Phase 5

Customer Churn Prediction

The objective, design thinking process, and development phases for a customer churn prediction project can be outlined as follows:

1. Objective:

The primary objective of a customer churn prediction project is to proactively identify customers who are likely to leave or stop using your products or services.

2. Design Thinking Process:

- ❖ Empathize: Understand the pain points and needs of your customers. Gather data on past churn cases and conduct surveys or interviews to get insights into customer behavior and reasons for churning. This phase involves:
  - Customer interviews
  - Surveys
  - Data collection and analysis

- ❖ Ideate: Generate ideas on how to address the problem. Explore potential solutions and approaches for churn prediction. This phase involves:
  - Brainstorming

- Identifying data sources
- Exploring different machine learning models

❖ Prototype: Create a prototype of the churn prediction model. Develop a small-scale model using historical data to test the feasibility of the approach. This phase involves:

- Data preprocessing
- Model development

❖ Test: Test the prototype to see if it can accurately predict customer churn. This phase involves:

- Model evaluation
- Performance metrics (e.g., accuracy, precision, recall).

❖ Implement: Once the model is satisfactory, implement it into your business operations. This phase involves integrating the model into your customer relationship management (CRM) system or other relevant systems.

❖ Monitor and Maintain: Continuously monitor the model's performance in real-time. Retrain the model periodically with new data to adapt to changing customer behavior.

3. Development Phases:

❖ Data Collection and Preprocessing:

Gather historical customer data, including demographics, transaction history, customer interactions, and any other relevant data.

Clean and preprocess the data, handling missing values, outliers, and feature engineering.

❖ Feature Selection and Engineering:

Identify the most relevant features for churn prediction.

Create new features that may improve the model's predictive power.

❖ Model Selection and Training:

Choose appropriate machine learning algorithms such as logistic regression, decision trees, random forests, or neural networks.

Split the data into training and testing sets for model training and evaluation.

❖ Model Evaluation and Validation:

Evaluate the model's performance using appropriate metrics.

Use techniques like cross-validation to ensure the model's generalizability.

❖ Deployment:

Deploy the trained model in a production environment where it can make real-time predictions.

❖ Monitoring and Maintenance:

Continuously monitor the model's performance, and retrain it as necessary with new data.

Keep the model up-to-date with changing customer behavior and business conditions.

❖ Feedback Loop:

Continuously gather feedback from the business and customer support teams to improve the model and its outcomes.

By following these design thinking principles and development phases, you can create an effective customer churn prediction system that helps your business reduce customer churn and improve overall customer satisfaction.

Analyzing customer churn and predicting it effectively involves several key steps, including defining analysis objectives, collecting data, visualizing data using tools like IBM Cognos, and building predictive models.

 Analysis Objectives:

The analysis objectives for customer churn prediction are crucial in guiding the project. Common objectives include:

1. Identifying factors influencing customer churn.

2. Building predictive models to estimate the likelihood of customer churn.

3. Developing actionable insights to reduce churn and retain valuable customers.

4. Monitoring and evaluating the performance of churn prediction models over time.

 Data Collection Process:

The data collection process involves the following steps:

1. Data Sources: Identify sources of customer data, which may include CRM databases, customer support records, transaction history, and customer feedback.

2. Data Extraction: Extract data from these sources, ensuring data integrity and quality.

3. Data Integration: Merge and consolidate data from different sources into a unified dataset.

4. Data Cleaning: Clean the data by handling missing values, outliers, and inconsistencies.

5. Feature Engineering: Create relevant features, such as customer tenure, purchase frequency, and customer interaction history.

6. Data Labeling: Label customers as "churn" or "non-churn" based on historical records.

- Data Visualization using IBM Cognos:

IBM Cognos is a powerful business intelligence and data visualization tool. You can use it to explore and understand your customer churn data.

1. Data Import: Import the cleaned and preprocessed data into IBM Cognos.

2. Data Exploration: Use Cognos to explore your data. Create reports, dashboards, and visualizations to gain insights into customer behavior and churn patterns.

3. Key Metrics: Identify and visualize key metrics that may influence churn, such as customer demographics, purchase history, customer service interactions, and customer feedback.

4. Segmentation: Segment your customer base to understand different groups of customers. Create visualizations that show how churn rates vary across these segments.

5. Trend Analysis: Use line charts and time-series visualizations to track churn trends over time. This can help identify seasonal patterns or changes in churn rates.

6. Drill-Down Analysis: Allow for drill-down capabilities in your visualizations to explore specific customer segments or time periods in more detail.

