Credit EDA

Loan Lending

Anviksha Singh
anviksha.singh0110s@gmail.com

Data Understanding

A. Load the application and previous application file

Loading the files provided in the data frames using the traditional pandas method below :

```
import warnings
warnings.filterwarnings("ignore")

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

```
app=pd.read_csv("application_data.csv")
app.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN
0	100002	1	Cash loans	M	N	Y	0
1	100003	0	Cash loans	F	N	N	0
2	100004	0	Revolving loans	M	Y	Y	0
3	100006	0	Cash loans	F	N	Y	0
4	100007	0	Cash loans	M	N	Y	0

5 rows × 122 columns

Data Understanding

B. Understanding the Application File

The Application file is stored in 'app' data frame and then we further use the pre defined functions.

```
app.shape
```

```
(307511, 122)
```

```
app.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
```

```
app.describe()
```

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05
max	456255.000000	1.000000	19.000000	1.170000e+08	4.050000e+06

8 rows × 106 columns

Data Understanding

C. Understanding the Previous Application File

The Application file is stored in 'papp' data frame and then we further use the pre defined functions.

```
papp.shape
```

```
(1670214, 37)
```

```
papp.describe()
```

	SK_ID_PREV	SK_ID_CURR	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT
count	1.670214e+06	1.670214e+06	1.297979e+06	1.670214e+06	1.670213e+06
mean	1.923089e+06	2.783572e+05	1.595512e+04	1.752339e+05	1.961140e+05
std	5.325980e+05	1.028148e+05	1.478214e+04	2.927798e+05	3.185746e+05
min	1.000001e+06	1.000010e+05	0.000000e+00	0.000000e+00	0.000000e+00
25%	1.461857e+06	1.893290e+05	6.321780e+03	1.872000e+04	2.416050e+04
50%	1.923110e+06	2.787145e+05	1.125000e+04	7.104600e+04	8.054100e+04
75%	2.384280e+06	3.675140e+05	2.065842e+04	1.803600e+05	2.164185e+05
max	2.845382e+06	4.562550e+05	4.180581e+05	6.905160e+06	6.905160e+06

8 rows × 21 columns

```
papp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 37 columns):
 #   Column                      Non-Null Count    Dtype
---  ------                      --------------    -----
 0   SK_ID_PREV                  1670214 non-null  int64
 1   SK_ID_CURR                  1670214 non-null  int64
 2   NAME_CONTRACT_TYPE          1670214 non-null  object
 3   AMT_ANNUITY                 1297979 non-null  float64
 4   AMT_APPLICATION             1670214 non-null  float64
```

Data Understanding

D. Understanding Column Structure & Values

- Understanding Column Structure & Values contained in Columns of Application File
- Storing the Column which have nulls greater than 45% in a list to delete column dynamically from the data frame

```
((app.isnull().sum()/app.shape[0])*100)>45
```

```
SK_ID_CURR                    False
TARGET                        False
NAME_CONTRACT_TYPE            False
CODE_GENDER                   False
FLAG_OWN_CAR                  False
                              ...
AMT_REQ_CREDIT_BUREAU_DAY     False
AMT_REQ_CREDIT_BUREAU_WEEK    False
AMT_REQ_CREDIT_BUREAU_MON     False
AMT_REQ_CREDIT_BUREAU_QRT     False
AMT_REQ_CREDIT_BUREAU_YEAR    False
Length: 122, dtype: bool
```

```
colNullMore50=app.columns[((app.isnull().sum()/app.shape[0])*100)>45]
```

```
delcol=colNullMore50.tolist()
```

```
type(delcol)
```

```
list
```

```
len(delcol)
```

```
49
```

Data Understanding

D. Understanding Column Structure & Values

- Understanding Column Structure & Values contained in Columns of Previous Application File
- Storing the Column which have nulls greater than 50% in a list to delete column dynamically from the data frame

```
colNullMore50_prev=papp.columns[((papp.isnull().sum()/papp.shape[0])*100)>50]
delcol_prev=colNullMore50_prev.tolist()
delcol_prev
```

```
['AMT_DOWN_PAYMENT',
 'RATE_DOWN_PAYMENT',
 'RATE_INTEREST_PRIMARY',
 'RATE_INTEREST_PRIVILEGED']
```

```
len(delcol_prev)
```

```
4
```

```
colOnlyInApp=[]
colCommonAppPre=[]
```

```
for col in app.columns:
    if col not in papp.columns:
        colOnlyInApp.append(col)
    else:
        colCommonAppPre.append(col)
```

```
len(colOnlyInApp)
len(colCommonAppPre)
colCommonAppPre
```

```
['SK_ID_CURR',
 'NAME_CONTRACT_TYPE',
 'AMT_CREDIT',
 'AMT_ANNUITY',
 'AMT_GOODS_PRICE',
 'NAME_TYPE_SUITE',
 'WEEKDAY_APPR_PROCESS_START',
 'HOUR_APPR_PROCESS_START']
```

Data Understanding

D. Understanding Column Structure & Values

Handling Nulls in Application File – There were 49 Columns having null values greater than 45%. Thus we plan to drop them after relevant analysis

```
nullValues=app[app.columns[((app.isnull().sum()/app.shape[0])*100)>45]]
nullValues.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 49 columns):
 #   Column                        Non-Null Count   Dtype
---  ------                        --------------   -----
 0   OWN_CAR_AGE                   104582 non-null  float64
 1   EXT_SOURCE_1                  134133 non-null  float64
 2   APARTMENTS_AVG                151450 non-null  float64
 3   BASEMENTAREA_AVG              127568 non-null  float64
 4   YEARS_BEGINEXPLUATATION_AVG   157504 non-null  float64
 5   YEARS_BUILD_AVG               103023 non-null  float64
 6   COMMONAREA_AVG                92646 non-null   float64
 7   ELEVATORS_AVG                 143620 non-null  float64
 8   ENTRANCES_AVG                 152683 non-null  float64
 9   FLOORSMAX_AVG                 154491 non-null  float64
 10  FLOORSMIN_AVG                 98869 non-null   float64
 11  LANDAREA_AVG                  124921 non-null  float64
 12  LIVINGAPARTMENTS_AVG          97312 non-null   float64
 13  LIVINGAREA_AVG                153161 non-null  float64
 14  NONLIVINGAPARTMENTS_AVG       93997 non-null   float64
 15  NONLIVINGAREA_AVG             137829 non-null  float64
 16  APARTMENTS_MODE               151450 non-null  float64
```

Data Understanding

D. Understanding Column Structure & Values

```
colNullMore50=app.columns[((app.isnull().sum()/app.shape[0])*100)>45]
```

```
delcol=colNullMore50.tolist()
len(delcol)
```

```
49
```

```
delcol
```

