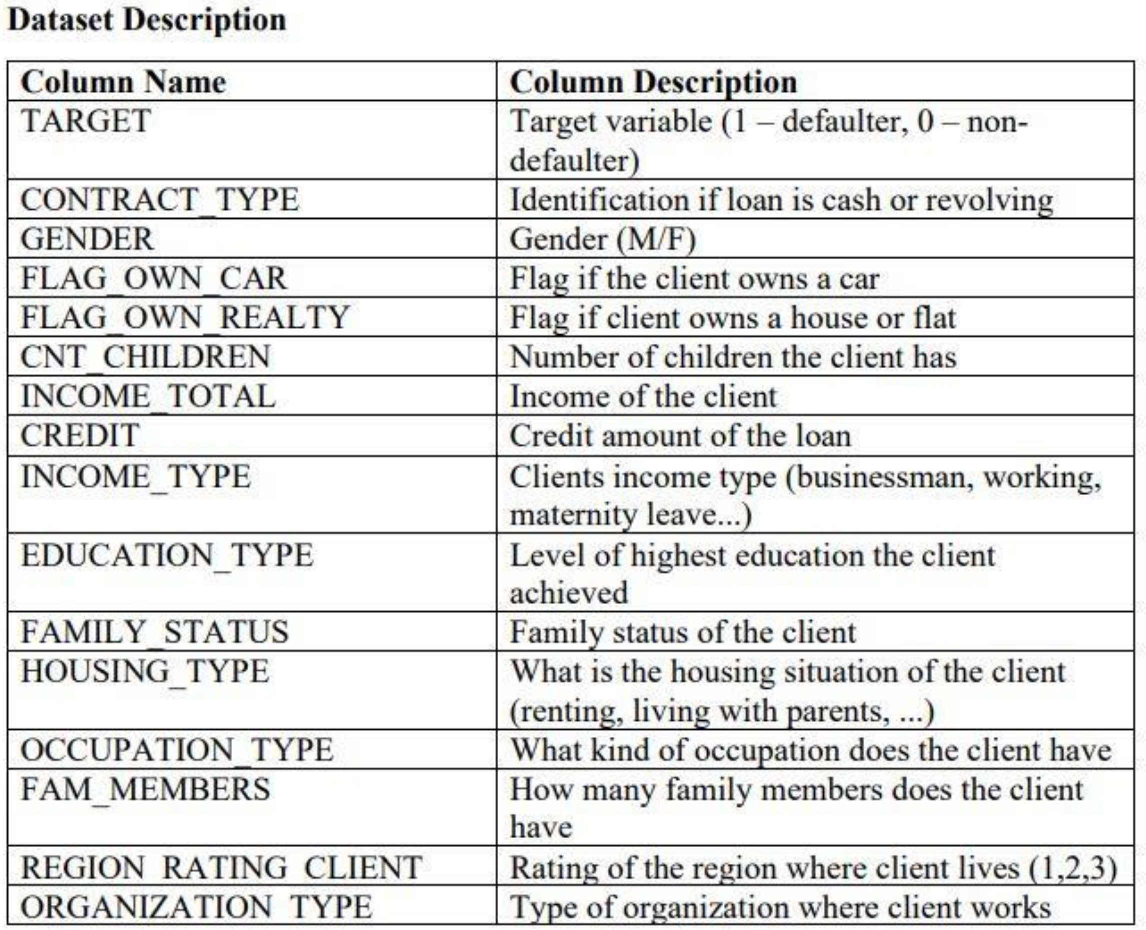
Data Mining Project Report

Credit Card Defaulter DataSet:

A dataset of credit card holders, we have to identify them as defaulters/non defaulters (loan paid or not) based on their respective attributes.

Dataset:

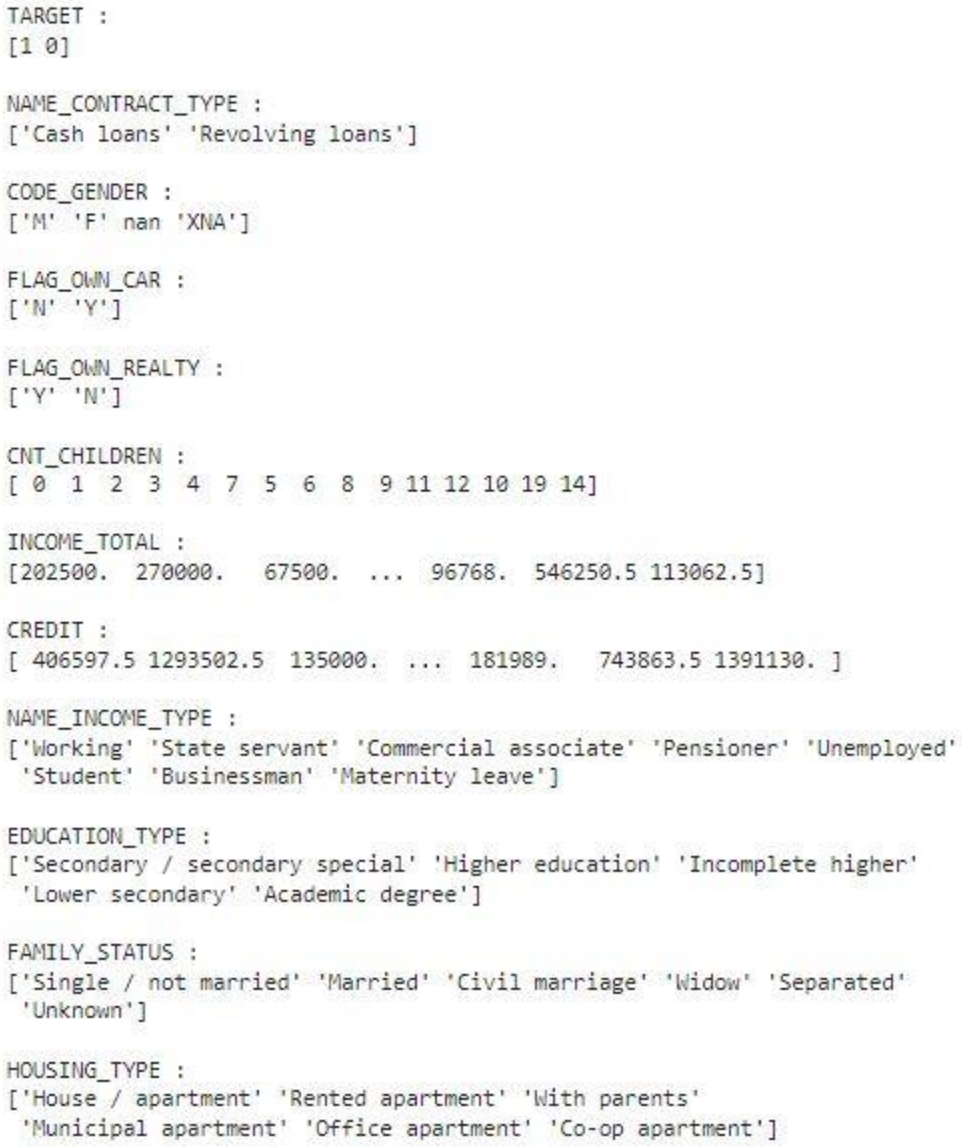
Our data has the following columns:



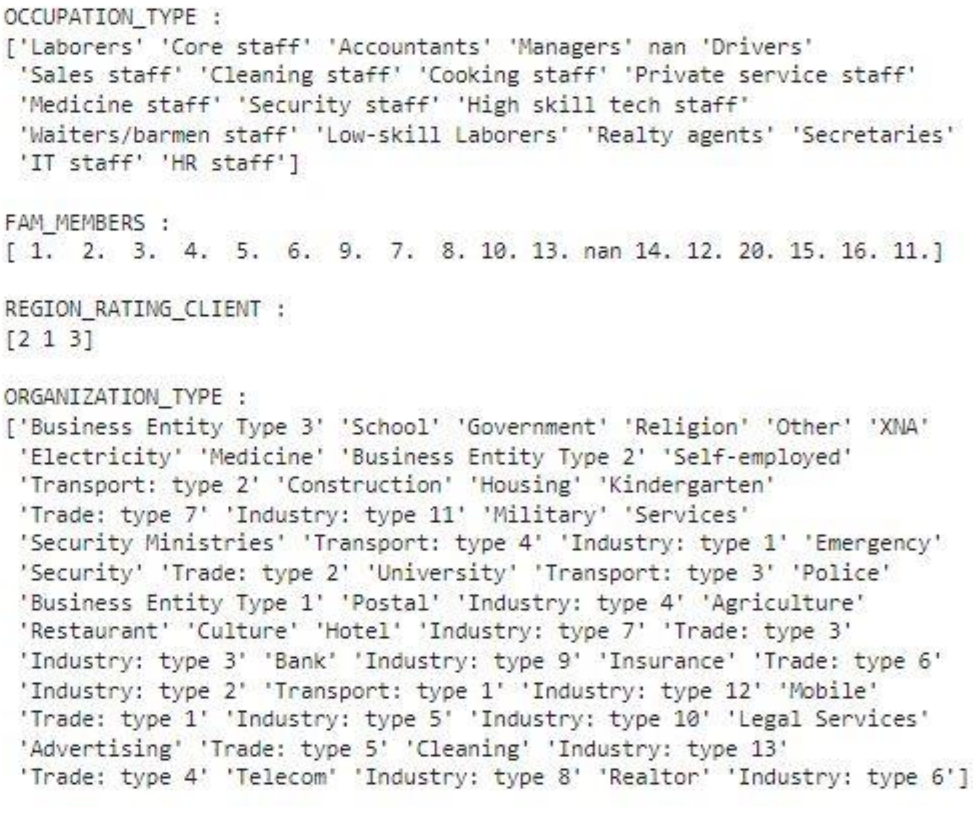
Dataset Description

Column Name	Column Description
TARGET	Target variable (1 – defaulter, 0 – non-defaulter)
CONTRACT_TYPE	Identification if loan is cash or revolving
GENDER	Gender (M/F)
FLAG_OWN_CAR	Flag if the client owns a car
FLAG_OWN_REALTY	Flag if client owns a house or flat
CNT_CHILDREN	Number of children the client has
INCOME_TOTAL	Income of the client
CREDIT	Credit amount of the loan
INCOME_TYPE	Clients income type (businessman, working, maternity leave...)
EDUCATION_TYPE	Level of highest education the client achieved
FAMILY_STATUS	Family status of the client
HOUSING_TYPE	What is the housing situation of the client (renting, living with parents, ...)
OCCUPATION_TYPE	What kind of occupation does the client have
FAM_MEMBERS	How many family members does the client have
REGION_RATING_CLIENT	Rating of the region where client lives (1,2,3)
ORGANIZATION_TYPE	Type of organization where client works

