Preprocessing Training Data

```
import numpy as np
import pandas as pd
import seaborn as sns
df=pd.read_csv('/content/churn-bigml-80.csv')
df
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	T min
0	KS	128	415	No	Yes	25	265.1	110	45.07	1
1	OH	107	415	No	Yes	26	161.6	123	27.47	1
2	NJ	137	415	No	No	0	243.4	114	41.38	1
3	OH	84	408	Yes	No	0	299.4	71	50.90	
4	OK	75	415	Yes	No	0	166.7	113	28.34	1
...	...	...	...	...	...	...	...	...	...	
2661	SC	79	415	No	No	0	134.7	98	22.90	1
2662	AZ	192	415	No	Yes	36	156.2	77	26.55	2
2663	WV	68	415	No	No	0	231.1	57	39.29	1
2664	RI	28	510	No	No	0	180.8	109	30.74	2
2665	TN	74	415	No	Yes	25	234.4	113	39.85	2

2666 rows × 20 columns

```
#First 5 observation print
df.head()
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Tota ev minute
0	KS	128	415	No	Yes	25	265.1	110	45.07	197.
1	OH	107	415	No	Yes	26	161.6	123	27.47	195.
2	NJ	137	415	No	No	0	243.4	114	41.38	121.
3	OH	84	408	Yes	No	0	299.4	71	50.90	61.

```
#Last 5 observation print
df.tail()
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	T min
2661	SC	79	415	No	No	0	134.7	98	22.90	1
2662	AZ	192	415	No	Yes	36	156.2	77	26.55	2
2663	WV	68	415	No	No	0	231.1	57	39.29	1
2664	RI	28	510	No	No	0	180.8	109	30.74	2
2665	TN	74	415	No	Yes	25	234.4	113	39.85	2

```
#Column heading print
df.columns
```

```
Index(['State', 'Account length', 'Area code', 'International plan',
       'Voice mail plan', 'Number vmail messages', 'Total day minutes',
       'Total day calls', 'Total day charge', 'Total eve minutes',
       'Total eve calls', 'Total eve charge', 'Total night minutes',
       'Total night calls', 'Total night charge', 'Total intl minutes',
       'Total intl calls', 'Total intl charge', 'Customer service calls',
       'Churn'],
      dtype='object')
```

```
#Each column types
df.dtypes
```

```
State                      object
Account length              int64
Area code                   int64
International plan         object
Voice mail plan            object
Number vmail messages       int64
Total day minutes         float64
Total day calls             int64
Total day charge          float64
Total eve minutes         float64
Total eve calls             int64
Total eve charge          float64
Total night minutes       float64
Total night calls           int64
Total night charge        float64
Total intl minutes        float64
Total intl calls            int64
Total intl charge         float64
Customer service calls      int64
Churn                        bool
dtype: object
```

```
#To find Missing values
df.isna().sum()
```

```
State                     0
Account length            0
Area code                 0
International plan        0
Voice mail plan           0
Number vmail messages     0
Total day minutes         0
Total day calls           0
Total day charge          0
Total eve minutes         0
Total eve calls           0
Total eve charge          0
Total night minutes       0
Total night calls         0
Total night charge        0
Total intl minutes        0
Total intl calls          0
Total intl charge         0
Customer service calls    0
Churn                     0
dtype: int64
```

Finding Count Of Each String Value Columns and plotting Graphs

```
#state counts
df['State'].value_counts()
```

```
WV    88
MN    70
NY    68
VA    67
AL    66
OH    66
WY    66
OR    62
NV    61
WI    61
MD    60
UT    60
CO    59
CT    59
MI    58
VT    57
ID    56
NC    56
TX    55
FL    54
IN    54
MT    53
OK    52
MA    52
KS    52
MO    51
DE    51
NJ    50
SC    49
SD    49
ME    49
GA    49
RI    48
MS    48
WA    48
AR    47
IL    45
DC    45
AZ    45
NE    45
HI    44
```

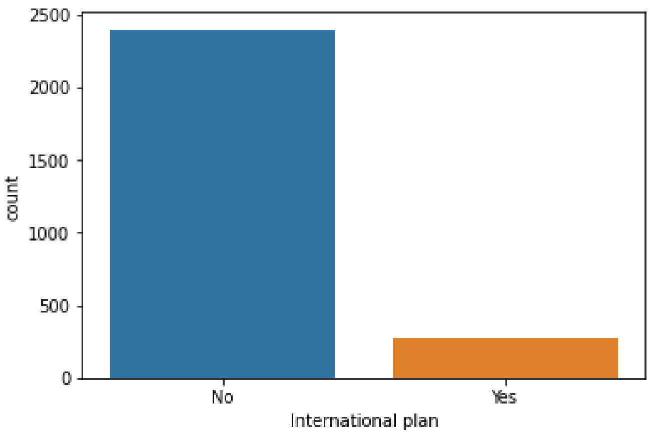
```
NM    44
ND    44
AK    43
KY    43
NH    43
TN    41
IA    38
PA    36
LA    35
CA    24
Name: State, dtype: int64
```

```
#International plan count
df['International plan'].value_counts()
```

```
No     2396
Yes     270
Name: International plan, dtype: int64
```

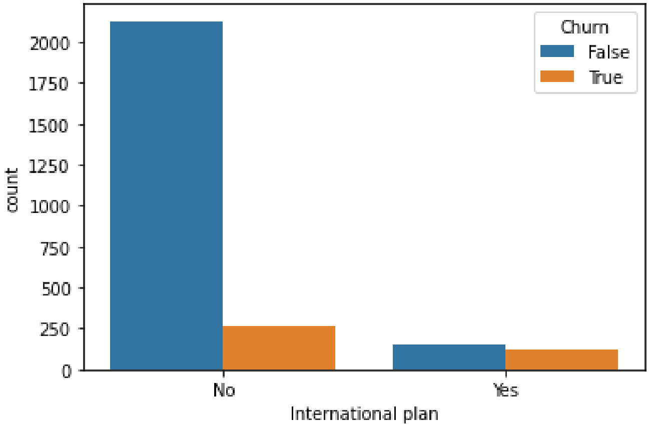
```
#International plan count graph
sns.countplot('International plan',data=df)
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='International plan', ylabel='count'>
```



```
#How International plan affect Churn(Output_Variables) graph
sns.countplot('International plan',data=df,hue='Churn')
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='International plan', ylabel='count'>
```



