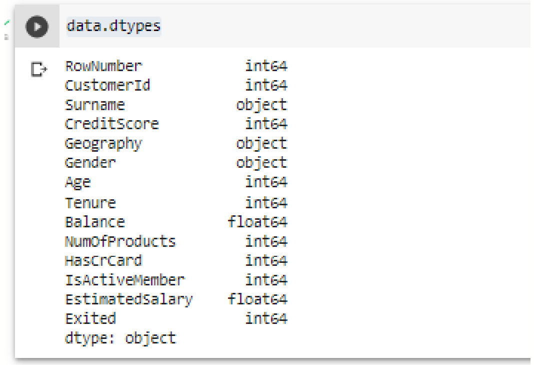
Assignment Date	21st September 2022
Student Name	Roshni RR
Student Roll No	211519104129
Maximum Marks	10 Marks

Q1:Perform Below Visualizations.

- **Univariate Analysis**
- **Bi - Variate Analysis**
- **Multi - Variate Analysis**

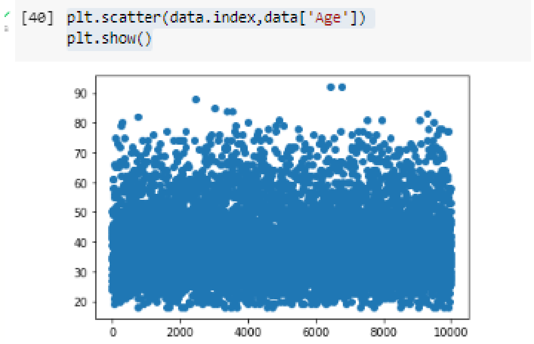
```
import matplotlib.pyplot as plt

import seaborn as sns

data.dtypes

plt.scatter(data.index,data['Age'])

plt.show()
```

UNIVARIATE ANALYSIS



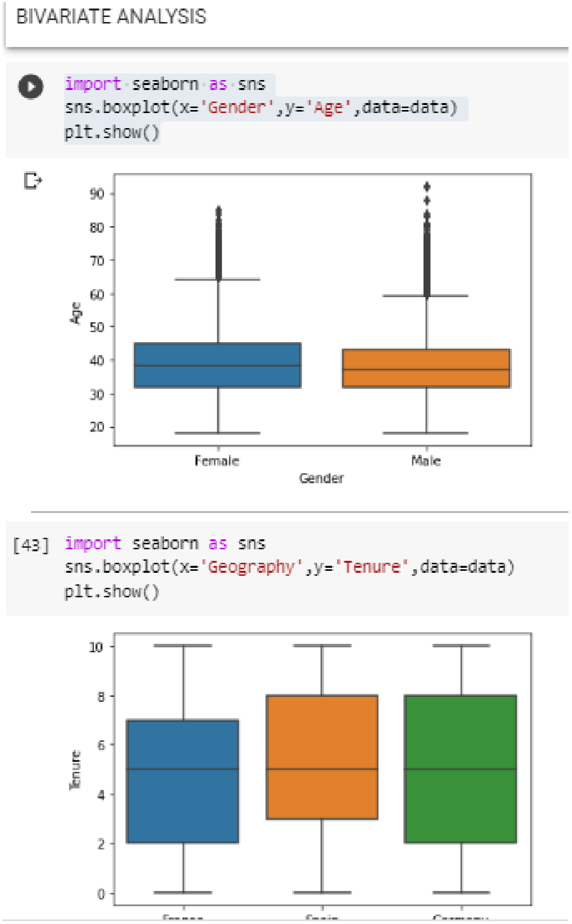
```
data.dtypes
```

```
RowNumber            int64
CustomerId           int64
Surname             object
CreditScore          int64
Geography           object
Gender              object
Age                  int64
Tenure               int64
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
EstimatedSalary    float64
Exited               int64
dtype: object
```



```
[40] plt.scatter(data.index,data['Age'])
     plt.show()
```

```
import seaborn as sns

sns.boxplot(x='Gender',y='Age',data=data)

plt.show()
```