7. Heatmaps and Scatter Plots: Utilize heatmaps to identify correlations between different customer attributes and churn. Scatter plots can help visualize relationships between two or more variables.

- Predictive Modeling:

Predictive modeling is a critical step in customer churn prediction. You can build machine learning models to estimate the likelihood of a customer churning.

1. Data Split: Split the data into training and testing sets to evaluate model performance.

2. Model Selection: Choose appropriate machine learning algorithms for churn prediction. Common choices include logistic regression, decision trees, random forests, and gradient boosting.

3. Feature Selection: Select the most relevant features that have the most significant impact on churn prediction.

4. Model Training: Train the selected model on the training dataset.

5. Model Evaluation: Evaluate the model using appropriate performance metrics, such as accuracy, precision, recall, F1-score, and ROC AUC.

6. Model Deployment: Deploy the trained model in a production environment where it can make real-time predictions on new data.

8. Actionable Insights: Use the model's predictions to take proactive actions, such as targeted marketing campaigns, personalized offers, or customer support interventions to reduce churn.

9. Feedback Loop: Gather feedback from business and customer support teams to continuously improve the model's accuracy and effectiveness.

Insights derived from data analysis and predictive models can be invaluable for businesses in reducing customer churn.

1. Identifying High-Risk Customers: Predictive models can identify customers with a high likelihood of churning in the near future. Businesses can focus their efforts on retaining these customers by offering personalized incentives or addressing their concerns promptly.

2. Segmentation and Targeted Marketing: Insights from data analysis and predictive models can help businesses segment their customer base effectively.

3. Personalized Offers and Recommendations: Predictive models can suggest personalized offers, discounts, or product recommendations to individual customers.

4. Early Intervention: By identifying potential churn indicators early, businesses can take proactive measures to prevent churn. This might include offering special support, resolving complaints, or reaching out to customers to address their issues before they decide to leave.

5. Customer Feedback Integration: Insights from customer feedback can be integrated with predictive models. If customers express dissatisfaction or specific concerns, the business can use this information to proactively address issues and improve overall customer satisfaction.

7. Product/Service Improvement: Understanding why customers leave can lead to product or service improvements. If common issues are related to the quality or performance of the product, addressing these issues can help reduce churn.

8. Cost Reduction: Reducing churn is not only about retaining customers but also about saving costs. Acquiring new customers is often more expensive than retaining existing ones.

9. Data-Driven Decision-Making: Using data-driven insights and predictive models, businesses can make more informed decisions. This can lead to more effective allocation of resources and budget toward retention efforts.

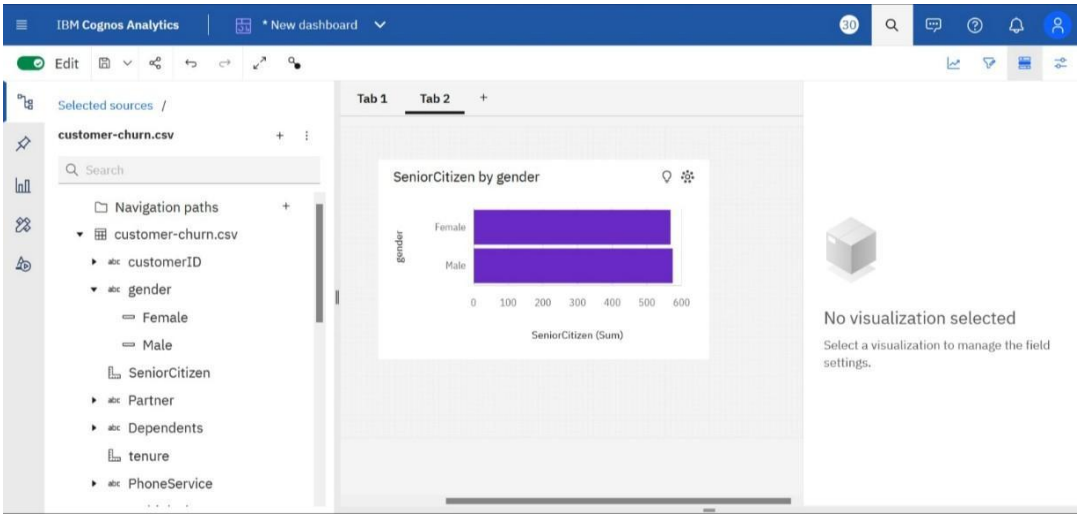
10. Competitive Advantage: A business that effectively reduces churn and retains its customers gains a competitive advantage. Loyal

customers are more likely to refer friends and family, positively review the business, and continue making purchases, contributing to long-term growth and profitability.

