## ANAND INSTITUTE OF HIGHER TECHNOLOGY OLD MAHABALIPURAM ROAD,

**KALASALINGAM NAGAR, KAZHIPATTUR – 603103**



**CUSTOMER CHURN PREDICTION**

**WITH**

**DATA ANALYTICS USING COGNOS**

**PHASE – 5**

### NAME : MOUNIKA V

### REG No. : 310121104065

**BRANCH : COMPUTER SCIENCE & ENGINEERING**

**YEAR/SEM : III / V**

**IBM DATA ANALYTICS WITH**

**COGNOS**

**TEAM NAME :** Proj\_229798\_Team\_1

**PROJECT :** 3101-Customer Churn Prediction

**TEAM MEMBERS :**

1. Nokitha V M
2. Mounika V
3. Yasmeen U
4. Roshini

**LEADER :** Nokitha V M

**Customer Churn Prediction**

# Definition :

The project involves using IBM Cognos to predict customer churn and identify factors influencing customer retention. The goal is to help businesses reduce customer attrition by understanding the patterns and reasons behind customers leaving. This project includes defining analysis objectives, collecting customer data, designing relevant

visualizations in IBM Cognos, and building a predictive model.

**Data Collection and Preprocessing :**

1. **Identify Key Data :**

Dive into the essential data points required to effectively predict customer churn. Understand the importance of data quality and data cleansing techniques

1. **Data Preprocessing :**

Explore the steps involved in preparing the data for analysis. Learn about techniques such as data normalization, feature engineering, and handling missing values

1. **Feature Selection** :

Discover various methods to identify the most relevant features for predicting churn. Understand the impact of feature selection on model performance.

**Exploratory Data Analysis :**

**Visualizing Data:**

Learn how to effectively visualize and analyze data to gain insights into customer behavior and potential churn indicators. Use charts and graphs to tell the story.

**Sentiment Analysis :**

Uncover the power of sentiment analysis in predicting

customer churn. Learn how to extract meaningful information from customer feedback and reviews.

**Customer Journey Mapping :**

Map out the customer journey to identify critical touchpoints and moments of truth. Understand the key stages where customers are at risk of churning.

**Design Thinking:**

* + Analysis Objectives: Define the specific objectives of predicting customer churn, such as identifying potential

churners and understanding the key factors contributing to churn.

* + Data Collection: Determine the sources and methods for collecting customer data, including customer

demographics, usage behavior, and historical interactions.

* + Visualization Strategy: Plan how to visualize the insights using IBM Cognos, showcasing factors affecting churn and retention rates.
  + Predictive Modeling: Decide on the machine learning algorithms and features to use for predicting customer churn.

**NumPy :**

NumPy, which stands for "Numerical Python," provides support for working with large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy is a core library in the Python data.

* + - Data Loading
    - Data Preparation
    - Data Exploration
    - Numerical Operations

**Pandas DataFrame :**

A pandas DataFrame is a two-dimensional, labelled data structure commonly used in data analysis and manipulation

within the Python programming language. It is one of the fundamental data structures provided by the pandas library.

* + - Data Import
    - Data Exploration
    - Data Manipulation
    - Time Series Analysis

**Matplotlib :**

Matplotlib is a widely-used Python library for creating static and interactive visualizations and plots. It is a powerful tool for data visualization and is particularly popular in the fields of data analysis, scientific computing, and machine

learning.

* + - Data Visualization
    - Exploratory Data Analysis(EDA)
    - Performance Metrics
    - Customization and Interactivity

**Jupyter Notebook Interface**

**Machine Learning Algorithms in Jupyter Notebook :**

Showcasing the implementation of KNN, CNN, and Gradient Descent algorithms in Jupyter Notebook.

**Executing Code in Jupyter Notebook :**

Witness the step-by-step execution of code snippets within the Jupyter Notebook environment.

**Visualizing Data in Jupyter Notebook :**

Demonstrating the power of data visualization using Jupyter Notebook's interactive capabilities.

**Using Machine Learning Algorithms :**

* + KNN, CNN, and Gradient Descent are some of the most popular machine learning algorithms used for data analysis. In this section, we will discuss how these algorithms are useful to transportation efficiency data's analysis and their results.
  + KNN can help in real-time traffic management and route optimization, CNN can provide automated insights from visual data for better traffic monitoring, and Gradient Descent plays a crucial role in training predictive models for various transportation-related applications.
  + The results of these algorithms can lead to more efficient transportation systems, reduced congestion, lower costs, and improved overall transportation experiences.

**DEVELOPMENT PART**

* Start building the customer churn prediction using IBM Cognos

for visualization. Define the analysis objectives and collect customer data from the source shared.

* Process and clean the collected data to ensure its quality and accuracy.
* This process involves collecting, cleaning, transforming, reduction of null values, visualization, scalability, efficiency and structuring raw data to make it suitable for analysis.
* Both qualitative and quantitative customer data are usually needed to start building an effective churn prediction model. To ensure that predictions aren't being made by arbitrary human guesses, these models are often built by a data scientist using machine learning.

# DATA COLLECTION:

CUSTOMER CHURN PREDICTION is done by using the Dataset of “Telco Customer Churn” provided by the dataset site

[www.Kaggle.com](http://www.kaggle.com/)

**Dataset Link:**

[**https://www.kaggle.com/datasets/blastchar/telco-customer-**](https://www.kaggle.com/datasets/blastchar/telco-customer-churn)

[**churn**](https://www.kaggle.com/datasets/blastchar/telco-customer-churn)

**DATASET AND ITS DETAILS :**

**TITLE:** CUSTOMER CHURN PREDICTION

**LIBRARIES:** sklearn, Matplotlib, pandas, seaborn, and NumPy

**Context:**

The dataset “CUSTOMER CHURN PREDICTION” on Kaggle is a collection of data related to the a Machine Learning Model That Can Predict Customers Who Will Leave The Company "Predict behavior to retain customers. You can analyze all relevant customer data and develop focused customer retention programs."

**Content:**

Each row represents a customer, each column contains

customer’s attributes described on the column Metadata.

**SIGNIFICANCE OF LOADING AND PREPROCESSING THE DATASET:**

* + Data Loading is defined as copying data from one electronic file or database into another. Data loading implies converting from one format into another; for example, from one type of production database into a decision support database from a different vendor.
  + Data preprocessing is essential before its actual use. Data preprocessing is the concept of changing the raw data into a clean data set. The dataset is preprocessed in order to check missing values, noisy data, and other inconsistencies before executing it to the algorithm.

# IMPORT AND LOAD THE DATASET :

Use Pandas to read the dataset file you downloaded into a DataFrame:

### CODING AND ITS OUTPUT: LOAD DATA:

**import** numpy **as** np

**import** pandas **as** pd

*# Visualization*

**import** plotly.express **as** px **import** matplotlib.pyplot **as** plt **import** seaborn **as** sns

**%matplotlib** inline

**import** warnings warnings**.**filterwarnings('ignore')



### IMPORT DATA:

df = pd.read\_csv(r"E:\nokitha nan mudhalvan\WA\_Fn-UseC\_-Telco- Customer-Churn.csv")

df.head()



### DATA UNDERSTANDING:

df.info()



### Check the Duplicate:

print(df.duplicated().value\_counts())

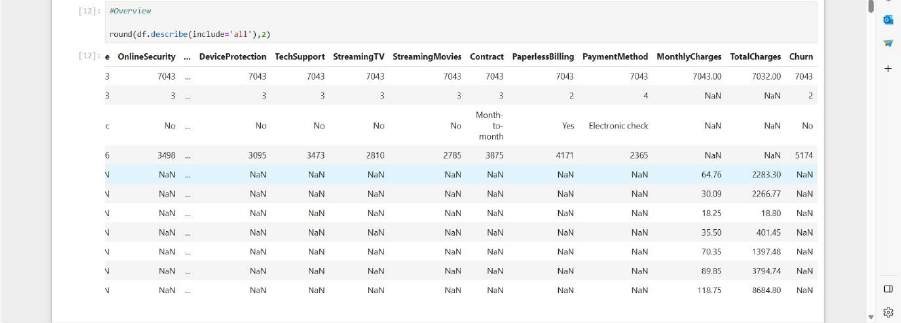
### Check the missing values:

df**.**isnull()**.**values**.**any()



### Overview:

round(df**.**describe(include**=**'all'),2)



### Target value:

*# Count the oocurance of unique values in the 'Churn' Column*

churn\_counts **=** df**.**Churn**.**value\_counts(normalize**=True**) churn\_counts



*# Calculate the percentage of 'Yes' and 'No' label*

total\_count **=** churn\_counts**.**sum()

percentage\_yes **=** (churn\_counts['Yes']**/** total\_count) **\*** 100 percentage\_no **=** (churn\_counts['No']**/** total\_count) **\***100

*# Plot the target value*

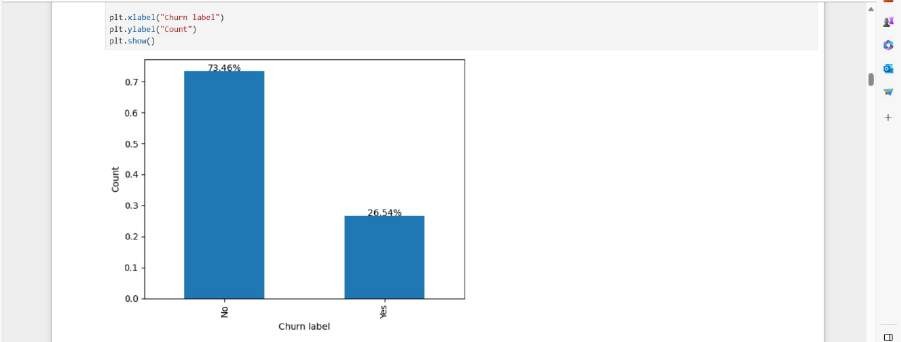
ax **=** churn\_counts**.**plot(kind**=**'bar')

*# Annotate the bars with percentages*

**for** i, count **in** enumerate(churn\_counts):

percentage **=** percentage\_yes **if** i **==** 1 **else** percentage\_no ax**.**annotate(f'{percentage:.2f}%', xy**=**(i, count), ha**=**'center')

plt**.**xlabel("Churn label") plt**.**ylabel("Count") plt**.**show()



# EDA - Exploratory Data Analysis on each feature:

It is observed that the dataset exhibits a significant class imbalance, with a larger amount of data representing non-churners.

