DSC680_CCPB_Data_Preprocessing

January 30, 2022

[27]:
```
import pandas as pd
```

[28]:
```
# loading the csv dataset

ccpb_df = pd.read_csv('C:
↪\BU\DSC680\project2\Customer_Churn_Prediction_in_banking\data\Churn_Modelling.
↪csv')
```

[29]:
```
print ("\n\n*********"+"\033[1m"+" Customer Churn Dataset"+␣
↪"\033[0m"+"**************** \n\n")
ccpb_df.info()
```

```
********* Customer Churn Dataset****************


<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
```

[30]:
```
# To check the number of unique classes of each attributes
print ("\n\n*********"+"\033[1m"+" number of unique classes of each␣
 ↪attributes"+ "\033[0m"+"**************** \n\n")
display (ccpb_df.nunique())
```

********* number of unique classes of each attributes****************

```
RowNumber          10000
CustomerId         10000
Surname             2932
CreditScore          460
Geography              3
Gender                 2
Age                   70
Tenure                11
Balance             6382
NumOfProducts          4
HasCrCard              2
IsActiveMember         2
EstimatedSalary     9999
Exited                 2
dtype: int64
```

[31]:
```
# Statistical description of the dataset
print ("\n\n*********"+"\033[1m"+" description of the dataset"+␣
 ↪"\033[0m"+"**************** \n\n")
display (ccpb_df.describe())
```

********* description of the dataset****************

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

             Balance  NumOfProducts    HasCrCard  IsActiveMember  \
```

```
count   10000.000000   10000.000000  10000.00000    10000.000000
mean    76485.889288       1.530200      0.70550        0.515100
std     62397.405202       0.581654      0.45584        0.499797
min         0.000000       1.000000      0.00000        0.000000
25%         0.000000       1.000000      0.00000        0.000000
50%     97198.540000       1.000000      1.00000        1.000000
75%    127644.240000       2.000000      1.00000        1.000000
max    250898.090000       4.000000      1.00000        1.000000

       EstimatedSalary        Exited
count     10000.000000  10000.000000
mean     100090.239881      0.203700
std       57510.492818      0.402769
min          11.580000      0.000000
25%       51002.110000      0.000000
50%      100193.915000      0.000000
75%      149388.247500      0.000000
max      199992.480000      1.000000
```

1 Describing each column feature

1. RowNumber: Row number for the row in the data table.
2. CustomerId: Unique Identification number of the customer.
3. Surname: Surname (Lastname) of the customer.
4. CreditScore: Credit Score of the customer.
5. Geography: Geographical location (country) of the customer.
6. Gender: Gender of the customer (Male / Female).
7. Age: Age of the customer.
8. Tenure: Number of years the customers has been associated with the bank.
9. Balance: The amount of balance in the customer's account.
10. NumOfProducts: Number of products that a customer has purchased through the bank during their tenure.
11. HasCrCard: Denotes if the customer owns a credit card with the bank.
12. IsActiveMember: Denotes if the customer is active with the bank.
13. EstimatedSalary: Estimated salary of the customer.
14. Exited: Denotes if the customer has churned (exited) from the bank or not.

[32]:
```
# check sample of the data
print ("\n\n*********"+"\033[1m"+" Sample Data from file"+␣
 ↪"\033[0m"+"**************** \n\n")
display(ccpb_df.head())
```

```
********* Sample Data from file****************


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

   Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0       2       0.00              1          1               1
1       1   83807.86              1          0               1
2       8  159660.80              3          1               0
3       1       0.00              2          0               0
4       2  125510.82              1          1               1

   EstimatedSalary  Exited
0        101348.88       1
1        112542.58       0
2        113931.57       1
3         93826.63       0
4         79084.10       0
```

[33]:
```
# data type of each columns feature variable

ccpb_df.dtypes
```

[33]:
```
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
```

[34]:
```
# The columns CustomerId,RowNumber are related to column  Surname and it is␣
 ↪considered as PII.
# [CustomerId,RowNumber,Surname] columns do not contain a significance on␣
 ↪quantitative analysis.
# Hence, we are good to drop these unused columns.

print ("\n\n*********"+"\033[1m"+" dropped columns␣
 ↪[CustomerId,RowNumber,Surname]"+ "\033[0m"+"**************** \n\n")
ccpb_df.drop(['RowNumber', 'CustomerId', 'Surname'], axis=1, inplace=True)
```

```
********* dropped columns [CustomerId,RowNumber,Surname]****************
```

[35]:
```
display(ccpb_df.dtypes)
```

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

[36]:
```
exitdf=ccpb_df[ccpb_df.Exited==0]
exitdf.IsActiveMember.value_counts()
```

[36]:
```
1    4416
0    3547
Name: IsActiveMember, dtype: int64
```

Observation: around 3547 people are not active and not exited. this will be a Potential feature for EDA

[37]:
```
# Check number of NaN or NULL
print ("\n\n*********"+"\033[1m"+"Check number of NaN or NULL"+␣
 ↪"\033[0m"+"**************** \n\n")
display(ccpb_df.isna().sum())
```

```
*********Check number of NaN or NULL****************
```

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

[ ]: