```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from matplotlib import cluster,tree
from sklearn.preprocessing import scale
```

Load the Dataset

```
df = pd.read_csv('Churn_Modelling.csv')
df.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0

```
df.shape
```

```
(10000, 14)
```

```
df.isnull().any()
```

```
RowNumber        False
CustomerId       False
Surname          False
CreditScore      False
Geography        False
Gender           False
Age              False
Tenure           False
Balance          False
NumOfProducts    False
HasCrCard        False
IsActiveMember   False
EstimatedSalary  False
Exited           False
dtype: bool
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
RowNumber        10000 non-null: int64
CustomerId       10000 non-null: int64
Surname          10000 non-null: object
CreditScore      10000 non-null: int64
Geography        10000 non-null: object
Gender           10000 non-null: object
Age              10000 non-null: int64
Tenure           10000 non-null: int64
Balance          10000 non-null: float64
NumOfProducts    10000 non-null: int64
HasCrCard        10000 non-null: int64
IsActiveMember   10000 non-null: int64
EstimatedSalary  10000 non-null: float64
Exited           10000 non-null: int64
dtypes: float64(2), int64(9), object(3)
memory usage: 1.0 MB
```

```
df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	0.203700
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581954	0.45584	0.499797	57510.492818	0.402769
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	0.000000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000	0.000000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000	0.000000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500	0.000000
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000	1.000000

```
df.Geography.value_counts()
```

```
France  5014
Germany 2509
Spain   2477
Name: Geography, dtype: int64
```

```
df.Surname.value_counts()
```

```
Smith    32
Scott    29
Martin   29
Walker   28
Brown    26
         ..
Cowles    1
Bold      1
Bishop    1
Patterson 1
Austin    1
Name: Surname, Length: 2932, dtype: int64
```

Visualizations.

Univariate Analysis

```
sns.distplot(df.Tenure)
```

```
<seaborn.axisgrid.FacetGrid at 0x12b4d7b610>
```



```
plt.pie(df.Geography.value_counts(),[0,0.1],autopct='%1.1f%%',labels=['France','Germany','Spain'],shadow=False,colors=['blue','red','green'])
plt.show()
```



```
sns.kdeplot(df.Age)
```

```
<Axes:xlabel= Age', ylabel='Count'>
```

Bi - Variate Analysis

```
plt.barchf.Tenure,df.CreditScore)
<BarContainer object of 10000 artists>
```


```
plt.barchf.Age,df.EstimatedSalary)
<BarContainer object of 10000 artists>
plt.ylabel('EstimatedSalary')
plt.title('Barplot')
```

```
Text(0.5, 1.0, 'Barplot')
```

Multi - Variable Analysis

```
df.hist(figsize=(20,15))
```

```
array([[<matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>,
        <matplotlib.axes._subplots.AxesSubplot:>]],
dtype=object)
```

Descriptive statistics on the dataset.

```
df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	0.203700
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581954	0.45584	0.499797	57510.492818	0.402769
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	0.000000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000	0.000000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000	0.000000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500	0.000000
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000	1.000000

```
df corr()
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_8796\4116472045.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=True' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
RowNumber	1.000000	0.004202	0.005840	0.000783	-0.007321	-0.009067	0.007246	0.000599	-0.012044	-0.005988	-0.016571
CustomerId	0.004202	1.000000	0.000308	0.009497	-0.014883	-0.012419	0.016972	-0.014025	0.001665	0.015271	-0.006248
CreditScore	0.005840	0.000308	1.000000	-0.003965	0.000842	0.006268	-0.002415	0.002081	-0.027084	0.062244	-0.027094
Age	0.000783	-0.009497	-0.003965	1.000000	-0.009997	0.028308	-0.030680	-0.011721	0.085472	-0.007201	0.285323
Tenure	-0.007321	-0.014883	0.000842	-0.009997	1.000000	-0.012254	0.013444	0.022583	-0.028362	0.007369	-0.014001
Balance	-0.009067	-0.012419	0.006268	0.028308	-0.012254	1.000000	-0.304180	-0.014858	-0.010084	0.012797	0.118533
NumOfProducts	0.007246	0.016972	-0.002415	-0.030680	0.013444	-0.304180	1.000000	0.003183	0.009612	0.014204	-0.047820
HasCrCard	0.000599	0.014025	-0.002081	-0.011721	0.022583	-0.014858	0.003183	1.000000	-0.011866	-0.009933	-0.007138
IsActiveMember	0.012044	0.001665	-0.027084	0.085472	-0.028362	-0.010084	0.009612	-0.011866	1.000000	-0.011421	-0.156128
EstimatedSalary	-0.005988	0.015271	-0.062244	-0.007201	0.007369	0.012797	0.014204	-0.009933	-0.011421	1.000000	0.012097
Exited	-0.016571	-0.006248	-0.027094	0.285323	-0.014001	0.118533	-0.047820	-0.007138	-0.156128	0.012097	1.000000

```
df.CreditScore.mean()
```

```
650.5288
```

```
df.mean()
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_8796\3390014374.py:1: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=True' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.mean()
```

```
RowNumber        5000.50000
CustomerId       1.569094e+07
CreditScore       650.528800
Age                38.921800
Tenure              5.012800
Balance         76485.889288
NumOfProducts       1.530200
HasCrCard           0.705500
IsActiveMember      0.515100
EstimatedSalary    100090.239881
Exited              0.203700
dtype: float64
```