11. Long-Term Customer Value: Retained customers have a higher lifetime value to a business than constantly acquiring new ones. Reducing churn is a long-term strategy that can significantly impact a company's bottom line.

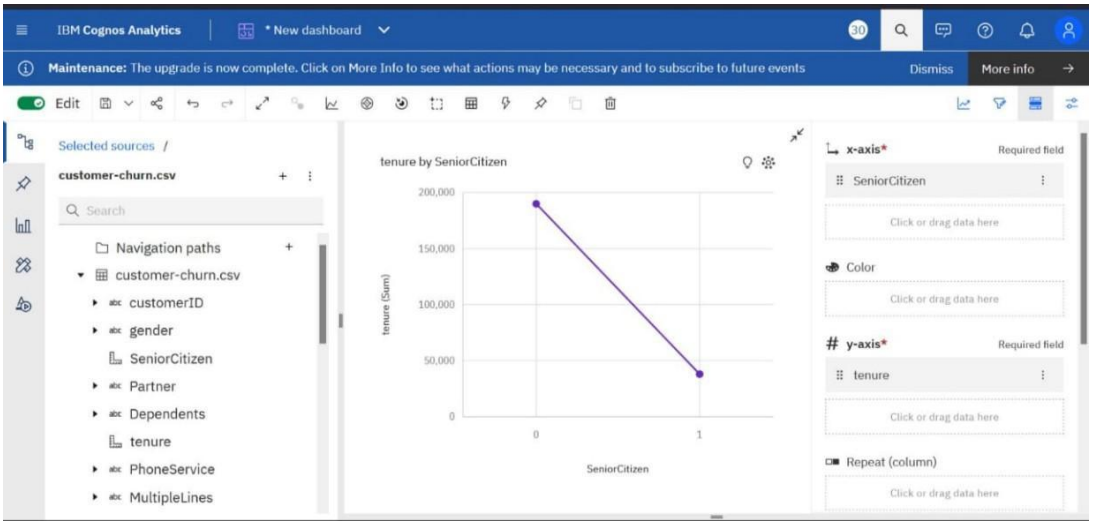
- Visualization using IBM Cognos:

1. **Bar Chart**: A bar chart or bar graph is a chart or graph that presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent. The bars can be plotted vertically or horizontally. A vertical bar chart is sometimes called a column chart.

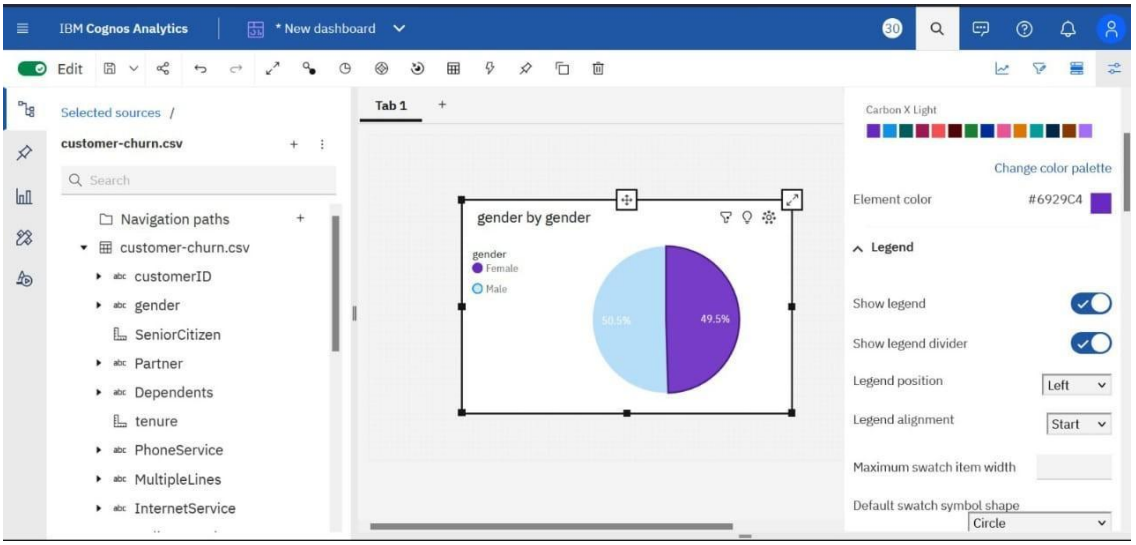


2. **Line Chart:** A line chart is a type of chart used to show information that changes over time. Line charts are created by plotting a series of several points

and connecting them with a straight line. Line charts are used to track changes over short and long periods.