Is there a correlation between churn and factors such as monthly charges and total charges?

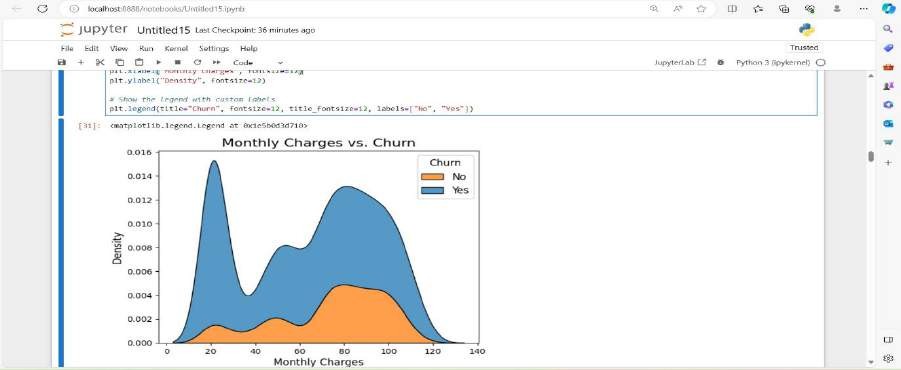
df[['MonthlyCharges','TotalCharges']] df.groupby(['MonthlyCharges','TotalCharges'])['Churn'].size()

sns.kdeplot(data=df, x="MonthlyCharges",hue="Churn",multiple="stack")

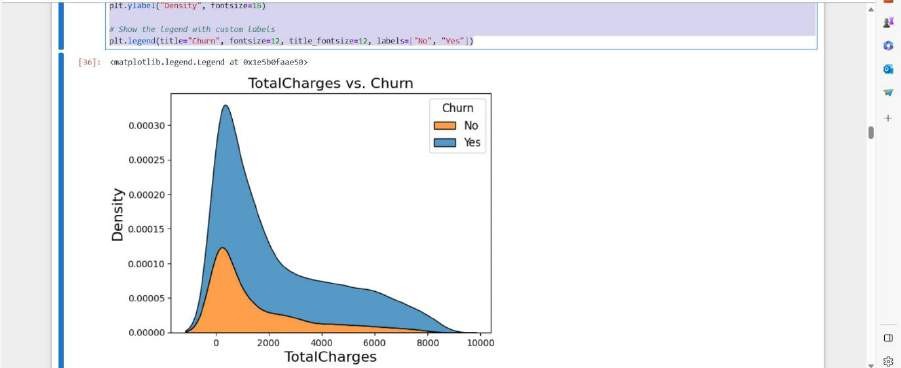
# Customize the plot appearance plt.title("Monthly Charges vs. Churn", fontsize=16) plt.xlabel("Monthly Charges", fontsize=12) plt.ylabel("Density", fontsize=12)

# Show the legend with custom labels

plt.legend(title="Churn", fontsize=12, title\_fontsize=12, labels=["No", "Yes"])



* + Note: it's noticeable that as monthly charges increase within the range of 60 to 120, the density also rises. This trend indicates a higher rate of churn as monthly charges increase.
  + sns.kdeplot(data=df, x="TotalCharges",hue="Churn",multiple="stack")



* + # Customize the plot appearance
  + plt.title("TotalCharges vs. Churn", fontsize=16)
  + plt.xlabel("TotalCharges", fontsize=16)
  + plt.ylabel("Density", fontsize=16)
  + # Show the legend with custom labels
  + plt.legend(title="Churn", fontsize=12, title\_fontsize=12, labels=["No", "Yes"])



* + Note: High churn rates are associated with lower total charges,

with the highest churning occurring in the 0-2000 total charges range.

1. How does the length of a customer's tenure with the company influence their likelihood of churning?

grouped\_data = df.groupby(['tenure', 'Churn']).size().reset\_index(name='count')

grouped\_data[30:40]

# Separate churn and non-churn counts

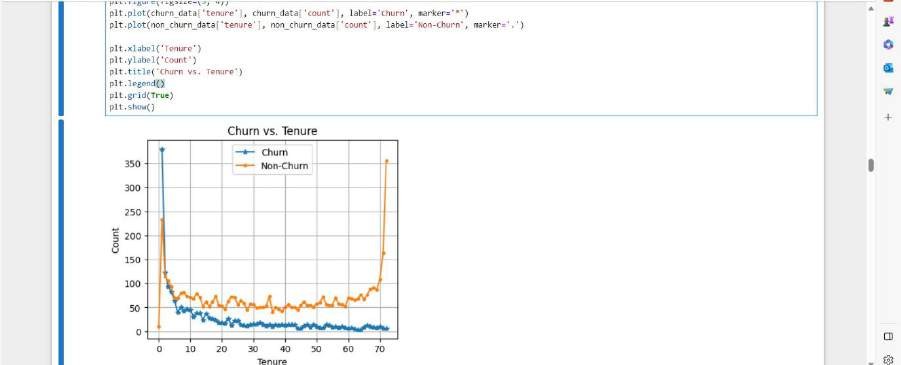
churn\_data = grouped\_data[grouped\_data['Churn'] == 'Yes'] non\_churn\_data = grouped\_data[grouped\_data['Churn'] == 'No'] # Create a line chart for churn and non-churn counts plt.figure(figsize=(5, 4))

plt.plot(churn\_data['tenure'], churn\_data['count'], label='Churn', marker='\*')

plt.plot(non\_churn\_data['tenure'], non\_churn\_data['count'], label='Non-Churn', marker='.')

plt.xlabel('Tenure') plt.ylabel('Count') plt.title('Churn vs. Tenure') plt.legend()

plt.grid(True) plt.show()



Note:

Customers with the low tenure **"0-10”** has the highest rate of churning,these range can be crucial for business decisions.

In general the **Non-Churn** line has small fluctuation and

remains **relatively stable**, with longer tenure tend to stay with the company.

The Churn line decreases or remains relatively stable as tenure increases, it suggests that **customer loyalty** increases with longer tenure

1. connection between gender, partner status, and churn df.gender.value\_counts(normalize=True)

# Create the DataFrame

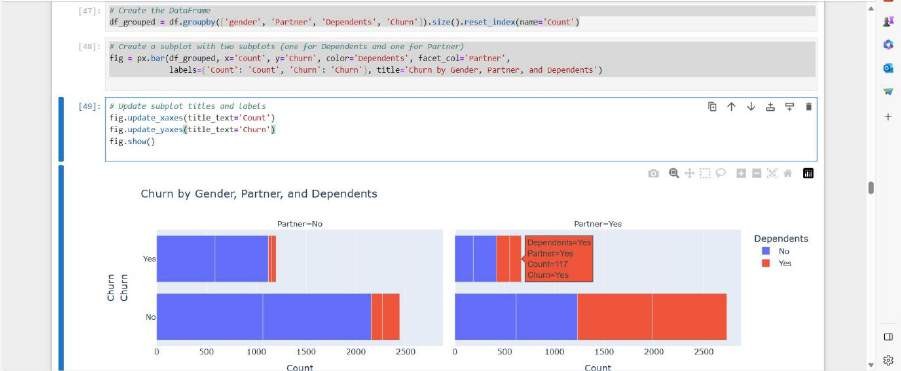
df\_grouped = df.groupby(['gender', 'Partner', 'Dependents', 'Churn']).size().reset\_index(name='Count')

# Create a subplot with two subplots (one for Dependents and one for Partner)

fig = px.bar(df\_grouped, x='Count', y='Churn', color='Dependents', facet\_col='Partner',

labels={'Count': 'Count', 'Churn': 'Churn'}, title='Churn by Gender, Partner, and Dependents')

# Update subplot titles and labels fig.update\_xaxes(title\_text='Count') fig.update\_yaxes(title\_text='Churn') fig.show()



Note:

Regardless of gender, individuals who lack a partner or dependents are at a higher risk of churning.

