Project : Customer Churn Dataset

Problem Statement:

You are the data scientist at a telecom company named "Neo" whose customers are churning out to its competitors. You have to analyze the data of your company and find insights and stop your customers from churning out to other telecom companies.

Customer_churn Dataset:

The details regarding this 'customer_churn' dataset are present in the data dictionary

customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
7590-VHVEG	Female	0	Yes	No	1	No	No phone service
5575-GNVDE	Male	0	No	No	34	Yes	No
3668-QPYBK	Male	0	No	No	2	Yes	No
7795-CFOCW	Male	0	No	No	45	No	No phone service
9237-HQITU	Female	0	No	No	2	Yes	No
9305-CDSKC	Female	0	No	No	8	Yes	Yes
1452-KIOVK	Male	0	No	Yes	22	Yes	Yes
6713-OKOMC	Female	0	No	No	10	No	No phone service
7892-POOKP	Female	0	Yes	No	28	Yes	Yes
6388-TABGU	Male	0	No	Yes	62	Yes	No

Lab Environment: Anaconda

Domain: Telecom

Tasks To Be Performed:

1. Data Manipulation:

- Extract the 5th column and store it in 'customer_5'
- Extract the 15th column and store it in 'customer_15'
- Extract all the male senior citizens whose payment method is electronic check and store the result in 'senior_male_electronic'
- Extract all those customers whose tenure is greater than 70 months or their monthly charges is more than $100 and store the result in 'customer_total_tenure'

- Extract all the customers whose contract is of two years, payment method is mailed check and the value of churn is 'Yes' and store the result in 'two_mail_yes'
- Extract 333 random records from the customer_churndataframe and store the result in 'customer_333'
- Get the count of different levels from the 'Churn' column

2. Data Visualization:
- Build a bar-plot for the 'InternetService' column:
    a. Set x-axis label to 'Categories of Internet Service'
    b. Set y-axis label to 'Count of Categories'
    c. Set the title of plot to be 'Distribution of Internet Service'
    d. Set the color of the bars to be 'orange'

- Build a histogram for the 'tenure' column:
    a. Set the number of bins to be 30
    b. Set the color of the bins to be 'green'
    c. Assign the title 'Distribution of tenure'

- Build a scatter-plot between 'MonthlyCharges' and 'tenure'. Map 'MonthlyCharges' to the y-axis and 'tenure' to the 'x-axis':
    a. Assign the points a color of 'brown'
    b. Set the x-axis label to 'Tenure of customer'
    c. Set the y-axis label to 'Monthly Charges of customer'
    d. Set the title to 'Tenure vs Monthly Charges'
    e. Build a box-plot between 'tenure' & 'Contract'. Map 'tenure' on the y-axis &
    f. 'Contract' on the x-axis.

3. Linear Regression:
- Build a simple linear model where dependent variable is 'MonthlyCharges' and independent variable is 'tenure':
    a. Divide the dataset into train and test sets in 70:30 ratio.
    b. Build the model on train set and predict the values on test set
    c. After predicting the values, find the root mean square error
    d. Find out the error in prediction & store the result in 'error'
    e. Find the root mean square error

4. Logistic Regression:
- Build a simple logistic regression model where dependent variable is 'Churn' and independent variable is 'MonthlyCharges':
    a. Divide the dataset in 65:35 ratio
    b. Build the model on train set and predict the values on test set
    c. Build the confusion matrix and get the accuracy score
    d. Build a multiple logistic regression model where dependent variable is 'Churn' and independent variables are 'tenure' and 'MonthlyCharges'
    e. Divide the dataset in 80:20 ratio
    f. Build the model on train set and predict the values on test set
    g. Build the confusion matrix and get the accuracy score

5. Decision Tree:
- Build a decision tree model where dependent variable is 'Churn' and independent variable is 'tenure':
    a. Divide the dataset in 80:20 ratio
    b. Build the model on train set and predict the values on test set
    c. Build the confusion matrix and calculate the accuracy