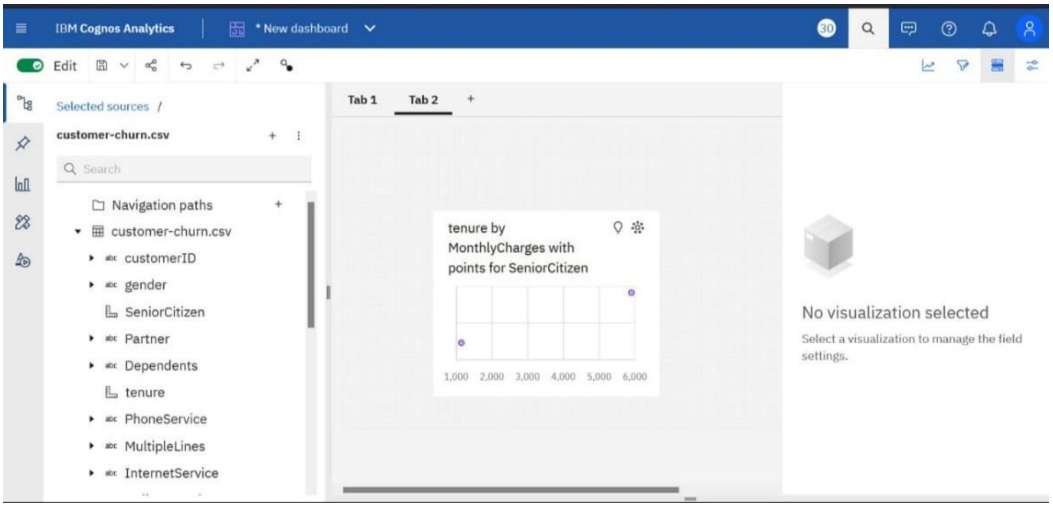


3. Pie Chart: A pie chart is a type of graph that represents the data in the circular graph. The slices of pie show the relative size of the data, and it is a type of pictorial representation of data. A pie chart requires a list of categorical variables and numerical variables. Here, the term “pie” represents the whole, and the “slices” represent the parts of the whole.

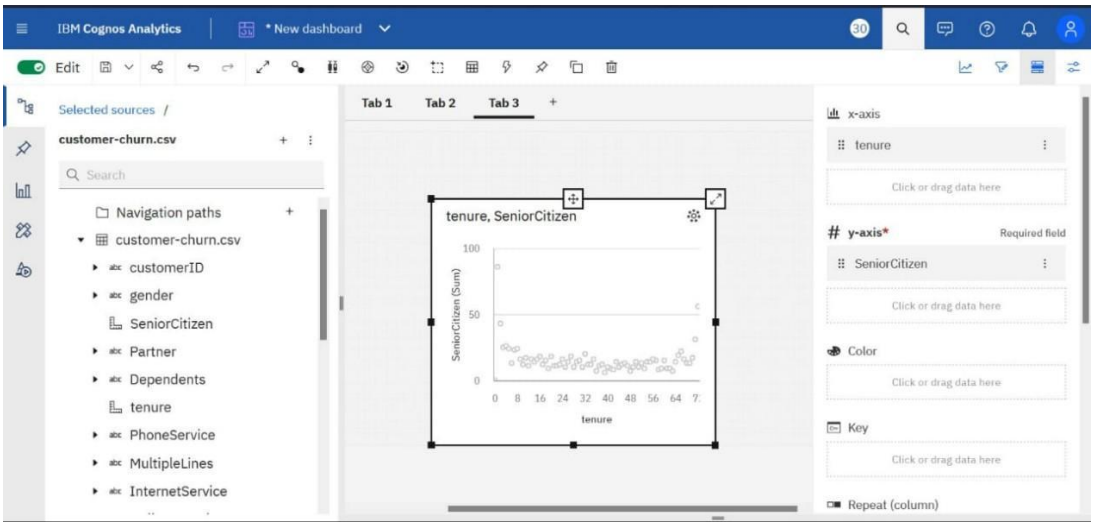


4. Box Plot: A box and whisker plot—also called a box plot—displays the five-number summary of a set of data. The five-number summary is the minimum, first quartile, median, third quartile, and maximum.

In a box plot, we draw a box from the first quartile to the third quartile. A vertical line goes through the box at the median. The whiskers go from each quartile to the minimum or maximum.



5. Scatter Plot: A scatter plot (aka scatter chart, scatter graph) uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. Scatter plots are used to observe relationships between variables.