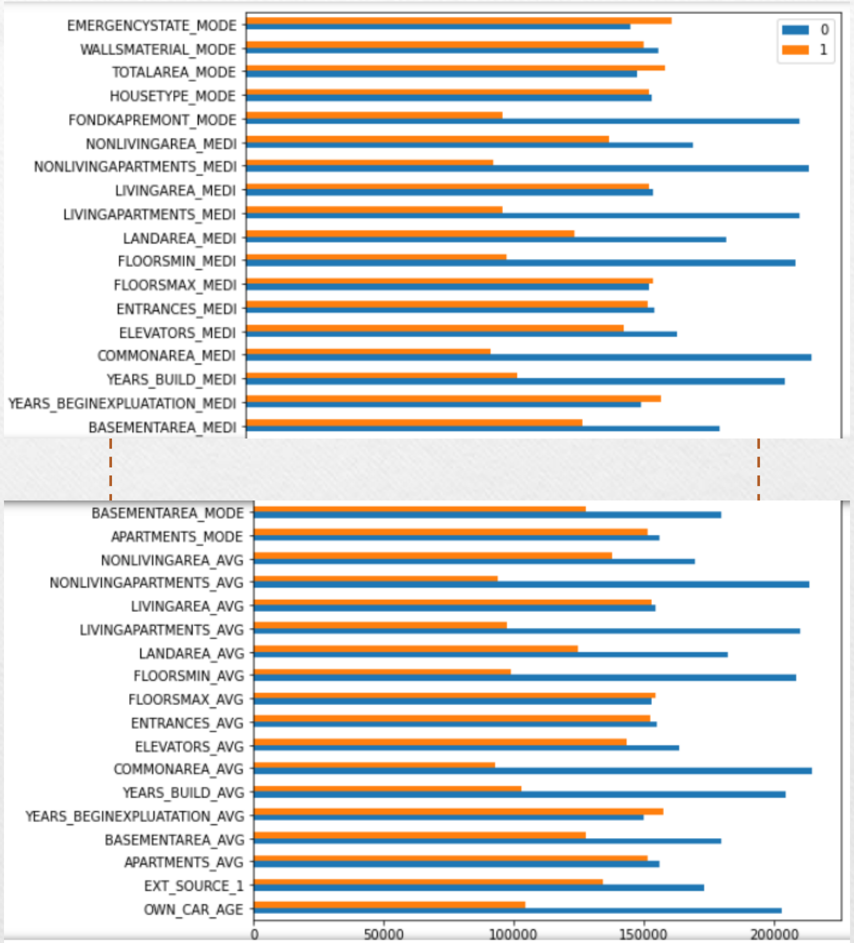
```
['OWN_CAR_AGE',
 'EXT_SOURCE_1',
 'APARTMENTS_AVG',
 'BASEMENTAREA_AVG',
 'YEARS_BEGINEXPLUATATION_AVG',
 'YEARS_BUILD_AVG',
 'COMMONAREA_AVG',
 'ELEVATORS_AVG',
 'ENTRANCES_AVG',
 'FLOORSMAX_AVG',
 'FLOORSMIN_AVG',
 'LANDAREA_AVG',
 'LIVINGAPARTMENTS_AVG',
 'LIVINGAREA_AVG',
 'NONLIVINGAPARTMENTS_AVG',
 'NONLIVINGAREA_AVG',
 'APARTMENTS_MODE',
 'BASEMENTAREA_MODE',
 'YEARS_BEGINEXPLUATATION_MODE',
 'YEARS_BUILD_MODE',
 'COMMONAREA_MODE',
 'ELEVATORS_MODE',
 'ENTRANCES_MODE',
 'FLOORSMAX_MODE',
 'FLOORSMIN_MODE',
 'LANDAREA_MODE',
 'LIVINGAPARTMENTS_MODE',
 'LIVINGAREA_MODE',
 'NONLIVINGAPARTMENTS_MODE',
 'NONLIVINGAREA_MODE',
 'APARTMENTS_MEDI',
 'BASEMENTAREA_MEDI',
 'YEARS_BEGINEXPLUATATION_MEDI',
 'YEARS_BUILD_MEDI',
 'COMMONAREA_MEDI',
 'ELEVATORS_MEDI',
 'ENTRANCES_MEDI',
 'FLOORSMAX_MEDI',
 'FLOORSMIN_MEDI',
 'LANDAREA_MEDI',
 'LIVINGAPARTMENTS_MEDI',
 'LIVINGAREA_MEDI',
 'NONLIVINGAPARTMENTS_MEDI',
 'NONLIVINGAREA_MEDI',
 'FONDKAPREMONT_MODE',
 'HOUSETYPE_MODE',
 'TOTALAREA_MODE',
 'WALLSMATERIAL_MODE',
 'EMERGENCYSTATE_MODE']
```

Reviewing the 49 column list is important because you would not want to drop a column which would help in your analysis – may be only we a set of row. Example Target = 1

Data Cleaning and Manipulation

A. Handling Nulls in Application File

```
a.T.plot.barh(figsize=[8,16])
plt.xlabel("Total No. of Records")
plt.ylabel("Columns with Null Values")
plt.show()
```



Key
0 - Null Values
1 - Non Null Values

The graph very clearly shows the difference between null values for individual column(49)

Dotted line in between the graph denotes that there are more column which cannot be displayed in ppt but are available in Jupiter notebook

Data Cleaning and Manipulation

B. Dropping Columns

Post dropping columns with null greater than 45% in Application File the data is stored in new data frame called “New_app”. This data frame will be used through the next implementations/ analysis

```
New_app=app.drop(delcol,axis=1)
```

```
New_app
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR
0	100002	1	Cash loans	M	N
1	100003	0	Cash loans	F	N
2	100004	0	Revolving loans	M	Y
3	100006	0	Cash loans	F	N
4	100007	0	Cash loans	M	N
...	...	...	...	...	...
307506	456251	0	Cash loans	M	N
307507	456252	0	Cash loans	F	N
307508	456253	0	Cash loans	F	N
307509	456254	1	Cash loans	F	N
307510	456255	0	Cash loans	F	N

307511 rows × 73 columns

Data Cleaning and Manipulation

C. Imputing Nulls with Relevant Value

Finding out the no. of columns and columns which need to be imputed with relevant value since the columns have null 0 - 45% of total volume in Application File

```
a=New_app.columns[New_app.isnull().sum() > 0].tolist()
```

```
len(a)
```

```
18
```

```
a
```

```
['AMT_ANNUITY',
 'AMT_GOODS_PRICE',
 'NAME_TYPE_SUITE',
 'OCCUPATION_TYPE',
 'CNT_FAM_MEMBERS',
 'EXT_SOURCE_2',
 'EXT_SOURCE_3',
 'OBS_30_CNT_SOCIAL_CIRCLE',
 'DEF_30_CNT_SOCIAL_CIRCLE',
 'OBS_60_CNT_SOCIAL_CIRCLE',
 'DEF_60_CNT_SOCIAL_CIRCLE',
 'DAYS_LAST_PHONE_CHANGE',
 'AMT_REQ_CREDIT_BUREAU_HOUR',
 'AMT_REQ_CREDIT_BUREAU_DAY',
 'AMT_REQ_CREDIT_BUREAU_WEEK',
 'AMT_REQ_CREDIT_BUREAU_MON',
 'AMT_REQ_CREDIT_BUREAU_QRT',
 'AMT_REQ_CREDIT_BUREAU_YEAR']
```

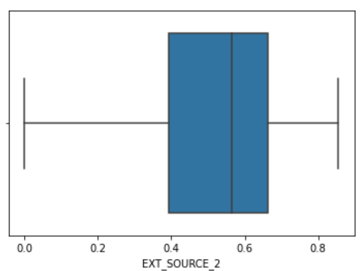
Data Cleaning and Manipulation

C. Imputing Nulls with Relevant Value

- Find out the relevant imputing values for each column having nulls in Application File
- Since Box plot is the best way to find out if the columns containing nulls have outliers or not - the box plots for the column which would require value imputation are built below

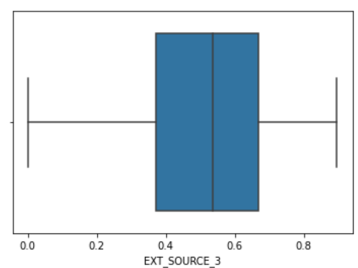
```
sns.boxplot(x=New_app["EXT_SOURCE_2"])
```

<AxesSubplot:xlabel='EXT_SOURCE_2'>



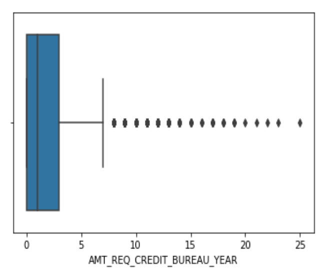

```
sns.boxplot(x=New_app["EXT_SOURCE_3"])
```

<AxesSubplot:xlabel='EXT_SOURCE_3'>



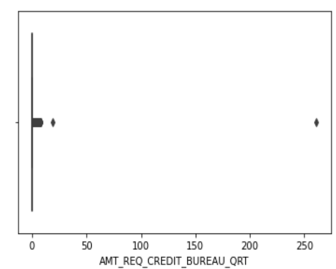

```
sns.boxplot(x=New_app["AMT_REQ_CREDIT_BUREAU_YEAR"])
```

<AxesSubplot:xlabel='AMT_REQ_CREDIT_BUREAU_YEAR'>



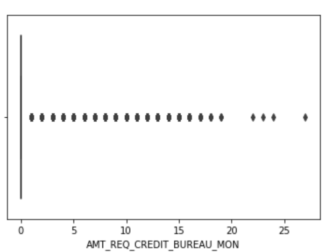

```
sns.boxplot(x=New_app["AMT_REQ_CREDIT_BUREAU_QRT"])
```

<AxesSubplot:xlabel='AMT_REQ_CREDIT_BUREAU_QRT'>



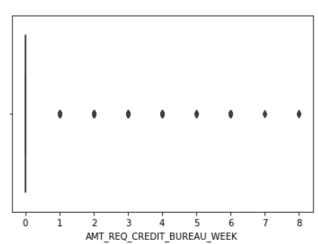

```
sns.boxplot(x=New_app["AMT_REQ_CREDIT_BUREAU_MON"])
```

<AxesSubplot:xlabel='AMT_REQ_CREDIT_BUREAU_MON'>



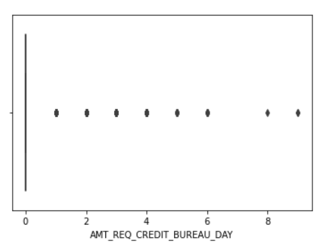

```
sns.boxplot(x=New_app["AMT_REQ_CREDIT_BUREAU_WEEK"])
```

<AxesSubplot:xlabel='AMT_REQ_CREDIT_BUREAU_WEEK'>



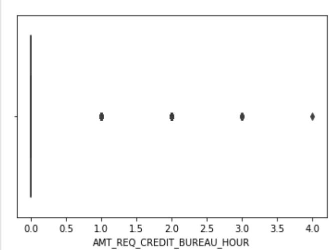

```
sns.boxplot(x=New_app["AMT_REQ_CREDIT_BUREAU_DAY"])
```

<AxesSubplot:xlabel='AMT_REQ_CREDIT_BUREAU_DAY'>