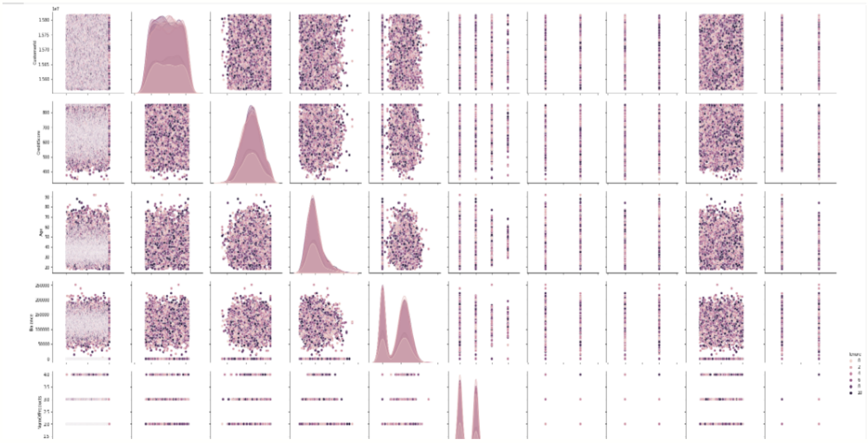


BIVARIATE ANALYSIS

```
import seaborn as sns
sns.boxplot(x='Gender',y='Age',data=data)
plt.show()
```

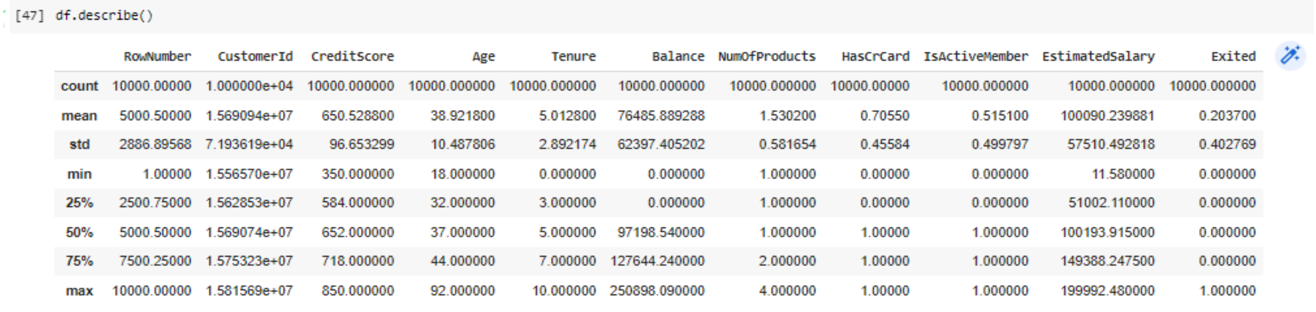
```
[43] import seaborn as sns
sns.boxplot(x='Geography',y='Tenure',data=data)
plt.show()
```

```
import seaborn as sns

sns.pairplot(data,hue="Tenure",height=3)

plt.show()
```



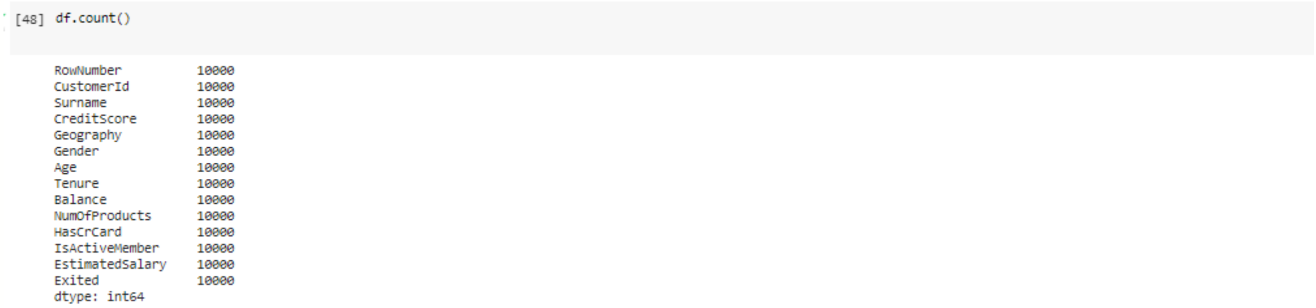
Q2: Perform descriptive statistics on the dataset.

```
import pandas as pd

import numpy as np

df = pd.DataFrame(data)

print (df)

df.describe()

df.count()
```



