Problem Statement:

You are the Data Scientist at a telecom company “Neo” whose customers are churning out to its competitors. You have to analyse the data of your company and find insights and stop your customers from churning out to other telecom companies.

Lab Environment: Anaconda Domain – Telecom

Tasks to be done: A) Data Manipulation: a. Extract the 5th column & store it in ‘customer_5’ b. Extract the 15th column & store it in ‘customer_15’ c. Extract all the male senior citizens whose Payment Method is Electronic check & store the result in ‘senior_male_electronic’ d. Extract all those customers whose tenure is greater than 70 months or their Monthly charges is more than 100$ & store the result in ‘customer_total_tenure’ e. Extract all the customers whose Contract is of two years, payment method is Mailed check & the value of Churn is ‘Yes’ & store the result in ‘two_mail_yes’ f. Extract 333 random records from the customer_churndataframe& store the result in ‘customer_333’ g. Get the count of different levels from the ‘Churn’ column

In [106…
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection  import train_test_split
```

In [107…
```
data = pd.read_csv(r'C:\Users\LENOVO\Downloads\datasets for python\Customer_churn.csv')
data.head()
```

Out[107]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No	No	One year	No	Mailed check	56.95	1889.5	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes

5 rows × 21 columns

In [108…
```
data1 = data.loc[:,'tenure']
data1
```

Out[108]:
```
0        1
1       34
2        2
3       45
4        2
        ..
7038    24
7039    72
7040    11
7041     4
7042    66
Name: tenure, Length: 7043, dtype: int64
```

In [109…
```
data2 = data.iloc[:,14]
data2.head()
```

Out[109]:
```
0    No
1    No
2    No
3    No
4    No
Name: StreamingMovies, dtype: object
```

In [110…
```
senior_male_electronic=data[(data['gender']=="Male") & (data['SeniorCitizen']== 1) & (data['PaymentMethod'] == "Electronic check")]