```
sns.boxplot(x=New_app["AMT_REQ_CREDIT_BUREAU_HOUR"])
```

<AxesSubplot:xlabel='AMT_REQ_CREDIT_BUREAU_HOUR'>



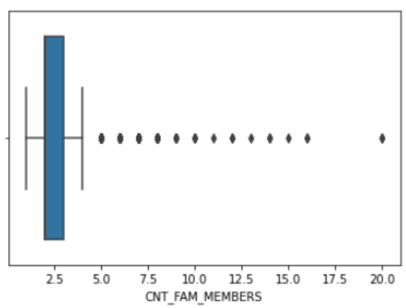
Data Cleaning and Manipulation

C. Imputing Nulls with Relevant Value

- Since Box plot is the best way to find out if the columns containing nulls have outliers or not - the box plots for the column which would require value imputation are built below

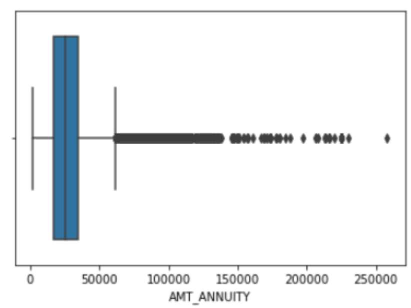
```
sns.boxplot(x=New_app["CNT_FAM_MEMBERS"])
```

<AxesSubplot:xlabel='CNT_FAM_MEMBERS'>



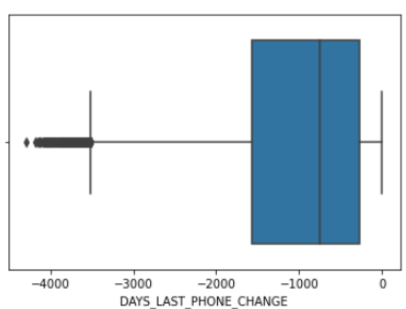

```
sns.boxplot(x=New_app["AMT_ANNUITY"])
```

<AxesSubplot:xlabel='AMT_ANNUITY'>



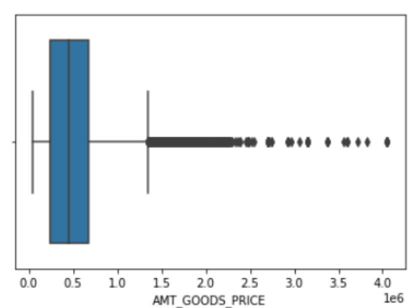

```
sns.boxplot(x=New_app["DAYS_LAST_PHONE_CHANGE"])
```

<AxesSubplot:xlabel='DAYS_LAST_PHONE_CHANGE'>



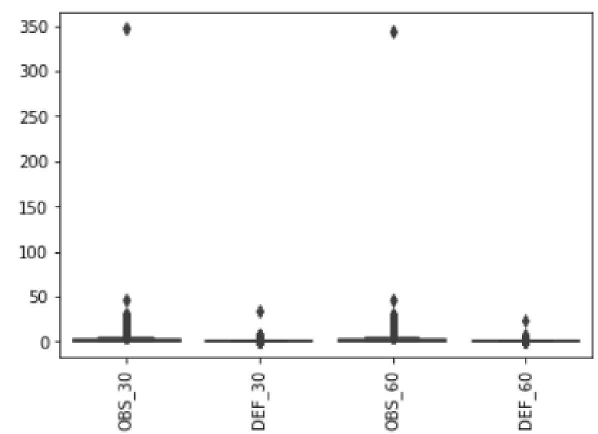

```
sns.boxplot(x=New_app["AMT_GOODS_PRICE"])
```

<AxesSubplot:xlabel='AMT_GOODS_PRICE'>




```
ax=sns.boxplot(data= New_app[['OBS_30_CNT_SOCIAL_CIRCLE',
'DEF_30_CNT_SOCIAL_CIRCLE',
'OBS_60_CNT_SOCIAL_CIRCLE',
'DEF_60_CNT_SOCIAL_CIRCLE'
]])

ax.set_xticklabels(['OBS_30',
'DEF_30',
'OBS_60',
'DEF_60'
], rotation=90)
plt.show()
```



Data Cleaning and Manipulation

C. Imputing Nulls with Relevant Value

Checking Null population occurrence in different columns shows there are 18 columns in total whose null values need to be imputed out of which 8 columns have nulls greater than 10%

```
New_app[New_app.columns[((New_app.isnull().sum()/app.shape[0])*100)>0]].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 18 columns):
 #   Column                      Non-Null Count   Dtype
     ------                      --------------   -----
 0   AMT_ANNUITY                 307499 non-null  float64
 1   AMT_GOODS_PRICE             307233 non-null  float64
 2   NAME_TYPE_SUITE             306219 non-null  object
 3   OCCUPATION_TYPE             211120 non-null  object
 4   CNT_FAM_MEMBERS             307509 non-null  float64
 5   EXT_SOURCE_2                306851 non-null  float64
 6   EXT_SOURCE_3                246546 non-null  float64
 7   OBS_30_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 8   DEF_30_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 9   OBS_60_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 10  DEF_60_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 11  DAYS_LAST_PHONE_CHANGE      307510 non-null  float64
 12  AMT_REQ_CREDIT_BUREAU_HOUR  265992 non-null  float64
 13  AMT_REQ_CREDIT_BUREAU_DAY   265992 non-null  float64
 14  AMT_REQ_CREDIT_BUREAU_WEEK  265992 non-null  float64
 15  AMT_REQ_CREDIT_BUREAU_MON   265992 non-null  float64
 16  AMT_REQ_CREDIT_BUREAU_QRT   265992 non-null  float64
 17  AMT_REQ_CREDIT_BUREAU_YEAR  265992 non-null  float64
dtypes: float64(16), object(2)
memory usage: 42.2+ MB
```

```
New_app[New_app.columns[((New_app.isnull().sum()/app.shape[0])*100)>10]].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 8 columns):
 #   Column                      Non-Null Count   Dtype
     ------                      --------------   -----
 0   OCCUPATION_TYPE             211120 non-null  object
 1   EXT_SOURCE_3                246546 non-null  float64
 2   AMT_REQ_CREDIT_BUREAU_HOUR  265992 non-null  float64
 3   AMT_REQ_CREDIT_BUREAU_DAY   265992 non-null  float64
 4   AMT_REQ_CREDIT_BUREAU_WEEK  265992 non-null  float64
 5   AMT_REQ_CREDIT_BUREAU_MON   265992 non-null  float64
 6   AMT_REQ_CREDIT_BUREAU_QRT   265992 non-null  float64
 7   AMT_REQ_CREDIT_BUREAU_YEAR  265992 non-null  float64
dtypes: float64(7), object(1)
memory usage: 18.8+ MB
```

Data Cleaning and Manipulation

C. Imputing Nulls with Relevant Value

As this is big dataset - we cannot be hard coding to write one statement for each column to impute values - we have built an easier, faster and dynamic way to find out what is the imputation required and finally do the imputation. The categorisation is done as below:

```
New_app[New_app.columns[((New_app.isnull().sum()/app.shape[0])*100)>0]].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 18 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   AMT_ANNUITY                 307499 non-null  float64
 1   AMT_GOODS_PRICE             307233 non-null  float64
 2   NAME_TYPE_SUITE             306219 non-null  object
 3   OCCUPATION_TYPE             211120 non-null  object
 4   CNT_FAM_MEMBERS             307509 non-null  float64
 5   EXT_SOURCE_2                306851 non-null  float64
 6   EXT_SOURCE_3                246546 non-null  float64
 7   OBS_30_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 8   DEF_30_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 9   OBS_60_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 10  DEF_60_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 11  DAYS_LAST_PHONE_CHANGE      307510 non-null  float64
 12  AMT_REQ_CREDIT_BUREAU_HOUR  265992 non-null  float64
 13  AMT_REQ_CREDIT_BUREAU_DAY   265992 non-null  float64
 14  AMT_REQ_CREDIT_BUREAU_WEEK  265992 non-null  float64
 15  AMT_REQ_CREDIT_BUREAU_MON   265992 non-null  float64
 16  AMT_REQ_CREDIT_BUREAU_QRT   265992 non-null  float64
 17  AMT_REQ_CREDIT_BUREAU_YEAR  265992 non-null  float64
dtypes: float64(16), object(2)
memory usage: 42.2+ MB
```

```
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 2   NAME_TYPE_SUITE             306219 non-null  object
 3   OCCUPATION_TYPE             211120 non-null  object
```