```
#voice mail plan count
df['Voice mail plan'].value_counts()
```

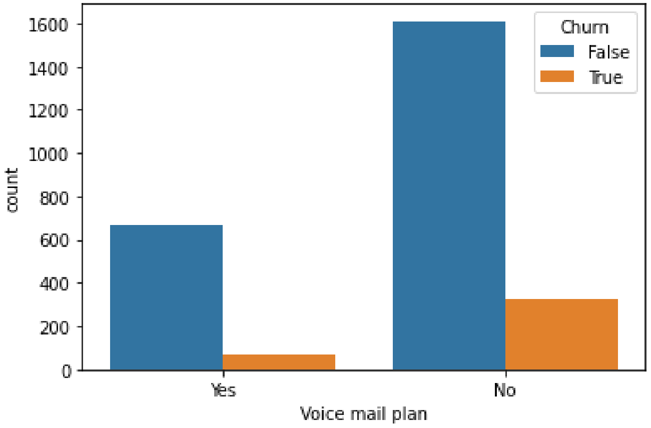
```
No     1933
Yes     733
Name: Voice mail plan, dtype: int64
```

```
#voice mail plan count graph
sns.countplot('Voice mail plan',data=df)
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass 
  warnings.warn(
<AxesSubplot:xlabel='Voice mail plan', ylabel='count'>
```

```
#How Voice mail plan affect Churn(Output_Variables) graph
sns.countplot('Voice mail plan',data=df,hue='Churn')
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass 
  warnings.warn(
<AxesSubplot:xlabel='Voice mail plan', ylabel='count'>
```

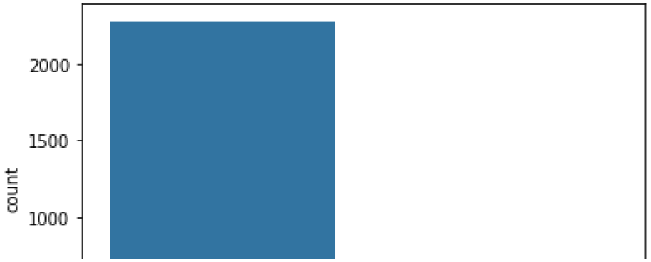


```
#Churn count
df['Churn'].value_counts()
```

```
False    2278
True      388
Name: Churn, dtype: int64
```

```
#Churn counts graph
sns.countplot('Churn',data=df)
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass 
  warnings.warn(
<AxesSubplot:xlabel='Churn', ylabel='count'>
```



Correlation

```
df.corr()
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	To ca
Account length	1.000000	-0.008620	-0.002996	0.002847	0.038862	0.002843	-0.015923	0.018
Area code	-0.008620	1.000000	-0.000584	-0.023134	-0.009629	-0.023130	0.000679	-0.018
Number vmail messages	-0.002996	-0.000584	1.000000	0.019027	-0.009622	0.019027	0.011401	0.005
Total day minutes	0.002847	-0.023134	0.019027	1.000000	0.016780	1.000000	0.003999	0.009
Total day calls	0.038862	-0.009629	-0.009622	0.016780	1.000000	0.016787	-0.026003	0.006
Total day charge	0.002843	-0.023130	0.019027	1.000000	0.016787	1.000000	0.004008	0.009
Total eve minutes	-0.015923	0.000679	0.011401	0.003999	-0.026003	0.004008	1.000000	-0.007
Total eve calls	0.018552	-0.018602	0.005131	0.009059	0.006473	0.009056	-0.007654	1.000

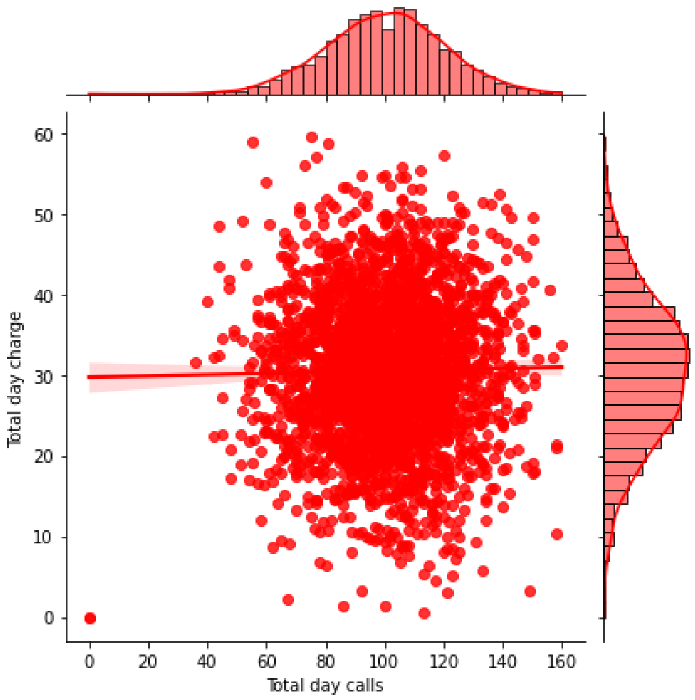
```
#HeatMap-correlation showing
sns.heatmap(df.corr())
```

```
<AxesSubplot:>
```

JOINTPLOTGRAPH

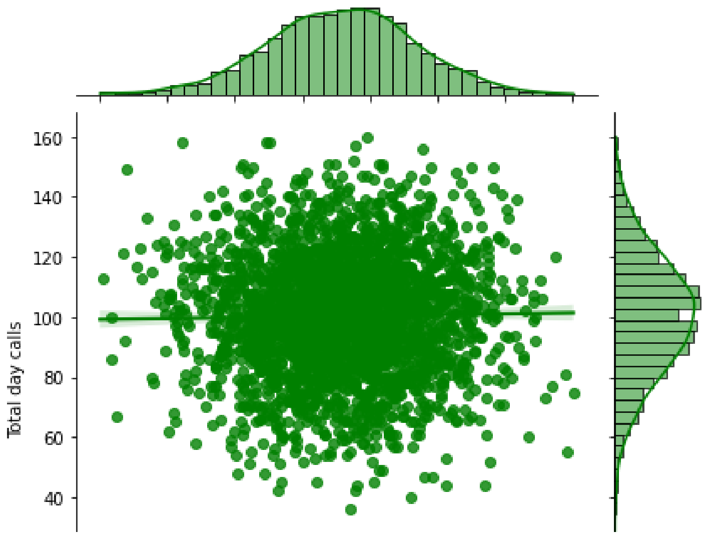
```
#How Total day calls affect Total day charge
sns.jointplot(x='Total day calls',y='Total day charge',data=df,kind='reg',color='red')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76183a40d0>
```



```
#How Total day mins affect Total day calls
sns.jointplot(x='Total day minutes',y='Total day calls',data=df,kind='reg',color='green')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76181451c0>
```