The values in each column are as followed:



```
TARGET :
[1 0]

NAME_CONTRACT_TYPE :
['Cash loans' 'Revolving loans']

CODE_GENDER :
['M' 'F' nan 'XNA']

FLAG_OWN_CAR :
['N' 'Y']

FLAG_OWN_REALTY :
['Y' 'N']

CNT_CHILDREN :
[ 0  1  2  3  4  7  5  6  8  9 11 12 10 19 14]

INCOME_TOTAL :
[202500.  270000.   67500.  ...  96768.  546250.5 113062.5]

CREDIT :
[ 406597.5 1293502.5  135000.  ...  181989.   743863.5 1391130. ]

NAME_INCOME_TYPE :
['Working' 'State servant' 'Commercial associate' 'Pensioner' 'Unemployed'
 'Student' 'Businessman' 'Maternity leave']

EDUCATION_TYPE :
['Secondary / secondary special' 'Higher education' 'Incomplete higher'
 'Lower secondary' 'Academic degree']

FAMILY_STATUS :
['Single / not married' 'Married' 'Civil marriage' 'Widow' 'Separated'
 'Unknown']

HOUSING_TYPE :
['House / apartment' 'Rented apartment' 'With parents'
 'Municipal apartment' 'Office apartment' 'Co-op apartment']
```



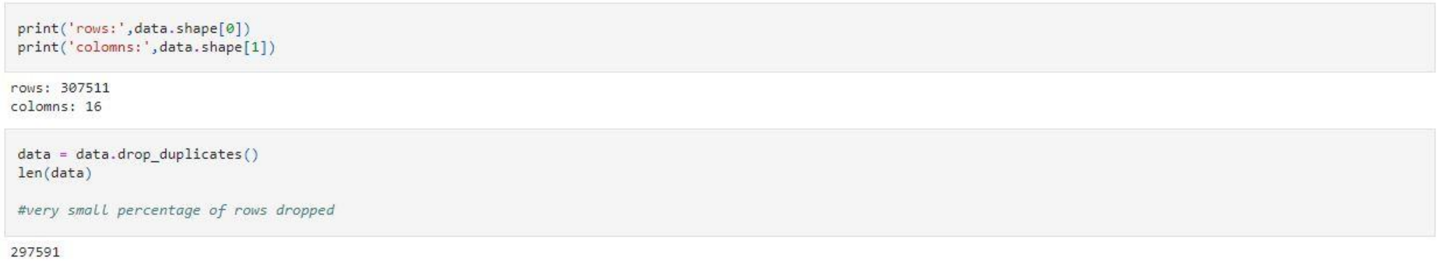
```
OCCUPATION_TYPE :
['Laborers' 'Core staff' 'Accountants' 'Managers' nan 'Drivers'
 'Sales staff' 'Cleaning staff' 'Cooking staff' 'Private service staff'
 'Medicine staff' 'Security staff' 'High skill tech staff'
 'Waiters/barmen staff' 'Low-skill Laborers' 'Realty agents' 'Secretaries'
 'IT staff' 'HR staff']

FAM_MEMBERS :
[ 1.  2.  3.  4.  5.  6.  9.  7.  8. 10. 13. nan 14. 12. 20. 15. 16. 11.]

REGION_RATING_CLIENT :
[2 1 3]

ORGANIZATION_TYPE :
['Business Entity Type 3' 'School' 'Government' 'Religion' 'Other' 'XNA'
 'Electricity' 'Medicine' 'Business Entity Type 2' 'Self-employed'
 'Transport: type 2' 'Construction' 'Housing' 'Kindergarten'
 'Trade: type 7' 'Industry: type 11' 'Military' 'Services'
 'Security Ministries' 'Transport: type 4' 'Industry: type 1' 'Emergency'
 'Security' 'Trade: type 2' 'University' 'Transport: type 3' 'Police'
 'Business Entity Type 1' 'Postal' 'Industry: type 4' 'Agriculture'
 'Restaurant' 'Culture' 'Hotel' 'Industry: type 7' 'Trade: type 3'
 'Industry: type 3' 'Bank' 'Industry: type 9' 'Insurance' 'Trade: type 6'
 'Industry: type 2' 'Transport: type 1' 'Industry: type 12' 'Mobile'
 'Trade: type 1' 'Industry: type 5' 'Industry: type 10' 'Legal Services'
 'Advertising' 'Trade: type 5' 'Cleaning' 'Industry: type 13'
 'Trade: type 4' 'Telecom' 'Industry: type 8' 'Realtor' 'Industry: type 6']
```

Data Cleaning:

Before we can derive meaningful insights and inferences from our data, we need to validate its correctness and ensure that it is in a standardized and usable format. Thus, we will divide the cleaning process into various stages before moving to the analysis. So, let’s charge into it. We will take things one at a time. Each pre-processing part is divided into a different section as shown below.

Dropping Duplicates:

Duplicate entries are problematic for multiple reasons. First off, when an entry appears more than once, it receives a disproportionate weight during training. Additionally, duplicate entries can ruin the split between train, validation and test sets in cases where identical entries are not all in the same set. This can lead to biased performance estimates that will lead to disappointing models in production.



```
print('rows:',data.shape[0])
print('colomns:',data.shape[1])
```

```
rows: 307511
colomns: 16
```

```
data = data.drop_duplicates()
len(data)

#very small percentage of rows dropped
```

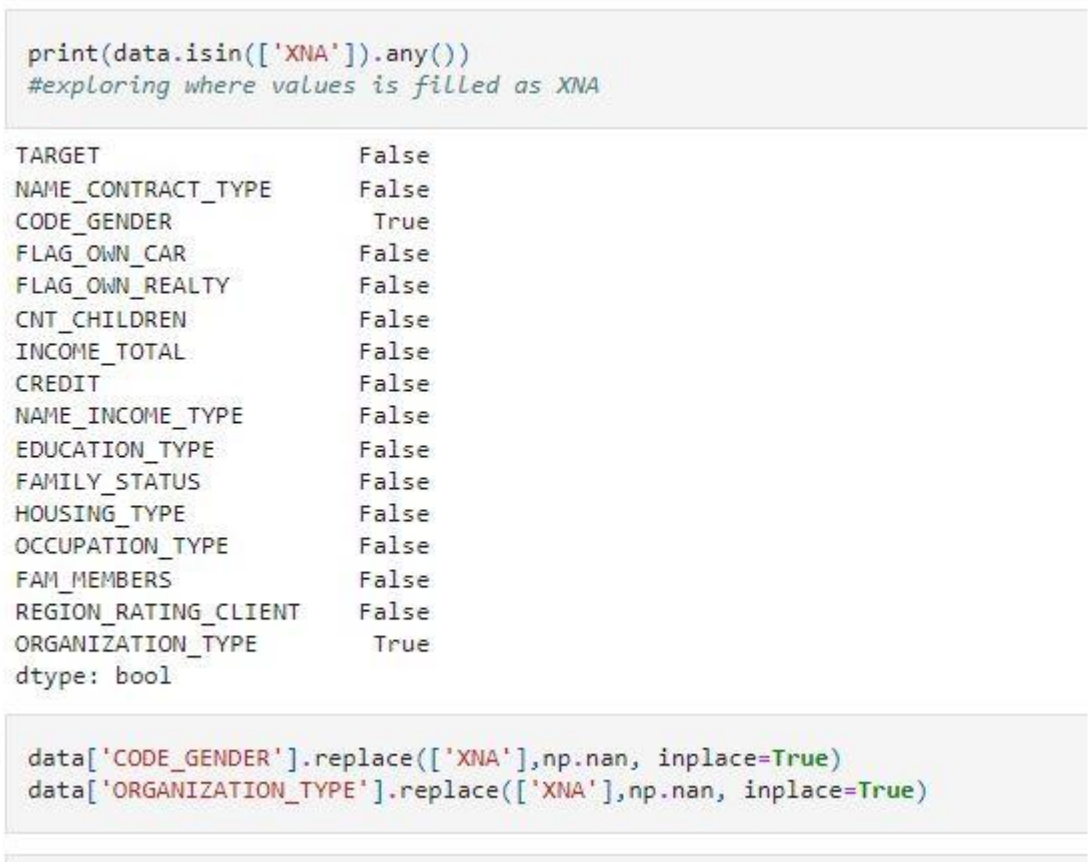
```
297591
```

So in our data a very small number of data was repeated our rows went from 307511 to 297591.