senior_male_electronic.head()
```

Out[110]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
20	8779-QRDMV	Male	1	No	No	1	No	No phone service	DSL	No	...	Yes	No	No	Yes	Month-to-month	Yes	Electronic check	39.65	39.65	Yes
55	1658-BYGOY	Male	1	No	No	18	Yes	Yes	Fiber optic	No	...	No	No	Yes	Yes	Month-to-month	Yes	Electronic check	95.45	1752.55	Yes
57	5067-XJQFU	Male	1	Yes	Yes	66	Yes	Yes	Fiber optic	No	...	Yes	Yes	Yes	Yes	One year	Yes	Electronic check	108.45	7076.35	No
78	0191-ZHSKZ	Male	1	No	No	30	Yes	No	DSL	Yes	...	No	No	Yes	Yes	Month-to-month	Yes	Electronic check	74.75	2111.3	No
91	2424-WVHPL	Male	1	No	No	1	Yes	No	Fiber optic	No	...	No	Yes	No	No	Month-to-month	No	Electronic check	74.70	74.7	No

5 rows × 21 columns

In [111…
```
customer_total_tenure= data[(data['tenure']>70) | (data['MonthlyCharges']>100)]
customer_total_tenure.head()
```

Out[111]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
8	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	Fiber optic	No	...	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	104.80	3046.05	Yes
12	8091-TTVAX	Male	0	Yes	No	58	Yes	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	One year	No	Credit card (automatic)	100.35	5681.1	No
13	0280-XJGEX	Male	0	No	No	49	Yes	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	Month-to-month	Yes	Bank transfer (automatic)	103.70	5036.3	Yes
14	5129-JLPIS	Male	0	No	No	25	Yes	No	Fiber optic	Yes	...	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	105.50	2686.05	No
15	3655-SNQYZ	Female	0	Yes	Yes	69	Yes	Yes	Fiber optic	Yes	...	Yes	Yes	Yes	Yes	Two year	No	Credit card (automatic)	113.25	7895.15	No

5 rows × 21 columns

In [112…
```
two_mail_yes= data[(data['Contract']=="Month-to-month") & (data['PaymentMethod']=="Mailed check") & (data['Churn']=="Yes")]
two_mail_yes.head()
```

Out[112]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
22	1066-JKSGK	Male	0	No	No	1	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service	Month-to-month	No	Mailed check	20.15	20.15	Yes
29	8773-HHUOZ	Female	0	No	Yes	17	Yes	No	DSL	No	...	No	No	Yes	Yes	Month-to-month	Yes	Mailed check	64.70	1093.1	Yes
97	0278-YXOOG	Male	0	No	No	5	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service	Month-to-month	No	Mailed check	21.05	113.85	Yes
139	0390-DCFDQ	Female	1	Yes	No	1	Yes	No	Fiber optic	No	...	No	No	No	No	Month-to-month	Yes	Mailed check	70.45	70.45	Yes

5 rows × 21 columns

In [113…
```
customer_333 = data.sample(n=333)
customer_333.head()
```

Out[113]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
5548	4589-IUAJB	Male	0	Yes	No	70	Yes	Yes	No	No internet service	...	No internet service	No internet service	No internet service	No internet service	Two year	No	Bank transfer (automatic)	24.25	1724.15	No
6699	3662-FXJFO	Female	0	No	No	54	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service	Two year	Yes	Mailed check	20.50	1035.7	No
2538	7758-UJWYS	Male	0	Yes	Yes	34	No	No phone service	DSL	No	...	No	Yes	Yes	No	Two year	Yes	Electronic check	40.55	1325.85	No
6075	5081-NWSUP	Female	0	No	No	10	Yes	No	DSL	No	...	No	Yes	No	Yes	One year	No	Mailed check	64.90	685.55	No
5423	5228-EXCET	Male	0	No	No	13	Yes	No	Fiber optic	No	...	No	No	No	Yes	Month-to-month	Yes	Electronic check	80.85	1008.7	Yes