5.Does the availability of technical support play a role in influencing customer churn? and does the duration of being customer

# Create subplots to compare the distribution fig, axes = plt.subplots(1, 2, figsize=(10, 5))

# Plot the distribution of 'TechSupport' for 'Yes' category

sns.countplot(data=Receive\_techSup, x='TechSupport', hue='Churn', ax=axes[0])

axes[0].set\_title('TechSupport = Yes') axes[0].set\_xlabel('TechSupport') axes[0].set\_ylabel('Count')

# Plot the distribution of 'TechSupport' for 'No' category

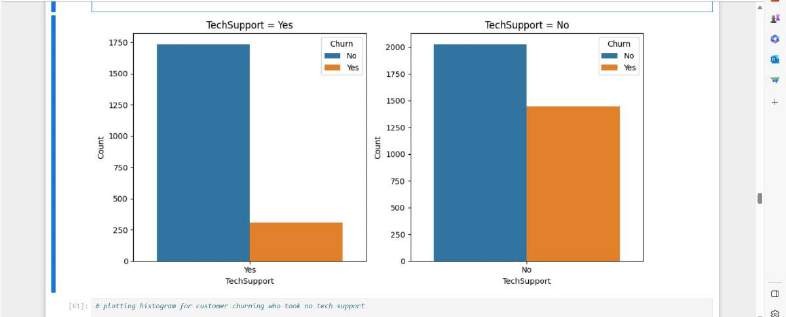
sns.countplot(data=NotReceive\_techSup, x='TechSupport', hue='Churn', ax=axes[1])

axes[1].set\_title('TechSupport = No') axes[1].set\_xlabel('TechSupport') axes[1].set\_ylabel('Count')

# Adjust layout plt.tight\_layout() plt.show()

Note:

Customers who do not receive tech support are more likely to churn.



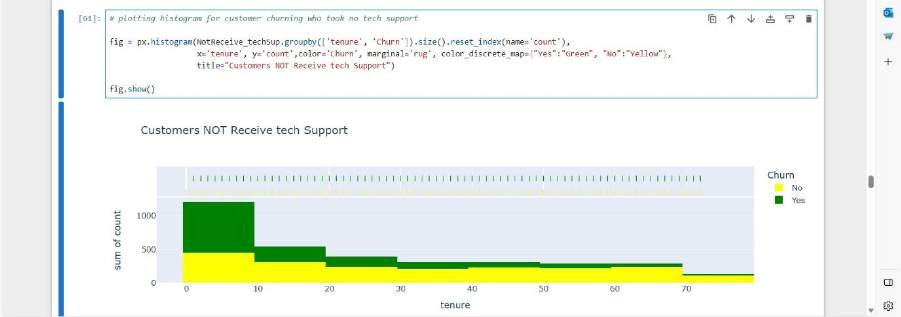
# plotting histogram for customer churning who took no tech support

fig = px.histogram(NotReceive\_techSup.groupby(['tenure', 'Churn']).size().reset\_index(name='count'),

x='tenure', y='count',color='Churn', marginal='rug', color\_discrete\_map={"Yes":"Green", "No":"Yellow"},

title="Customers NOT Receive tech Support")

fig.show()



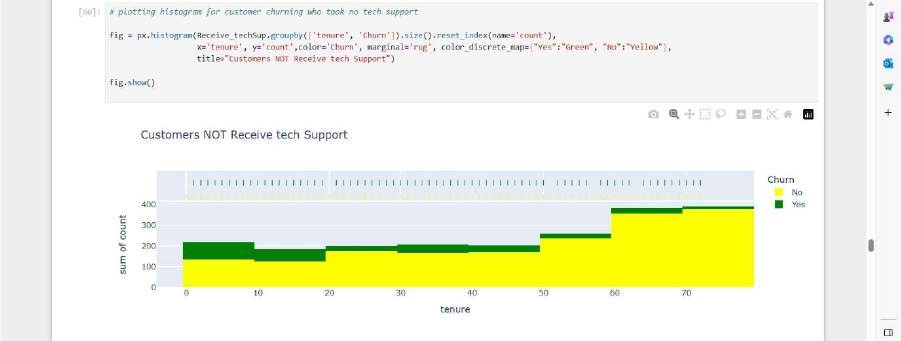
# plotting histogram for customer churning who took no tech support

fig = px.histogram(Receive\_techSup.groupby(['tenure', 'Churn']).size().reset\_index(name='count'),

x='tenure', y='count',color='Churn', marginal='rug', color\_discrete\_map={"Yes":"Green", "No":"Yellow"},

title="Customers NOT Receive tech Support")

fig.show()



Note:

The data indicates that churn rates are highest within the first year of service, especially among customers without tech support. Conversely, longer tenure is associated with increased customer loyalty.

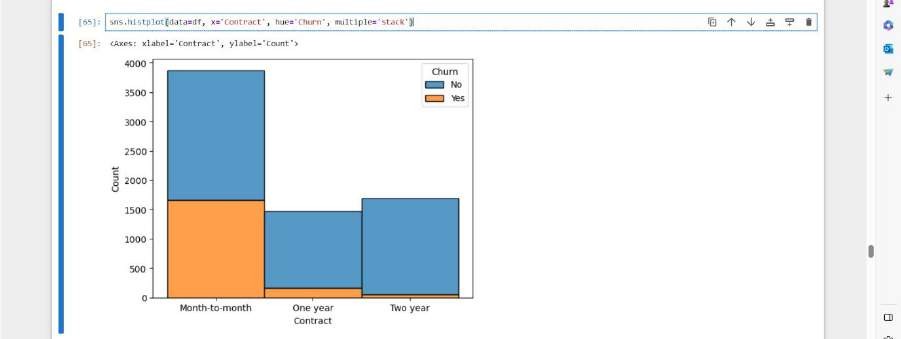
1. .Which aspect of the contract has the most significant impact on the business?

df.Contract.value\_counts(normalize =True) Contract\_condition =

df.groupby(['Churn','Contract']).size().reset\_index(name='count')

Contract\_condition

sns.histplot(data=df, x='Contract', hue='Churn', multiple='stack')



Note:

It is evident that customers with month-to-month contracts have the highest churn rates.

1. .How does the quality of service differ for customers who have opted for streaming services?

df.StreamingTV.value\_counts()

Streming = df.groupby

(['Churn','StreamingTV']).size().reset\_index(name='count')

Streming

ig, axes = plt.subplots(1,2,figsize = (10,5)) # Plot the distribution of 'StreamingTV'

sns.histplot(data=df, x='StreamingTV', hue='Churn', multiple='stack',ax=axes[0])

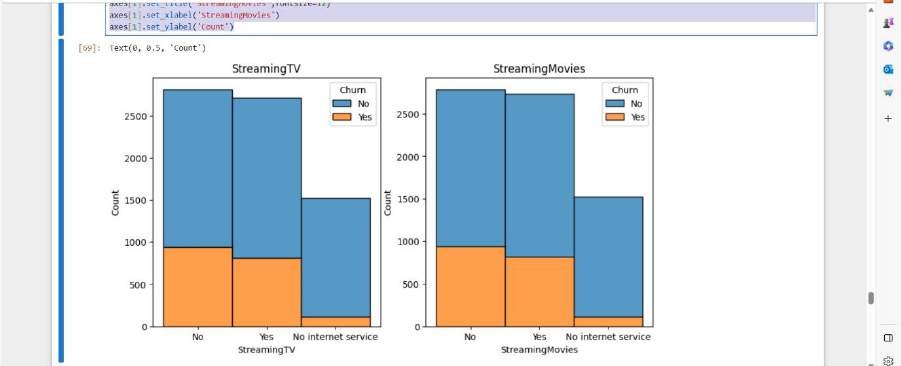
axes[0].set\_title('StreamingTV',fontsize=12) axes[0].set\_xlabel('StreamingTV')

axes[0].set\_ylabel('Count')

# Plot the distribution of 'StreamingMovies' sns.histplot(data=df, x='StreamingMovies', hue='Churn', multiple='stack',ax=axes[1])

axes[1].set\_title('StreamingMovies',fontsize=12) axes[1].set\_xlabel('StreamingMovies')

axes[1].set\_ylabel('Count')



Note: Churn rates are similar for both the 'Yes' and 'No' groups in terms of whether customers are connected to StreamingTV and StreamingMovies.

7. Given that the dataset pertains to the telecom industry, what insights can we uncover regarding phone and internet services?

df.InternetService.value\_counts(normalize=True) df.PhoneService.value\_counts(normalize=True)

# Group the data by "InternetService", "phoneService" and "Churn" and count the number of records

service\_counts = df.groupby(['InternetService','PhoneService', 'Churn']).size().reset\_index(name='Count')

service\_counts

plt.figure(figsize=(10, 5))

# Create two subplots plt.subplot(121)

# 1 row, 2 columns, the first plot for Phone Service

sns.barplot(data=service\_counts[service\_counts['InternetService'] == 'DSL'], x='PhoneService', y='Count', hue='Churn', palette='viridis') plt.xlabel('Phone Service')

plt.ylabel('Count')

plt.title('Counts by Phone Service (DSL Internet)') plt.subplot(122)

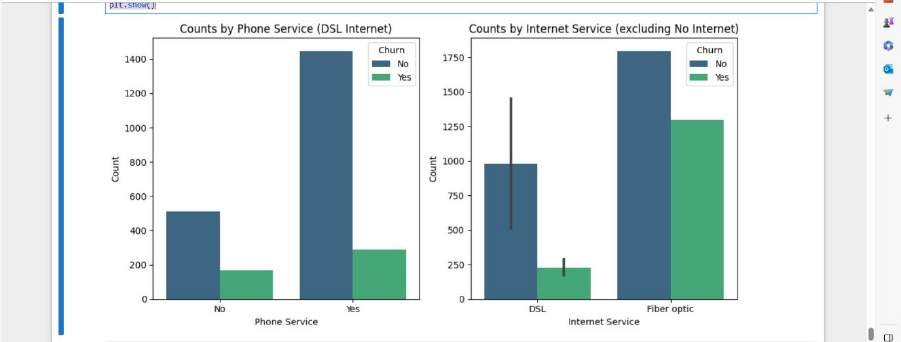
# 1 row, 2 columns, the second plot for Internet Service

sns.barplot(data=service\_counts[service\_counts['InternetService'] != 'No'], x='InternetService', y='Count', hue='Churn', palette='viridis') plt.xlabel('Internet Service')

plt.ylabel('Count')

plt.title('Counts by Internet Service (excluding No Internet)')

plt.tight\_layout() plt.show()



Note:

The plot suggests that the customers with phone service are loyal and have not churned.(the blue bar is significantly taller than green).