Finding Null Values:

Lets see if null values exist in our data

Some Null data in our file was filled as ‘XNA’ so we replaced the with null and checked the null values of our data

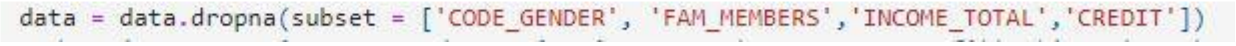


```
print(data.isin(['XNA']).any())
#exploring where values is filled as XNA
```

```
TARGET                  False
NAME_CONTRACT_TYPE      False
CODE_GENDER              True
FLAG_OWN_CAR            False
FLAG_OWN_REALTY         False
CNT_CHILDREN            False
INCOME_TOTAL            False
CREDIT                  False
NAME_INCOME_TYPE        False
EDUCATION_TYPE          False
FAMILY_STATUS           False
HOUSING_TYPE            False
OCCUPATION_TYPE         False
FAM_MEMBERS             False
REGION_RATING_CLIENT    False
ORGANIZATION_TYPE        True
dtype: bool
```

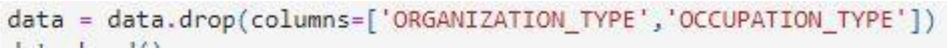
```
data['CODE_GENDER'].replace(['XNA'],np.nan, inplace=True)
data['ORGANIZATION_TYPE'].replace(['XNA'],np.nan, inplace=True)
```

We decided to drop null values in CODE_GENDER INCOME_TOTAL CREDIT FAM_MEMBERS as very few values compared to size of dataset



```
data = data.dropna(subset = ['CODE_GENDER', 'FAM_MEMBERS','INCOME_TOTAL','CREDIT'])
```

dropped Occupation type and Organization type column as cannot fill this column because alot of nan values dataset will skew if mode used, and as values come from large.



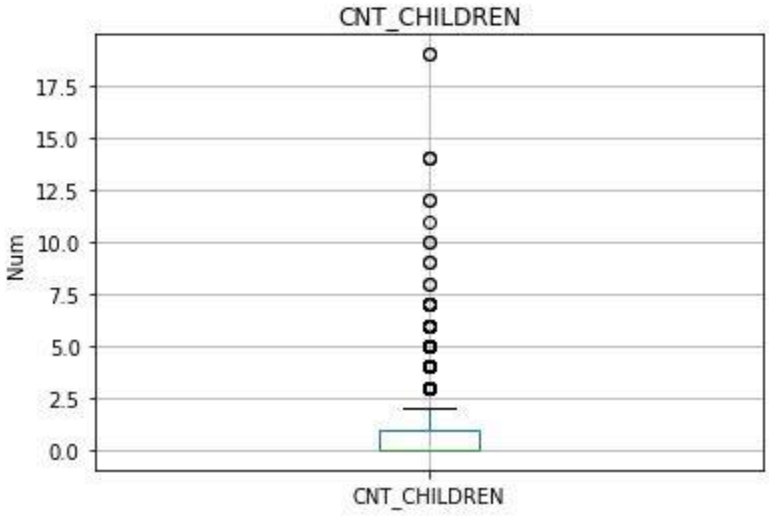
```
data = data.drop(columns=['ORGANIZATION_TYPE','OCCUPATION_TYPE'])
```

Data Visualization:

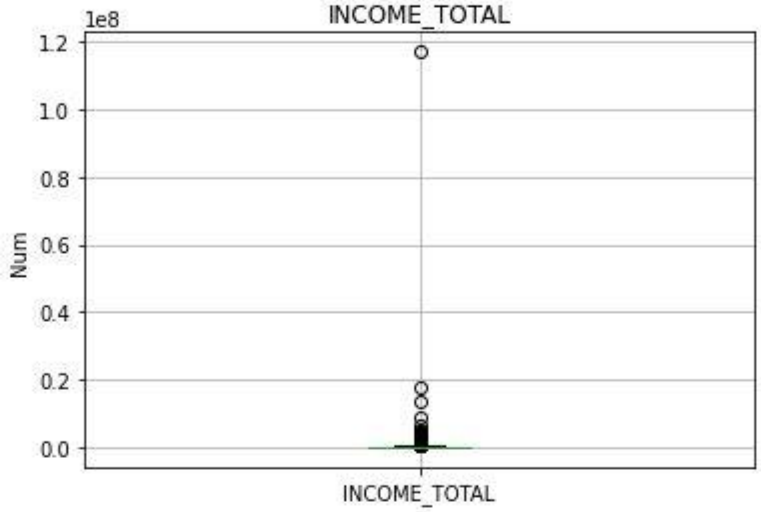
Data visualization helps to tell stories by curating data into a form easier to understand, highlighting the trends and outliers.

Box Plots:

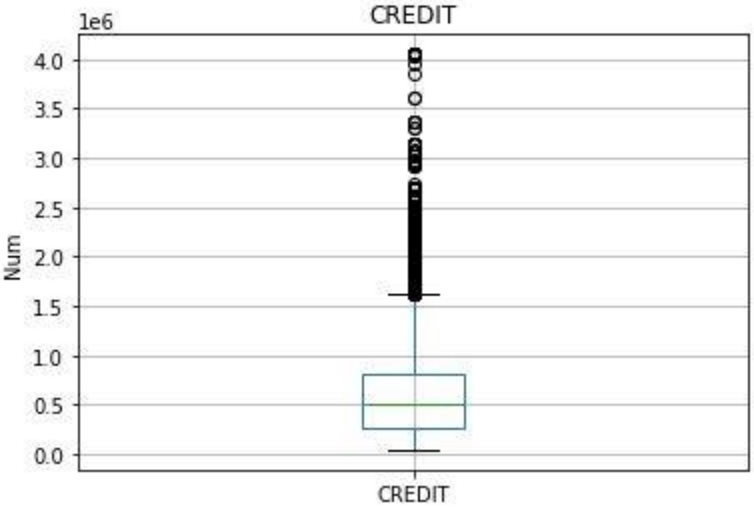
box plot is a method for graphically depicting groups of numerical data through their quartiles. The box extends from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of box to show the range of the data. We have extended whisker 1.5*IQR



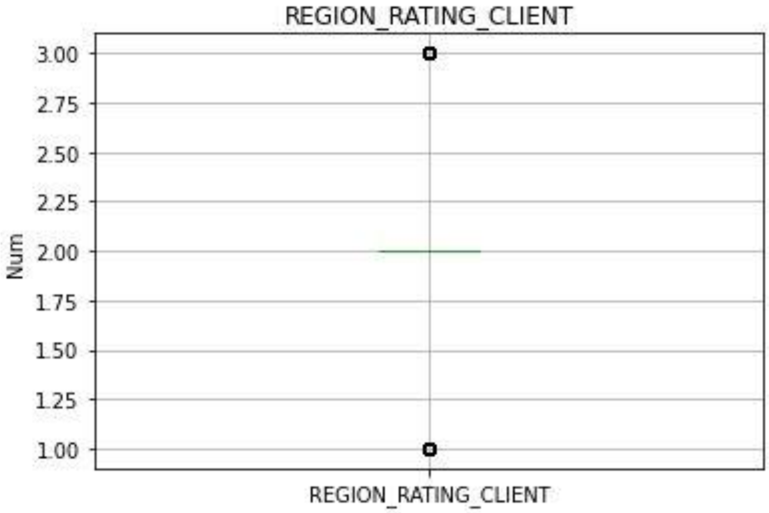
The graph shows that generally individuals in our data set have 2 children, but there are some people who lie outside our whiskers.



Their lies a huge outlier in ourdata set who could potentially damage our analysis. It would be recommended to remove this person from our dataset



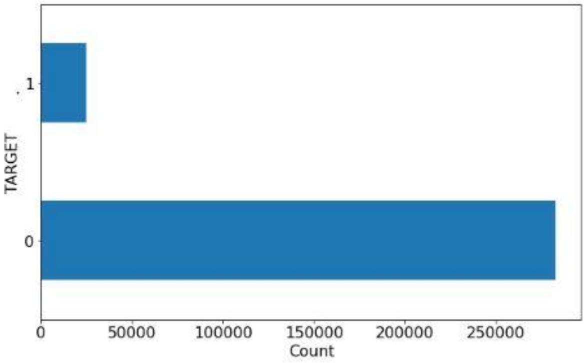
Alot of people have credit outside our whisker range

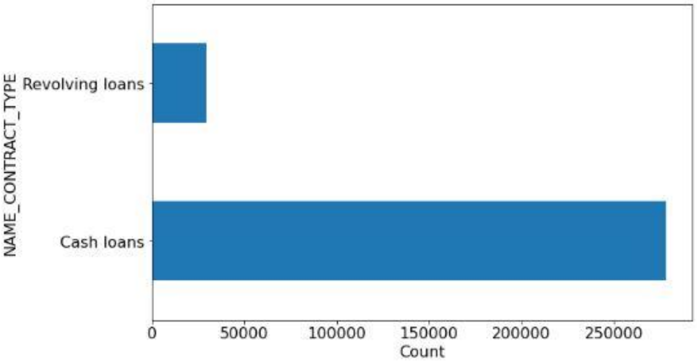


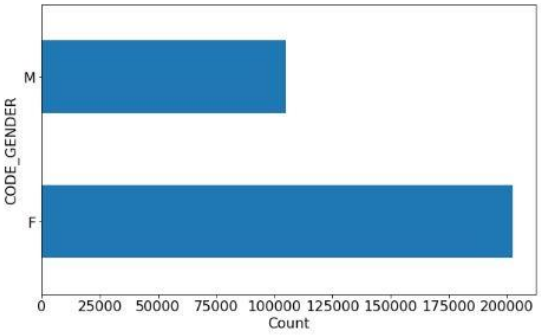
The Region rating has nearly everybody on 2 rated region, just 2 outliers one in 1 rated region one in 3 rated region.