5 rows × 21 columns

In [114…
```
data['Churn'].value_counts()
```

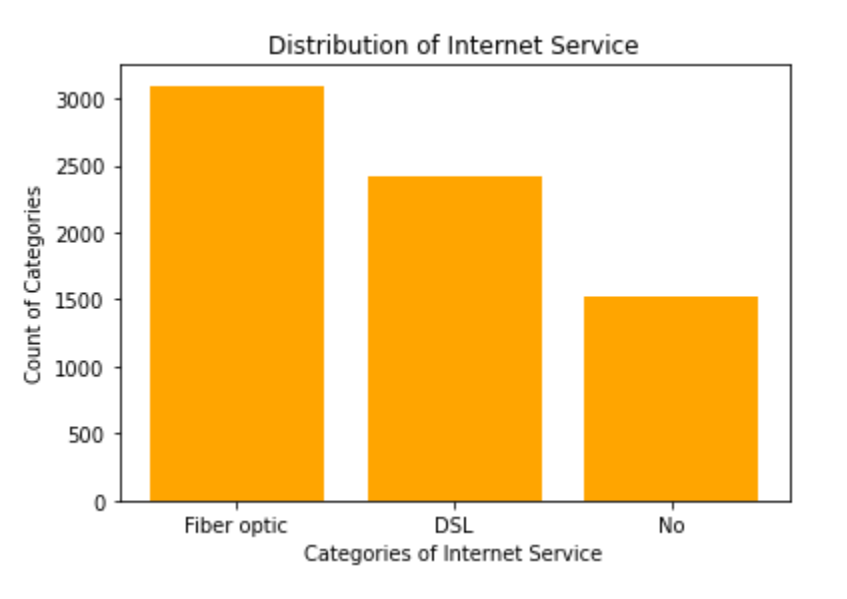
Out[114]:
```
No     5174
Yes    1869
Name: Churn, dtype: int64
```

B) Data Visualization:

a. Build a bar-plot for the ‘InternetService’ column: i. Set x-axis label to ‘Categories of Internet Service’ ii. Set y-axis label to ‘Count of Categories’ iii. Set the title of plot to be ‘Distribution of Internet Service’ iv. Set the color of the bars to be ‘orange’ b. Build a histogram for the ‘tenure’ column: i. Set the number of bins to be 30 ii. Set the color of the bins to be ‘green’ iii. Assign the title ‘Distribution of tenure’ c. Build a scatter-plot between ‘MonthlyCharges’ & ‘tenure’. Map ‘MonthlyCharges’ to the y-axis & ‘tenure’ to the ‘x-axis’: i. Assign the points a color of ‘brown’ ii. Set the x-axis label to ‘Tenure of customer’ iii. Set the y-axis label to ‘Monthly Charges of customer’ iv. Set the title to ‘Tenure vs Monthly Charges’ d. Build a box-plot between ‘tenure’ & ‘Contract’. Map ‘tenure’ on the y-axis & ‘Contract’ on the x-axis.

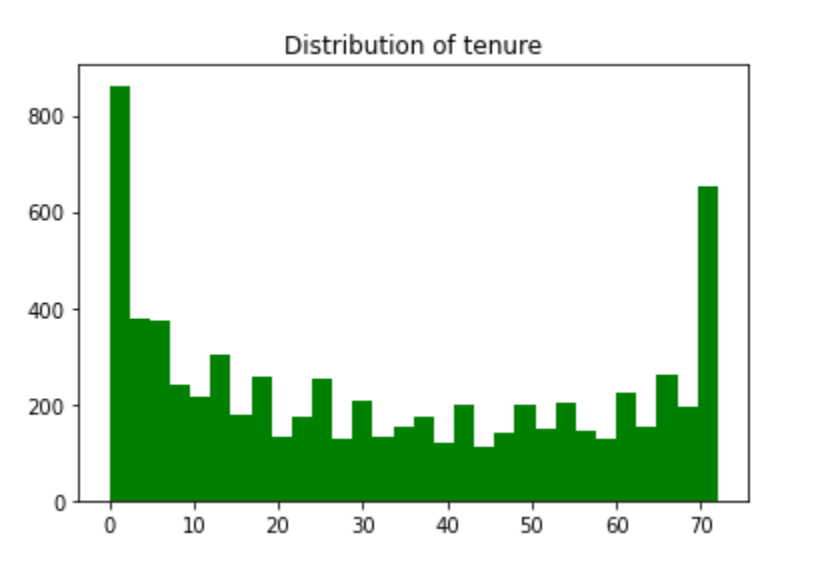
In [115…
```
plt.bar(data['InternetService'].value_counts().keys().tolist(),data['InternetService'].value_counts().tolist(),color="orange")
plt.xlabel("Categories of Internet Service")
plt.ylabel("Count of Categories")
plt.title("Distribution of Internet Service")
```

Out[115]:
```
Text(0.5, 1.0, 'Distribution of Internet Service')
```



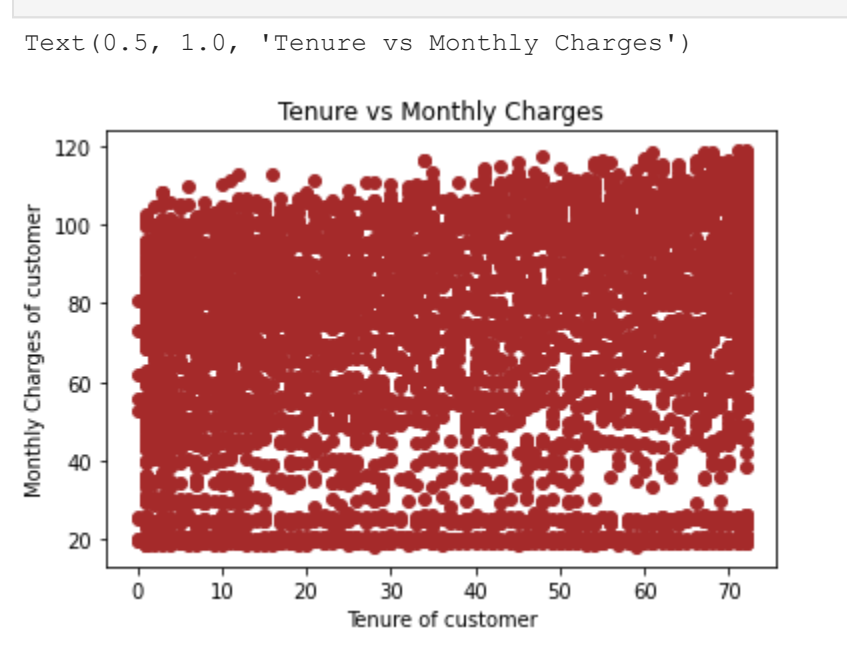
In [116…
```
plt.hist(data['tenure'],bins=30,color="green")
plt.title("Distribution of tenure")
```

Out[116]:
```
Text(0.5, 1.0, 'Distribution of tenure')
```



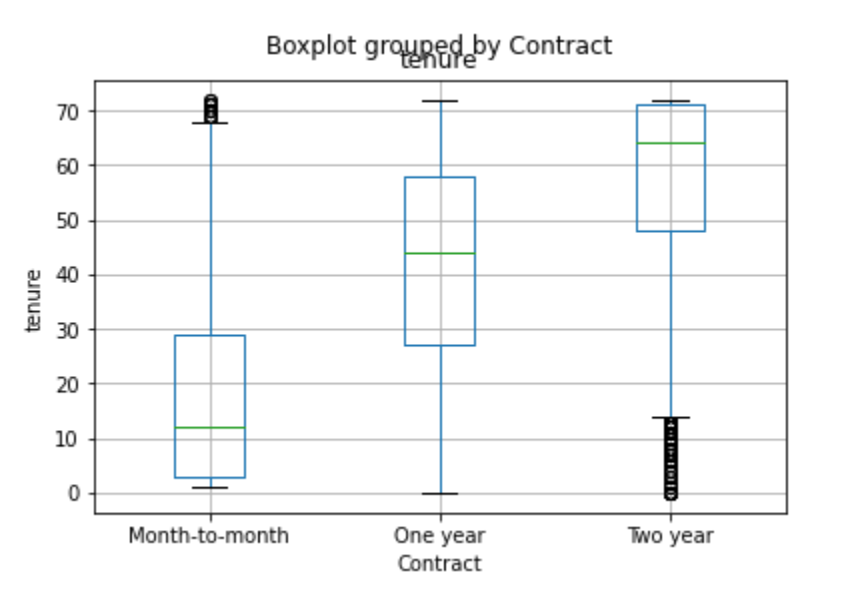
In [117…
```
plt.scatter(x=data['tenure'], y=data['MonthlyCharges'],color="brown")
plt.xlabel('Tenure of customer')
plt.ylabel('Monthly Charges of customer')
plt.title('Tenure vs Monthly Charges')
```

Out[117]:
```
Text(0.5, 1.0, 'Tenure vs Monthly Charges')
```



In [118…
```
data.boxplot(column=['tenure'], by= ['Contract'])
plt.xlabel('Contract')
plt.ylabel('tenure')
```

Out[118]:
```
Text(0, 0.5, 'tenure')
```



C) Linear Regression:

a. Build a simple linear model where dependent variable is ‘MonthlyCharges’ and independent variable is ‘tenure’ i. Divide the dataset into train and test sets in 70:30 ratio. ii. Build the model on train set and predict the values on test set iii. After predicting the values, find the root mean square error iv. Find out the error in prediction & store the result in ‘error’ v. Find the root mean square error

In [119…
```
from sklearn.linear_model import LogisticRegression
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.model_selection  import train_test_split
```

In [120…
```
x= data[['MonthlyCharges']]
y= data[['tenure']]
x
y
```

Out[120]:

	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

In [121…
```
x_train,x_test,y_train, y_test =train_test_split(x,y,test_size=0.30,random_state=0)
```

In [122…
```
x_train.shape,y_train.shape,x_test.shape,y_test.shape
```

Out[122]:
```
((4930, 1), (4930, 1), (2113, 1), (2113, 1))
```

In [123…
```
regression= LinearRegression()