6. Random Forest:
- Build a Random Forest model where dependent variable is 'Churn' and independent variables are 'tenure' and 'MonthlyCharges':
    a. Divide the dataset in 70:30 ratio
    b. Build the model on train set and predict the values on test set
    c. Build the confusion matrix and calculate the accuracy

jupyter Churn_project Last Checkpoint: 8 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[3]:
```
df=pd.read_csv(r"C:\Users\Shanawaz khan\Downloads\customer_churn (1) (1).csv")
```

[4]:
```
df
```

[4]:

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

jupyter Churn_project Last Checkpoint: 8 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

...	HQITU	...	...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	No
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	No
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	No	No
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes	...	Yes	Yes

7043 rows × 21 columns

[5]:
```
df.shape
```

[5]:
```
(7043, 21)
```

[6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
```

jupyter Churn_project Last Checkpoint: 9 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[5]:
```
(7043, 21)
```

[6]:
```
df.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
```

jupyter Churn_project Last Checkpoint: 9 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
memory usage: 1.1+ MB
```

[8]:
```
df.isnull().sum()
```

[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        0
Churn               0
dtype: int64
```

[10]:
```
#1.Extract the 5th column and store it in 'customer_5'
customer_5=df["Dependents"]
customer_5
```

jupyter Churn_project Last Checkpoint: 10 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
TotalCharges        0
Churn               0
dtype: int64
```

[10]:
```
#1.Extract the 5th column and store it in 'customer_5'
customer_5=df["Dependents"]
customer_5
```

[10]:
```
0        No
1        No
2        No
3        No
4        No
       ...
7038    Yes
7039    Yes
7040    Yes
7041     No
7042     No
Name: Dependents, Length: 7043, dtype: object
```

[11]:
```
customer_5=df.loc[:,["Dependents"]]
customer_5
```

[11]:

	Dependents
0	No
1	No
2	No

jupyter Churn_project Last Checkpoint: 10 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
Name: Dependents, Length: 7043, dtype: object
```

[11]:
```
customer_5=df.loc[:,["Dependents"]]
customer_5
```

[11]:

	Dependents
0	No
1	No
2	No
3	No
4	No
...	...
7038	Yes
7039	Yes
7040	Yes
7041	No
7042	No

7043 rows × 1 columns

[13]:
```
#3.Extract the 15th column and store it in 'customer_15'
```

jupyter Churn_project Last Checkpoint: 10 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

```
[13]: #3.Extract the 15th column and store it in 'customer_15'
customer_15=df.loc[:,["StreamingMovies"]]
customer_15
```

[13]:

	StreamingMovies
0	No
1	No
2	No
3	No
4	No
...	...
7038	Yes
7039	Yes
7040	No
7041	No
7042	Yes

7043 rows × 1 columns

```
[15]: #4.Extract all the male senior citizens whose payment method is electroniccheck and store the result in 'senior_male_electronic'
senior_male_electronic=df[(df["gender"]=="Male")&(df["SeniorCitizen"]==1)&(df["PaymentMethod"]=="Electronic check")]
senior_male_electronic
```

[15]:

	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	Monthly
	No phone service	DSL	No	...	Yes	No	No	Yes	Month-to-month	Yes	Electronic check	
	Yes	Fiber optic	No	...	No	No	Yes	Yes	Month-to-month	Yes	Electronic check	
	Yes	Fiber optic	No	...	Yes	Yes	Yes	Yes	One year	Yes	Electronic check	
	No	DSL	Yes	...	No	No	Yes	Yes	Month-to-month	Yes	Electronic check	
	No	Fiber optic	No	...	No	Yes	No	No	Month-to-month	No	Electronic check	
	...	...	...	...	...	...	...	...	...	...	...	
	No	Fiber optic	No	...	No	No	Yes	No	Month-to-month	Yes	Electronic check	

jupyter Churn_project Last Checkpoint: 11 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