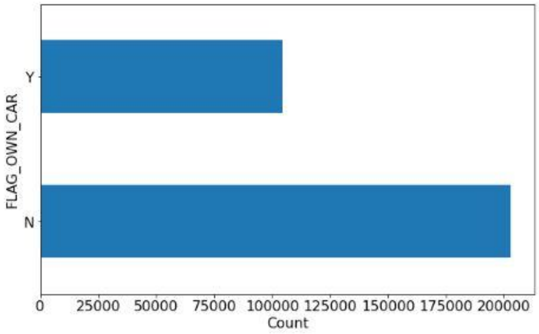
Bar Graphs:

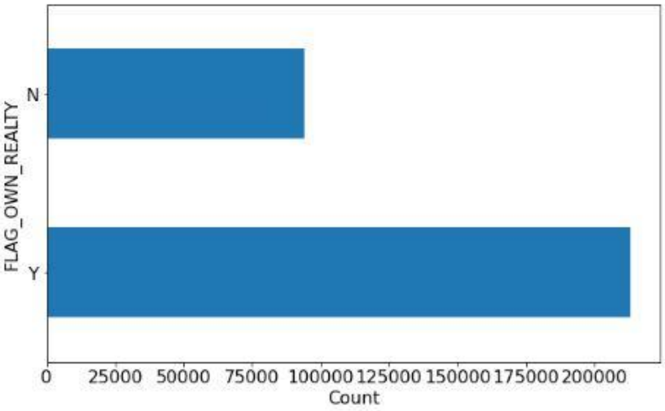
We used Bar graphs to check the disparity of counts in binary features.


TARGET
1
0
0
50000
100000
150000
200000
250000
Count


NAME_CONTRACT_TYPE
Revolving loans
Cash loans
0
50000
100000
150000
200000
250000
Count

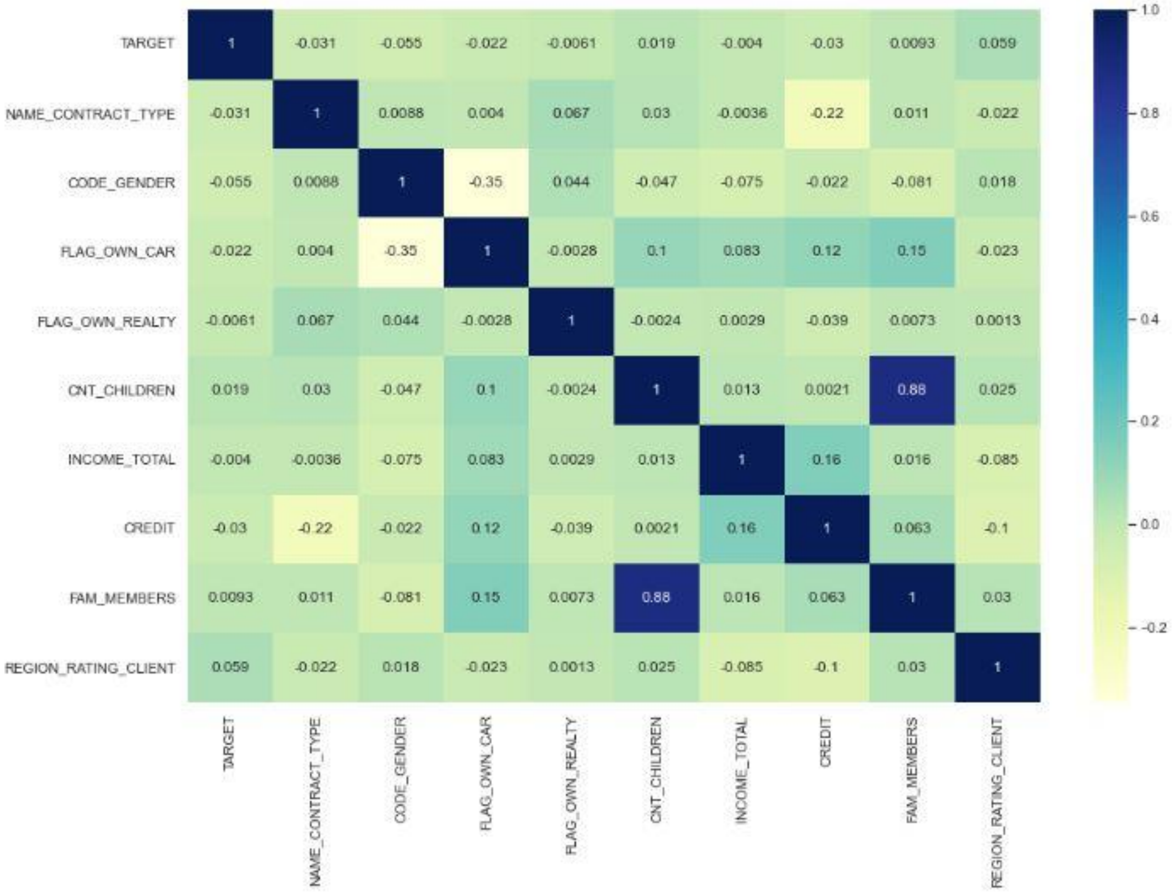

CODE_GENDER
M
F
0
25000
50000
75000
100000
125000
150000
175000
200000
Count


FLAG_OWN_CAR
Y
N
0
25000
50000
75000
100000
125000
150000
175000
200000
Count



These graphs show which binary features have a majority in our data set. And in case of Cash Loans to Revolving Loans and Non-Defaulters to Defaulters we have got an Overwhelming Majority.

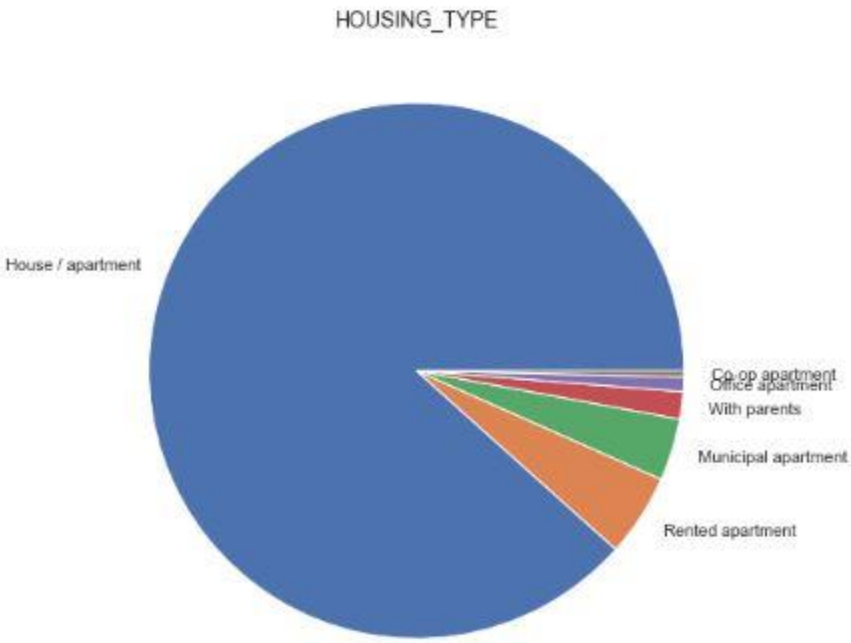
Correlation HeatMap:

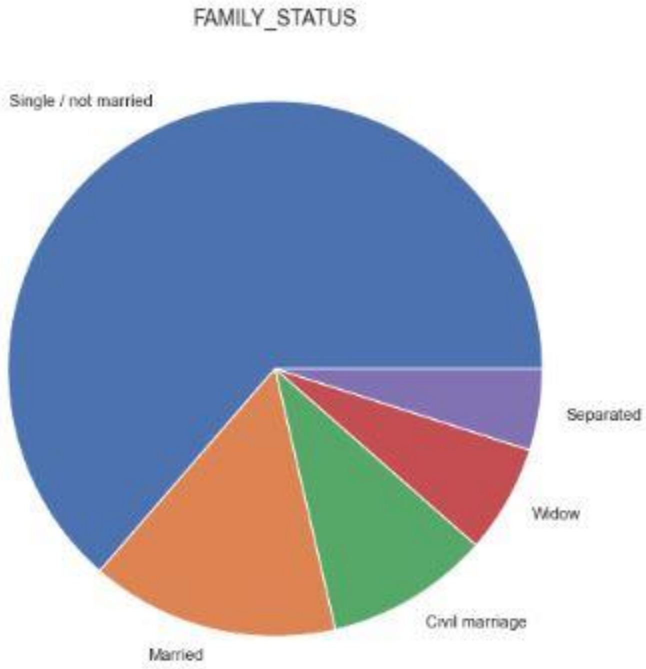


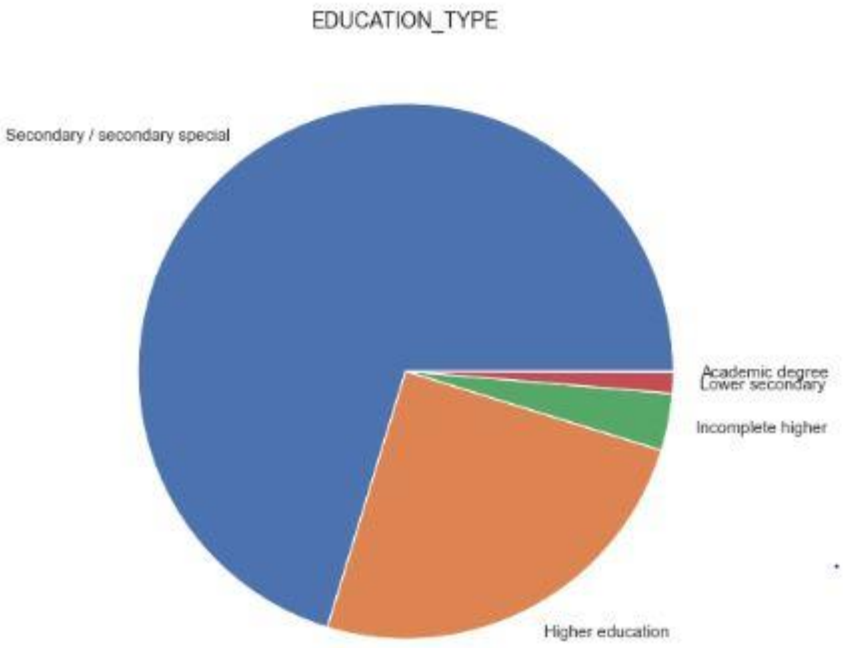
Our Heat map does not give us much Relevant data, that could be helped in making decisions about the data. The biggest correlation is between CNT_CHILDREN and FAM_MEMBERS the reason behind which is easy to see. These are redundant features and one of them could be removed.
Except that there are not any major correlations.