 Predictive Modeling using Python:

1. importing the modules and packages.

```
[3] import pandas as pd
    import numpy as np
    import seaborn as sns
    import plotly.express as px
    import plotly.graph_objects as go
    from plotly.subplots import make_subplots
    import warnings
    warnings.filterwarnings('ignore')

    from sklearn.preprocessing import LabelEncoder
    from sklearn.preprocessing import StandardScaler


    from sklearn.model_selection import train_test_split
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.preprocessing import StandardScaler
    from sklearn.preprocessing import LabelEncoder

    from sklearn.tree import DecisionTreeClassifier
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.naive_bayes import GaussianNB
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.svm import SVC
    from sklearn.neural_network import MLPClassifier
    from sklearn.ensemble import AdaBoostClassifier
    from sklearn.ensemble import GradientBoostingClassifier
    from sklearn.ensemble import ExtraTreesClassifier
    from sklearn.linear_model import LogisticRegression
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import accuracy_score
    from xgboost import XGBClassifier

    from sklearn import metrics
    from sklearn.metrics import roc_curve
    from sklearn.metrics import recall_score, confusion_matrix, precision_score, f1_score, accuracy_score, classification_report

    from sklearn.ensemble import VotingClassifier
```

2. getting the information of the dataset

```
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
```

```
[9] data.shape
```

```
(7043, 21)
```

```
[10] import missingno as msno
msno.matrix(data)
```

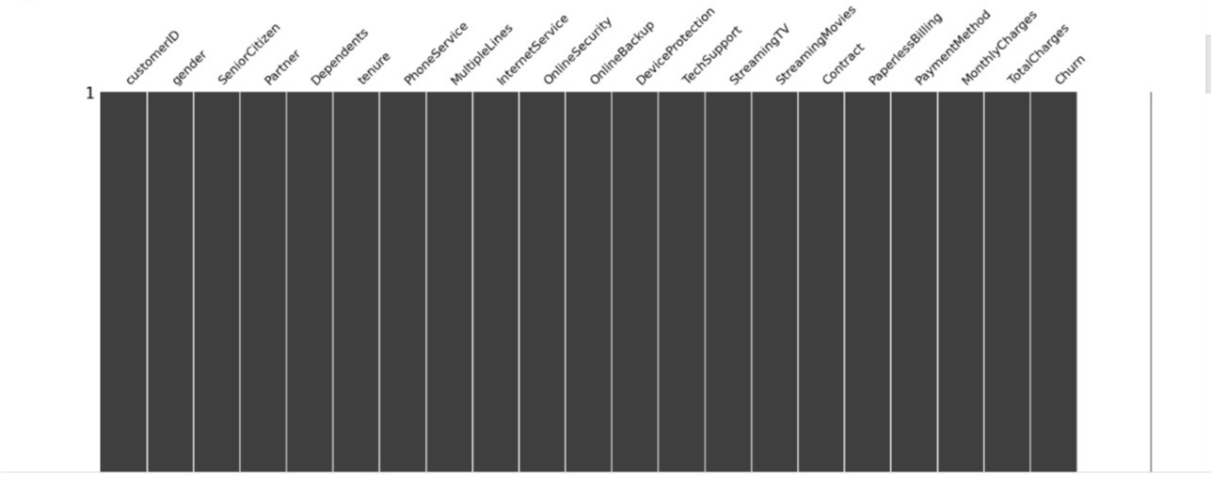
3. creating the matrix of the dataset

```
[9] data.shape
```

```
(7043, 21)
```

```
import missingno as msno
msno.matrix(data)
```