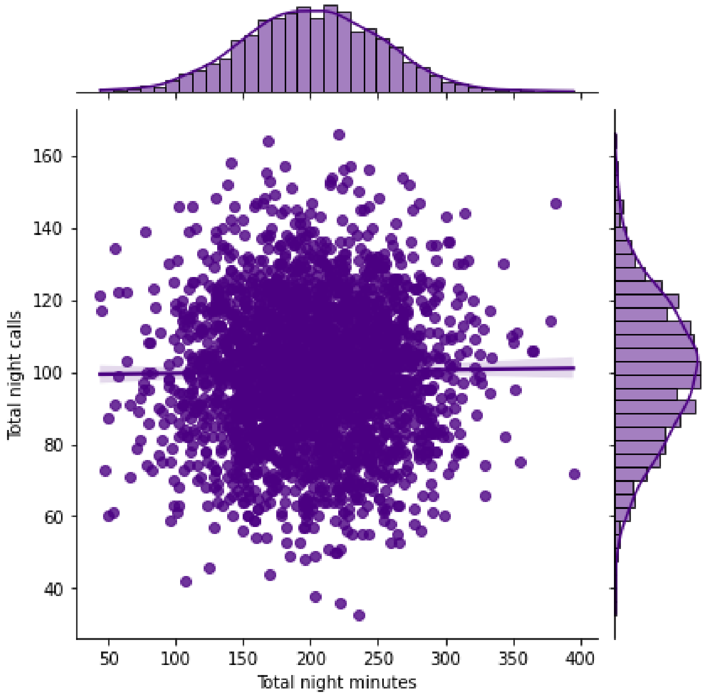


```
#How Total night calls affect Total night charge
sns.jointplot(x='Total night calls',y='Total night charge',data=df,kind='reg',color='black')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76186aec10>
```

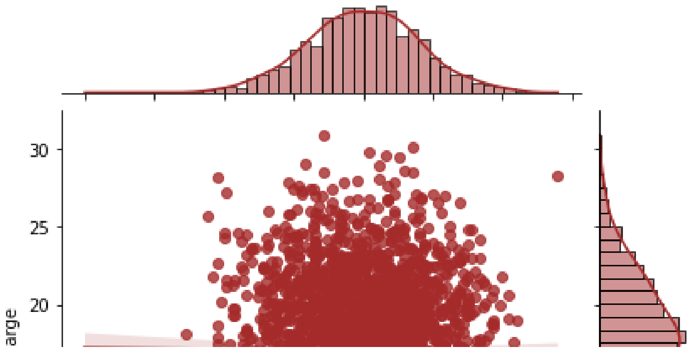
```
#How Total night minutes affect Total night calls
sns.jointplot(x='Total night minutes',y='Total night calls',data=df,kind='reg',color='indigo')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76180329d0>
```



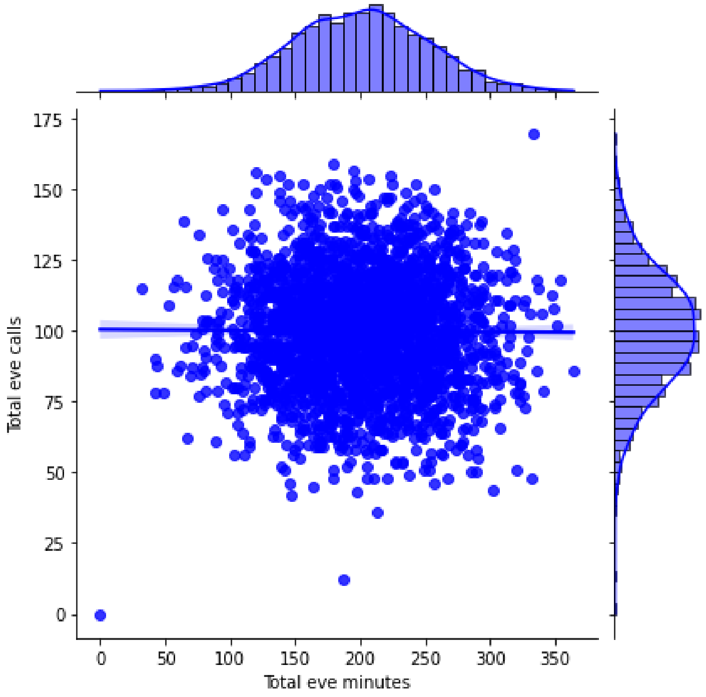
```
#How Total eve calls affect Total eve charge
sns.jointplot(x='Total eve calls',y='Total eve charge',data=df,kind='reg',color='brown')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7617f7f2e0>
```



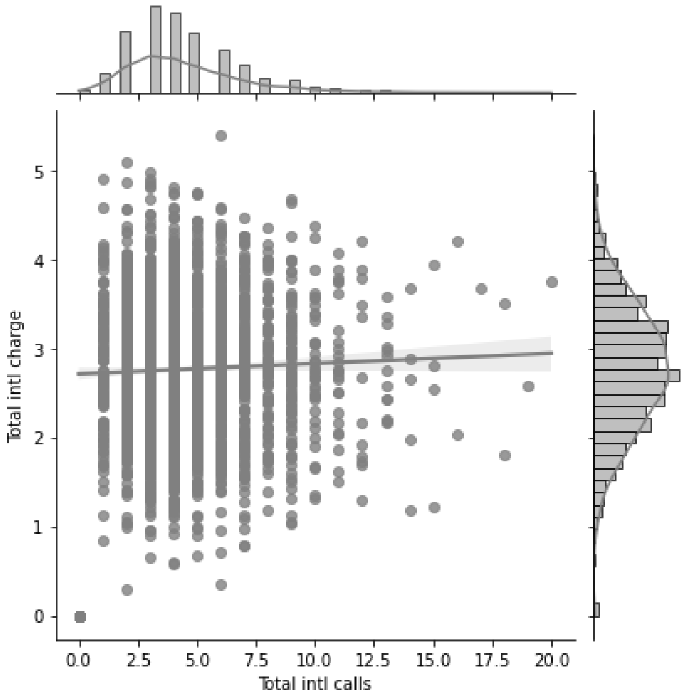
```
#How Total eve minutes affect Total eve calls
sns.jointplot(x='Total eve minutes',y='Total eve calls',data=df,kind='reg',color='blue')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7617c9f370>
```



```
#How Total intl calls affect Total intl charge
sns.jointplot(x='Total intl calls',y='Total intl charge',data=df,kind='reg',color='grey')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7618491580>
```