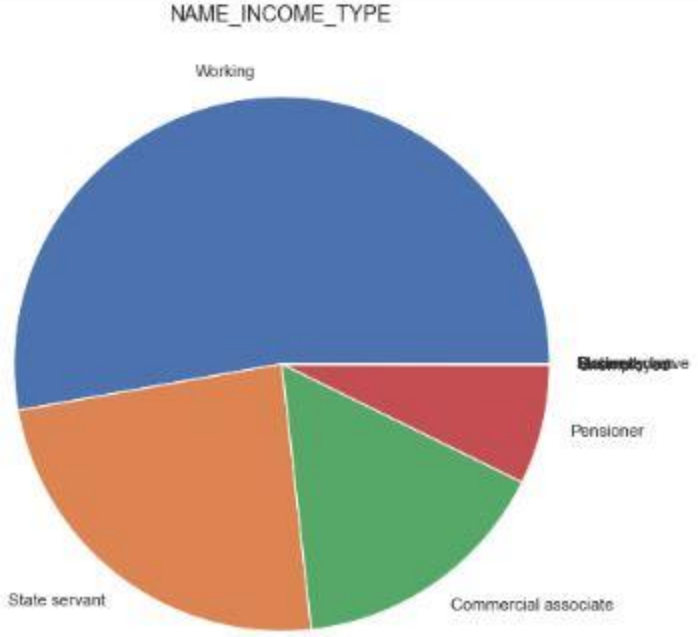
Pie Charts:

The pie charts have been used to see the counts of non binary features. Which value in the features dominates the data set.


HOUSING_TYPE
House / apartment
Co-op apartment
Office apartment
With parents
Municipal apartment
Rented apartment

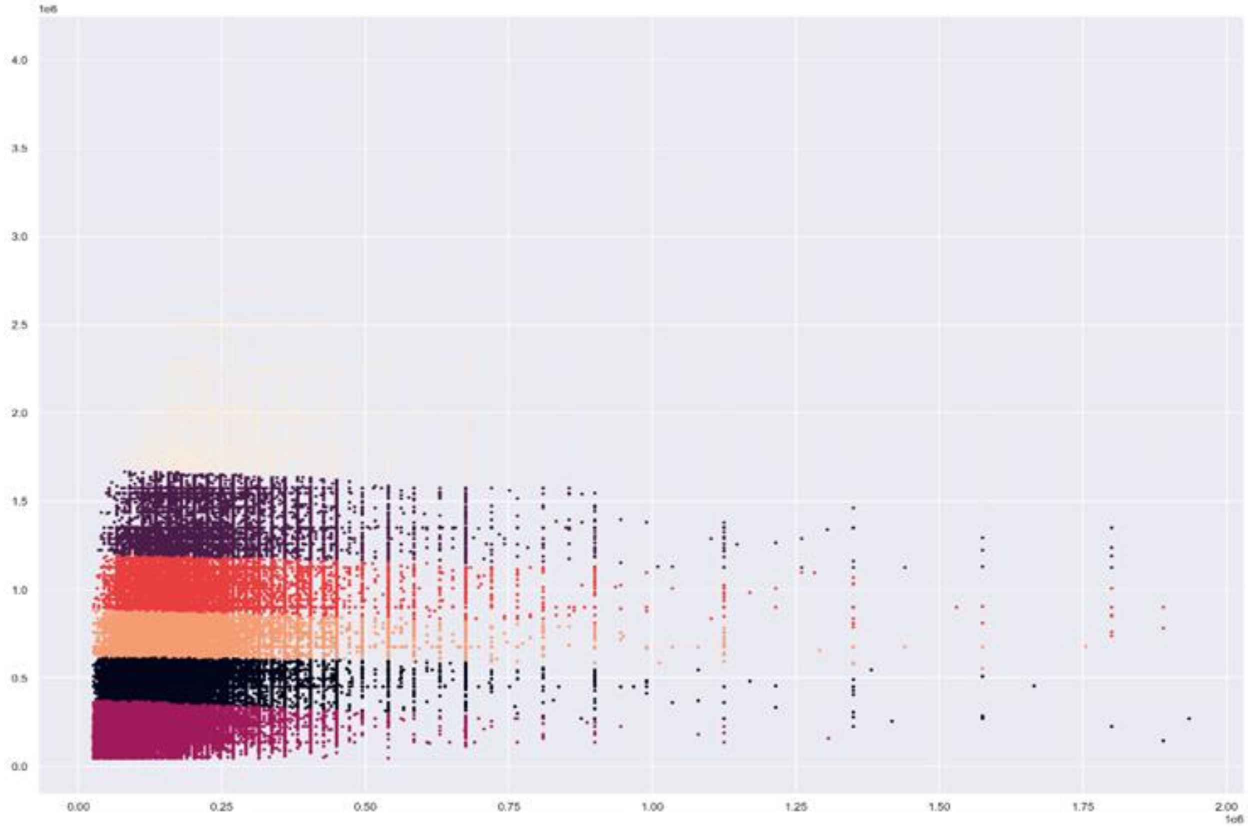

FAMILY_STATUS
Single / not married
Separated
Widow
Civil marriage
Married


EDUCATION_TYPE
Secondary / secondary special
Academic degree
Lower secondary
Incomplete higher
Higher education