```
[16]: #4.Extract all those customers whose tenure is greater than 70 months or their monthly charges is more than $100 and store the result in
#'customer_total_tenure'
customer_total_tenure=df[(df["tenure"]>70) | (df["MonthlyCharges"]>100)]
customer_total_tenure
```

[16]:

	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	Monthly
	Yes	Fiber optic	No	...	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	
	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	One year	No	Credit card (automatic)	
	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	Month-to-month	Yes	Bank transfer (automatic)	
	No	Fiber optic	Yes	...	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	
	Yes	Fiber optic	Yes	...	Yes	Yes	Yes	Yes	Two year	No	Credit card (automatic)	
	...	...	...	...	...	...	...	...	...	...	...	
	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	Month-to-month	Yes	Electronic check	

jupyter Churn_project Last Checkpoint: 11 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[18]:
```
x=df["tenure"]
x.sort_values().head(50)
```

[18]:
```
1340    0
936     0
6670    0
4380    0
5218    0
3826    0
753     0
6754    0
3331    0
488     0
1082    0
4275    1
6623    1
6620    1
1373    1
1371    1
4290    1
5724    1
1383    1
0       1
6633    1
6612    1
4242    1
6609    1
6607    1
4236    1
1428    1
```

jupyter Churn_project Last Checkpoint: 11 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[19]:
```
#5.Get the count of different levels from the 'Churn' column
df["Churn"].value_counts()
```

[19]:
```
Churn
No     5174
Yes    1869
Name: count, dtype: int64
```

[21]:
```
#2. Data Visualization:
#• Build a bar-plot for the 'InternetService' column:
#a. Set x-axis label to 'Categories of Internet Service'
#b. Set y-axis label to 'Count of Categories'
#c. Set the title of plot to be 'Distribution of Internet Service'
#d. Set the color of the bars to be 'orange'
df["InternetService"].value_counts()
```

[21]:
```
InternetService
Fiber optic    3096
DSL            2421
No             1526
Name: count, dtype: int64
```

[23]:
```
X=df["InternetService"].value_counts().keys().tolist()
X
```

[23]:
```
['Fiber optic', 'DSL', 'No']
```

jupyter Churn_project Last Checkpoint: 12 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

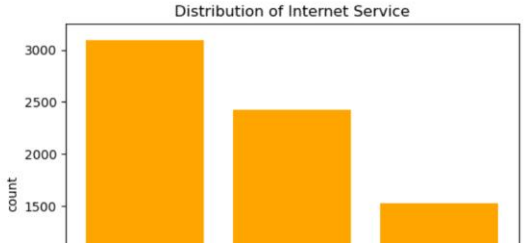
[23]:
```
['Fiber optic', 'DSL', 'No']
```

[24]:
```
Y=df["InternetService"].value_counts().tolist()
Y
```

[24]:
```
[3096, 2421, 1526]
```

[25]:
```
plt.bar(X,Y,color="ORANGE")
plt.title("Distribution of Internet Service")
plt.xlabel("Catogories")
plt.ylabel("count")
```

[25]:
```
Text(0, 0.5, 'count')
```



jupyter Churn_project Last Checkpoint: 12 minutes ago

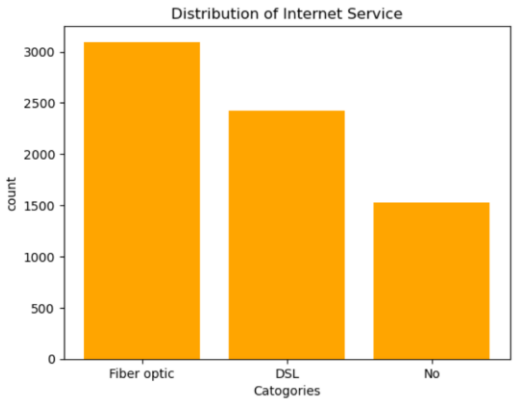
File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
[25]: Text(0, 0.5, 'count')
```




```
[28]: #Build a scatter-plot between 'MonthlyCharges' and 'tenure'. Map'MonthlyCharges' to the y-axis and 'tenure' to the 'x-axis':
      #a. Assign the points a color of 'brown'
```