```
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   AMT_ANNUITY                 307499 non-null  float64
 1   AMT_GOODS_PRICE             307233 non-null  float64
 4   CNT_FAM_MEMBERS             307509 non-null  float64
 5   EXT_SOURCE_2                306851 non-null  float64
 6   EXT_SOURCE_3                246546 non-null  float64
 7   OBS_30_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 8   DEF_30_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 9   OBS_60_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 10  DEF_60_CNT_SOCIAL_CIRCLE    306490 non-null  float64
 11  DAYS_LAST_PHONE_CHANGE      307510 non-null  float64
 12  AMT_REQ_CREDIT_BUREAU_HOUR  265992 non-null  float64
 13  AMT_REQ_CREDIT_BUREAU_DAY   265992 non-null  float64
 14  AMT_REQ_CREDIT_BUREAU_WEEK  265992 non-null  float64
 15  AMT_REQ_CREDIT_BUREAU_MON   265992 non-null  float64
 16  AMT_REQ_CREDIT_BUREAU_QRT   265992 non-null  float64
 17  AMT_REQ_CREDIT_BUREAU_YEAR  265992 non-null  float64
```

Data Cleaning and Manipulation

C. Imputing Nulls with Relevant Value

We are calculating the outliers columns and then taking a decision whether to take a mean or median. The Categorical col null values are replaced with string 'Missing'

```
MissingInput={}
```

```
for item in a:
    if (New_app[item].dtypes == 'float64' or New_app[item].dtypes == 'int32' or New_app[item].dtypes == 'int64' ):
        q1=New_app[item].describe()[4]
        q3=New_app[item].describe()[6]
        iqr=q3-q1
        lb=q1-(1.5*iqr)
        ub=q3+(1.5*iqr)
        HasOutlier=np.where((New_app[item]>ub).sum() or (New_app[item]<lb).sum(),1,0)
        if(HasOutlier.tolist()==1):
            MissingInput[item]="Median"
        elif(HasOutlier.tolist()==0):
            MissingInput[item]="Mean"
    else:
        MissingInput[item]="Missing"
```

```
MissingInput
```

```
{'AMT_ANNUITY': 'Median',
 'AMT_GOODS_PRICE': 'Median',
 'NAME_TYPE_SUITE': 'Missing',
 'OCCUPATION_TYPE': 'Missing',
 'CNT_FAM_MEMBERS': 'Median',
 'EXT_SOURCE_2': 'Mean',
 'EXT_SOURCE_3': 'Mean',
 'OBS_30_CNT_SOCIAL_CIRCLE': 'Median',
 'DEF_30_CNT_SOCIAL_CIRCLE': 'Median',
 'OBS_60_CNT_SOCIAL_CIRCLE': 'Median',
 'DEF_60_CNT_SOCIAL_CIRCLE': 'Median',
 'DAYS_LAST_PHONE_CHANGE': 'Median',
 'AMT_REQ_CREDIT_BUREAU_HOUR': 'Median',
 'AMT_REQ_CREDIT_BUREAU_DAY': 'Median',
 'AMT_REQ_CREDIT_BUREAU_WEEK': 'Median',
 'AMT_REQ_CREDIT_BUREAU_MON': 'Median',
 'AMT_REQ_CREDIT_BUREAU_QRT': 'Median',
 'AMT_REQ_CREDIT_BUREAU_YEAR': 'Median'}
```

```
for key in MissingInput :
    if (MissingInput[key]=='Missing'):
        New_app[key]=New_app[key].fillna("Missing")
    elif(MissingInput[key]=='Mean'):
        New_app[key]=New_app[key].fillna(New_app[key].mean())
    elif(MissingInput[key]=='Median'):
        New_app[key]=New_app[key].fillna(New_app[key].median())
```

```
New_app.columns[New_app.isnull().sum()>0]
```

```
Index([], dtype='object')
```

Data Analysis

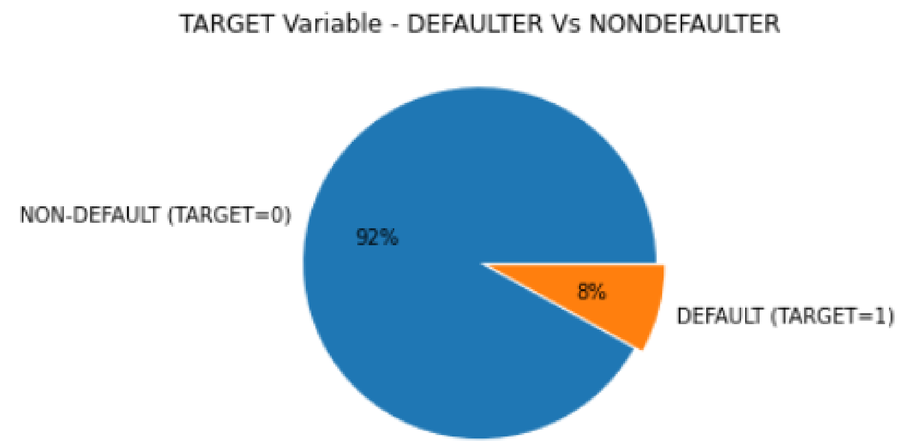
A. Target Percentage

In the application file dataset - there are 92% applicants who are non defaulters and 8% of applicants are defaulters

```
plt.pie(New_app['TARGET'].value_counts(normalize=True)*100,labe
plt.title('TARGET Variable - DEFAULTER Vs NONDEFAULTER')
print("Precentage of Non Defaulters",round((Non_Defaulter.shape
print("Precentage of Defaulters",round((Defaulter.shape[0]/ (No