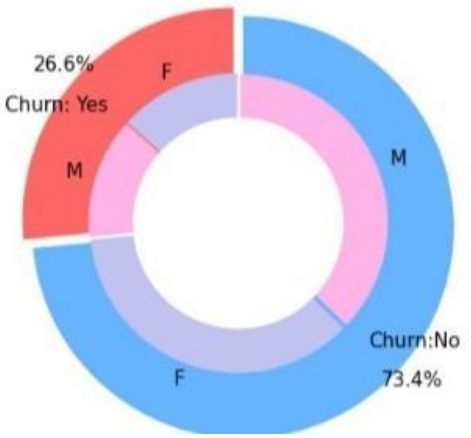
```
<Axes: >
```



4. Analysing our dataset with the visualization of pie chart model.

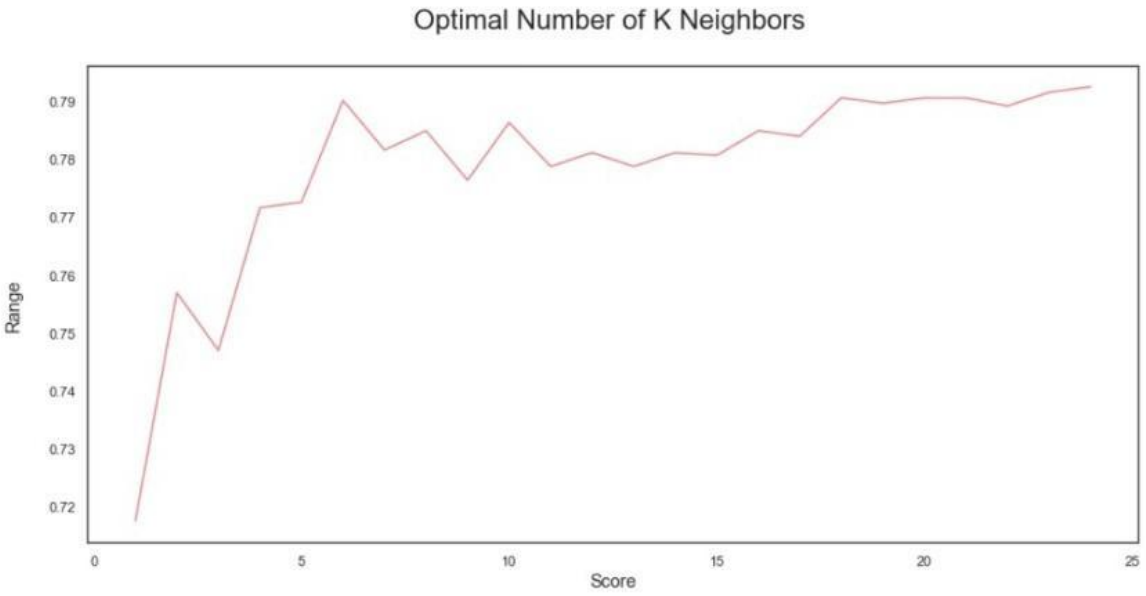
```
plt.figure(figsize=(6, 6))
labels =["Churn: Yes","Churn:No"]
values = [1869,5163]
labels_gender = ["F","M","F","M"]
sizes_gender = [939,930 , 2544,2619]
colors = ['#ff6666', '#66b3ff']
colors_gender = ['#c2c2f0','#ffb3e6', '#c2c2f0','#ffb3e6']
explode = (0.3,0.3)
explode_gender = (0.1,0.1,0.1,0.1)
textprops = {"fontsize":15}
#Plot
plt.pie(values, labels=labels,autopct='%1.1f%%',pctdistance=1.08, labeldistance=0.8,colors=colors, startangle=90,frame=True
plt.pie(sizes_gender,labels=labels_gender,colors=colors_gender,startangle=90, explode=explode_gender,radius=7, textprops =t
#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()
```

Churn Distribution w.r.t Gender: Male(M), Female(F)



5. The k-Nearest Neighbors (KNN) algorithm is a versatile and simple machine learning model that can be applied to various classification and regression tasks. It operates on the principle of similarity, where it classifies or predicts based on the majority class or average value of its k-nearest data points in the feature space.Using the k-nearest the dataset of the customer churn is predicted.

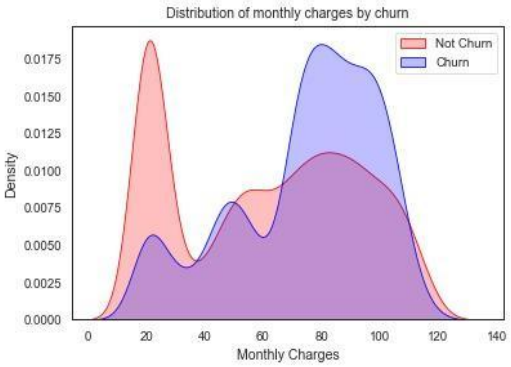
```
fig = plt.figure(figsize=(15, 7))
plt.plot(range(1,25),score_array, color = '#ec838a')
plt.ylabel('Range\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.xlabel('Score\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")

plt.title('Optimal Number of K Neighbors \n',horizontalalignment="center", fontstyle = "normal",fontsize = "22", fontfami
#plt.legend(loc='top right', fontsize = "medium")

plt.xticks(rotation=0, horizontalalignment="center")
plt.yticks(rotation=0, horizontalalignment="right")
plt.show()
```