jupyter Churn_project Last Checkpoint: 12 minutes ago

File Edit View Run Kernel Settings Help

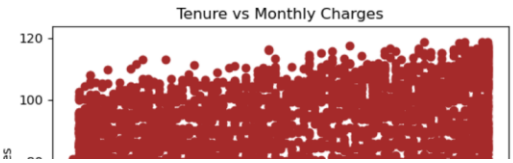
Trusted

Code

JupyterLab Python 3 (ipykernel)

```
[28]: #Build a scatter-plot between 'MonthlyCharges' and 'tenure'. Map'MonthlyCharges' to the y-axis and 'tenure' to the 'x-axis':
      #a. Assign the points a color of 'brown'
      #b. Set the x-axis label to 'Tenure of customer'
      #c. Set the y-axis label to 'Monthly Charges of customer'
      #d. Set the title to 'Tenure vs Monthly Charges'
      #e. Build a box-plot between 'tenure' & 'Contract'. Map 'tenure' on the
      #y-axis &
      #f. 'Contract' on the x-axis.
      plt.scatter(x=df["tenure"],y=df["MonthlyCharges"],color="brown")
      plt.title("Tenure vs Monthly Charges")
      plt.xlabel("tenure")
      plt.ylabel("MonthlyCharges")
```

```
[28]: Text(0, 0.5, 'MonthlyCharges')
```



jupyter Churn_project Last Checkpoint: 13 minutes ago

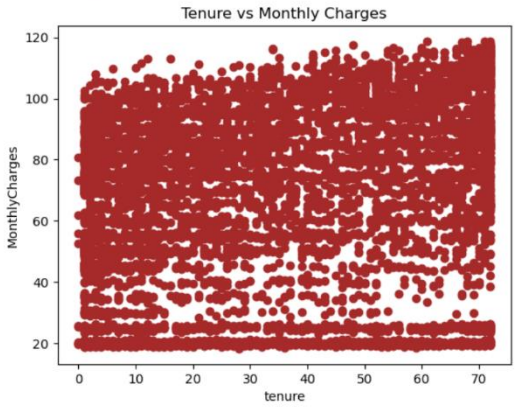
File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
[28]: Text(0, 0.5, 'MonthlyCharges')
```




```
[29]: plt.scatter(x=df["tenure"].head(35),y=df["MonthlyCharges"].head(35),color="brown")
```

jupyter Churn_project Last Checkpoint: 13 minutes ago

File Edit View Run Kernel Settings Help

Trusted

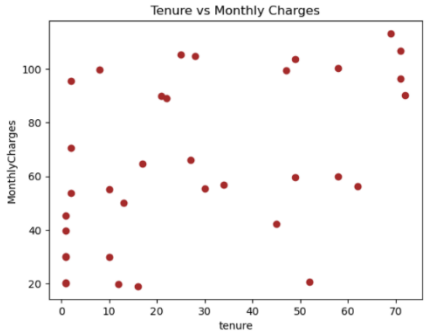
Code

JupyterLab Python 3 (ipykernel)

tenure

```
[29]: plt.scatter(x=df["tenure"].head(35),y=df["MonthlyCharges"].head(35),color="brown")
      plt.title("Tenure vs Monthly Charges")
      plt.xlabel("tenure")
      plt.ylabel("MonthlyCharges")
```