plt.show()
```

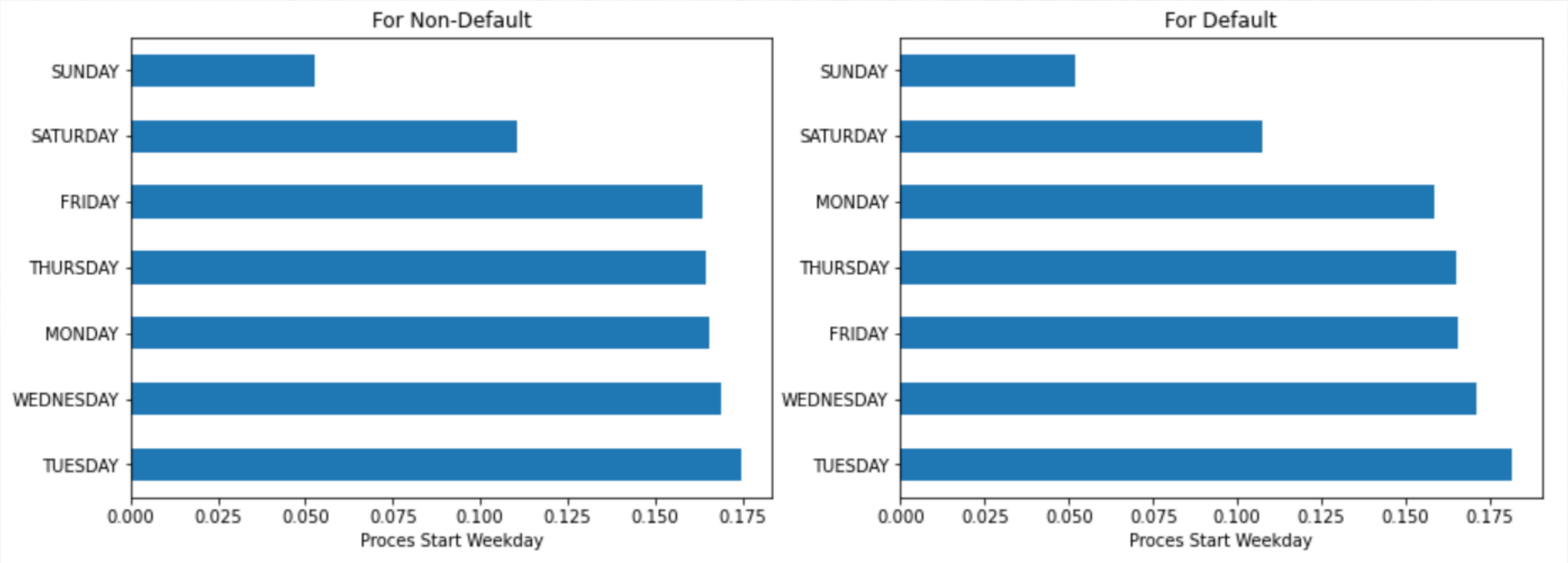
```
Precentage of Non Defaulters 91.93
Precentage of Defaulters 8.07
```



Data Analysis

B. Univariant Categorical Analysis

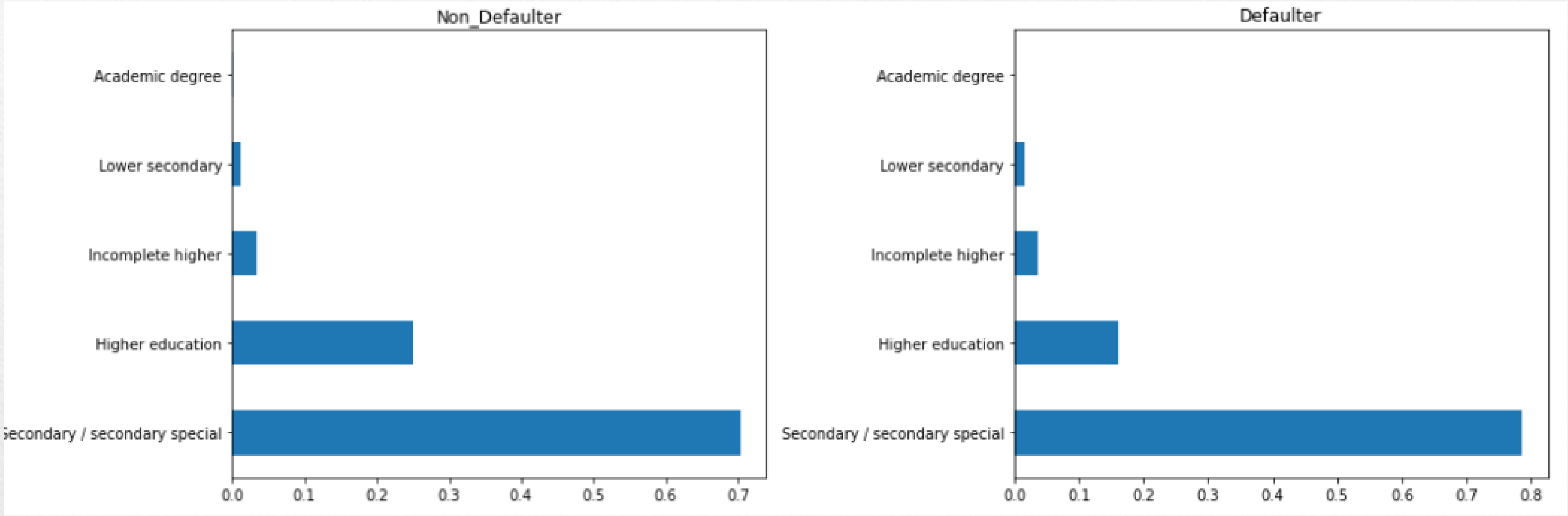
Observations : From the below graph we can conclude that application starting processes are generally less in Saturday and Sunday



Data Analysis

B. Univariant Categorical Analysis

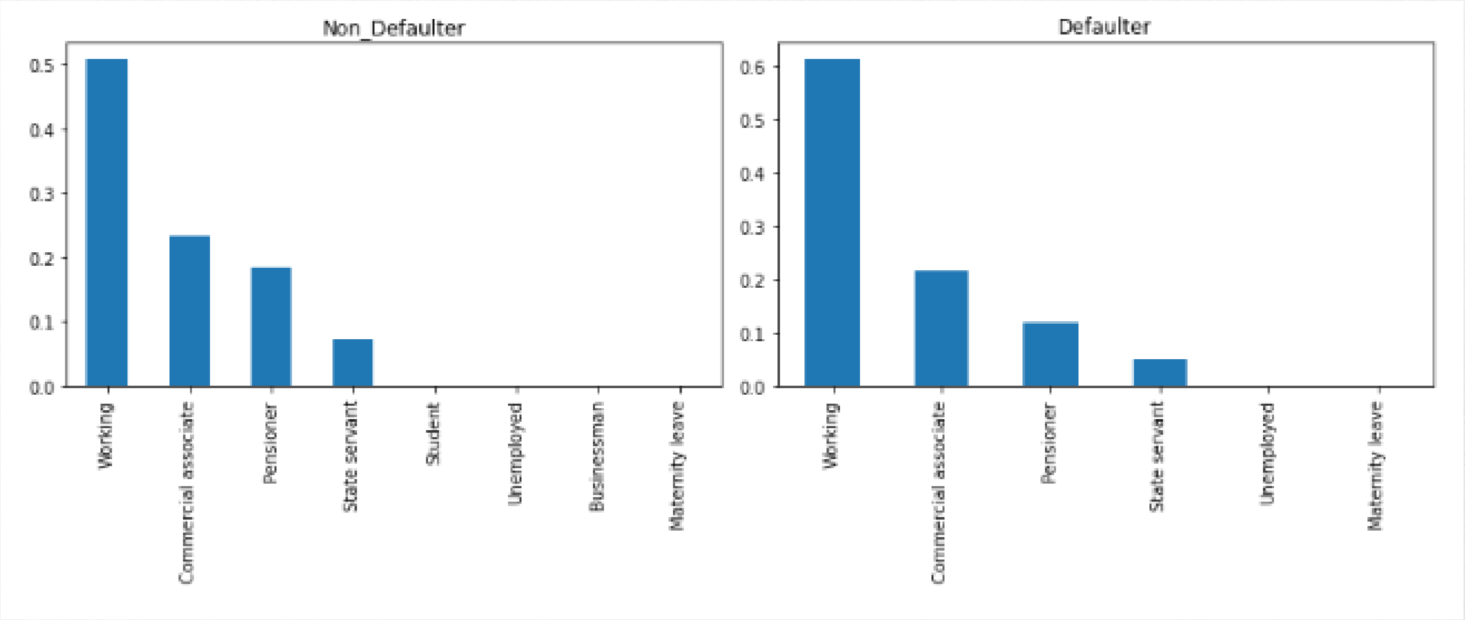
Observations : From the plot below we can conclude that secondary/special educated people are applying loans high in number. Academic degree educated people are applying loan in least count.



Data Analysis

B. Univariant Categorical Analysis

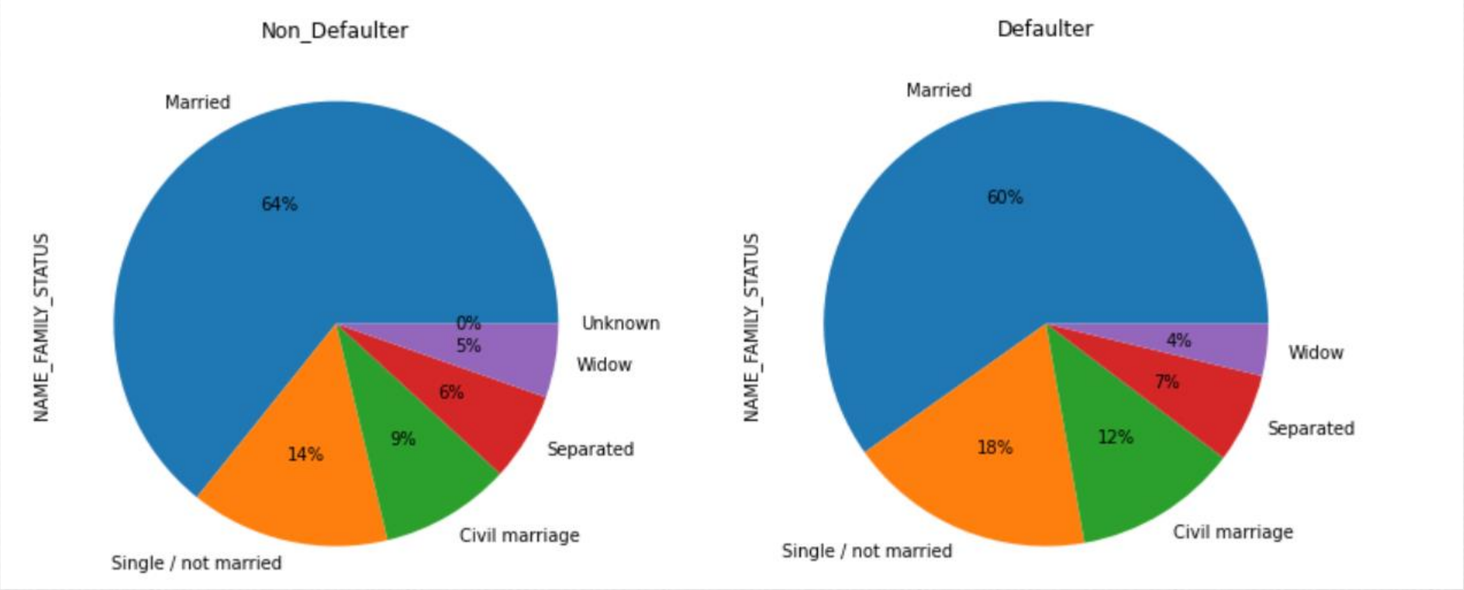
Observations : From the plot below we can notice that the students don't default. The reason could be they are not required to pay during the time they are students. We can also see that the Business Men never default. Most of the loans are distributed to working class people. We also see that working class people contribute ~50% to non defaulters while they contribute to ~60% of the defaulters. Thus, the chances of defaulting are more in working class case.



Data Analysis

B. Univariant Categorical Analysis

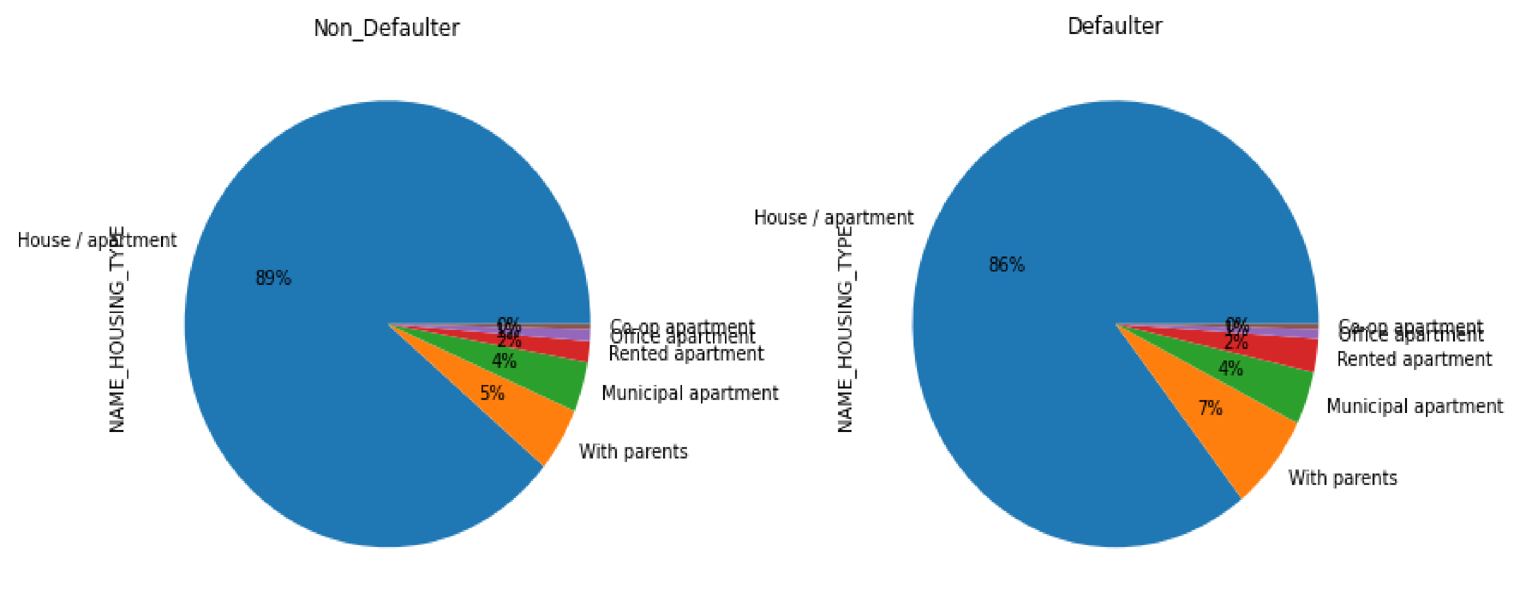
Observations : The order of both defaulter and not defaulter customers is same i.e., Married, Single/not married, civil marriage, separated, widow. It also shows that there exists few(1 or 2) unknown values in not default client family status. We can say more married people tend to take more Loan as compared to other categories and being married is not impacting default and not defaulting. We can see that Single/Not Married, Civil Marriage applicants have defaulted more