For customers without phone service, the blue bar for non-churn is still higher than the green bar. This indicate that even among those without phone service, a significant portion has not churned, and the absence of phone service is associated with a lower churn rate.

**PROBLEM DEFINITION**

Customer churn is a crucial metric impacting a company’s profitability and growth. Losing existing customers leads to revenue decline and incurs costs associated with acquiring new customers to replace the lost ones. High customer churn can also damage a company’s reputation and hinder its long-term sustainability.

**Required steps:**

**1 .**univariate Analysis:

Univariate data requires to analyze each variable separately. Data is gathered for the purpose of answering a question.

1. Bivariate Analysis:

Bivariate analysis is stated to be an analysis of any concurrent relation between two variables or attributes.

1. Time series analysis:

Time Series models are based on data where we observe predictors and churn simultaneously as they occur period to period.

1. Dashboard creation:

A customer churn dashboard is a visual representation of key metrics and data related to customer churn within a business.

1. heatmap:

A Churn Heatmap is a visual summary of every account journey of your churned accounts - from customer acquisition to customer exit - scoring the company's performance at each step with a color code.

1. Cohort analysis:

The examination of different groups of users based on how they interact with your product.

## Use machine learning algorithms to build a predictive model that identifies potential churners based on

**historical data and relevant features.**

# Create separate box plots for 'tenure', 'TotalCharges', and 'MonthlyCharges'

plt.figure(figsize=(18, 6))

# Box Plot of 'tenure'

plt.subplot(131) # 1 row, 3 columns, plot 1 sns.boxplot(x=df['tenure'], color='#66b3ff') plt.title("Box Plot of Tenure")

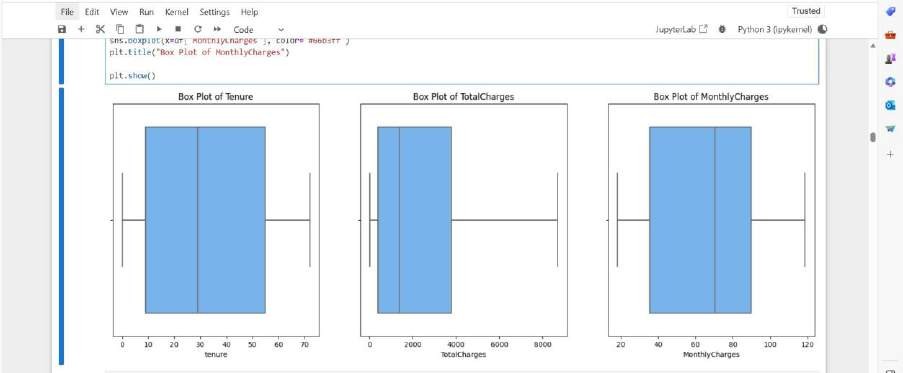
# Box Plot of 'TotalCharges' plt.subplot(132) # 1 row, 3 columns, plot 2

sns.boxplot(x=df['TotalCharges'], color='#66b3ff') plt.title("Box Plot of TotalCharges")

# Box Plot of 'MonthlyCharges' plt.subplot(133) # 1 row, 3 columns, plot 3

sns.boxplot(x=df['MonthlyCharges'], color='#66b3ff') plt.title("Box Plot of MonthlyCharges")

plt.show()



g\_labels = ['Male', 'Female'] c\_labels = ['No', 'Yes']

# Create subplots: use 'domain' type for Pie subplot

fig = make\_subplots(rows=1, cols=2, specs=[[{'type':'domain'},

{'type':'domain'}]])

fig.add\_trace(go.Pie(labels=g\_labels, values=df['gender'].value\_counts(), name="Gender"),

1, 1)

fig.add\_trace(go.Pie(labels=c\_labels, values=df['Churn'].value\_counts(), name="Churn"),

1, 2)

# Use `hole` to create a donut-like pie chart

fig.update\_traces(hole=.4, hoverinfo="label+percent+name", textfont\_size=16)

fig.update\_layout(

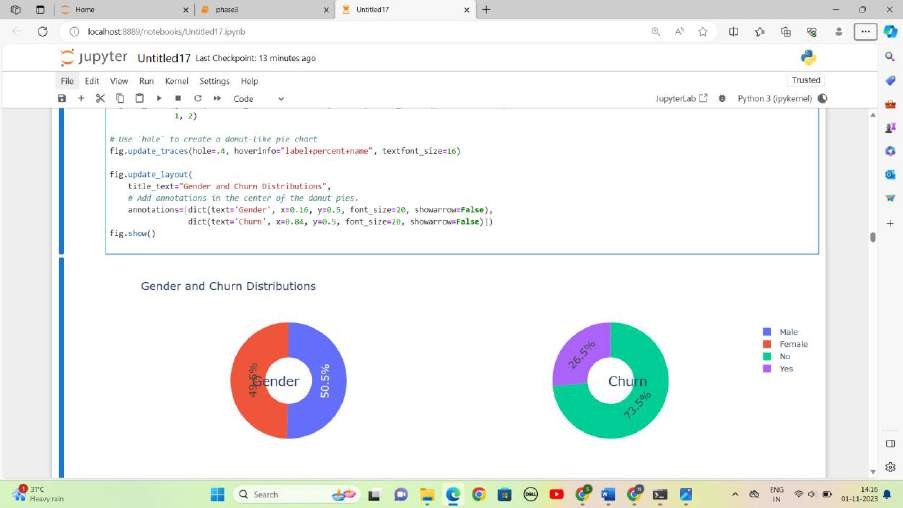
title\_text="Gender and Churn Distributions",

# Add annotations in the center of the donut pies. annotations=[dict(text='Gender', x=0.16, y=0.5, font\_size=20,

showarrow=False),

dict(text='Churn', x=0.84, y=0.5, font\_size=20, showarrow=False)])

fig.show()



plt.figure(figsize=(6, 6))

labels =["Churn: Yes","Churn:No"] values = [1869,5163] labels\_gender = ["F","M","F","M"]

sizes\_gender = [939,930 , 2544,2619] colors = ['#ff6666', '#66b3ff']

colors\_gender = ['#c2c2f0','#ffb3e6', '#c2c2f0','#ffb3e6'] explode = (0.3,0.3)

explode\_gender = (0.1,0.1,0.1,0.1) textprops = {"fontsize":15}

#Plot

plt.pie(values, labels=labels,autopct='%1.1f%%',pctdistance=1.08,

labeldistance=0.8,colors=colors, startangle=90,frame=True, explode=explode,radius=10, textprops =textprops, counterclock = True, )

plt.pie(sizes\_gender,labels=labels\_gender,colors=colors\_gender,s tartangle=90, explode=explode\_gender,radius=7, textprops

=textprops, counterclock = True, ) #Draw circle

centre\_circle = plt.Circle((0,0),5,color='black', fc='white',linewidth=0)

fig = plt.gcf() fig.gca().add\_artist(centre\_circle)

plt.title('Churn Distribution w.r.t Gender: Male(M), Female(F)', fontsize=15, y=1.1)

# show plot

plt.axis('equal') plt.tight\_layout() plt.show()