```
[29]: Text(0, 0.5, 'MonthlyCharges')
```




```
[30]: #• Build a histogram for the 'tenure' column:
```

jupyter Churn_project Last Checkpoint: 16 minutes ago

File Edit View Run Kernel Settings Help

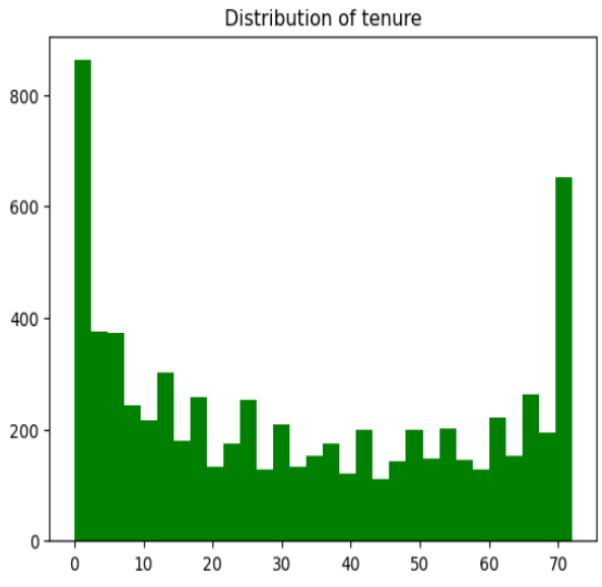
Trusted

Code

JupyterLab Python 3 (ipykernel)

```
[30]: #• Build a histogram for the 'tenure' column:
      #a. Set the number of bins to be 30
      #b. Set the color of the bins to be 'green'
      #c. Assign the title 'Distribution of tenure'
      plt.hist(df["tenure"],color="green",bins=30)
      plt.title("Distribution of tenure")
```

```
[30]: Text(0.5, 1.0, 'Distribution of tenure')
```



jupyter Churn_project Last Checkpoint: 16 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[32]:
```
#Linear Regression
#Build a simple linear model where dependent variable is 'MonthlyCharges'and independent variable is 'tenure':
#a. Divide the dataset into train and test sets in 70:30 ratio.
#b. Build the model on train set and predict the values on test set
#c. After predicting the values, find the root mean square error
#d. Find out the error in prediction & store the result in 'error'
#e. Find the root mean square error
x=df.loc[:,["tenure"]]
y=df.loc[:,["MonthlyCharges"]]
```

[33]:
```
x
```

[33]:

	tenure
0	1
1	34
2	2
3	45
4	2
...	...
7038	24
7039	72
7040	11
7041	4
7042	66

7043 rows × 1 columns

[34]:
```
y
```

[34]:

	MonthlyCharges
0	29.85
1	56.95

jupyter Churn_project Last Checkpoint: 17 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[34]:
```
y
```

[34]:

	MonthlyCharges
0	29.85
1	56.95
2	53.85
3	42.30
4	70.70
...	...
7038	84.80
7039	103.20
7040	29.60
7041	74.40
7042	105.65

7043 rows × 1 columns

[37]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import *
```

[38]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=3)
```

[53]:
```
Lin_reg=LinearRegression()
len(x_train)
```

[53]:
```
4930
```

[43]:
```
Lin_reg.fit(x_train,y_train)
```

[43]:
```
LinearRegression
```

jupyter Churn_project Last Checkpoint: 17 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[43]:
```
Lin_reg.fit(x_train,y_train)
```

[43]:
```
LinearRegression()
```

[46]:
```
y_pred=Lin_reg.predict(x_test)
y_pred
```

[46]:
```
array([[56.96721279],
       [61.09200198],
       [68.75232477],
       ...,
       [59.02960739],
       [76.11801976],
       [64.33290778]])
```

[49]:
```
mse=mean_squared_error(y_test,y_pred)
mse
```

[49]:
```
859.0613594181958
```

[56]:
```
rmse=np.sqrt(mse)
rmse
```

[56]:
```
29.309748538979242
```

[55]:
```
4930/859
```

[55]:
```
5.739231664726426
```

[57]:
```
#4. Logistic Regression:
#• Build a simple logistic regression model where dependent variable is‘Churn’ and independent variable is ‘MonthlyCharges’:
#a. Divide the dataset in 65:35 ratio
#b. Build the model on train set and predict the values on test set
#c. Build the confusion matrix and get the accuracy score
#d. Build a multiple logistic regression model where dependent variableis ‘Churn’ and independent variables are ‘tenure’ and‘MonthlyCharges’
#e. Divide the dataset in 80:20 ratio
#f. Build the model on train set and predict the values on test set
#g. Build the confusion matrix and get the accuracy score
df.columns
```

jupyter Churn_project Last Checkpoint: 17 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[57]:
```
#4. Logistic Regression:
#• Build a simple logistic regression model where dependent variable is‘Churn’ and independent variable is ‘MonthlyCharges’:
#a. Divide the dataset in 65:35 ratio
#b. Build the model on train set and predict the values on test set
#c. Build the confusion matrix and get the accuracy score
#d. Build a multiple logistic regression model where dependent variableis ‘Churn’ and independent variables are ‘tenure’ and‘MonthlyCharges’
#e. Divide the dataset in 80:20 ratio
#f. Build the model on train set and predict the values on test set
#g. Build the confusion matrix and get the accuracy score
df.columns
```

[57]:
```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