regression.fit(x_train,y_train)
```

Out[123]:
```
LinearRegression()
```

In [124…
```
y_pridict= regression.predict(x_test)
```

In [125…
```
from sklearn.metrics import mean_squared_error

np.sqrt(mean_squared_error(y_test,y_pridict))
```

Out[125]:
```
23.816464447907975
```

D) Logistic Regression:

a. Build a simple logistic regression modelwhere dependent variable is ‘Churn’ & independent variable is ‘MonthlyCharges’ i. Divide the dataset in 65:35 ratio ii. Build the model on train set and predict the values on test set iii. Build the confusion matrix and get the accuracy score b. Build a multiple logistic regression model where dependent variable is ‘Churn’ & independent variables are ‘tenure’ & ‘MonthlyCharges’ i. Divide the dataset in 80:20 ratio ii. Build the model on train set and predict the values on test set iii. Build the confusion matrix and get the accuracy score

In [126…
```
x=data[['MonthlyCharges']]
y=data[['Churn']]
x
y
```

Out[126]:

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

7043 rows × 1 columns

In [127…
```
x_train,x_test,y_train, y_test =train_test_split(x,y,test_size=0.35,random_state=0)
```

In [128…
```
print(x_train.shape , x_test.shape ,y_train.shape, y_test.shape)
```
```
(4577, 1) (2466, 1) (4577, 1) (2466, 1)
```

In [129…
```
from sklearn.linear_model import LogisticRegression
```

In [130…
```
log = LogisticRegression()
```

In [131…
```
log.fit(x_train, y_train)
```
```
C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\utils\validation.py:993: 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)
```

Out[131]:
```
LogisticRegression()
```

In [132…
```
y_pred  = log.predict(x_test)
y_pred
```

Out[132]:
```
array(['No', 'No', 'No', ..., 'No', 'No', 'No'], dtype=object)
```

In [133…
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [134…
```
confusion_matrix(y_test, y_pred), accuracy_score(y_test, y_pred)
```

Out[134]:
```
(array([[1815,    0],
        [ 651,    0]], dtype=int64),
 0.7360097323600974)
```

In [135…
```
(1815)/(1815+651)
```

Out[135]:
```
0.7360097323600974
```

In [136…
```
x=data[['MonthlyCharges','tenure']]
y=data[['Churn']]
x
y
```

Out[136]:

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

7043 rows × 1 columns

In [137…
```
x_train,x_test,y_train, y_test =train_test_split(x,y,test_size=0.20,random_state=0)
```

In [138…
```
print(x_train.shape , x_test.shape ,y_train.shape, y_test.shape)
```
```
(5634, 2) (1409, 2) (5634, 1) (1409, 1)
```

In [139…
```
from sklearn.linear_model import LogisticRegression
```

In [140…
```
log = LogisticRegression()