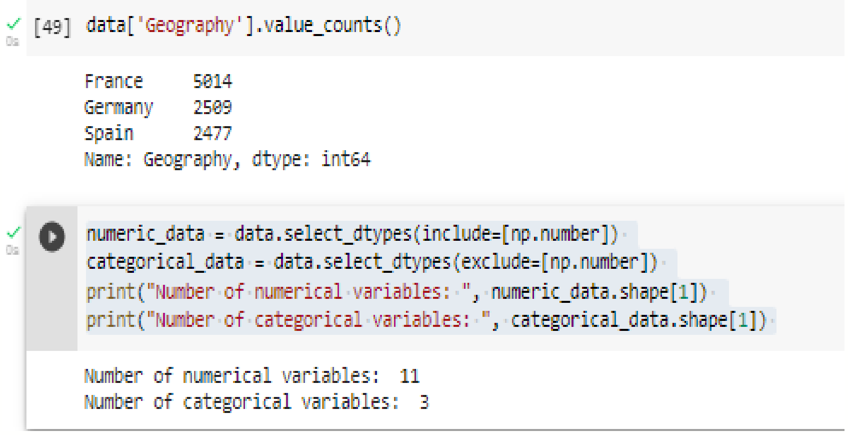
```
[47] df.describe()
```

	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	10000.00000	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	0.581654	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



```
[48] df.count()

RowNumber          10000
CustomerId         10000
Surname            10000
CreditScore        10000
Geography          10000
Gender             10000
Age                10000
Tenure             10000
Balance            10000
NumOfProducts      10000
HasCrCard          10000
IsActiveMember     10000
EstimatedSalary    10000
Exited             10000
dtype: int64
```

```
data['Geography'].value_counts()

numeric_data = data.select_dtypes(include=[np.number])

categorical_data = data.select_dtypes(exclude=[np.number])

print("Number of numerical variables: ", numeric_data.shape[1])

print("Number of categorical variables: ", categorical_data.shape[1])
```

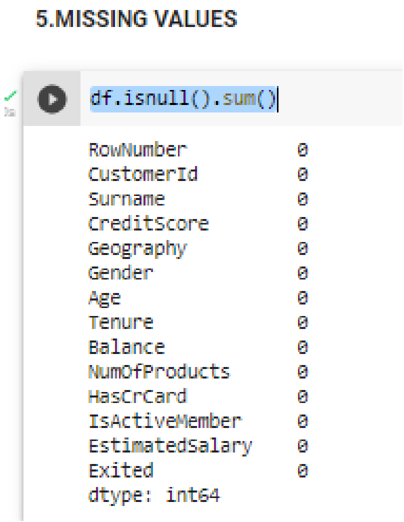


```
[49] data['Geography'].value_counts()

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

```
numeric_data = data.select_dtypes(include=[np.number])
categorical_data = data.select_dtypes(exclude=[np.number])
print("Number of numerical variables: ", numeric_data.shape[1])
print("Number of categorical variables: ", categorical_data.shape[1])

Number of numerical variables:  11
Number of categorical variables:  3
```

Q3:Handle the Missing values.