Encoding string to numeric using getdummies

```
df1=pd.get_dummies(df[['State','International plan','Voice mail plan']],drop_first=True)
df1
```

	State_AL	State_AR	State_AZ	State_CA	State_CO	State_CT	State_DC	State_DE	S
0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	

concatination-combining

```
dfe=pd.concat([df,df1],axis=1)
dfe
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	T min
0	KS	128	415	No	Yes	25	265.1	110	45.07	1
1	OH	107	415	No	Yes	26	161.6	123	27.47	1
2	NJ	137	415	No	No	0	243.4	114	41.38	1
3	OH	84	408	Yes	No	0	299.4	71	50.90	
4	OK	75	415	Yes	No	0	166.7	113	28.34	1
...	...	...	...	...	...	...	...	...	...	
2661	SC	79	415	No	No	0	134.7	98	22.90	1
2662	AZ	192	415	No	Yes	36	156.2	77	26.55	2
2663	WV	68	415	No	No	0	231.1	57	39.29	1
2664	RI	28	510	No	No	0	180.8	109	30.74	2
2665	TN	74	415	No	Yes	25	234.4	113	39.85	2

2666 rows × 72 columns

```
#Dropping unwanted columns
dfe.drop(['State','International plan','Voice mail plan'],axis=1,inplace=True)
dfe
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes
0	128	415	25	265.1	110	45.07	197.4	99	16.78	244.7
1	107	415	26	161.6	123	27.47	195.5	103	16.62	254.4
2	137	415	0	243.4	114	41.38	121.2	110	10.30	162.6
3	84	408	0	299.4	71	50.90	61.9	88	5.26	196.9
4	75	415	0	166.7	113	28.34	148.3	122	12.61	186.9
...	...	...	...	...	...	...	...	...	...	...
2661	79	415	0	134.7	98	22.90	189.7	68	16.12	221.4
2662	192	415	36	156.2	77	26.55	215.5	126	18.32	279.1
2663	68	415	0	231.1	57	39.29	153.4	55	13.04	191.3
2664	28	510	0	180.8	109	30.74	288.8	58	24.55	191.9
2665	74	415	25	234.4	113	39.85	265.9	82	22.60	241.4

2666 rows × 69 columns

```
#To find types of columns after encoding,concation,dropping
dfe.dtypes
```

```
Account length              int64
Area code                   int64
Number vmail messages       int64
Total day minutes         float64
Total day calls             int64
                           ...
```

```
    State_WI                    uint8
    State_WV                    uint8
    State_WY                    uint8
    International plan_Yes      uint8
    Voice mail plan_Yes         uint8
    Length: 69, dtype: object
```

```
#To find missing values after encoding,concation,dropping
dfe.isna().sum()
```

```
    Account length            0
    Area code                 0
    Number vmail messages     0
    Total day minutes         0
    Total day calls           0
                             ..
    State_WI                  0
    State_WV                  0
    State_WY                  0
    International plan_Yes    0
    Voice mail plan_Yes       0
    Length: 69, dtype: int64
```

```
#Seperate x_train
x_train=dfe.drop(['Churn'],axis=1)
x_train
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes
0	128	415	25	265.1	110	45.07	197.4	99	16.78	244.7
1	107	415	26	161.6	123	27.47	195.5	103	16.62	254.4
2	137	415	0	243.4	114	41.38	121.2	110	10.30	162.6
3	84	408	0	299.4	71	50.90	61.9	88	5.26	196.9

```
#Seperate y_train
y_train=dfe['Churn']
y_train
```

```
0       False
1       False
2       False
3       False
4       False
        ...  
2661    False
2662    False
2663    False
2664    False
2665    False
Name: Churn, Length: 2666, dtype: bool
```

Preprocessing Testing Data

```
import numpy as np
import pandas as pd
import seaborn as sns
dft=pd.read_csv('/content/churn-bigml-20.csv')
dft
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	To minu
0	LA	117	408	No	No	0	184.5	97	31.37	35
1	IN	65	415	No	No	0	129.1	137	21.95	22
2	NY	161	415	No	No	0	332.9	67	56.59	31
3	SC	111	415	No	No	0	110.4	103	18.77	13
4	HI	49	510	No	No	0	119.3	117	20.28	21
...	...	...	...	...	...	...	...	...	...	
662	WI	114	415	No	Yes	26	137.1	88	23.31	15
663	AL	106	408	No	Yes	29	83.6	131	14.21	20
664	VT	60	415	No	No	0	193.9	118	32.96	8
665	WV	159	415	No	No	0	169.8	114	28.87	19

```
#First 5 observation print
dft.head()
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Tota ev minute
0	LA	117	408	No	No	0	184.5	97	31.37	351.
1	IN	65	415	No	No	0	129.1	137	21.95	228.
2	NY	161	415	No	No	0	332.9	67	56.59	317.
3	SC	111	415	No	No	0	110.4	103	18.77	137.
4	HI	49	510	No	No	0	119.3	117	20.28	215.

```
#Last 5 observation print
dft.tail()
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	To minu
662	WI	114	415	No	Yes	26	137.1	88	23.31	15
663	AL	106	408	No	Yes	29	83.6	131	14.21	20
664	VT	60	415	No	No	0	193.9	118	32.96	8
665	WV	159	415	No	No	0	169.8	114	28.87	19
666	CT	184	510	Yes	No	0	213.8	105	36.35	15

```
#Each column types
dft.dtypes
```

```
State                      object
Account length              int64
Area code                   int64
International plan         object
Voice mail plan            object
Number vmail messages       int64
Total day minutes         float64
Total day calls             int64
Total day charge          float64
Total eve minutes         float64
Total eve calls             int64
Total eve charge          float64
Total night minutes       float64
Total night calls           int64
Total night charge        float64
Total intl minutes        float64
Total intl calls            int64
Total intl charge         float64
Customer service calls      int64
Churn                        bool
dtype: object
```

```
#Column heading print
dft.columns
```

```
Index(['State', 'Account length', 'Area code', 'International plan',
       'Voice mail plan', 'Number vmail messages', 'Total day minutes',
       'Total day calls', 'Total day charge', 'Total eve minutes',
       'Total eve calls', 'Total eve charge', 'Total night minutes',
       'Total night calls', 'Total night charge', 'Total intl minutes',
       'Total intl calls', 'Total intl charge', 'Customer service calls',
       'Churn'],
      dtype='object')
```

```
#To find missing values
dft.isna().sum()
```

```
State                     0
Account length            0
Area code                 0
International plan        0
Voice mail plan           0
Number vmail messages     0
Total day minutes         0
Total day calls           0
Total day charge          0
Total eve minutes         0
Total eve calls           0
Total eve charge          0
Total night minutes       0
Total night calls         0
Total night charge        0
Total intl minutes        0
Total intl calls          0
Total intl charge         0
Customer service calls    0
Churn                     0
dtype: int64
```