Data Analysis

B. Univariant Categorical Analysis

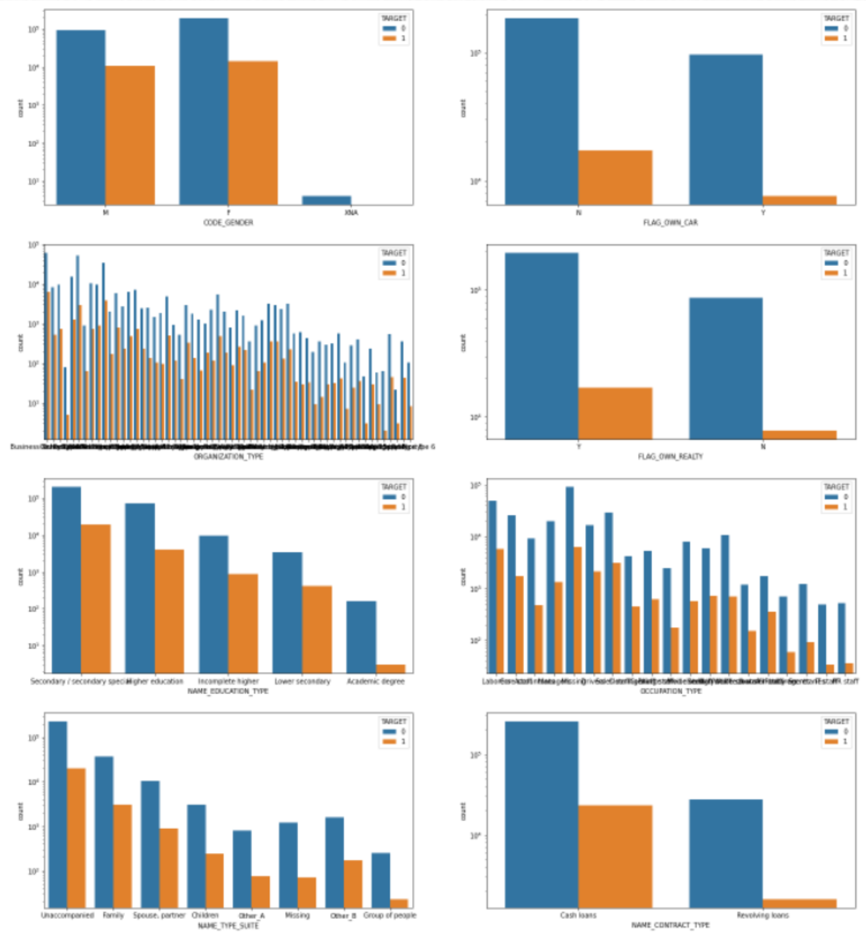
Observations : The order of both defaulter and not defaulter customers is same i.e. House/Apartment, With Parents, Municipal, Rented, Officer, Co-Op Apartment. Applicants living in last 4 category do not infer any information here since the measure is same. While we see applicant living in House / Apartment are more in non defaulters than defaulters, though they vary with less percentage. But since the we saw in 1st pie chat that 92% is non defaulter - so 89% of 92% is a good number to say Housing / Apartment are very less likely to default.



Data Analysis

B. Univariant Categorical Analysis

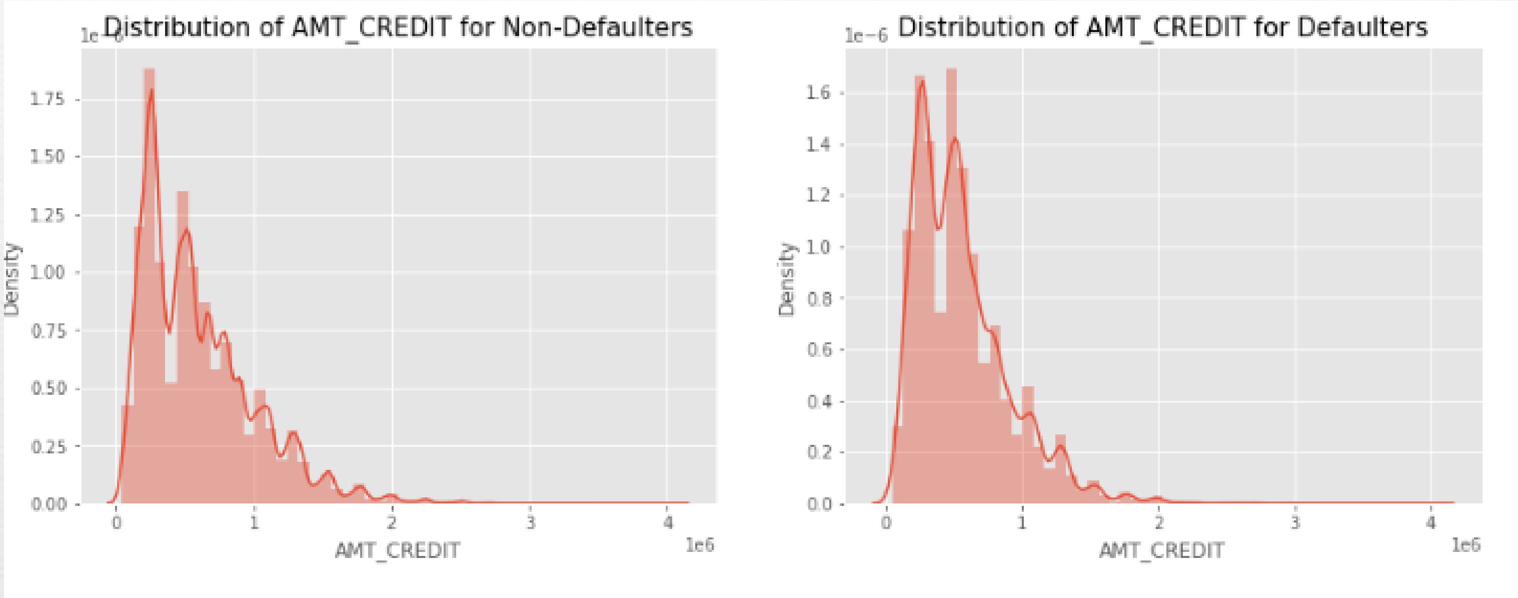
Observations from the graphs : Male not defaulting is similar to Women but Women default more than Men. People who don't own a car tends to take more loans. People tend to take more cash loans, and default percentage of revolving loans is less. People with real estate tends to take more loans. People who are not accompanied with anyone tend to take more loans. We can conclude that secondary/special educated people are applying loans in high in number.



Data Analysis

C. Univariant Continuous/ Numerical Analysis

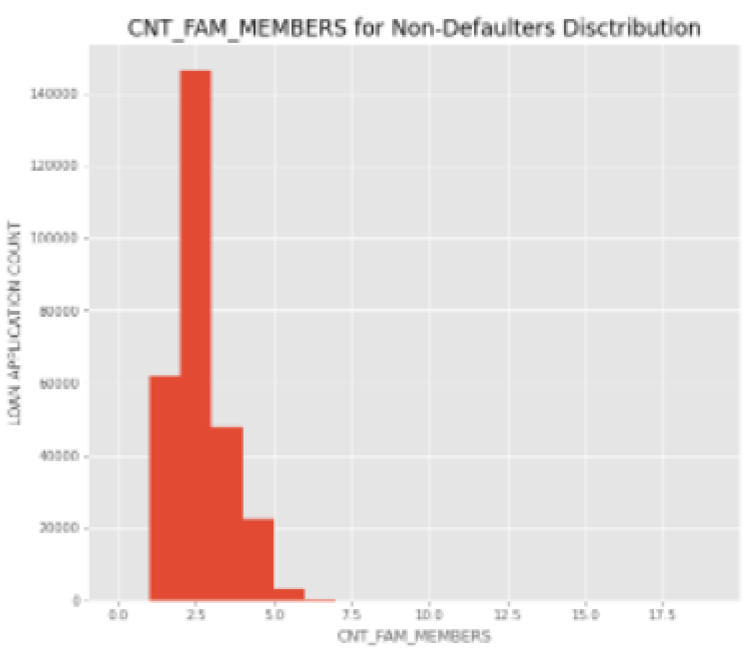
Observations from the graphs : Although there doesn't seem to be a clear distinguish between the group which defaulted vs the group which didn't but, we can see that when the AMT_CREDIT is more than 50, people default

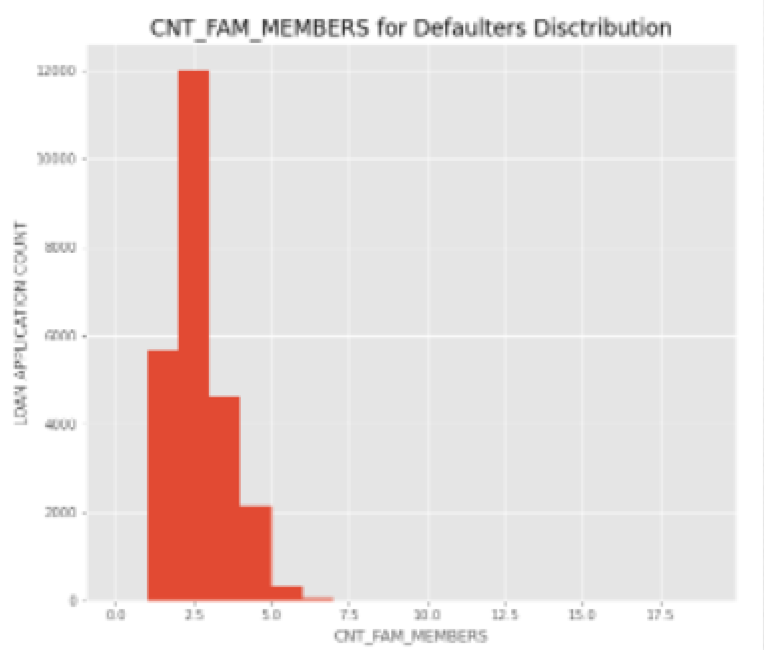


Data Analysis