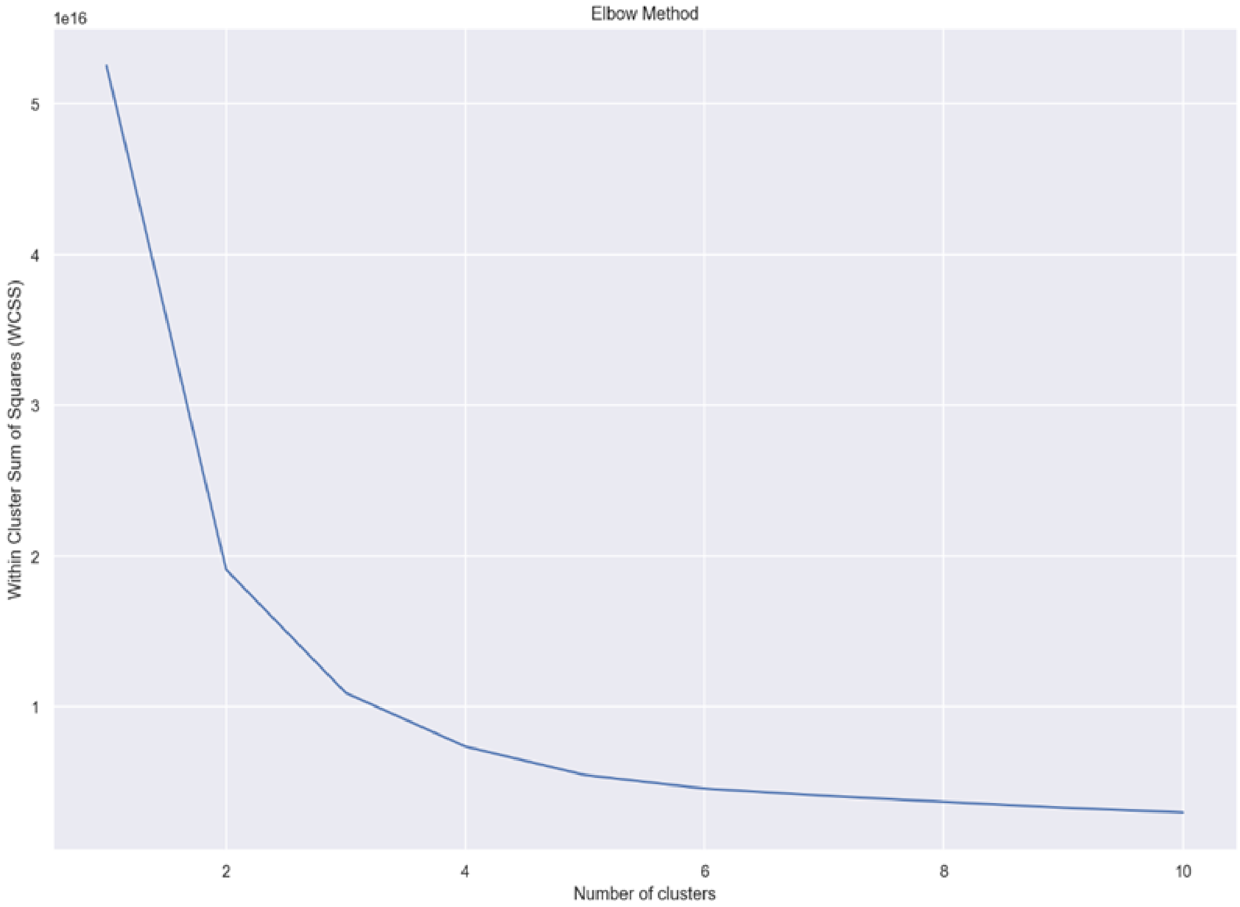

NAME_INCOME_TYPE
Working
Pensioner
State servant
Commercial associate

Clustering

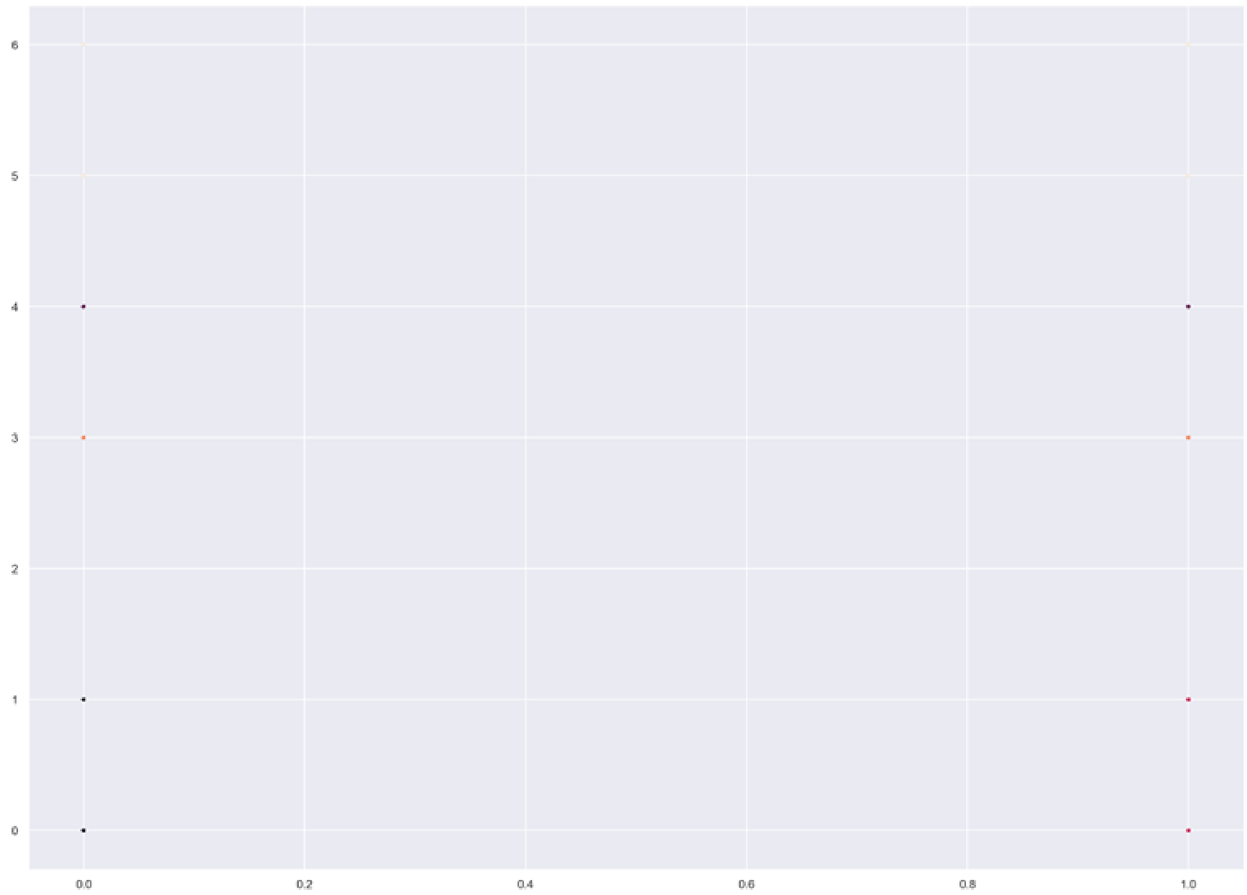
Income Total VS Credit Clustering:



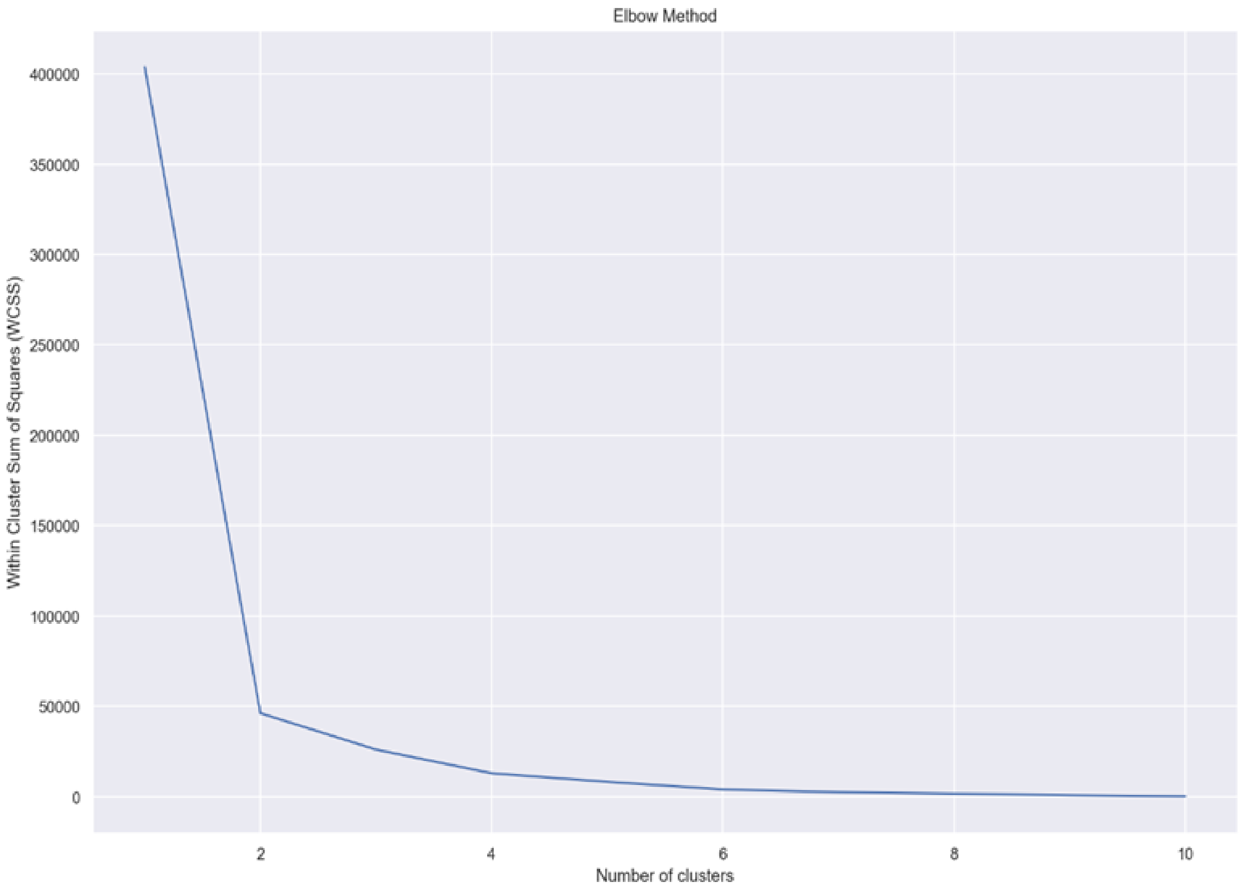
Elbow method used to find out the number of clusters