Finding Count Of Each String Value Columns and plotting Graphs

```
#State Counts
dft['State'].value_counts()
```

```
AZ    19
ND    18
WV    18
NJ    18
KS    18
```

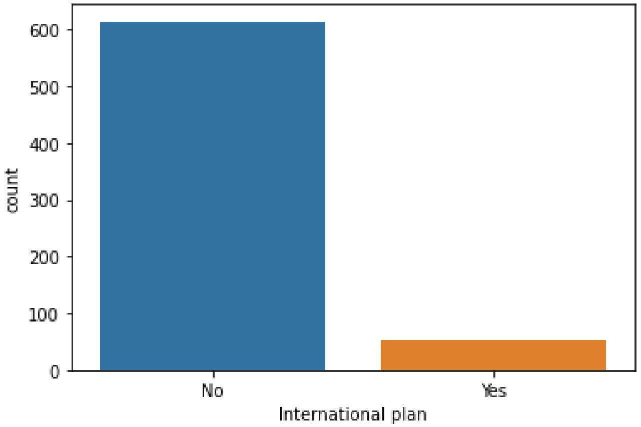
```
NM    18
WA    18
RI    17
ID    17
MS    17
WI    17
TX    17
IN    17
LA    16
NE    16
OR    16
KY    16
VT    16
NY    15
MI    15
MT    15
CT    15
AL    14
MN    14
MA    13
IL    13
ME    13
NH    13
TN    12
UT    12
NC    12
MO    12
OH    12
SD    11
WY    11
SC    11
CA    10
DE    10
VA    10
MD    10
FL     9
AK     9
DC     9
HI     9
PA     9
OK     9
AR     8
CO     7
IA     6
NV     5
GA     5
Name: State, dtype: int64
```

```
#International plan counts
dft['International plan'].value_counts()
```

```
No     614
Yes     53
Name: International plan, dtype: int64
```

```
#International plan counts graph
sns.countplot('International plan',data=dft)
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='International plan', ylabel='count'>
```



```
#How International plan affect Churn(Output_Variables) graph
sns.countplot('International plan',data=dft,hue='Churn')
```

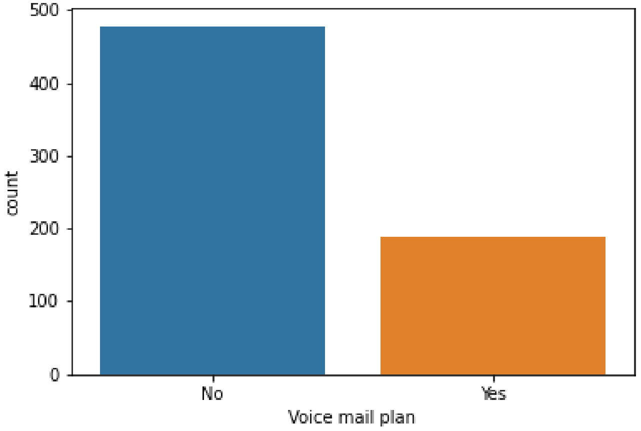
```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='International plan', ylabel='count'>
```

```
#Voice mail plan counts
dft['Voice mail plan'].value_counts()
```

```
No     478
Yes    189
Name: Voice mail plan, dtype: int64
```

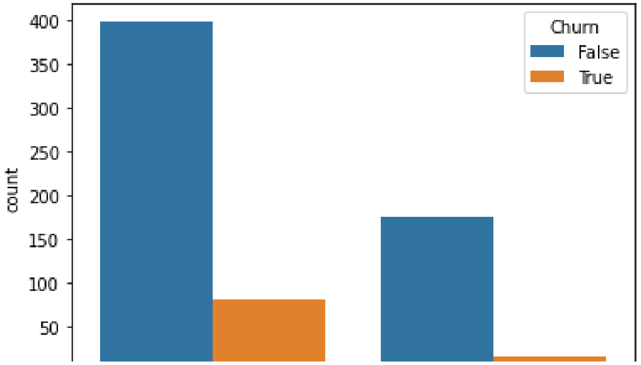
```
#Voice mail plan counts graph
sns.countplot('Voice mail plan',data=dft)
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='Voice mail plan', ylabel='count'>
```



```
#How Voice mail plan affect Churn(Output_Variables) graph
sns.countplot('Voice mail plan',data=dft,hue='Churn')
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='Voice mail plan', ylabel='count'>
```

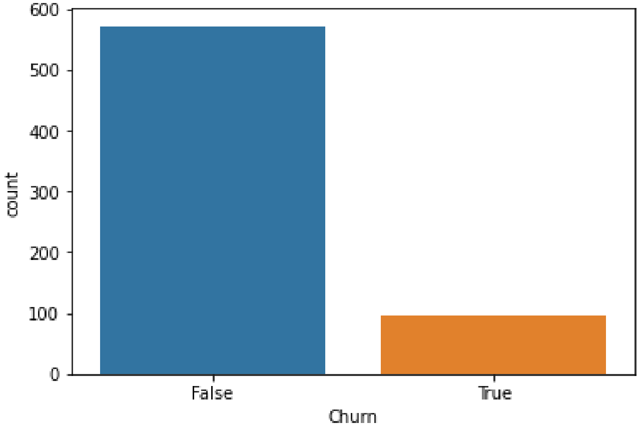


```
#churn counts
dft['Churn'].value_counts()
```

```
False    572
True      95
Name: Churn, dtype: int64
```

```
#Churn Counts graph
sns.countplot('Churn',data=dft)
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass
  warnings.warn(
<AxesSubplot:xlabel='Churn', ylabel='count'>
```