```
df.isnull().sum()
```

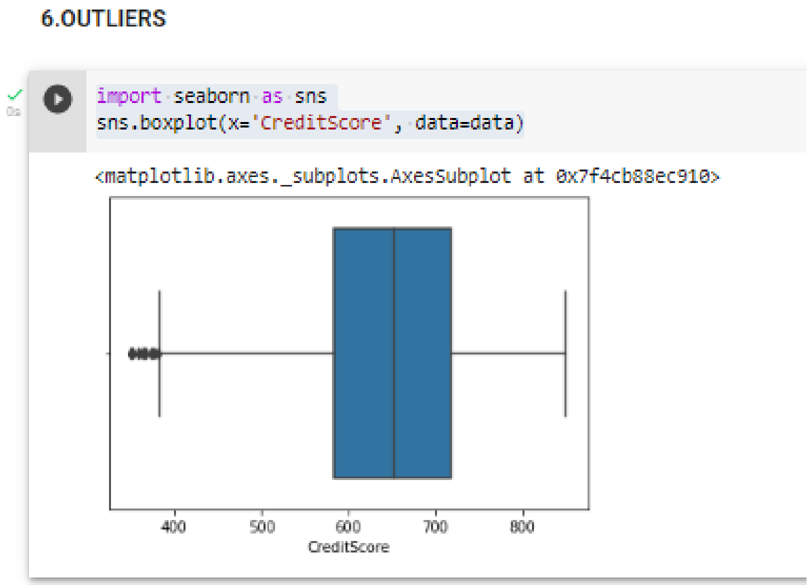


5.MISSING VALUES

```
df.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
```

Q4: Find the outliers and replace the outliers

```
import seaborn as sns

sns.boxplot(x='CreditScore', data=data)
```



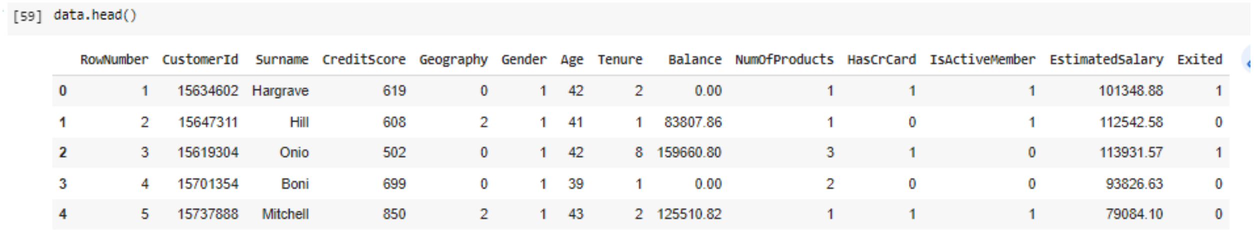
6.OUTLIERS

```
import seaborn as sns
sns.boxplot(x='CreditScore', data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4cb88ec910>
```

Q5:Check for Categorical columns and perform encoding.

```
print("Number of categorical variables: ", categorical_data.shape[1])

Cat_vars = list(categorical_data.columns)

Cat_vars


data['Geography'].value_counts()


data['Gender'].value_counts()

CleanGender = {"Gender": {"Male": 0, "Female": 2}}

data = data.replace(CleanGender)
```

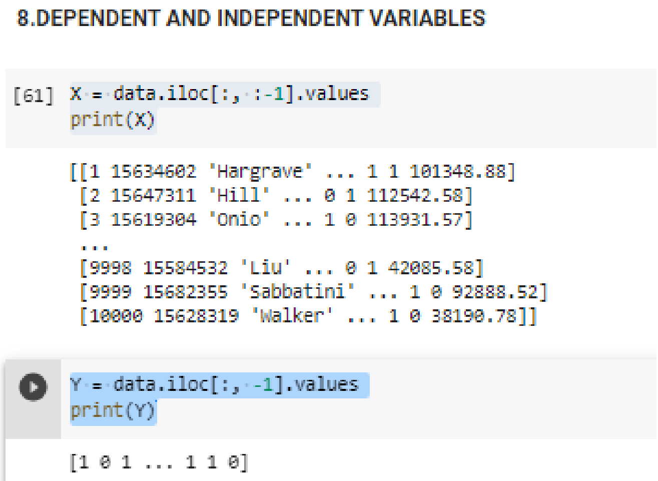


```
[59] data.head()
```

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

Q6:Split the data into dependent and independent variables.