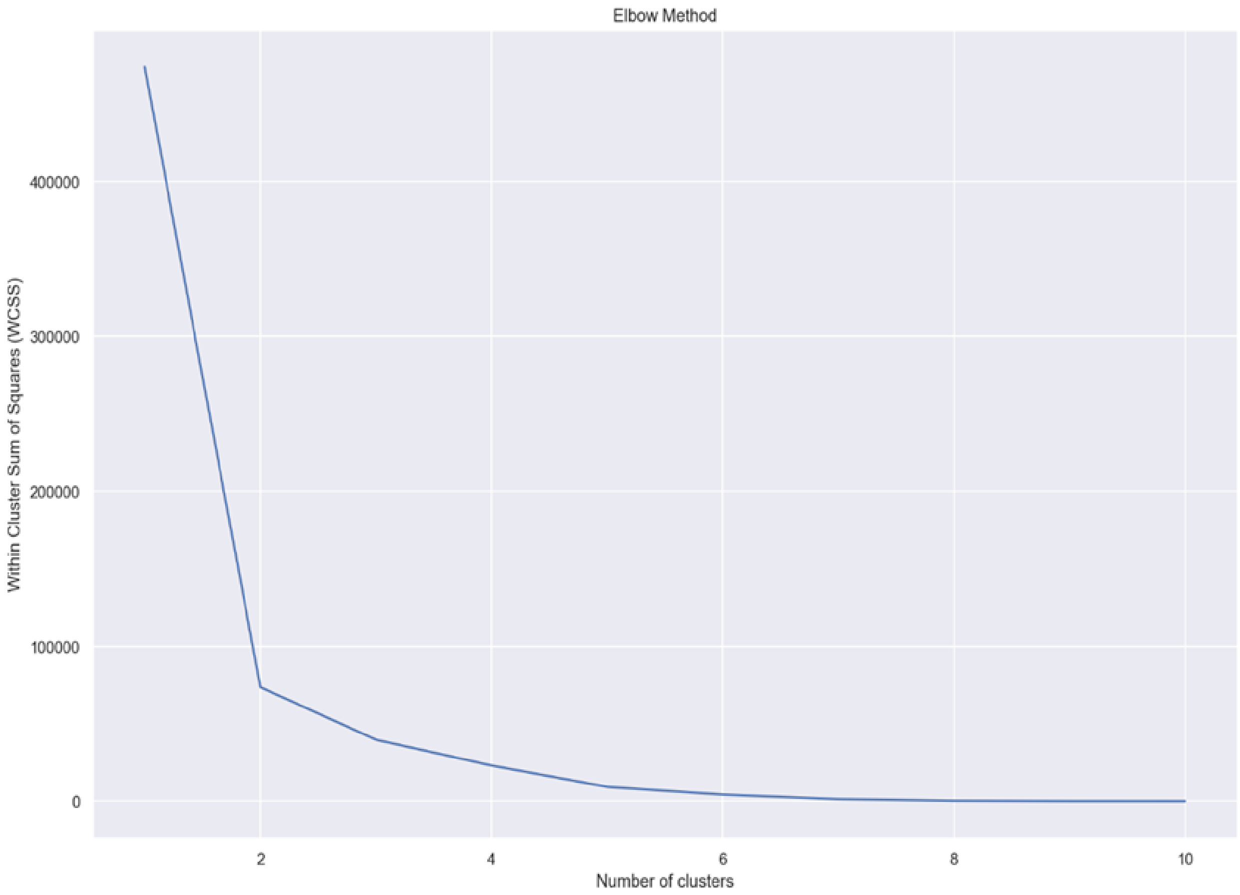
Target VS Housing type Clustering:

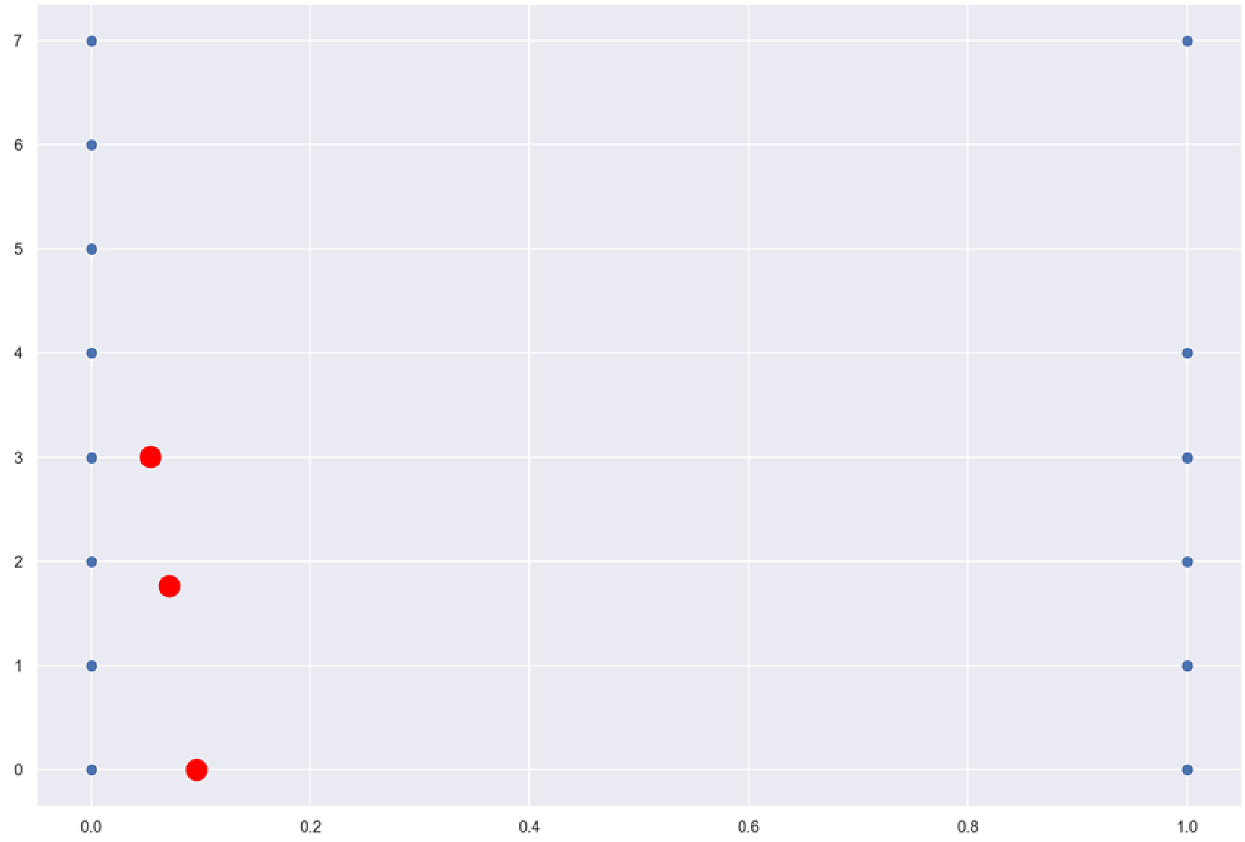


6
5
4
3
2
1
0
0.0
0.2
0.4
0.6
0.8
1.0



Target VS Name Income Type:

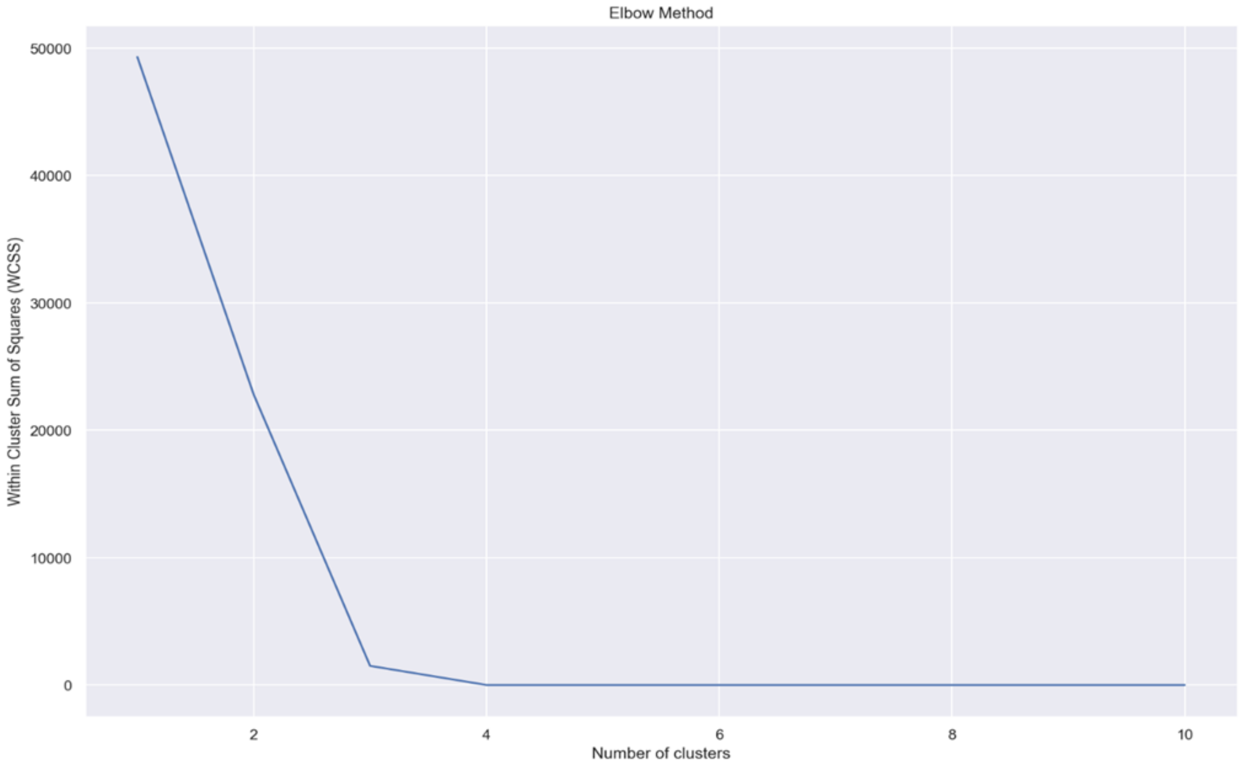

Elbow Method
Within Cluster Sum of Squares (WCSS)
400000
300000
200000
100000
0
2
4
6
8
10
Number of clusters