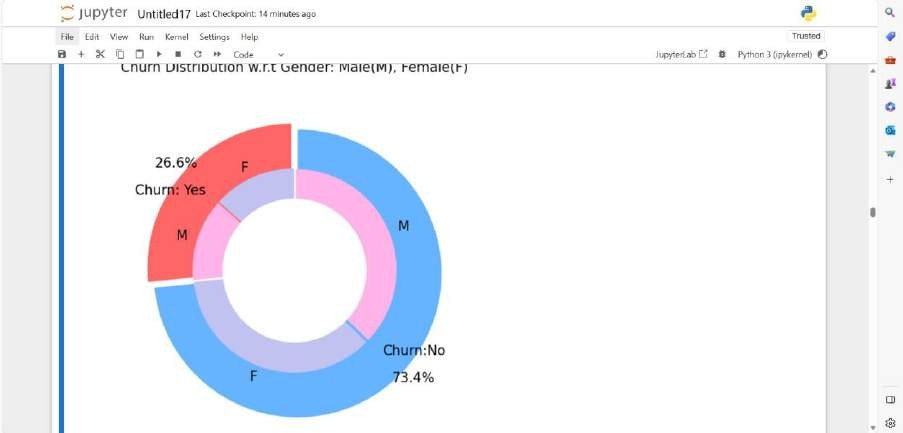
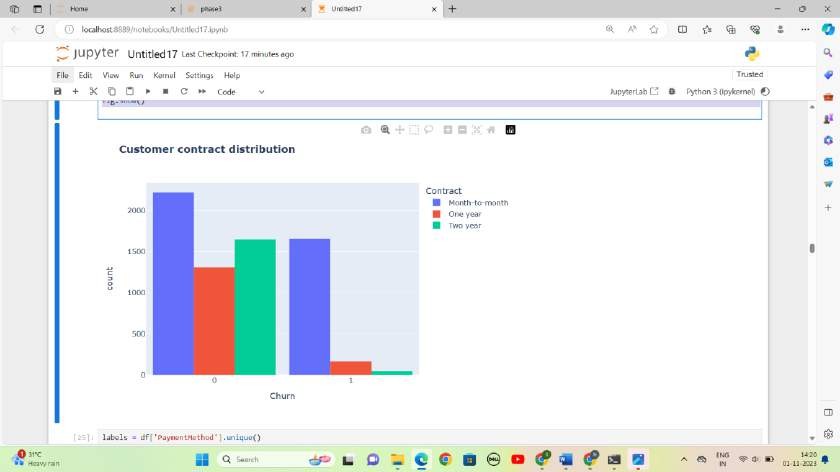


fig = px.histogram(df, x="Churn", color="Contract", barmode="group", title="<b>Customer contract distribution<b>")

fig.update\_layout(width=700, height=500, bargap=0.1) fig.show()

labels = df['PaymentMethod'].unique() values = df['PaymentMethod'].value\_counts()



sns.set\_context("paper", font\_scale=1.1) plt.figure(figsize=(8, 6))

# Line plot for customers who do not churn (Churn = 0)

sns.kdeplot(df.MonthlyCharges[df['Churn'] == 0], color='red', label='Not Churn', shade=True)

# Line plot for customers who churn (Churn = 1)

sns.kdeplot(df.MonthlyCharges[df['Churn'] == 1], color='blue', label='Churn', shade=True)

plt.xlabel('Monthly Charges') plt.ylabel('Density')

plt.title('Distribution of Monthly Charges by Churn')

plt.legend() plt.show()

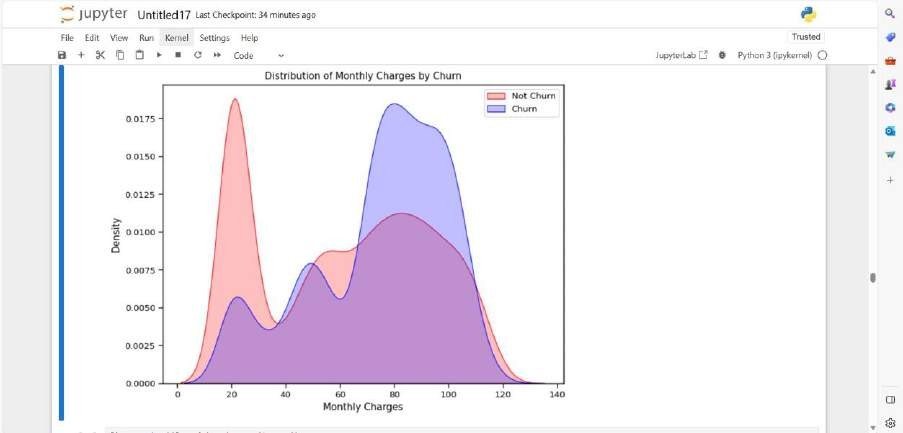


fig = px.box(df, x='Churn', y = 'tenure') # Update yaxis properties

fig.update\_yaxes(title\_text='Tenure (Months)', row=1, col=1) # Update xaxis properties fig.update\_xaxes(title\_text='Churn', row=1, col=1)

# Update size and title

fig.update\_layout(autosize=True, width=750, height=600, title\_font=dict(size=25, family='Courier'),

title='<b>Tenure vs Churn</b>',

)

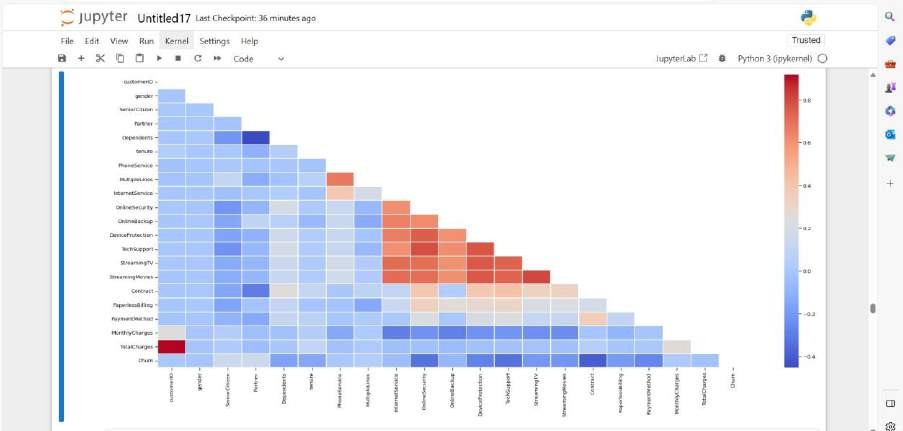
fig.show()



# Correlation between all variables plt.figure(figsize=(25, 10))

corr = df.apply(lambda x: pd.factorize(x)[0]).corr() mask = np.triu(np.ones\_like(corr, dtype=bool))

ax = sns.heatmap(corr, mask=mask, xticklabels=corr.columns, yticklabels=corr.columns, annot=True, linewidths=.2, cmap='coolwarm')



# Correlation between churn and selected boolean and numeric variables

plt.figure

ds\_corr = df[['SeniorCitizen', 'Partner', 'Dependents', 'tenure', 'PhoneService', 'PaperlessBilling', 'MonthlyCharges', 'TotalCharges']]

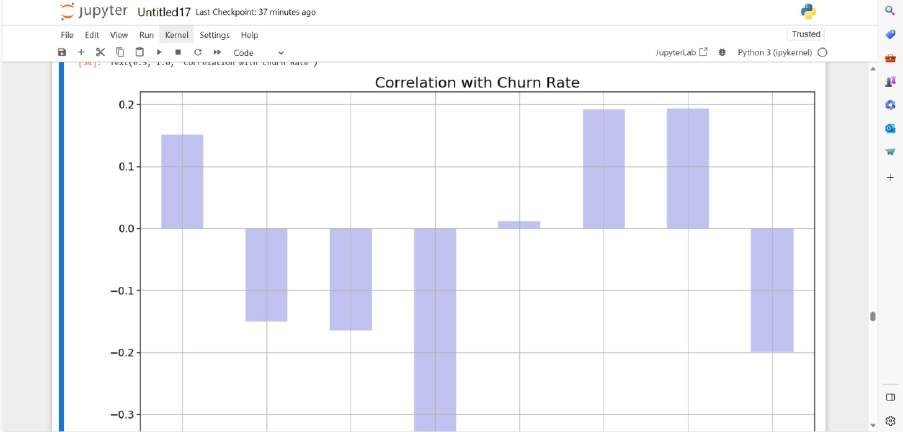
correlations = ds\_corr.corrwith(df.Churn) correlations = correlations[correlations!=1] correlations.plot.bar(

figsize = (18, 10),

fontsize = 15, color = '#c2c2f0',

rot = 45, grid = True)

plt.title('Correlation with Churn Rate', horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans")



knn\_model = KNeighborsClassifier(n\_neighbors = 10) knn\_model.fit(X\_train,y\_train)

# Evaluate model

accuracy\_knn = knn\_model.score(X\_test,y\_test) print("Accuracy of K-Nearest Neighbor: ", accuracy\_knn)

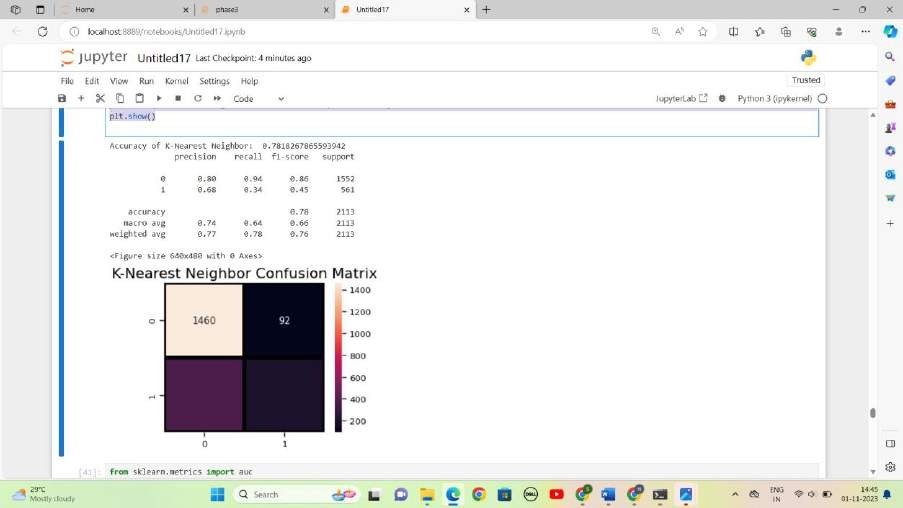
# Classification report

knn\_prediction = knn\_model.predict(X\_test) print(classification\_report(y\_test, knn\_prediction))

plt.figure(14) plt.figure(figsize=(4,3))

sns.heatmap(confusion\_matrix(y\_test, knn\_prediction), annot=True, fmt = "d", linecolor="k", linewidths=3)

plt.title("K-Nearest Neighbor Confusion Matrix", fontsize=16) plt.show()



a\_index = list(range(1, 15))

accuracies = [] # Initialize an empty list to store accuracy values

for i in a\_index:

model = KNeighborsClassifier(n\_neighbors=i) model.fit(X\_train, y\_train)

prediction = model.predict(X\_test)

accuracy = metrics.accuracy\_score(y\_test, prediction) accuracies.append(accuracy)

