In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


df=pd.read_csv('C:\\Users\\Vishal\\Downloads\\All Codes\\Classification\\WA_Fr
```

Decision Tree Model

import numpy as np import pandas as pd import matplotlib.pyplot as plt import seaborn as sns

In [2]:
```
df
```

Out[2]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLir
0	7590-VHVEG	Female	0	Yes	No	1	No	No phc serv
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	No phc serv
4	9237-HQITU	Female	0	No	No	2	Yes	
...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	`
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	`
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phc serv
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	`
7042	3186-AJIEK	Male	0	No	No	66	Yes	

7043 rows × 21 columns

In [3]:
```
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 
 20  Churn             7043 non-null   object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

In [4]:
```
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'],errors='coerce')
```

In [5]:
```
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      7032 non-null   float64
 20  Churn             7043 non-null   object 
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```

In [6]:
```
df.dropna(how='any',inplace=True)
```

In [7]:
```
df.head()
```

Out[7]:

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

5 rows × 21 columns

In [8]:
```
x = df.drop(['customerID','Churn'], axis=1)
y = df.Churn.values
```

In [9]:
```
x
```

Out[9]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetS
0	Female	0	Yes	No	1	No	No phone service	
1	Male	0	No	No	34	Yes	No	
2	Male	0	No	No	2	Yes	No	
3	Male	0	No	No	45	No	No phone service	
4	Female	0	No	No	2	Yes	No	Fib
...	...	...	...	...	...	...	...	
7038	Male	0	Yes	Yes	24	Yes	Yes	
7039	Female	0	Yes	Yes	72	Yes	Yes	Fib
7040	Female	0	Yes	Yes	11	No	No phone service	
7041	Male	1	Yes	No	4	Yes	Yes	Fib
7042	Male	0	No	No	66	Yes	No	Fib

7032 rows × 19 columns

In [10]:
```
y
```

Out[10]:
```
array(['No', 'No', 'Yes', ..., 'No', 'Yes', 'No'], dtype=object)
```

In [11]:
```
x = pd.get_dummies( x,columns=['gender','Partner', 'Dependents',
       'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'Contract', 'PaperlessBilling', 'PaymentMethod'],
    drop_first=True
)
```

In [12]:
```
x.head()
```

Out[12]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Male	Partner_Yes	Dependents_
0	0	1	29.85	29.85	False	True	F
1	0	34	56.95	1889.50	True	False	F
2	0	2	53.85	108.15	True	False	F
3	0	45	42.30	1840.75	True	False	F
4	0	2	70.70	151.65	False	False	F

5 rows × 30 columns

In [13]:
```
x = x.astype(int)
```

In [14]:
```
x
```

Out[14]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Male	Partner_Yes	Depender
0	0	1	29	29	0	1	
1	0	34	56	1889	1	0	
2	0	2	53	108	1	0	
3	0	45	42	1840	1	0	
4	0	2	70	151	0	0	
...	...	...	...	...	...	...	
7038	0	24	84	1990	1	1	
7039	0	72	103	7362	0	1	
7040	0	11	29	346	0	1	
7041	1	4	74	306	1	1	
7042	0	66	105	6844	1	0	

7032 rows × 30 columns

In [15]:
```
from sklearn.model_selection import train_test_split


x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
```

In [16]:
```
len(x_train)
```

Out[16]: 5625

In [17]:
```
len(x_test)
```

Out[17]:
```
1407
```

In [18]:
```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

In [19]:
```
x_train
```

Out[19]:
```
array([[-0.43874278, -0.99570377,  0.35323805, ..., -0.51744418,
         1.39574826, -0.54331175],
       [-0.43874278, -0.7923329 ,  0.78437207, ..., -0.51744418,
         1.39574826, -0.54331175],
       [-0.43874278, -1.28042299, -0.67485077, ..., -0.51744418,
        -0.71646158,  1.84056391],
       ...,
       [-0.43874278,  1.32272413,  0.65171545, ..., -0.51744418,
        -0.71646158, -0.54331175],
       [-0.43874278,  0.38721813, -0.80750739, ..., -0.51744418,
        -0.71646158, -0.54331175],
       [ 2.27923977,  1.40407248,  1.34816272, ..., -0.51744418,
         1.39574826, -0.54331175]])
```