```
X = data.iloc[:, :-1].values

print(X)


Y = data.iloc[:, -1].values

print(Y)
```



8.DEPENDENT AND INDEPENDENT VARIABLES

```
[61] X = data.iloc[:, :-1].values
     print(X)
```

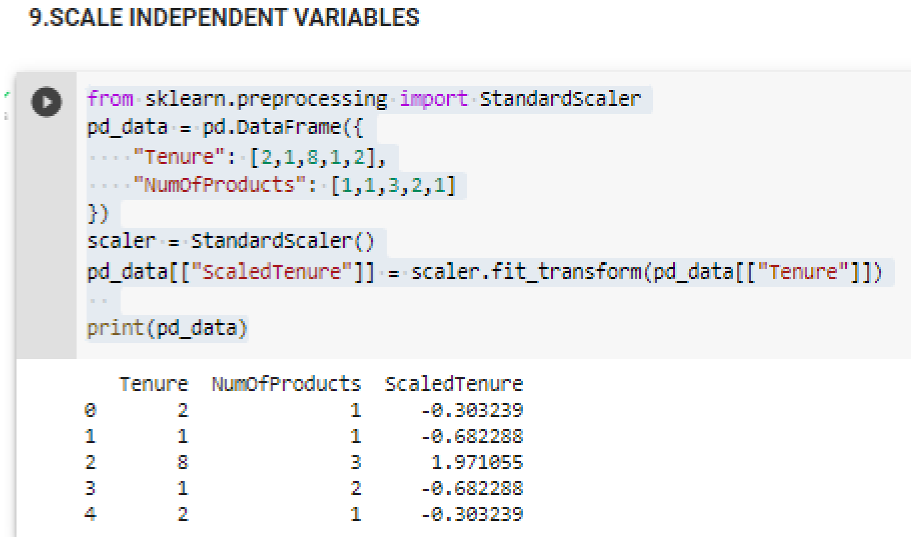
```
[[1 15634602 'Hargrave' ... 1 1 101348.88]
 [2 15647311 'Hill' ... 0 1 112542.58]
 [3 15619304 'Onio' ... 1 0 113931.57]
 ...
 [9998 15584532 'Liu' ... 0 1 42085.58]
 [9999 15682355 'Sabbatini' ... 1 0 92888.52]
 [10000 15628319 'Walker' ... 1 0 38190.78]]
```

```
Y = data.iloc[:, -1].values
print(Y)
```

```
[1 0 1 ... 1 1 0]
```

Q7:Scale the independent variables

```
from sklearn.preprocessing import StandardScaler

pd_data = pd.DataFrame({

     "Tenure": [2,1,8,1,2],

     "NumOfProducts": [1,1,3,2,1]

})

scaler = StandardScaler()

pd_data[["ScaledTenure"]] = scaler.fit_transform(pd_data[["Tenure"]])