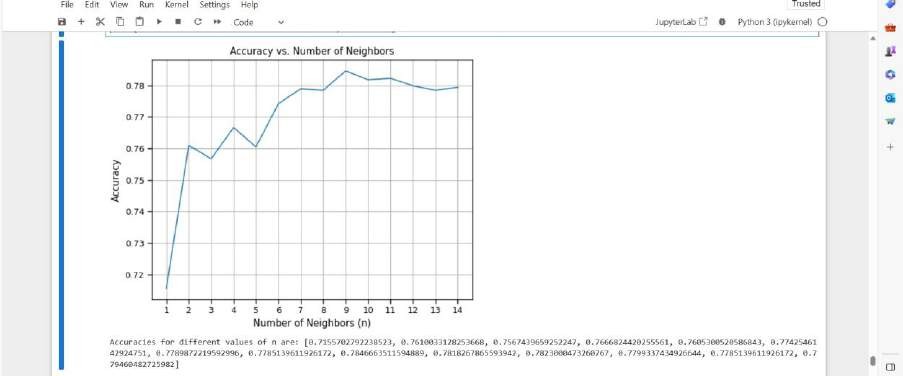
# Plot the accuracy values import matplotlib.pyplot as plt

plt.plot(a\_index, accuracies) plt.xticks(a\_index) plt.xlabel('Number of Neighbors (n)') plt.ylabel('Accuracy')

plt.title('Accuracy vs. Number of Neighbors') plt.grid(True)

plt.show()

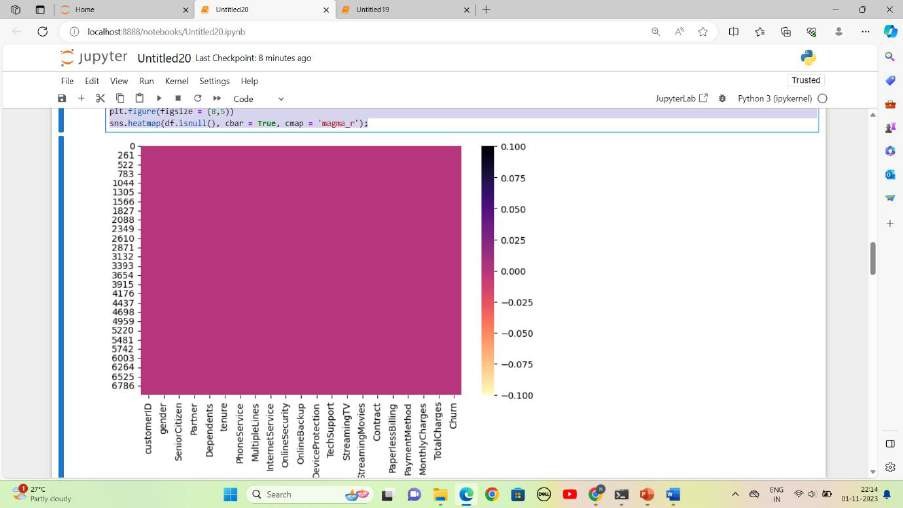
print('Accuracies for different values of n are:', accuracies)



# visualizing the missing values in 'TotalCharges'

plt.figure(figsize = (8,5))

sns.heatmap(df.isnull(), cbar = True, cmap = 'magma\_r');



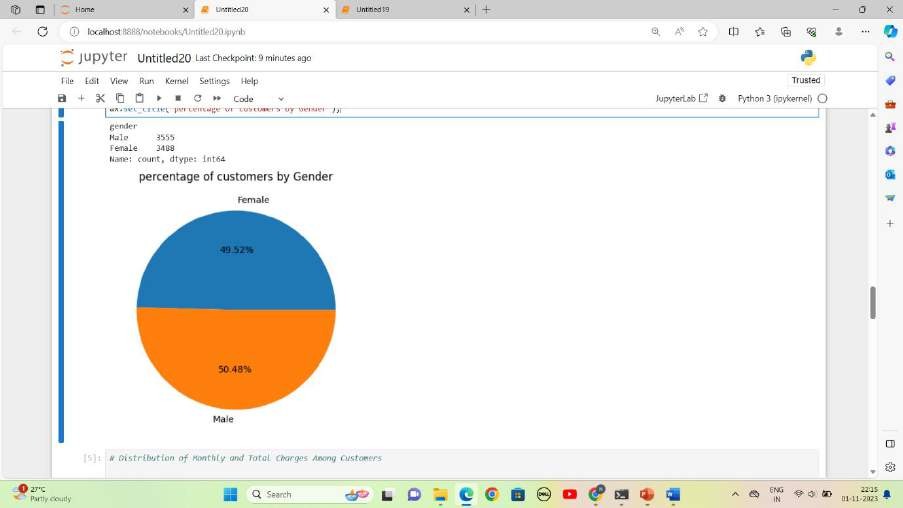
#Data Analysis #Univariate Analysis

# percentage of customers by Gender print(df['gender'].value\_counts())

fig, ax = plt.subplots(figsize = (5,5)) count = Counter(df['gender'])

ax.pie(count.values(), labels = count.keys(), autopct = lambda p:f'{p:.2f}%')

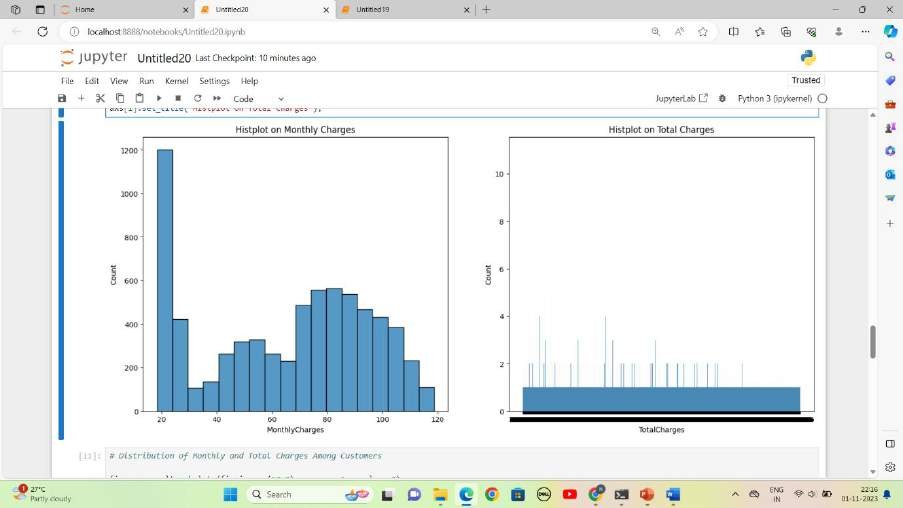
ax.set\_title('percentage of customers by Gender');



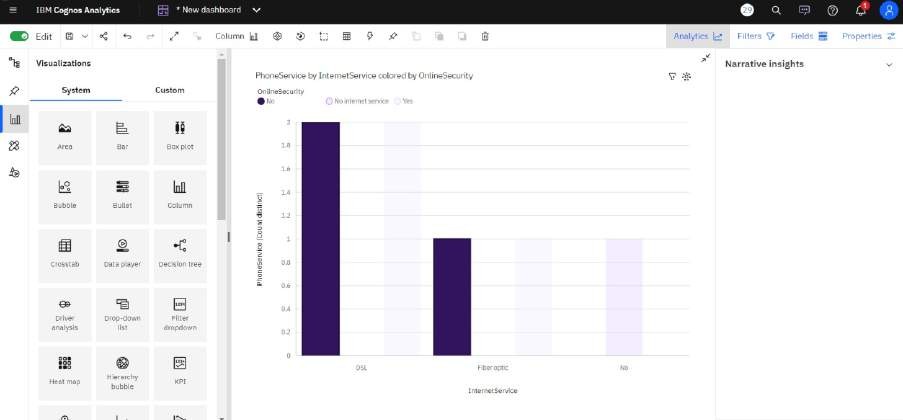
# Distribution of Monthly and Total Charges Among Customers fig, axs = plt.subplots(figsize = (17,7), nrows = 1, ncols = 2)

