1.Downloading the Dataset and importing theLibraries

```
# import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

2.load the datset

```
from google.colab import files
uploaded = files.upload()
```

Choose Files Churn_Modelling.csv

- **Churn_Modelling.csv**(text/csv) - 684858 bytes, last modified: 9/22/2022 - 100% done

```
Saving Churn_Modelling.csv to Churn_Modelling (1).csv
```

```
data= pd.read_csv("/content/Churn_Modelling.csv",encoding='latin1',low_memory=False)
```

```
data = pd.read_csv("/content/Churn_Modelling.csv",encoding='latin1',low_memory=False)
```

```
data.sample(20)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure
5938	5939	15679668	Yao	850	Spain	Male	38	7
4850	4851	15593094	Goddard	516	France	Male	27	9
1008	1009	15569050	Farrell	444	France	Male	45	6
1951	1952	15589793	Onwuamaeze	604	France	Male	53	8
3399	3400	15633352	Okwukwe	628	France	Female	31	6
4968	4969	15572158	Blackburn	604	Spain	Male	41	3
4740	4741	15618661	Chidubem	535	France	Male	30	6
8083	8084	15684011	Miller	576	Germany	Male	29	7
2171	2172	15747174	Hao	526	Germany	Male	58	9
35	36	15794171	Lombardo	475	France	Female	45	0

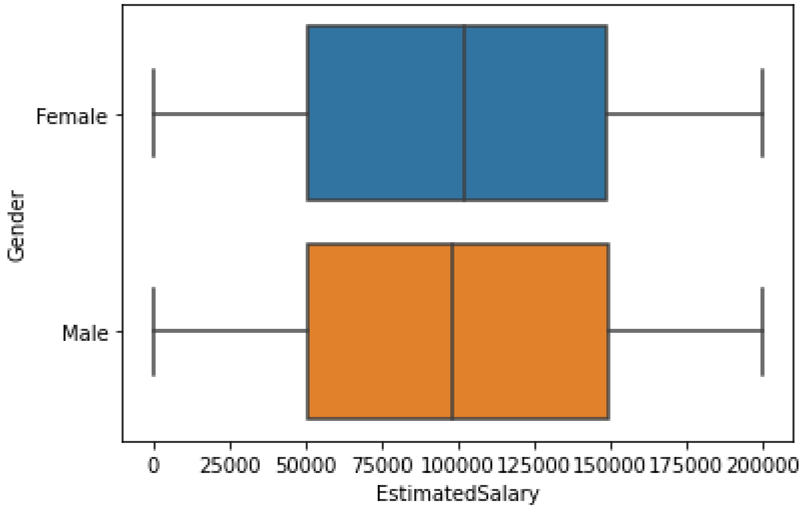
univarient

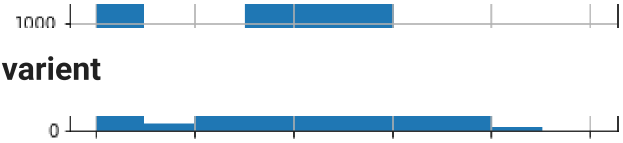
2013 2014 15742238 Bellucci 705 Germany Male 35 4

```
features =['Age',   'CreditScore',  'Balance']
data[features].hist(figsize=(13, 10));
```

bivarient

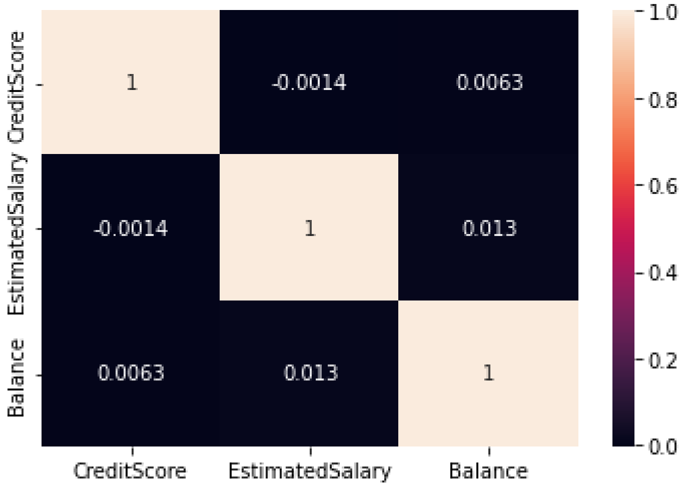
```
import seaborn as sns
sns.boxplot(x = data['EstimatedSalary'], y = data['Gender'] );
```