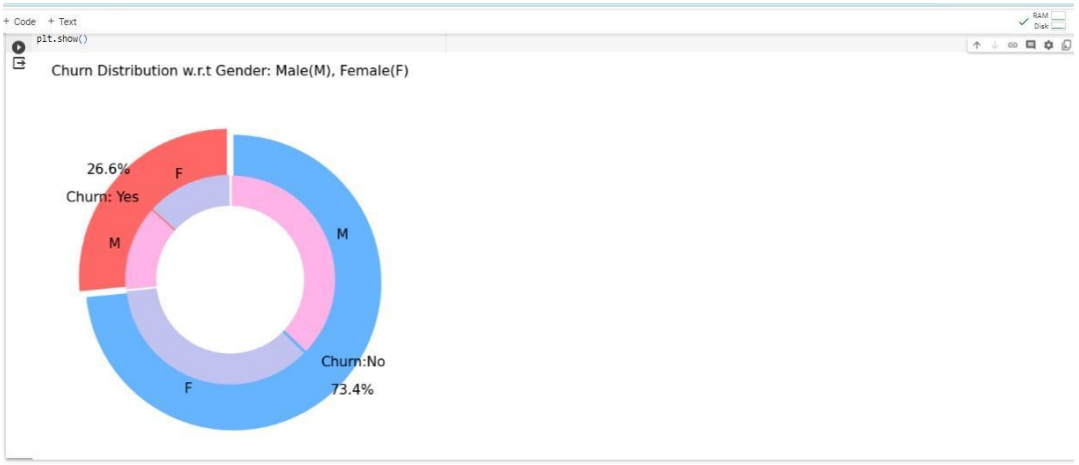
Visualization using Python:

In [327…

```
sns.set_context("paper",font_scale=1.1)
ax = sns.kdeplot(data.MonthlyCharges[(data["Churn"] == 'No') ],
                 color="Red", shade = True);
ax = sns.kdeplot(data.MonthlyCharges[(data["Churn"] == 'Yes') ],
                 ax =ax, color="Blue", shade= True);
ax.legend(["Not Churn","Churn"],loc='upper right');
ax.set_ylabel('Density');
ax.set_xlabel('Monthly Charges');
ax.set_title('Distribution of monthly charges by churn');
```



```
import matplotlib.pyplot as plt

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,startangle=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()
```

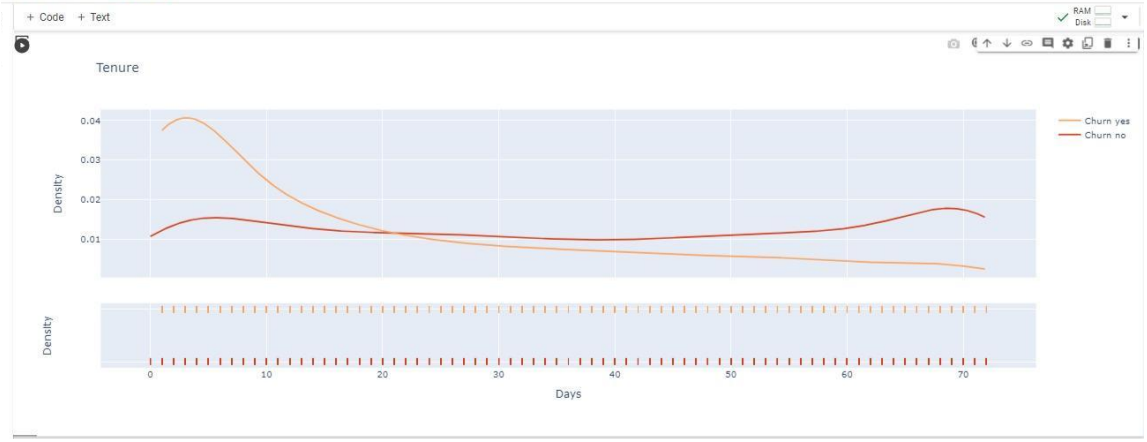


+ Code + Text

```
dtype='object')
```

[ ]

```
import plotly.figure_factory as ff #importing a new function from plotly
x1=df[df["Churn"]=='No']["tenure"]
x2=df[df["Churn"]=='Yes']["tenure"]
hist_data = [x1,x2]

group_labels = ['Churn no','Churn yes']
colors = ['#DC3912', '#FFA15A']

# Create distplot with curve_type set to 'normal'
distplot1 = ff.create_distplot(hist_data, group_labels, show_hist=False, colors=colors)

# Add title
distplot1.update_layout(title_text='Tenure')
distplot1.update_xaxes(title_text='Days')
distplot1.update_yaxes(title_text='Density')
distplot1.show()
```