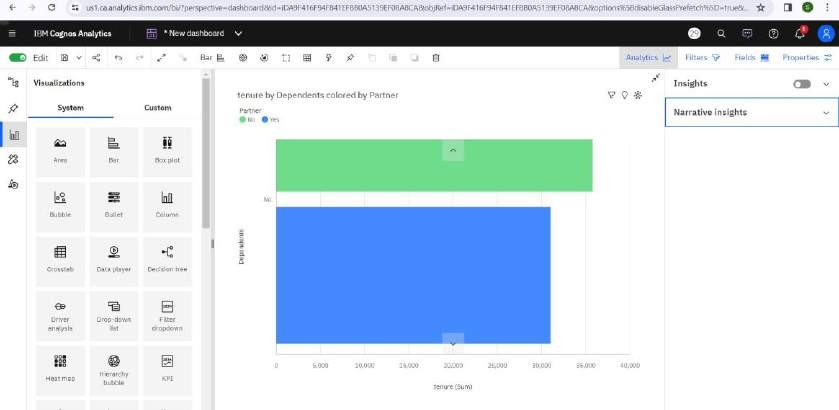
sns.histplot(x = 'MonthlyCharges', data = df, ax = axs[0]) axs[0].set\_title('Histplot on Monthly Charges')

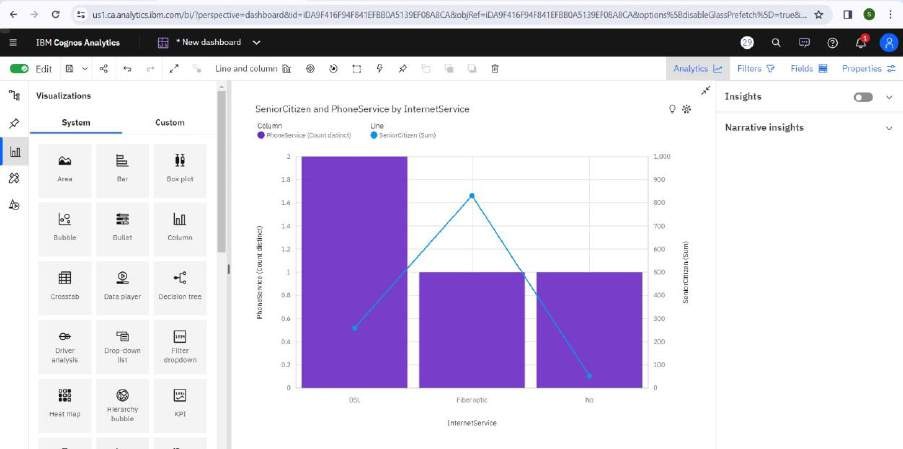
sns.histplot(x = 'TotalCharges', data = df, ax = axs[1]) axs[1].set\_title('Histplot on Total Charges');

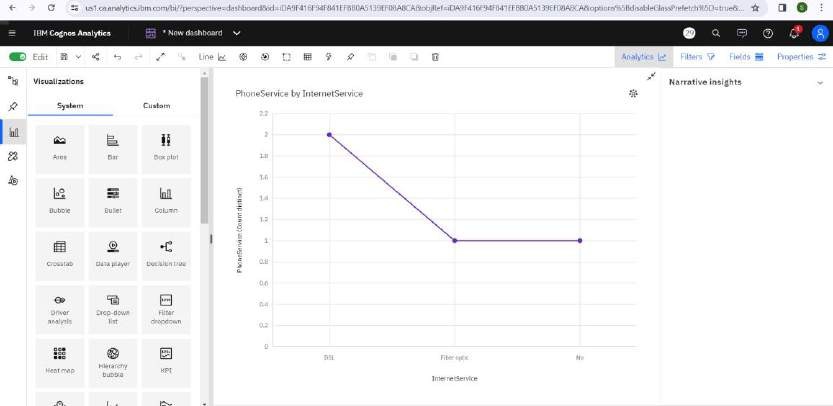


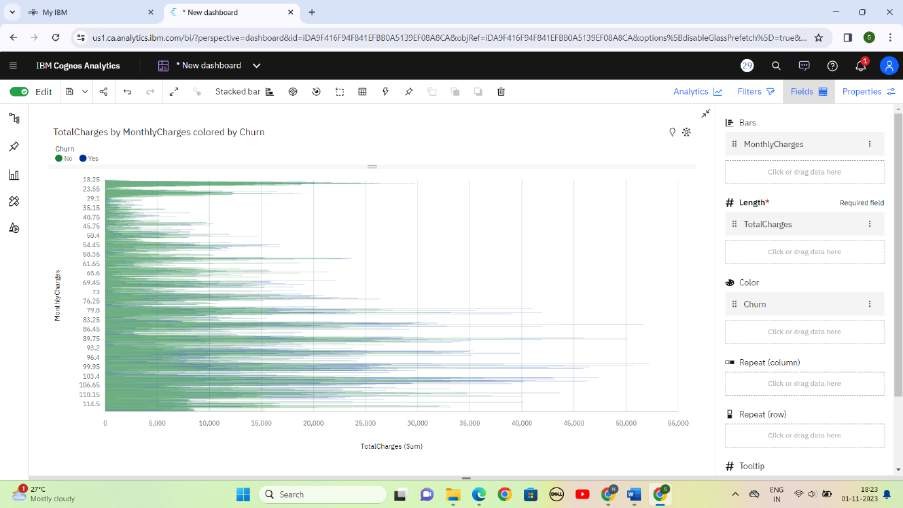
## Create interactive dashboards and reports in IBM Cognos to visualize churn patterns, retention rates, and key factors influencing churn.