C. Univariant Continuous/ Numerical Analysis

Observations from the graphs: We can see that a family of 3 applies loan more often than the other families

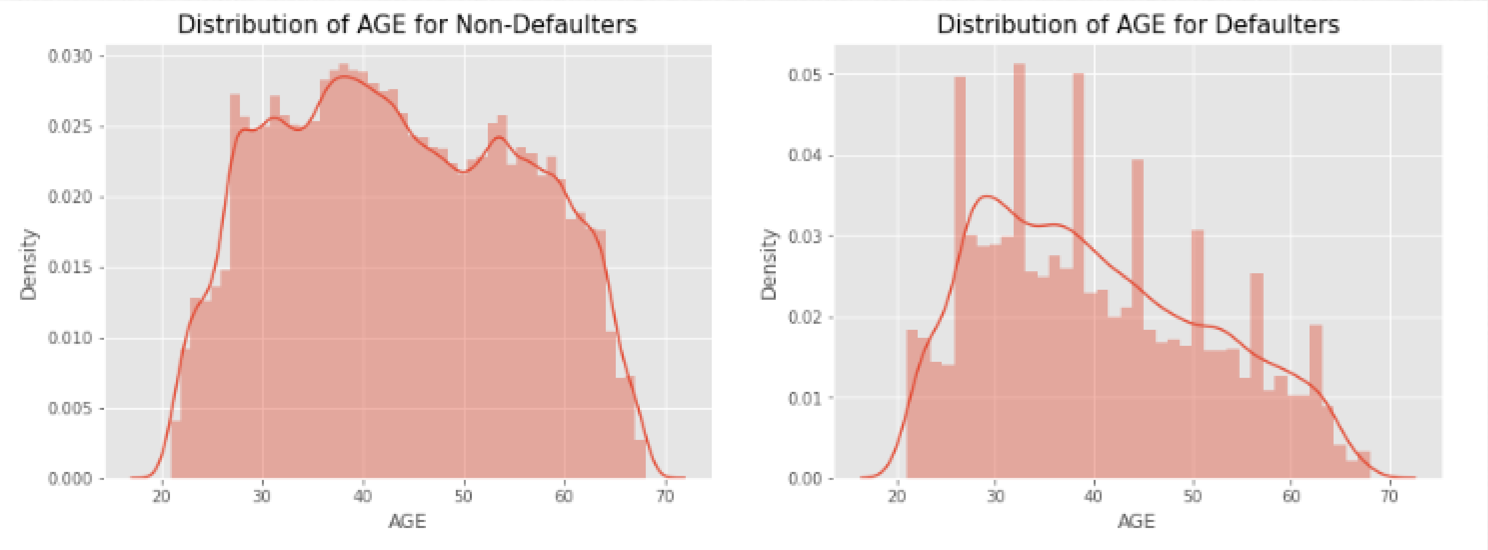




Data Analysis

C. Univariant Continuous/ Numerical Analysis

Observations from the graphs : People are more likely to default when they are in their mid age like 25-26,34-35,45,55-46. With the increase in age the defaulting behaviour of people decreases i.e. with the higher age has less defaulters

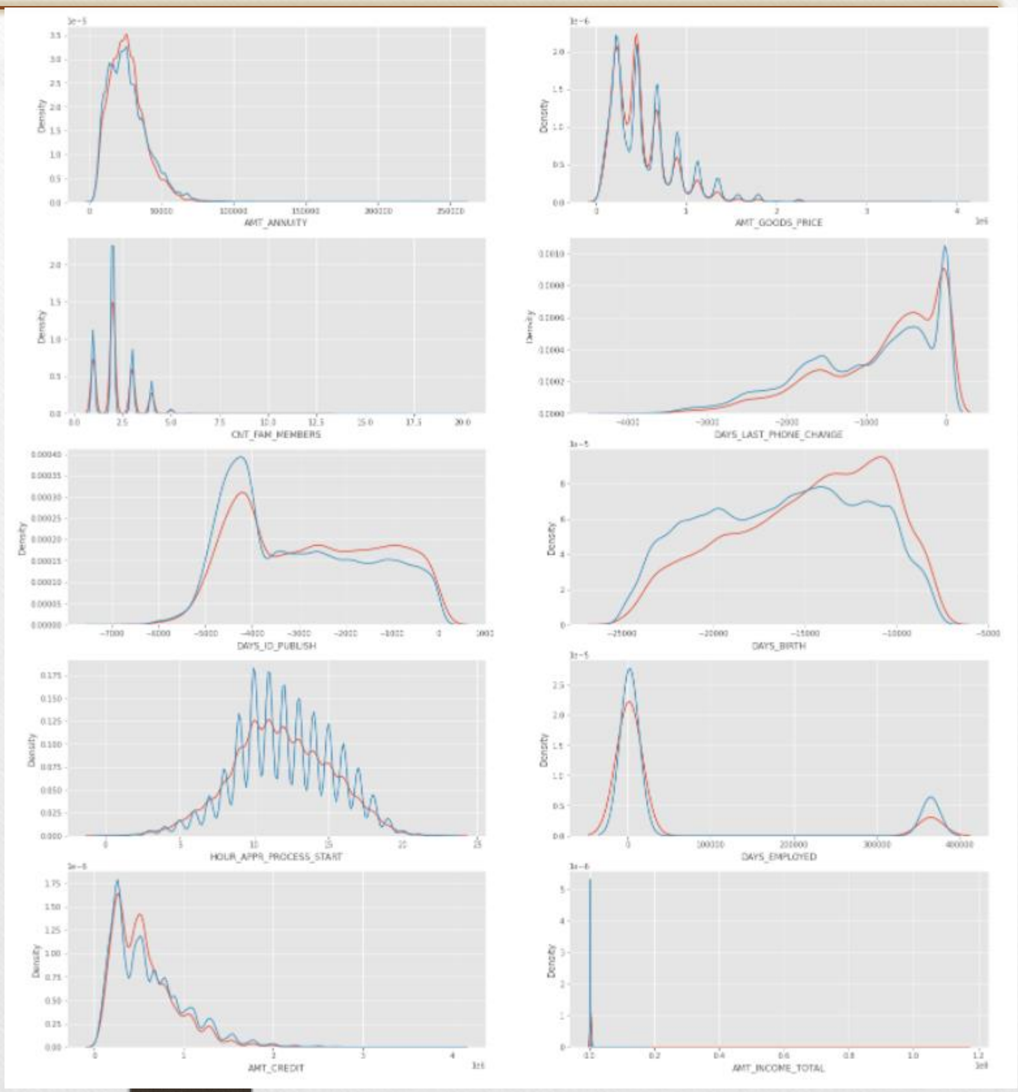


Data Analysis

C. Univariant Continuous/ Numerical Analysis

Observations from the graphs :

- People with lower total income are more likely to default.
- People who just got employed tends to take more loans.
- High number of applications are filed in 10 AM to 2 PM. People with age between 27yrs(10000-days) and 41(15000-days) yrs tend to take more loans.
- People whose id(s) got published between 4000 days and 5000 days ago tend to take more loans.
- Nuclear family tends to take more loans. Most no of loans are given for goods price below 10 lakhs.
- Credit amount of the loan is mostly less then 10 lakhs.
- The re-payers and defaulters distribution overlap in all the plots and hence we cannot use any of these variables in isolation to make a decision



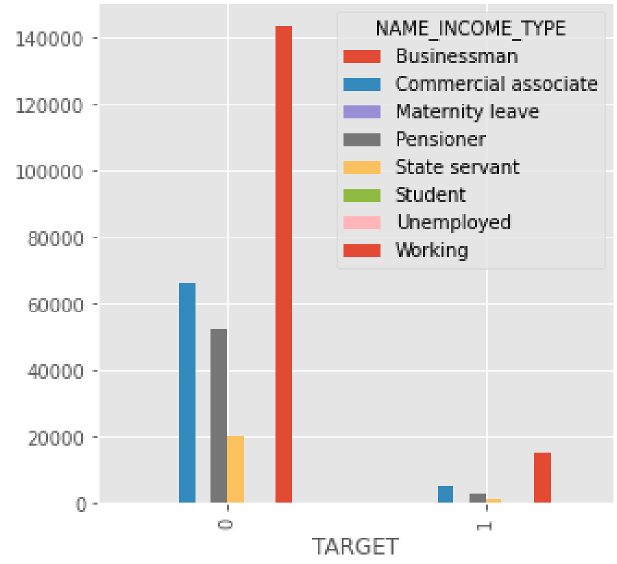
Data Analysis

D. Bi/Multi-Variate Categorical - Categorical Analysis

Observations :
Working people take more loans. Business Man, People on Maternity Leave take no loans at all. Students might take but they do not start to pay until employed thus out of consideration. Similar Unemployed is out of consideration