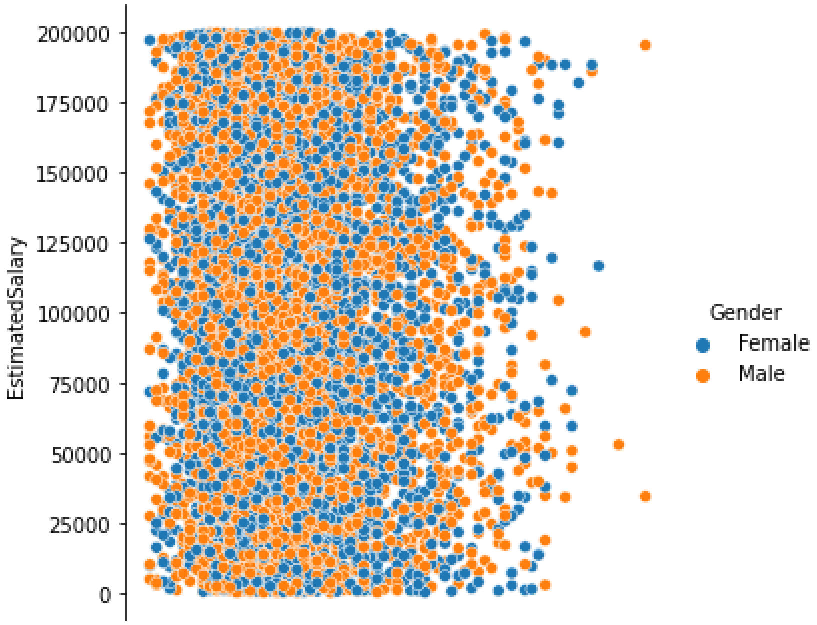
multivarient

```
df_1 = pd.DataFrame(data,columns=['CreditScore','EstimatedSalary','Balance'])
corrMatrix = df_1.corr()
sns.heatmap(corrMatrix, annot=True)
plt.show()
```



```
sns.relplot(x = "Age",y ="EstimatedSalary",hue="Gender",data=data)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f390d419bd0>
```



4. Performing descriptive statistics on the dataset.

```
data[['CreditScore','Balance','EstimatedSalary']].mean()
```

```
CreditScore           650.528800
Balance             76485.889288
EstimatedSalary    100090.239881
dtype: float64
```

```
data[['CreditScore','Balance','EstimatedSalary']].median()
```

```
CreditScore           652.000
Balance             97198.540
EstimatedSalary    100193.915
dtype: float64
```

```
data[['CreditScore','Balance','EstimatedSalary']].mode()
```

	CreditScore	Balance	EstimatedSalary
0	850	0.0	24924.92

```
data[['CreditScore','Balance','EstimatedSalary']].quantile()
```

```
CreditScore           652.000
Balance             97198.540
EstimatedSalary    100193.915
Name: 0.5, dtype: float64
```

```
data[['CreditScore','Balance','EstimatedSalary']].std()
```

```
     CreditScore           96.653299
     Balance            62397.405202
     EstimatedSalary    57510.492818
     dtype: float64
```

```
data[['CreditScore','Balance','EstimatedSalary']].min()
```

```
     CreditScore        350.00
     Balance              0.00
     EstimatedSalary     11.58
     dtype: float64
```

```
data[['CreditScore','Balance','EstimatedSalary']].max()
```

```
     CreditScore           850.00
     Balance            250898.09
     EstimatedSalary    199992.48
     dtype: float64
```

```
data[['CreditScore','Balance','EstimatedSalary']].skew()
```

```
     CreditScore       -0.071607
     Balance           -0.141109
     EstimatedSalary    0.002085
     dtype: float64
```

```
data.info()
```

```
     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 10000 entries, 0 to 9999
     Data columns (total 14 columns):
      #   Column           Non-Null Count  Dtype  
     ---  ------           --------------  -----  
      0   RowNumber        10000 non-null  int64  
      1   CustomerId       10000 non-null  int64  
      2   Surname          10000 non-null  object 
      3   CreditScore      10000 non-null  int64  
      4   Geography        10000 non-null  object 
      5   Gender           10000 non-null  object 
      6   Age              10000 non-null  int64  
      7   Tenure           10000 non-null  int64  
      8   Balance          10000 non-null  float64
      9   NumOfProducts    10000 non-null  int64  
      10  HasCrCard        10000 non-null  int64  
      11  IsActiveMember   10000 non-null  int64  
      12  EstimatedSalary  10000 non-null  float64
      13  Exited           10000 non-null  int64  
     dtypes: float64(2), int64(9), object(3)
     memory usage: 1.1+ MB
```

```
data.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000

5. Handling the Missing values.