print(pd_data)
```



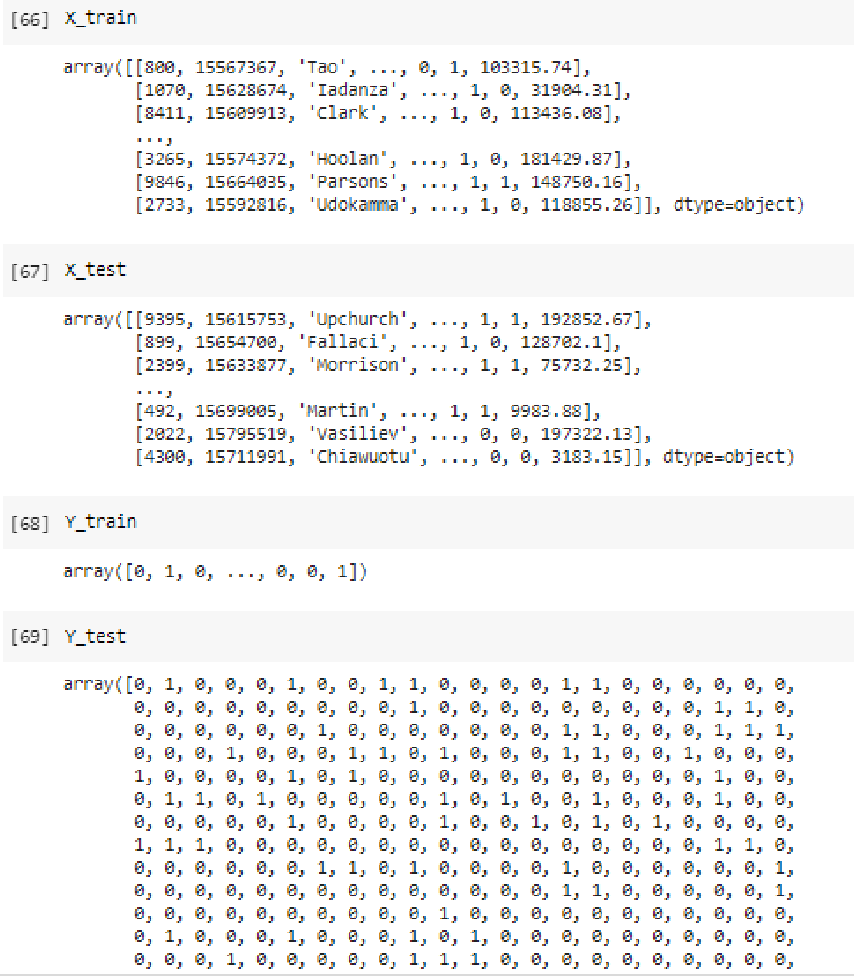
9.SCALE INDEPENDENT VARIABLES

```
from sklearn.preprocessing import StandardScaler
pd_data = pd.DataFrame({
    "Tenure": [2,1,8,1,2],
    "NumOfProducts": [1,1,3,2,1]
})
scaler = StandardScaler()
pd_data[["ScaledTenure"]] = scaler.fit_transform(pd_data[["Tenure"]])

print(pd_data)
```

```
   Tenure  NumOfProducts  ScaledTenure
0       2              1     -0.303239
1       1              1     -0.682288
2       8              3      1.971055
3       1              2     -0.682288
4       2              1     -0.303239
```

Q8:Split the data into training and testing

```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.05,
random_state=0)
```



```
[66] X_train

     array([[800, 15567367, 'Tao', ..., 0, 1, 103315.74],
            [1070, 15628674, 'Iadanza', ..., 1, 0, 31904.31],
            [8411, 15609913, 'Clark', ..., 1, 0, 113436.08],
            ...,
            [3265, 15574372, 'Hoolan', ..., 1, 0, 181429.87],
            [9846, 15664035, 'Parsons', ..., 1, 1, 148750.16],
            [2733, 15592816, 'Udokamma', ..., 1, 0, 118855.26]], dtype=object)
```

```
[67] X_test

     array([[9395, 15615753, 'Upchurch', ..., 1, 1, 192852.67],
            [899, 15654700, 'Fallaci', ..., 1, 0, 128702.1],
            [2399, 15633877, 'Morrison', ..., 1, 1, 75732.25],
            ...,
            [492, 15699005, 'Martin', ..., 1, 1, 9983.88],
            [2022, 15795519, 'Vasiliev', ..., 0, 0, 197322.13],
            [4300, 15711991, 'Chiawuotu', ..., 0, 0, 3183.15]], dtype=object)
```

```
[68] Y_train

     array([0, 1, 0, ..., 0, 0, 1])
```

```
[69] Y_test

     array([0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
            0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1,
            0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0,
            1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
            0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
            1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
            0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```