In [20]:
```
x_test
```

Out[20]:
```
array([[ 2.27923977, -1.11772629,  0.65171545, ..., -0.51744418,
        -0.71646158,  1.84056391],
       [-0.43874278,  1.56676917,  0.51905883, ...,  1.93257561,
        -0.71646158, -0.54331175],
       [-0.43874278,  1.44474665,  1.41449103, ..., -0.51744418,
         1.39574826, -0.54331175],
       ...,
       [-0.43874278, -1.11772629, -0.5753583 , ..., -0.51744418,
         1.39574826, -0.54331175],
       [-0.43874278, -0.34491699,  0.95019285, ..., -0.51744418,
         1.39574826, -0.54331175],
       [-0.43874278,  1.40407248,  0.98335701, ...,  1.93257561,
        -0.71646158, -0.54331175]])
```

In [44]:
```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(max_depth=2) # max_depth for visualization

model.fit(x_train,y_train)
```

Out[44]:

▾ DecisionTreeClassifier (https://scikit-learn.org/1.7/modules/generated/sklearn.tree.DecisionTreeC

Parameters

In [45]:
```
y_pred = model.predict(x_test)
```

In [46]:
```
y_pred
```

Out[46]:
```
array(['Yes', 'No', 'No', ..., 'No', 'No', 'No'], dtype=object)
```

In [47]:
```
data = [[0,2,87,180,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0]]
```

In [48]:
```
new_data = sc.transform(data)

single = model.predict(new_data)

print(single)
```

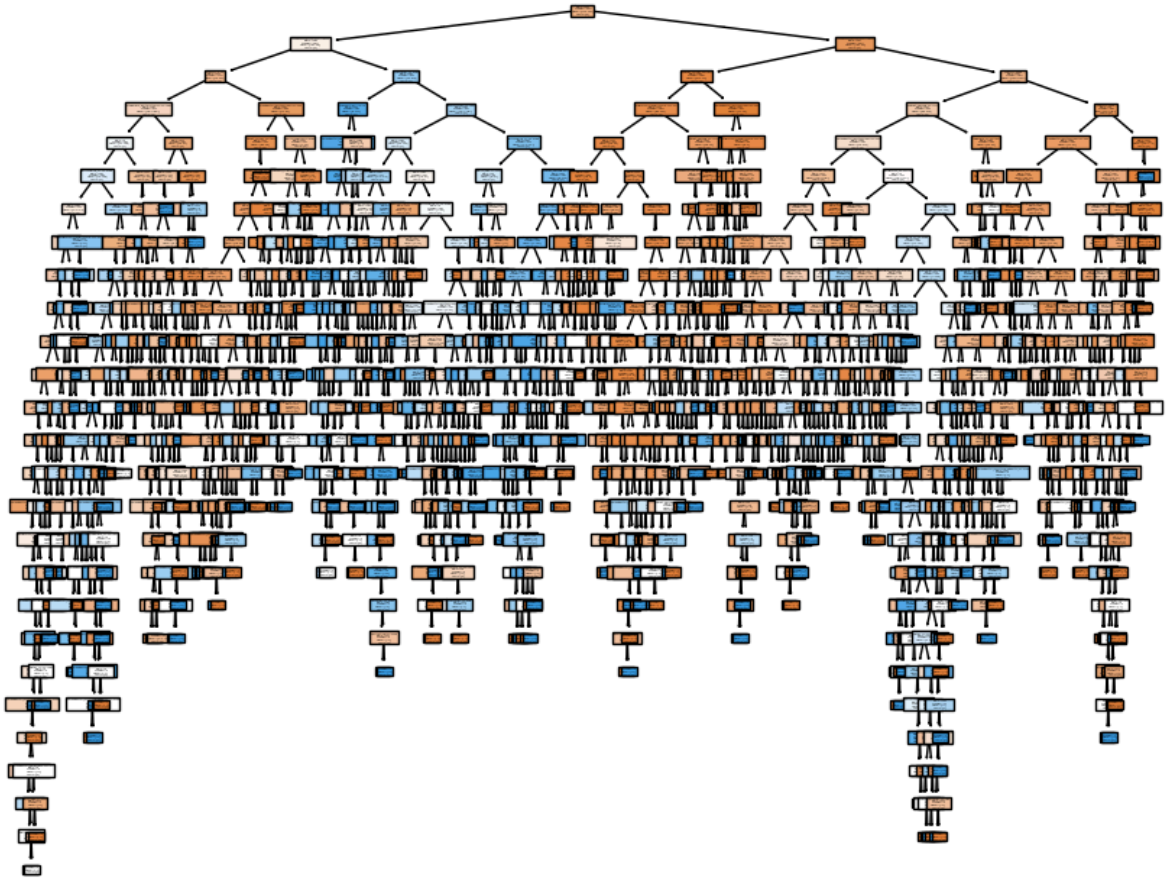
```
['No']

C:\Users\Vishal\anaconda3\Lib\site-packages\sklearn\utils\validation.py:274
9: UserWarning: X does not have valid feature names, but StandardScaler was
fitted with feature names
  warnings.warn(
```