```
data.isnull().sum()
```

```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

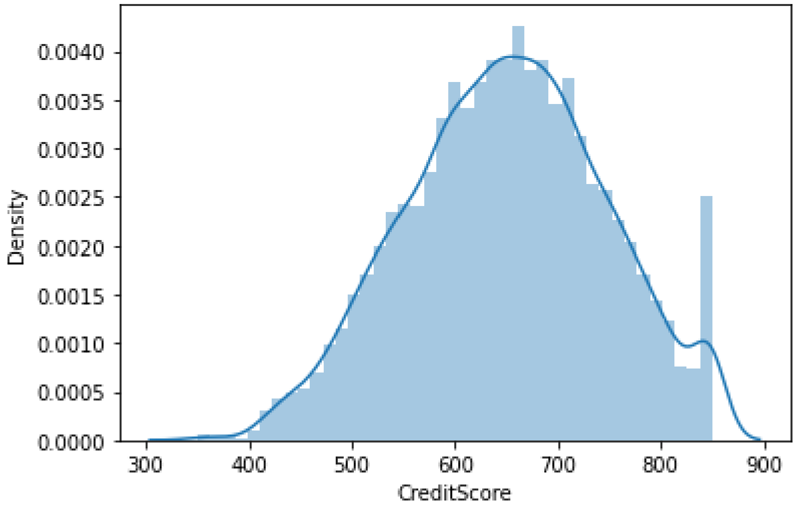
```
data.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.00000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.88928
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.00000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.00000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.54000

6.finding the outliers and replace the outlier

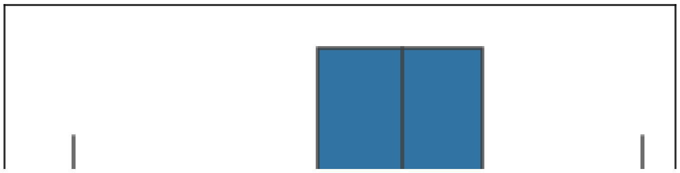
```
sns.distplot(data['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `d
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f3938dedd10>
```

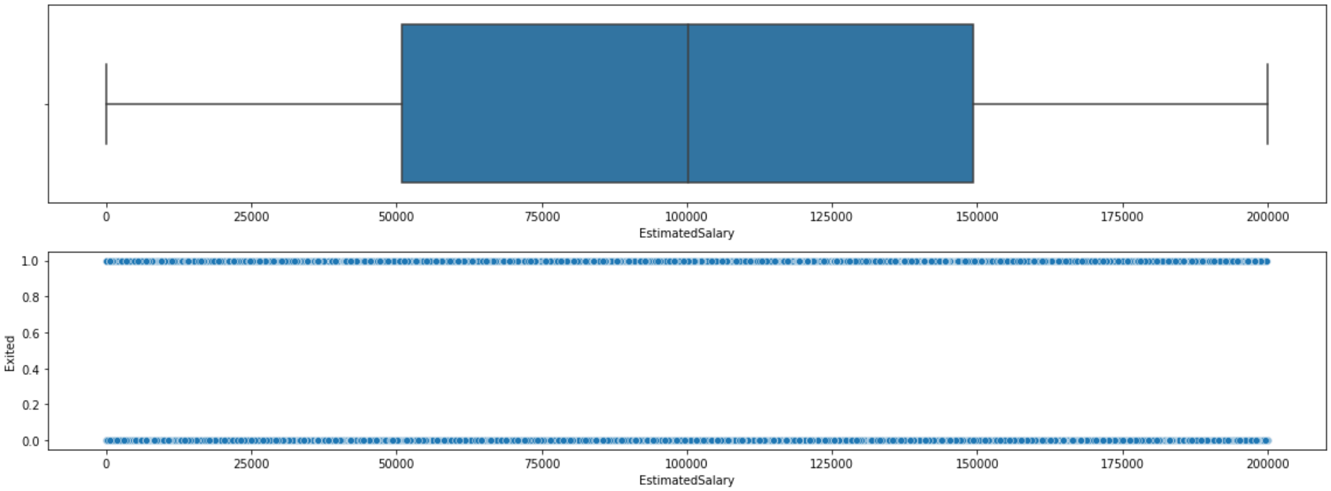


```
sns.boxplot(data['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass t
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f390dff3050>
```



```
box_scatter(data,'EstimatedSalary','Exited');
plt.tight_layout()
```



```
upper_limit = data['CreditScore'].mean() + 3*data['CreditScore'].std()
lower_limit = data['CreditScore'].mean() - 3*data['CreditScore'].std()
print('upper limit:', upper_limit)
print('lower limit:', lower_limit)
```