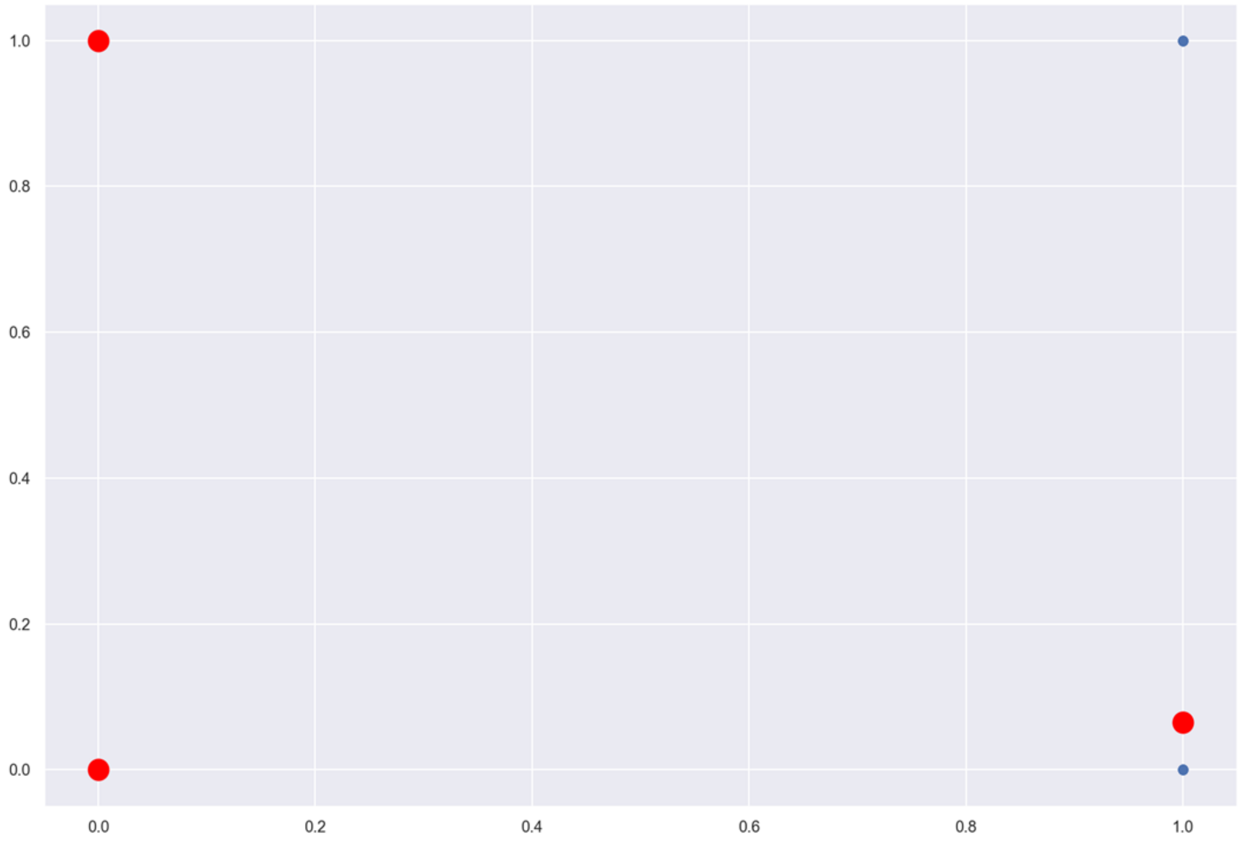


Explanation of the graph:

First of all we converted the non-numeric categorical and we convert them numeric values through replace method of the python. ['Working', 'State servant', 'Commercial associate', 'Pensioner', 'Unemployed', 'Student' , 'Businessman' , 'Maternity leave'],[0, 1 , 2 , 3 , 4 , 5 , 6 , 7]. These were the conversions we used. Also we used Elbow method to calculate the number of cluster beforehand .so from the graphs we can see that the clusters are formed around 0 2 and 3 there are three cluster in this relation and we can confirm that the non-defaulter are mostly form the Income type of “Working”, ‘commercial’ and “pensioner”

Target VS Name Contract Type:

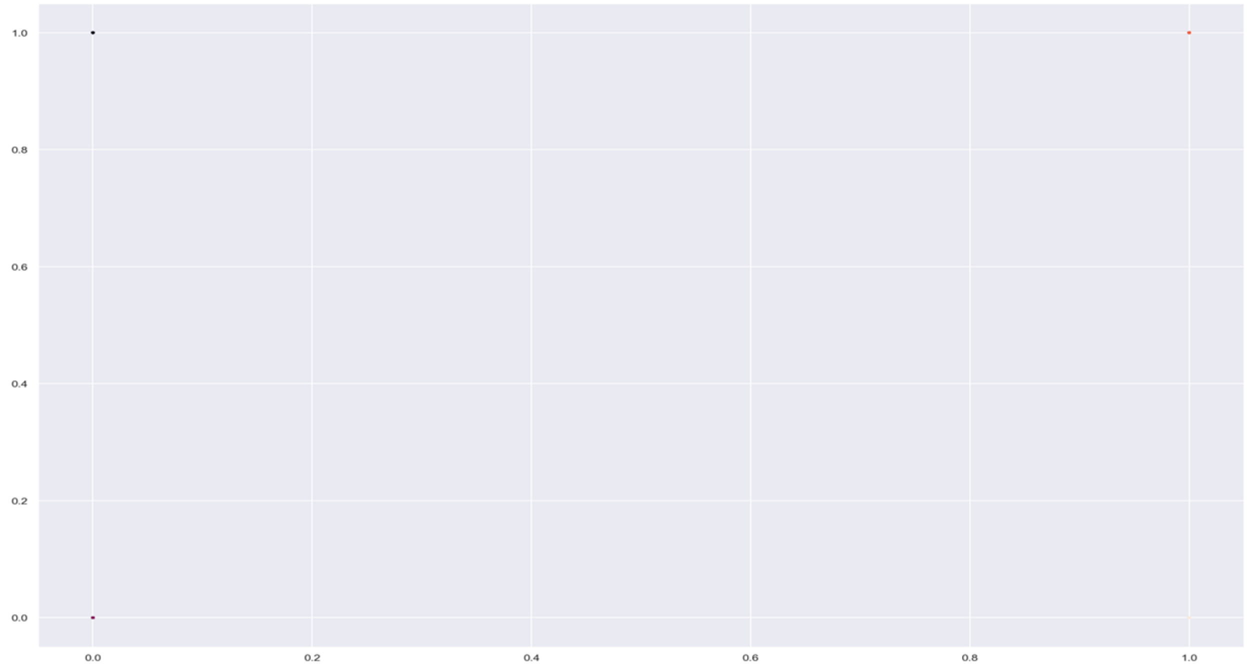

Elbow Method
Within Cluster Sum of Squares (WCSS)
50000
40000
30000
20000
10000
0
2
4
6
8
10
Number of clusters



Explanation of the graph:

We performed this clustering to find out the clustering relationship between our target variable that whether a person is defaulter or not and its relationship with variable contract type which tells us about the contract type of the person taking loan. Either it is cash loans or resolving loans. First of all we converted the non-numeric categorical and we convert them numeric values through replace method of the python. Replace(['Cash loans', 'Revolving loans'],[0, 1]. These were the conversions we used. Also we used Elbow method to calculate the number of cluster beforehand .so from the graphs we can see that the clusters are formed around 0,0 1,0 and 0,1 there are three cluster in this relation and we can confirm that the non-defaulter are mostly form the both contract type of “Cash Loans” and “resolving loans” but the defaulters are mostly clustered around the “resolving loans” field.

Target vs own_realty:



Explanation of the graph:

We performed this clustering to find out the clustering relationship between our target variable that whether a person is defaulter or not and its relationship with variable Flag_own_reality which tells us about real estate information of the person taking loan. Either it is cash loans or resolving loans. First of all we converted the non-numeric categorical and we convert them numeric values through replace method of the python. Replace(['Y', 'N'],[0, 1]. These were the conversions we used. Also we used Elbow method to calculate the number of cluster beforehand .so from the graphs we can see that the clusters are formed around 0,0 1,0 and 1,1 there are three cluster in this relation and we can confirm that the non-defaulter are mostly form the both “Y” and “N” but the defaulters are mostly clustered around the “Y” field. Which means most of the defaulters have real estates.

Frequent pattern mining:

We ran apriori on Defaulters and Non Defaulter separators to find common patterns in them.

Apriori on Defaulters

```
----Min Support---- 0.6

----Min Support---- 0.6
frozenset({'FC:N', 'Cash loans'})
frozenset({'Cash loans', 'FR:Y'})
frozenset({'Cash loans', 'House / apartment'})
frozenset({'Secondary / secondary special', 'Cash loans'})
frozenset({'FR:Y', 'House / apartment'})
frozenset({'Secondary / secondary special', 'House / apartment'})
frozenset({'Secondary / secondary special', 'Cash loans', 'House / apartment'})
----Min Support---- 0.7
frozenset({'Cash loans', 'House / apartment'})
frozenset({'Secondary / secondary special', 'Cash loans'})
----Min Support---- 0.8
frozenset({'Cash loans', 'House / apartment'})
```

Apriori on Non Defaulters:

```
----Min Support---- 0.6
frozenset({'Cash loans', 'FR:Y'})
frozenset({'Cash loans', 'House / apartment'})
frozenset({'Secondary / secondary special', 'Cash loans'})
frozenset({'FR:Y', 'House / apartment'})
frozenset({'Secondary / secondary special', 'House / apartment'})
----Min Support---- 0.7
frozenset({'Cash loans', 'House / apartment'})
----Min Support---- 0.8
frozenset({'Cash loans', 'House / apartment'})
```

The following sets are the only 2 different minimum support sets between defaulters and non defaulters.
{'Secondary / secondary special', 'Cash loans', 'House / apartment'} at Min Support 0.6
({'Secondary / secondary special', 'Cash loans'} at Min Support 0.7

So these patterns in a person's profile are more likely to make him defaulter.

Recommendations:

From our results of frequent pattern mining and clustering we can conclude that the people with income type working and own real estate have and having cash loans are most likely to be defaulters.

From our APriori we can conclude people having the following attributes together in their data are more likely to be defaulters.
{'Secondary / secondary special', 'Cash loans', 'House / apartment'}
({'Secondary / secondary special', 'Cash loans'}

So while given condition the banks can look for these attributes to figure out which type of person might become defaulter.