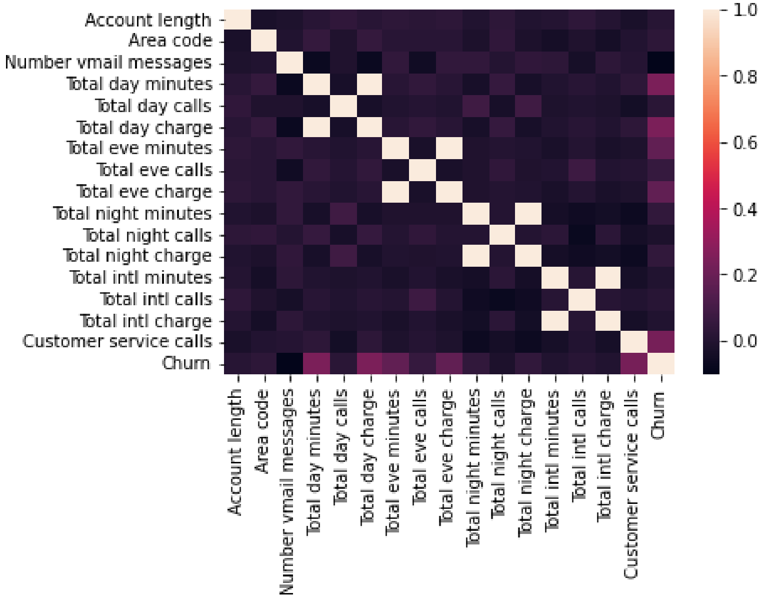
Correlation

```
dft.corr()
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	To ca
Account length	1.000000	-0.026327	-0.011993	0.017833	0.035703	0.017839	0.027043	0.02
Area code	-0.026327	1.000000	-0.006907	0.051507	-0.008972	0.051492	0.017160	0.01
Number vmail messages	-0.011993	-0.006907	1.000000	-0.069172	-0.009952	-0.069187	0.040865	-0.05
Total day minutes	0.017833	0.051507	-0.069172	1.000000	-0.032306	1.000000	0.017987	0.04

```
#Heatmap-correlation showing graph
sns.heatmap(dft.corr())
```

```
<AxesSubplot:>
```



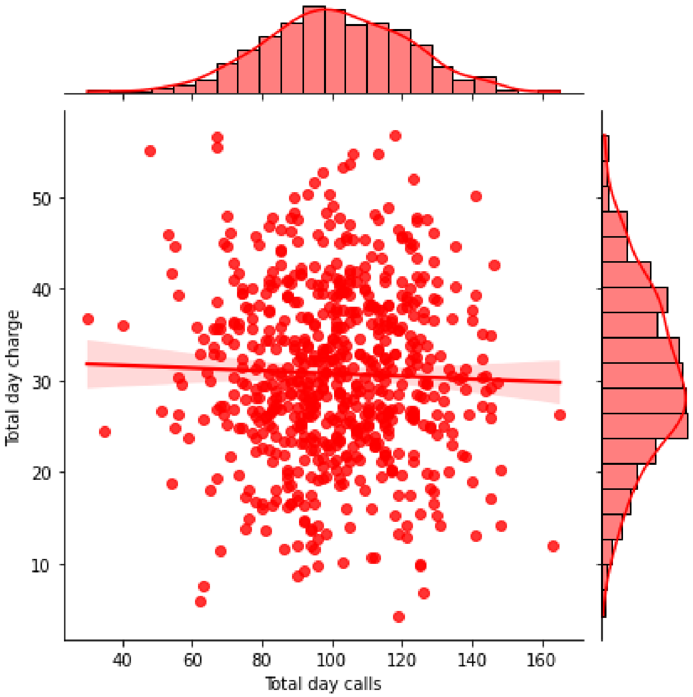
Total intl	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

JOINTPLOTGRAPH

Total intl	0.031279	-0.010530	-0.036847	0.016597	-0.005155	0.016582	0.002929	0.074

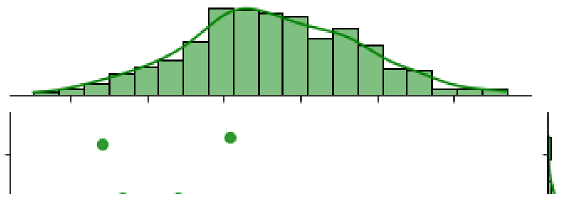
```
#How Total day calls affect Total day charge
sns.jointplot(x='Total day calls',y='Total day charge',data=dft,kind='reg',color='red')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76174cf370>
```



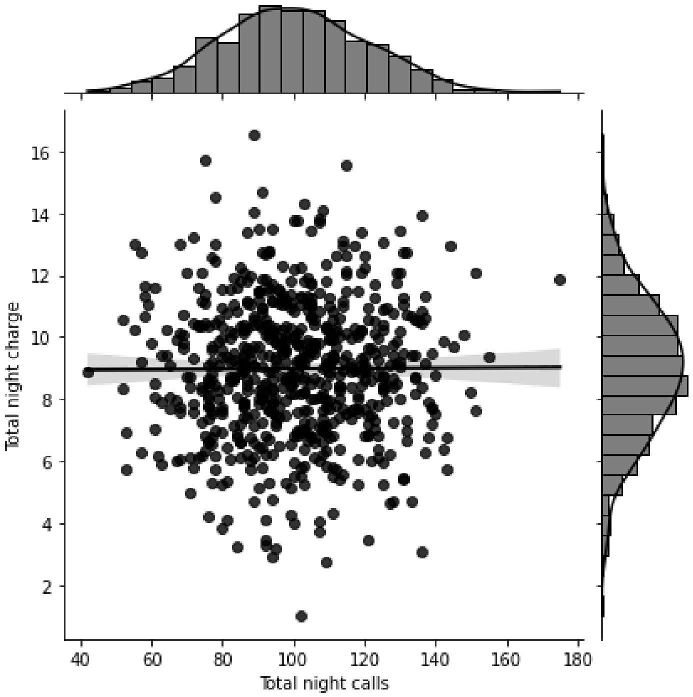
```
#How Total day mins affect Total day calls
sns.jointplot(x='Total day minutes',y='Total day calls',data=dft,kind='reg',color='green')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7617370fa0>
```



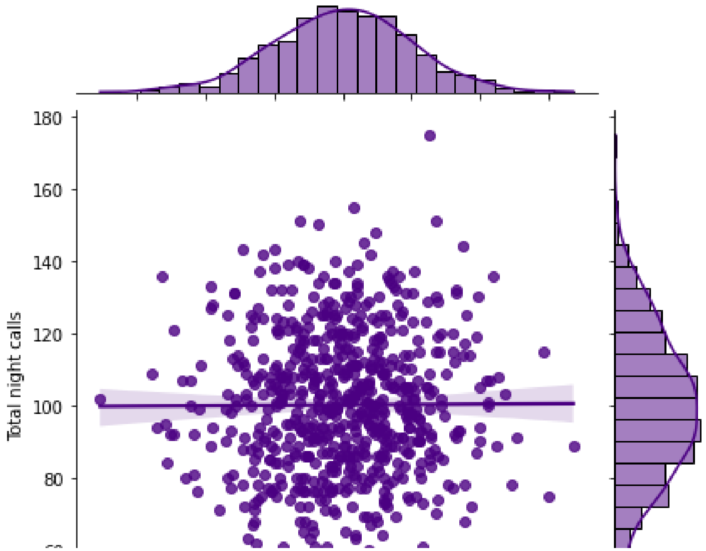
```
#How Total night calls affect Total night charge
sns.jointplot(x='Total night calls',y='Total night charge',data=dft,kind='reg',color='black')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76174c89a0>
```