```
upper limit: 940.488696208391
lower limit: 360.568903791609
```

```
data.loc[(data['CreditScore'] > upper_limit) | (data['CreditScore'] < lower_limit)]
```

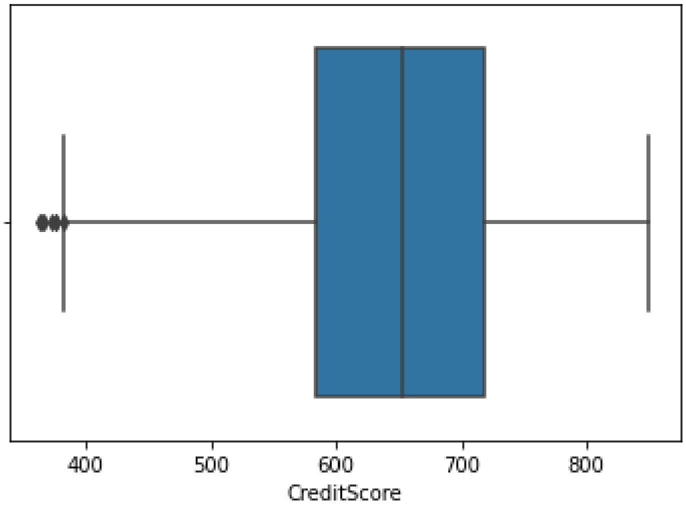
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	B
1405	1406	15612494	Panicucci	359	France	Female	44	6	12
1631	1632	15685372	Azubuike	350	Spain	Male	54	1	15
1838	1839	15758813	Campbell	350	Germany	Male	39	0	10

```
new_data = data.loc[(data['CreditScore'] <= upper_limit) & (data['CreditScore'] >= lower_limi
print('before removing outliers:', len(data))
print('after removing outliers:',len(new_data))
print('outliers:', len(data)-len(new_data))
```

```
before removing outliers: 10000
after removing outliers: 9992
outliers: 8
```

```
sns.boxplot(new_data['CreditScore'])
```

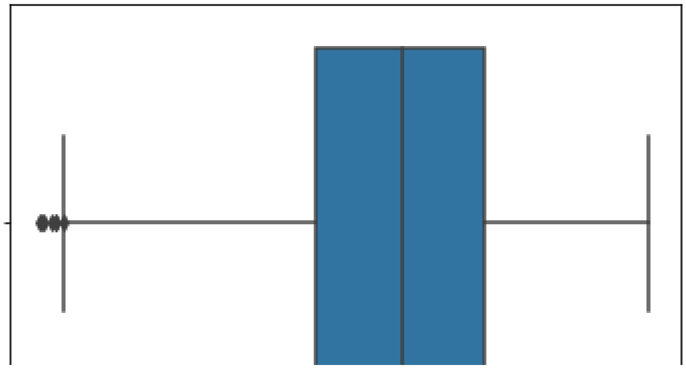
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass t
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f390e305110>
```



```
new_df = data.copy()
new_df.loc[(new_df['CreditScore']>=upper_limit), 'CreditScore'] = upper_limit
new_df.loc[(new_df['CreditScore']<=lower_limit), 'CreditScore'] = lower_limit
```

```
sns.boxplot(new_data['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass t
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f390e0bedd0>
```



```
len(new_data)
```

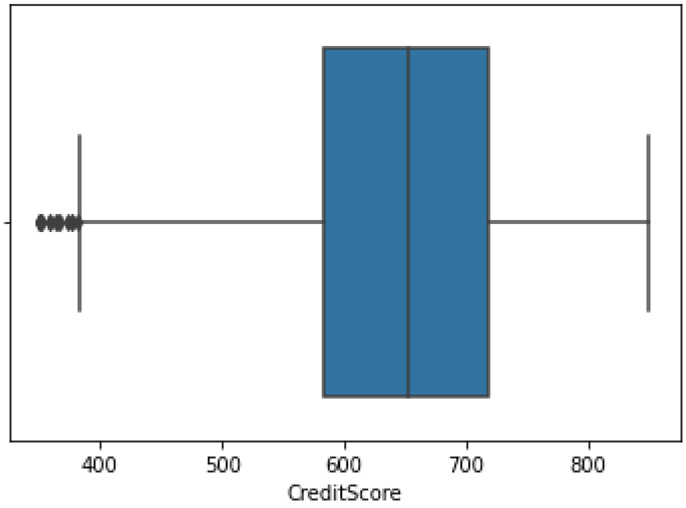
```
9992
```

```
upper_limit = data['CreditScore'].quantile(0.99)
lower_limit = data['CreditScore'].quantile(0.01)
print('upper limit:', upper_limit)
print('lower limit:', lower_limit)
```

```
upper limit: 850.0
lower limit: 432.0
```