In [49]:
```
from sklearn.metrics import accuracy_score

print(accuracy_score(y_test,y_pred)*100)
```

```
77.96730632551528
```

Visualization of decision Tree

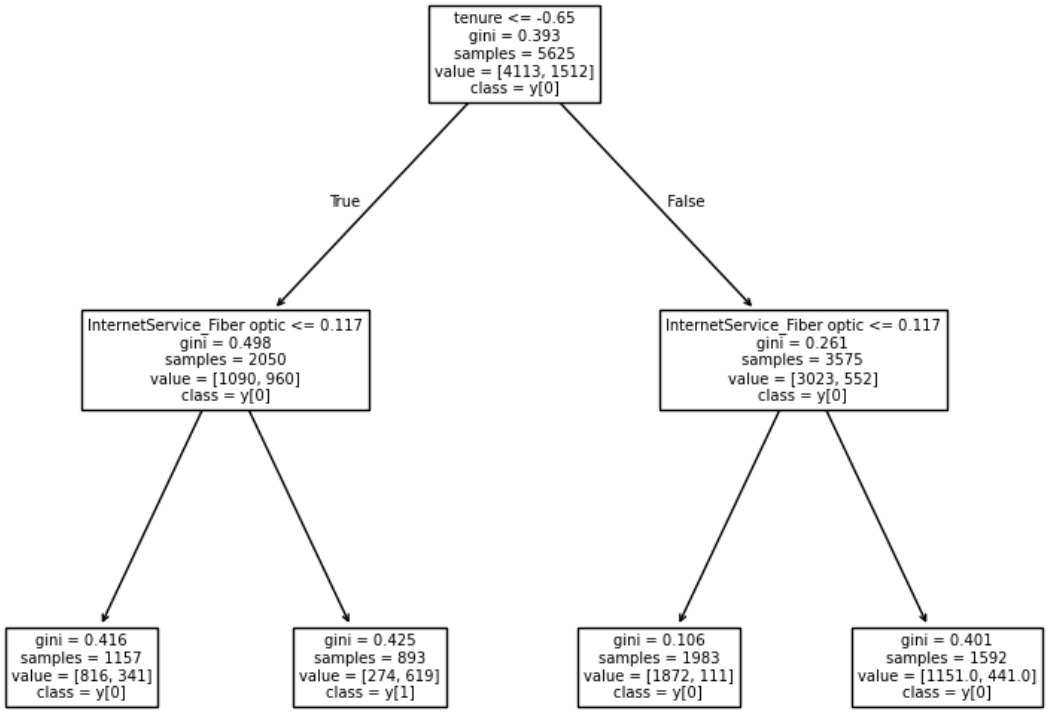
In [39]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(10,8))
plot_tree(model,filled=True,feature_names=x.columns,class_names=True)
plt.show()
```



lets do for max_depth = 2

change in above code at DecisionTreeClassifier(max_depth=2)

In [51]:
```
plt.figure(figsize=(10,8))
plot_tree(model,feature_names=x.columns,class_names=True)
plt.show()
```



In [ ]:
```
```