[59]:
```
x=df.loc[:,["MonthlyCharges"]]
y=df.loc[:,["Churn"]]
```

[65]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.35,random_state=3)
from sklearn.linear_model import LogisticRegression
```

[63]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import *
```

[61]:
```
logreg=LogisticRegression()
```

[66]:
```
logreg.fit(x_train,y_train)
```

```
C:\Users\Shanawaz khan\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

[66]:
```
LogisticRegression()
```

jupyter Churn_project Last Checkpoint: 18 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[66]:
```
LogisticRegression()
```

[67]:
```
y_pred=logreg.predict(x_test)
y_pred
```

[67]:
```
array(['No', 'No', 'No', ..., 'No', 'No', 'No'], dtype=object)
```

[68]:
```
from sklearn.metrics import accuracy_score,confusion_matrix
```

[69]:
```
confusion_matrix(y_test,y_pred)
```

[69]:
```
array([[1813,    0],
       [ 653,    0]], dtype=int64)
```

[70]:
```
accuracy_score(y_pred,y_test)
```

[70]:
```
0.735198702351987
```

[ ]:
```
#Decision Tree:
#● Build a decision tree model where dependent variable is 'Churn' and independent variable is 'tenure':
#a. Divide the dataset in 80:20 ratio
#b. Build the model on train set and predict the values on test set
#c. Build the confusion matrix and calculate the accuracy
x=df.loc[:,['MonthlyCharges","tenure"]]
y=df.loc[:,["Churn"]]
```

[71]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=3)
from sklearn.linear_model import LogisticRegression
```

[72]:
```
logreg=LogisticRegression()
```

[73]:
```
logreg.fit(x_train,y_train)
```

```
C:\Users\Shanawaz khan\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

[73]:
```
LogisticRegression()
```

jupyter Churn_project Last Checkpoint: 18 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[67]:
```
array(['No', 'No', 'No', ..., 'No', 'No', 'No'], dtype=object)
```

[68]:
```
from sklearn.metrics import accuracy_score,confusion_matrix
```

[69]:
```
confusion_matrix(y_test,y_pred)
```

[69]:
```
array([[1813,    0],
       [ 653,    0]], dtype=int64)
```

[70]:
```
accuracy_score(y_pred,y_test)
```

[70]:
```
0.735198702351987
```

[ ]:
```
#Decision Tree:
#● Build a decision tree model where dependent variable is 'Churn' and independent variable is 'tenure':
#a. Divide the dataset in 80:20 ratio
#b. Build the model on train set and predict the values on test set
#c. Build the confusion matrix and calculate the accuracy
x=df.loc[:,['MonthlyCharges","tenure"]]
y=df.loc[:,["Churn"]]
```

[71]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=3)
from sklearn.linear_model import LogisticRegression
```

[72]:
```
logreg=LogisticRegression()
```

[73]:
```
logreg.fit(x_train,y_train)
```

```
C:\Users\Shanawaz khan\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

[73]:
```
LogisticRegression()
```

[75]:
```
y_pred=logreg.predict(x_test)
```

[76]:
```
accuracy_score(y_test,y_pred)
```