```
#How Total night minutes affect Total night calls
sns.jointplot(x='Total night minutes',y='Total night calls',data=dft,kind='reg',color='indigo')
```

```
<seaborn.axisgrid.JointGrid at 0x7f76170cd460>
```

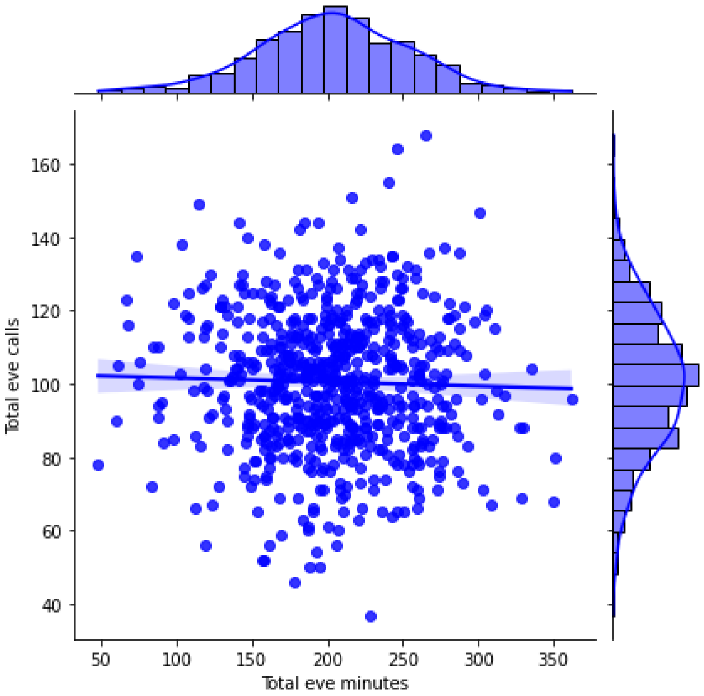


```
#How Total eve calls affect Total eve charge
sns.jointplot(x='Total eve calls',y='Total eve charge',data=dft,kind='reg',color='brown')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7616e94c40>
```

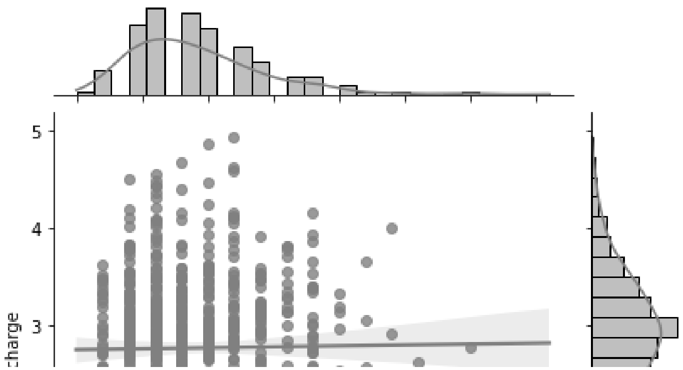
```
#How Total eve minutes affect Total eve calls
sns.jointplot(x='Total eve minutes',y='Total eve calls',data=dft,kind='reg',color='blue')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7616d90b80>
```



```
#How Total eve minutes affect Total eve calls
sns.jointplot(x='Total intl calls',y='Total intl charge',data=dft,kind='reg',color='grey')
```

```
<seaborn.axisgrid.JointGrid at 0x7f7616c16070>
```



Encoding string to numeric using getdummies

```
dft1=pd.get_dummies(dft[['State','International plan','Voice mail plan']],drop_first=True)
dft1
```

	State_AL	State_AR	State_AZ	State_CA	State_CO	State_CT	State_DC	State_DE	St
0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	
662	0	0	0	0	0	0	0	0	
663	1	0	0	0	0	0	0	0	
664	0	0	0	0	0	0	0	0	
665	0	0	0	0	0	0	0	0	
666	0	0	0	0	0	1	0	0	

667 rows × 52 columns

concatination-combining

```
dfte=pd.concat([dft,dft1],axis=1)
dfte
```

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	To minu
0	LA	117	408	No	No	0	184.5	97	31.37	35
1	IN	65	415	No	No	0	129.1	137	21.95	22
2	NY	161	415	No	No	0	332.9	67	56.59	31
3	SC	111	415	No	No	0	110.4	103	18.77	13
4	HI	49	510	No	No	0	119.3	117	20.28	21
...	...	...	...	...	...	...	...	...	...	
662	WI	114	415	No	Yes	26	137.1	88	23.31	15
663	AL	106	408	No	Yes	29	83.6	131	14.21	20
664	VT	60	415	No	No	0	193.9	118	32.96	8
665	WV	159	415	No	No	0	169.8	114	28.87	19
666	CT	184	510	Yes	No	0	213.8	105	36.35	15

667 rows × 72 columns