MODEL EVALUATION

- Mean

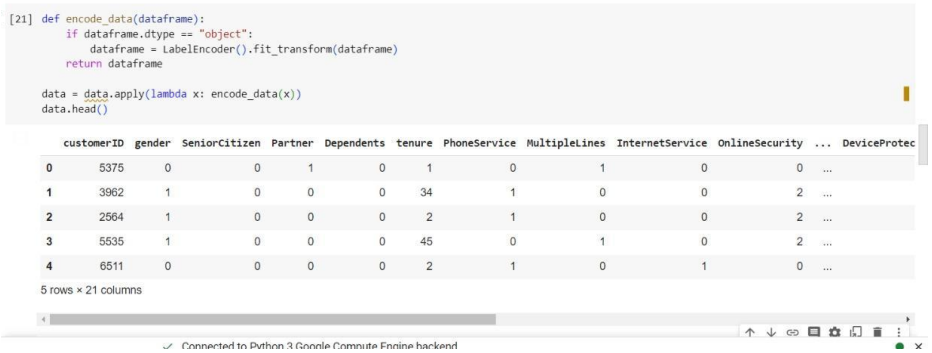
The statistics.mean() function is used to calculate the mean/average of input values or data set.

```
mean_total_charges = data['TotalCharges'].mean()
print("Mean of TotalCharges:", mean_total_charges)
```

```
Mean of TotalCharges: 2283.3004408418656
```

- VIF:

In Python, there are several ways to detect multicollinearity in a dataset, such as using the Variance Inflation Factor (VIF) or calculating the correlation matrix of the independent variables.



```
[21] def encode_data(dataframe):
         if dataframe.dtype == "object":
             dataframe = LabelEncoder().fit_transform(dataframe)
         return dataframe

     data = data.apply(lambda x: encode_data(x))
     data.head()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtec
0	5375	0	0	1	0	1	0	1	0	0	...	
1	3962	1	0	0	0	34	1	0	0	2	...	
2	2564	1	0	0	0	2	1	0	0	2	...	
3	5535	1	0	0	0	45	0	1	0	2	...	
4	6511	0	0	0	0	2	1	0	1	0	...	

5 rows × 21 columns

REPLICATION OF ANALYSIS:

- Data Preprocessing:

To load and preprocess a Customer Churn dataset for analysis, you can follow these general steps using Python and Pandas. Make sure you have a Customer Churn dataset in a suitable format available.

1. **Import Libraries**: Start by importing the necessary Python libraries, including Pandas, to load and preprocess the dataset.

> Import pandas as pd

2. **Load the Customer Churn Dataset**: Load the Customer_churn dataset into a Pandas DataFrame. You can use **pd.read_csv()** for CSV files, but the method may vary depending on the file format.

```
In [ ]: import pandas as pd
```

```
In [9]: import pandas as pd
        df = pd.read_csv("K:\Downloads\customer-churn.csv")
        df
```

Out[9]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechS
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	
...	...	...	...	...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	No	
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes	...	Yes	

7043 rows × 21 columns

3. **Data Inspection:** Before preprocessing, inspect the data to understand its structure and identify any potential issues.

```
In [10]: print(df.head())
```

```
   customerID  gender  SeniorCitizen Partner Dependents  tenure PhoneService  \
0  7590-VHVEG  Female              0     Yes         No       1           No
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]
```

```
In [11]: print(df.isnull().sum())
```

```
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
```

```
In [13]: print(df.dtypes)
```

```
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
```

4. **Data Preprocessing**:

a. Data Cleaning:

Handle missing values by either imputing them or removing rows with missing data

```
In [14]: df.fillna(0,inplace=True)
         df.dropna(inplace=True)
         df
```

Out[14]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechS
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	
...	...	...	...	...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	No	
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes	...	Yes	

7043 rows × 21 columns

b. **Data Transformation:** If necessary, transform the data to suit your analysis objectives. For instance, you may want to aggregate data by date or region.

```
In [17]: df=df.groupby('customerID').agg({'MonthlyCharges':'sum','TotalCharges':'sum'}).reset_index()
         df
```

Out[17]:

	customerID	MonthlyCharges	TotalCharges
0	0002-ORFBO	65.60	593.3
1	0003-MKNFE	59.90	542.4
2	0004-TLHLJ	73.90	280.85
3	0011-IGKFF	98.00	1237.85
4	0013-EXCHZ	83.90	267.4
...	...	...	...
7038	9987-LUTYD	55.15	742.9
7039	9992-RRAMN	85.10	1873.7
7040	9992-UJOEL	50.30	92.75
7041	9993-LHIEB	67.85	4627.65
7042	9995-HOTOH	59.00	3707.6

7043 rows × 3 columns