Handle the Missing values

```
df.isnull().any()
```

```
RowNumber        False
CustomerId       False
Surname          False
CreditScore      False
Geography        False
Gender           False
Age              False
Tenure           False
Balance          False
NumOfProducts    False
HasCrCard        False
IsActiveMember   False
EstimatedSalary  False
Exited           False
dtype: bool
```

Outliers and replace the outliers

```
df.shape
```

```
(10000, 14)
```

```
df describe()
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_8796\3390014374.py:1: FutureWarning: The default value of numeric_only in DataFrame.median is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=True' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.median()
```

```
RowNumber        5.000500e+03
CustomerId       1.569074e+07
CreditScore      6.520000e+02
Age              3.700000e+01
Tenure           5.000000e+00
Balance          9.719854e+04
NumOfProducts    1.000000e+00
HasCrCard        1.000000e+00
IsActiveMember   1.000000e+00
EstimatedSalary  1.001939e+05
Exited           0.000000e+00
dtype: float64
```

```
sns.boxplot(df.CreditScore)
```

```
<Axes:>
```



```
q25f CreditScore.quantile(0.25)
q75f CreditScore.quantile(0.75)
iqrf=q75
```

```
la_limit=q25-1.5*iqr
lu_limit=q75+1.5*iqr
```

```
df['CreditScore']=np.where(df['CreditScore']>lu_limit,lu_limit,df['CreditScore'])
```

```
sns.boxplot(df CreditScore)
```

```
<Axes:>
```


Categorical columns and perform encoding.

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
df Surname=le.fit_transform(df.Surname)
df Geography=le.fit_transform(df.Geography)
df Gender=le.fit_transform(df.Gender)
df head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	1115	619	0	0	42	2	0.00	1	1	1	101348.88	1
1	2	15647311	1177	608	2	0	41	1	83807.86	1	0	1	112542.58	0
2	3	15619304	2040	502	0	0	42	8	159660.80	3	1	0	113931.57	1
3	4	15701354	289	699	0	0	39	1	0.00	2	0	0	93826.63	0
4	5	15737888	1778	850	2	0	43	2	125510.82	1	1	1	79084.10	0

```
plt.figure(figsize=(8,6))
sns.heatmap(df.corr(),lanot=True)
```

```
<Axes:>
```


Split the data into dependent and independent variables.

```
y=df['EstimatedSalary']
y[0:5]
```

```
0    101348.88
1    112542.58
2    113931.57
3     93826.63
4     79084.10
Name: EstimatedSalary, dtype: float64
```

```
X=df.drop(columns=['EstimatedSalary'],axis=1)
X.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Exited
0	1	15634602	1115	619	0	0	42	2	0.00	1	1	1	1
1	2	15647311	1177	608	2	0	41	1	83807.86	1	0	1	0
2	3	15619304	2040	502	0	0	42	8	159660.80	3	1	0	1
3	4	15701354	289	699	0	0	39	1	0.00	2	0	0	0
4	5	15737888	1778	850	2	0	43	2	125510.82	1	1	1	0

Scale the independent variables

```
X_scaled=pd DataFrame(scale(X),columns=X.columns)
X_scaled.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Exited
0	-1.73478	-0.78202	-0.464165	-0.326362	-0.901886	-1.095988	0.293517	-1.041760	-1.225848	0.916202	0.646092	0.970243	1.977166
1	-1.73391	-0.606016	-0.456011	-0.44752	1.515067	-1.095988	0.198164	-1.387538	0.117663	-0.912283	-1.547768	0.970243	-0.505775
2	-1.73145	-0.995604	0.620968	-1.551762	-0.901886	-1.095988	0.293517	-1.041760	1.333053	2.527057	0.646092	-1.030670	1.977166
3	-1.72858	-0.144173	-0.918023	-0.608538	-0.901886	-1.095988	0.007457	-1.387538	-1.225848	0.807737	-1.547768	-1.030670	-0.505775
4	-1.72640	0.502806	2.072584	2.132564	1.515067	-1.095988	0.388871	-1.041760	0.785725	-0.912283	0.646092	0.970243	-0.505775