```
#Dropping unwanted columns
dfte.drop(['State','International plan','Voice mail plan'],axis=1,inplace=True)
dfte
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes
0	117	408	0	184.5	97	31.37	351.6	80	29.89	215.8
1	65	415	0	129.1	137	21.95	228.5	83	19.42	208.8
2	161	415	0	332.9	67	56.59	317.8	97	27.01	160.6
3	111	415	0	110.4	103	18.77	137.3	102	11.67	189.6
4	49	510	0	119.3	117	20.28	215.1	109	18.28	178.7
...	...	...	...	...	...	...	...	...	...	...
662	114	415	26	137.1	88	23.31	155.7	125	13.23	247.6
663	106	408	29	83.6	131	14.21	203.9	131	17.33	229.5
664	60	415	0	193.9	118	32.96	85.0	110	7.23	210.1
665	159	415	0	169.8	114	28.87	197.7	105	16.80	193.7
666	184	510	0	213.8	105	36.35	159.6	84	13.57	139.2

667 rows × 69 columns

```
#To find types of columns after encoding,concation,dropping
dfte.dtypes
```

```
Account length              int64
Area code                   int64
Number vmail messages       int64
Total day minutes         float64
Total day calls             int64
                           ...
State_WI                    uint8
State_WV                    uint8
State_WY                    uint8
International plan_Yes      uint8
Voice mail plan_Yes         uint8
Length: 69, dtype: object
```

```
#To find missing values after encoding,concation,dropping
dfte.isna().sum()
```

```
Account length           0
Area code                0
Number vmail messages    0
Total day minutes        0
Total day calls          0
                        ..
State_WI                 0
State_WV                 0
State_WY                 0
International plan_Yes   0
Voice mail plan_Yes      0
Length: 69, dtype: int64
```

```
#Seperate x_test
x_test=dfte.drop(['Churn'],axis=1)
x_test
```

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes
0	117	408	0	184.5	97	31.37	351.6	80	29.89	215.8
1	65	415	0	129.1	137	21.95	228.5	83	19.42	208.8
2	161	415	0	332.9	67	56.59	317.8	97	27.01	160.6
3	111	415	0	110.4	103	18.77	137.3	102	11.67	189.6
4	49	510	0	119.3	117	20.28	215.1	109	18.28	178.7
...	...	...	...	...	...	...	...	...	...	...
662	114	415	26	137.1	88	23.31	155.7	125	13.23	247.6
663	106	408	29	83.6	131	14.21	203.9	131	17.33	229.5
664	60	415	0	193.9	118	32.96	85.0	110	7.23	210.1
665	159	415	0	169.8	114	28.87	197.7	105	16.80	193.7
666	184	510	0	213.8	105	36.35	159.6	84	13.57	139.2

667 rows × 68 columns

```
#Seperate y_test
y_test=dfte['Churn']
y_test
```

```
0      False
1       True
2       True
3      False
4      False
       ...  
662    False
663    False
664    False
665    False
666    False
Name: Churn, Length: 667, dtype: bool
```

Normalization using Minmaxscaler

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(x_train)
x_train=scaler.fit_transform(x_train)
x_test=scaler.fit_transform(x_test)
x_train
```

```
array([[0.52479339, 0.06862745, 0.5       , ..., 0.        , 0.        ,
        1.        ],
       [0.43801653, 0.06862745, 0.52      , ..., 0.        , 0.        ,
        1.        ],
       [0.56198347, 0.06862745, 0.        , ..., 0.        , 0.        ,
        0.        ],
       ...,
       [0.2768595 , 0.06862745, 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.11157025, 1.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.30165289, 0.06862745, 0.5       , ..., 0.        , 0.        ,
        1.        ]])
```

```
x_test
```

```
array([[0.5021645 , 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.27705628, 0.06862745, 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.69264069, 0.06862745, 0.        , ..., 0.        , 0.        ,
        0.        ],
       ...,
       [0.25541126, 0.06862745, 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.68398268, 0.06862745, 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.79220779, 1.        , 0.        , ..., 0.        , 1.        ,
        0.        ]])
```

Model creation

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
K_model=KNeighborsClassifier(n_neighbors=5)
nb_model=MultinomialNB()
sv_model=SVC()
lsb_model=[K_model,nb_model,sv_model]
```

Perfomance Evaluation

Accuracy score,Classification_report

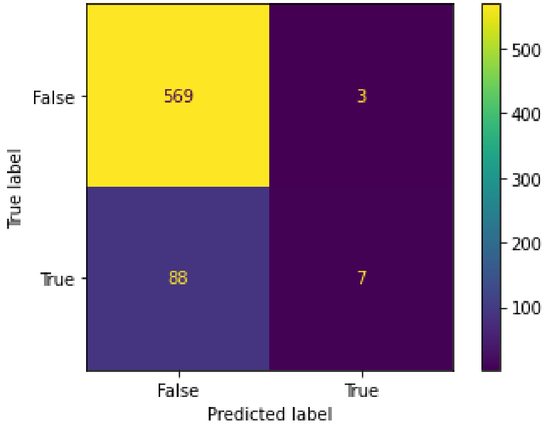
```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
for i in lsb_model:
  print(i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  result=confusion_matrix(y_test,y_pred)
  print(result)
  print("********************************************")
  print(accuracy_score(y_test,y_pred))
  print("********************************************")
  print(classification_report(y_test,y_pred))
```

```
KNeighborsClassifier()
[[566   6]
 [ 88   7]]
********************************************
0.8590704647676162
********************************************
              precision    recall  f1-score   support

       False       0.87      0.99      0.92       572
        True       0.54      0.07      0.13        95

    accuracy                           0.86       667
   macro avg       0.70      0.53      0.53       667
weighted avg       0.82      0.86      0.81       667

MultinomialNB()
[[569   3]
 [ 91   4]]
********************************************
0.8590704647676162
********************************************
              precision    recall  f1-score   support

       False       0.86      0.99      0.92       572
        True       0.57      0.04      0.08        95

    accuracy                           0.86       667
   macro avg       0.72      0.52      0.50       667
weighted avg       0.82      0.86      0.80       667

SVC()
[[569   3]
 [ 88   7]]
********************************************
0.863568215892054
********************************************
              precision    recall  f1-score   support

       False       0.87      0.99      0.93       572
        True       0.70      0.07      0.13        95

    accuracy                           0.86       667
   macro avg       0.78      0.53      0.53       667
weighted avg       0.84      0.86      0.81       667
```

Dislay Confusion metrics

```
from sklearn.metrics._plot.confusion_matrix import ConfusionMatrixDisplay
from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
cm=['False','True']
cmd=ConfusionMatrixDisplay(result,display_labels=cm)
cmd.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f7616abc4f0>
```


False
True
True label
False
True
Predicted label