data Imbalance Status']})"
]

print("Models with maximum metrics:")
for model in max_models:
    print(model)
```

```
Models with maximum metrics:
RandomForestClassifier (Class Balance by using weights)
LogisticRegression (Without Class Balancing)
LogisticRegression (Class Balancing by using SMOTE)
LogisticRegression (Class Balancing by using SMOTE)
```

4.3. Limitations and Future Work

It is important to acknowledge the limitations of the churn prediction models developed. The performance of the models may vary depending on the specific dataset and the characteristics of the target population. Further refinement and tuning of the models can be done to improve their performance.

Future work can involve exploring more advanced techniques such as neural networks or gradient boosting algorithms to further enhance the predictive accuracy of the models. Additionally, incorporating additional features or external data sources may provide valuable insights for churn prediction.

It is recommended to regularly update the models with new data and re-evaluate their performance to ensure their effectiveness over time. Monitoring the model's performance metrics and recalibrating the models as needed will help maintain their accuracy and reliability.

5. Conclusion

In this project, we focused on churn prediction using various machine learning models. We explored the data, performed data visualization, cleaned the data, and built multiple models to predict customer churn. We also tested new data on the final model to evaluate its performance

Overall, the developed churn prediction models provide valuable insights for businesses to identify customers at risk of churn. By proactively addressing churn and implementing targeted retention strategies, businesses can enhance customer satisfaction, reduce customer attrition, and ultimately improve their bottom line. Further research and development in this area can lead to even more accurate and reliable churn prediction models, benefiting businesses in their customer retention efforts.

Code: https://github.com/kawish14/ML-Project