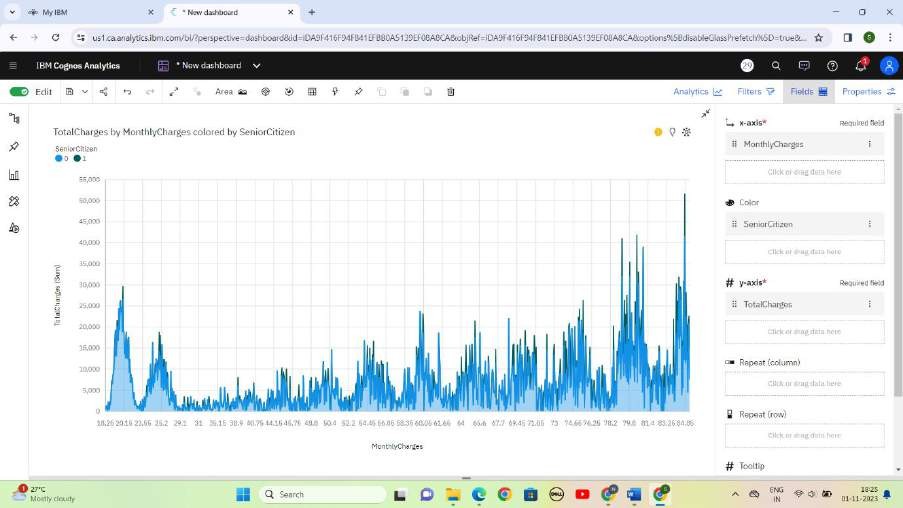


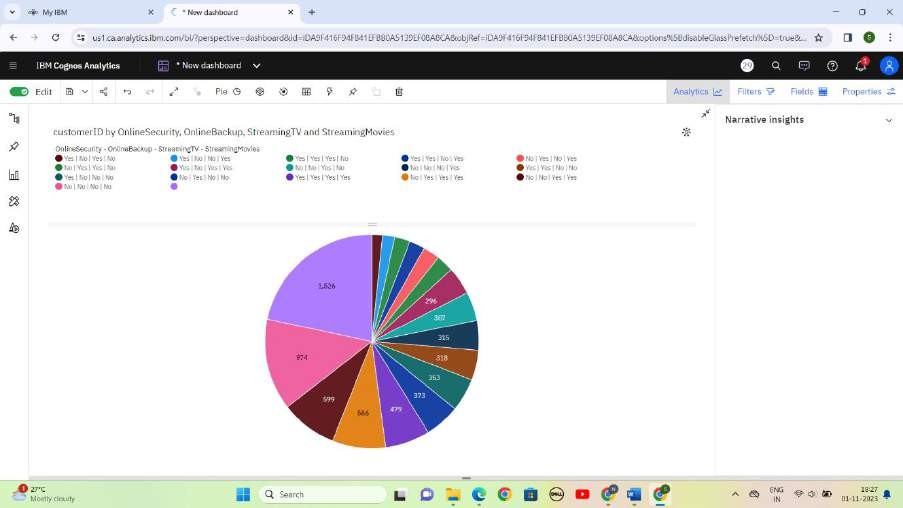


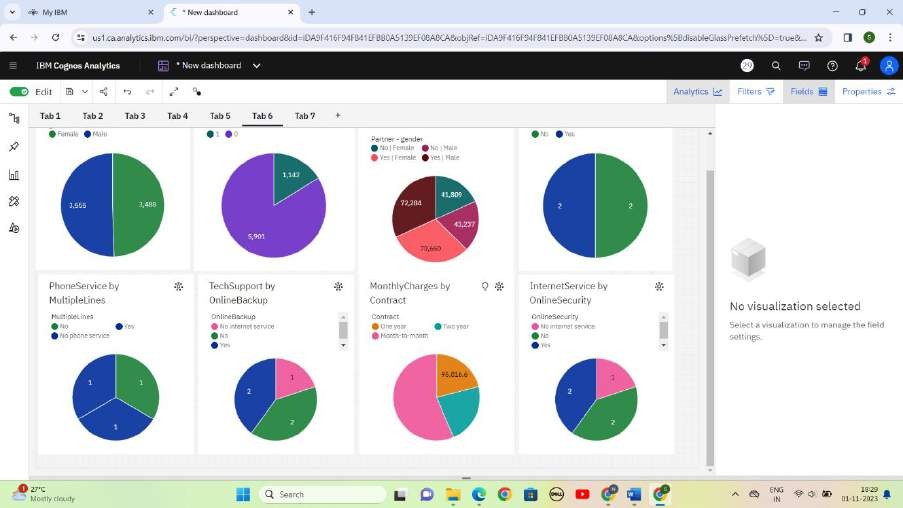


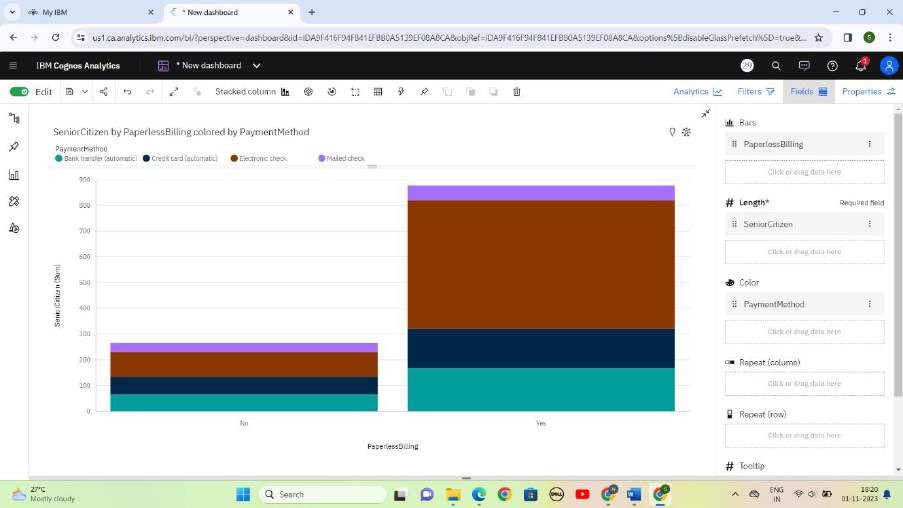


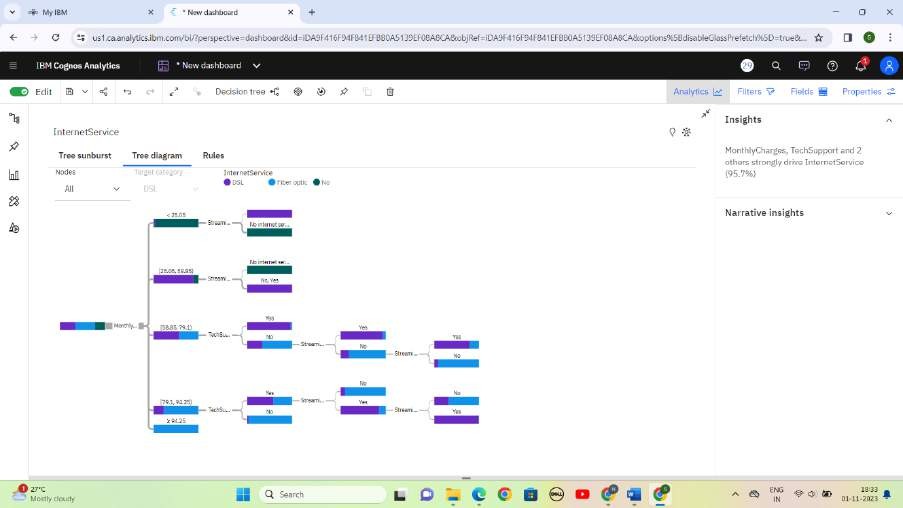












### Insights and prediction model to help businesses reduce customer churn.

1. Import the necessary libraries. import pandas as pd

from sklearn.model\_selection import train\_test\_split from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

from sklearn.preprocessing import LabelEncoder

# Load the dataset

data = pd.read\_csv(r"E:\nokitha nan mudhalvan\WA\_Fn-UseC\_- Telco-Customer-Churn.csv")

# Drop irrelevant columns

data = data.drop(['customerID'], axis=1)

# Convert categorical features to numerical

categorical\_cols = data.select\_dtypes(include=['object']).columns le = LabelEncoder()

for col in categorical\_cols:

data[col] = le.fit\_transform(data[col])

# Split the data into features and target variable X = data.drop('Churn', axis=1)

y = data['Churn']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Create and train a Random Forest classifier

rf\_classifier = RandomForestClassifier(n\_estimators=100, random\_state=42)

rf\_classifier.fit(X\_train, y\_train)

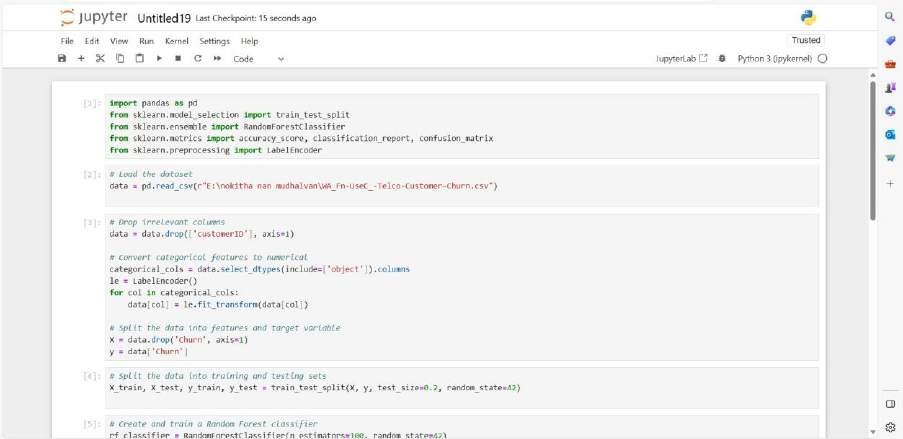
# Make predictions on the test set y\_pred = rf\_classifier.predict(X\_test)

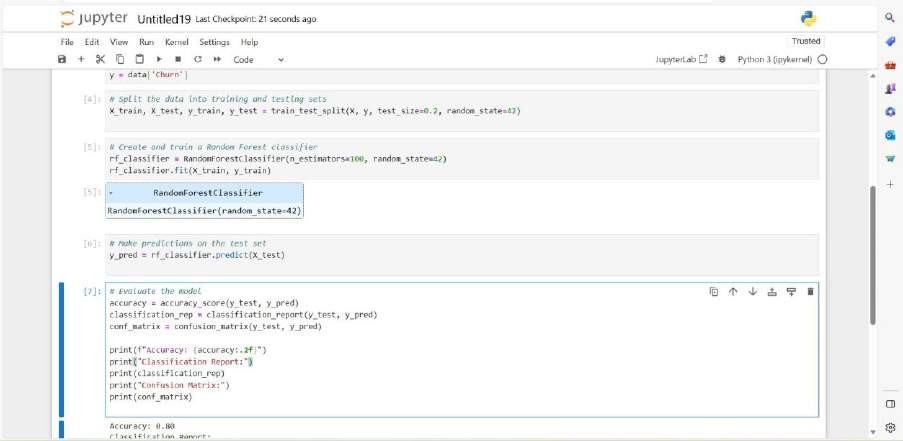
# Evaluate the model

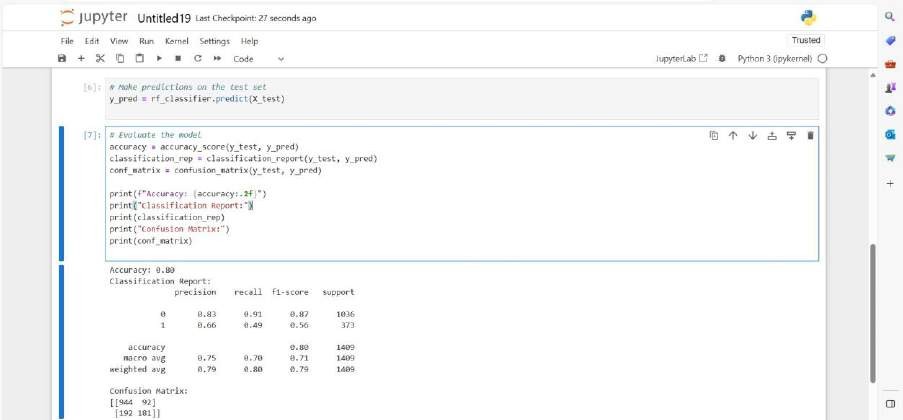
accuracy = accuracy\_score(y\_test, y\_pred) classification\_rep = classification\_report(y\_test, y\_pred) conf\_matrix = confusion\_matrix(y\_test, y\_pred)

print(f"Accuracy: {accuracy:.2f}") print("Classification Report:") print(classification\_rep)

print("Confusion Matrix:") print(conf\_matrix)







### CONCLUSION:

In conclusion, the telecom customer churn prediction model plays a pivotal role in today's highly competitive telecommunications industry. By harnessing the power of data and machine learning, telecom companies can proactively identify and retain valuable customers while reducing customer attrition. This model allows businesses to make data-driven decisions, deploy targeted retention strategies, and allocate resources more effectively. It not only enhances customer satisfaction but also contributes to the long-term profitability and sustainability of telecom providers. By continuously improving and fine-tuning such models, telecom companies can stay ahead of the curve, adapt to changing customer dynamics, and build stronger, lasting relationships with their customer base.