log.fit(x_train, y_train)
```
```
C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\utils\validation.py:993: 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)
```

Out[140]:
```
LogisticRegression()
```

In [141…
```
y_pred  = log.predict(x_test)
y_pred
```

Out[141]:
```
array(['No', 'No', 'No', ..., 'No', 'No', 'No'], dtype=object)
```

In [142…
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [143…
```
confusion_matrix(y_test, y_pred), accuracy_score(y_test, y_pred)
```

Out[143]:
```
(array([[934, 107],
        [212, 156]], dtype=int64),
 0.7735982966643009)
```

In [144…
```
(934+156)/(934+212+156+107)
```

Out[144]:
```
0.7735982966643009
```

E) Decision Tree:

a. Build a decision tree model where dependent variable is ‘Churn’ & independent variable is ‘tenure’ i. Divide the dataset in 80:20 ratio ii. Build the model on train set and predict the values on test set iii. Build the confusion matrix and calculate the accuracy

In [145…
```
x=data[['tenure']]
y=data[['Churn']]
x
y
```

Out[145]:

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

7043 rows × 1 columns

In [146…
```
from sklearn.tree import DecisionTreeClassifier

x_train, x_test, y_train, y_test =train_test_split(x,y,test_size=0.20,random_state=0)
```

In [147…
```
my_tree = DecisionTreeClassifier()
my_tree.fit(x_train,y_train)
```

Out[147]:
```
DecisionTreeClassifier()
```

In [148…
```
y_pred= my_tree.predict(x_test)
```

In [149…
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [150…
```
confusion_matrix(y_test, y_pred), accuracy_score(y_test, y_pred)
```

Out[150]:
```
(array([[965,  76],
        [281,  87]], dtype=int64),
 0.7466288147622427)
```

In [151…

```
(831+153)/(831+153+210+215)
```

Out[151]:

```
0.6983676366217175
```