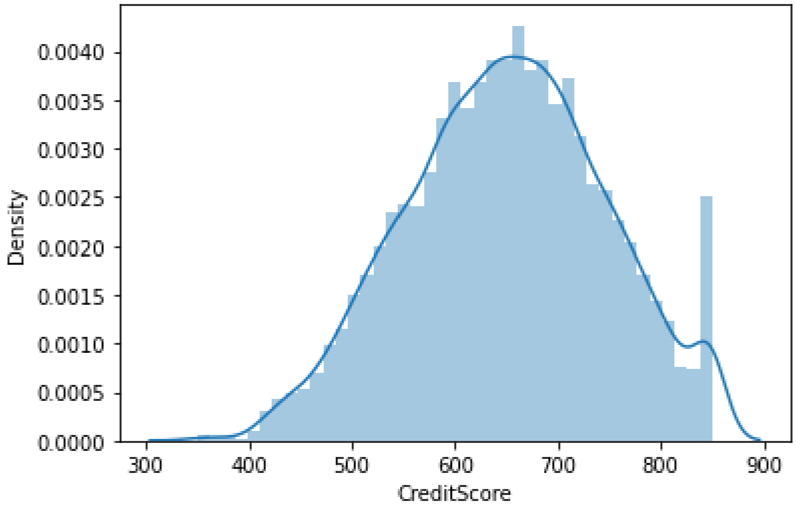
```
sns.boxplot(data['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass t
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f390df2af50>
```



```
sns.distplot(data['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `d
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f390de60810>
```



7.Checking for Categorical columns andperforming encoding.

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
for i in data:
if data[i].dtype=='object' or data[i].dtype=='category':data[i]=encoder.fit_transform(data[i]
```

8. Split the data into dependent andindependent variables

```
x = data.iloc[:,:-1]
x.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balar
0	1	15634602	Hargrave	619	France	Female	42	2	0.
1	2	15647311	Hill	608	Spain	Female	41	1	83807.
2	3	15619304	Onio	502	France	Female	42	8	159660.
3	4	15701354	Boni	699	France	Female	39	1	0.
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.

```
y=data.iloc[:-1]
```

```
y.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balan
0	1	15634602	Hargrave	619	France	Female	42	2	0.
1	2	15647311	Hill	608	Spain	Female	41	1	83807.
2	3	15619304	Onio	502	France	Female	42	8	159660.
3	4	15701354	Boni	699	France	Female	39	1	0.
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.

9.Scaling the independent variables

```
names=x.columns
names
```

```
    Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
           'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
           'IsActiveMember', 'EstimatedSalary'],
          dtype='object')
```

```
X = pd.DataFrame(x,columns = names)
X
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	E
0	1	15634602	Hargrave	619	France	Female	42	2	
1	2	15647311	Hill	608	Spain	Female	41	1	8

10. Splitting the data into Training andTesting

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
df=pd.read_csv("/content/Churn_Modelling (1).csv")
df
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	E
0	1	15634602	Hargrave	619	France	Female	42	2	
1	2	15647311	Hill	608	Spain	Female	41	1	8
2	3	15619304	Onio	502	France	Female	42	8	15
3	4	15701354	Boni	699	France	Female	39	1	
4	5	15737888	Mitchell	850	Spain	Female	43	2	12
...	...	...	...	...	...	...	...	...	
9995	9996	15606229	Obijiaku	771	France	Male	39	5	
9996	9997	15569892	Johnstone	516	France	Male	35	10	5
9997	9998	15584532	Liu	709	France	Female	36	7	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	7
9999	10000	15628319	Walker	792	France	Female	28	4	13

10000 rows × 14 columns

```
import numpy as np
```

```
x=np.array(df['CreditScore']).reshape(-1,1)
```

```
x.shape
```

```
(10000, 1)
```

```
print(x)
```

```
[[619]
 [608]
 [502]
 ...
 [709]
 [772]
 [792]]
```

```
y=np.array(df['CreditScore']).reshape(-1,1)
```

```
y.shape
```

```
(10000, 1)
```

```
print(y)
```

```
[[619]
 [608]
 [502]
 ...
 [709]
 [772]
 [792]]
```

```
print(type(y))
```

```
<class 'pandas.core.frame.DataFrame'>
```

! 0s completed at 6:47 AM ● ✕