```
NAME_INCOME_TYPE  Businessman  Commercial associate  Maternity leave  \
TARGET
0                          10                 66257                3
1                           0                  5360                2

NAME_INCOME_TYPE  Pensioner  State servant  Student  Unemployed  Working
TARGET
0                     52380          20454       18          14   143550
1                      2982           1249        0           8    15224
```

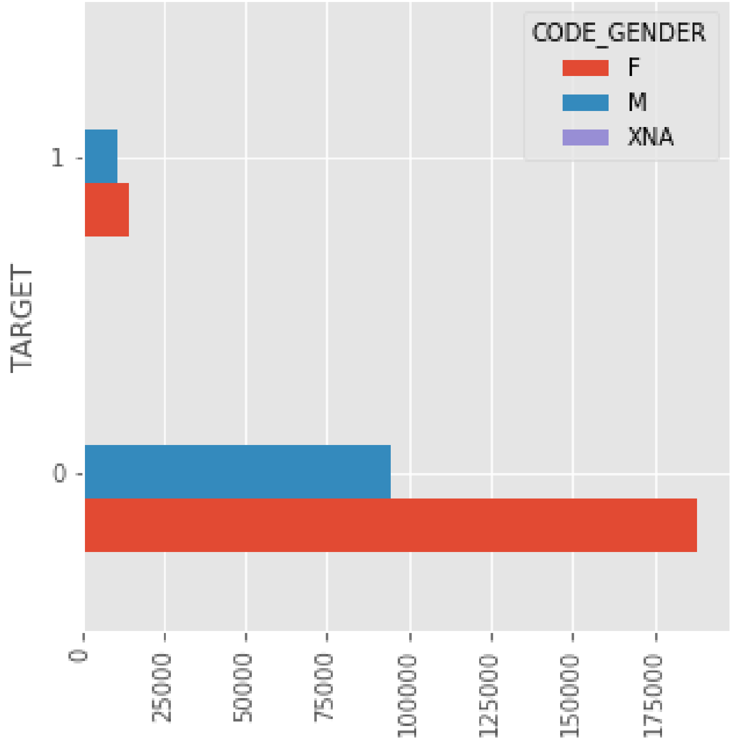


Data Analysis

D. Bi/Multi-Variate Categorical - Categorical Analysis

Observations : Females take more loans and Males default more.

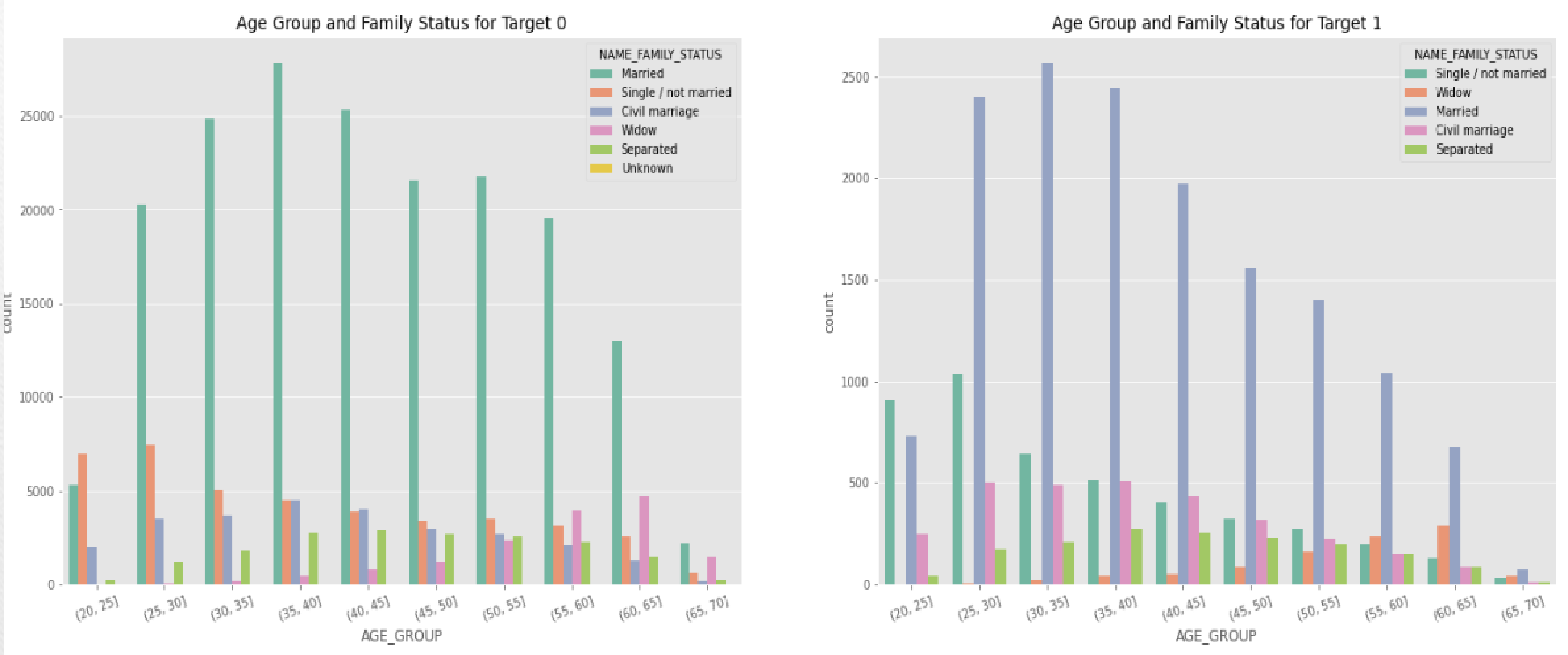
CODE_GENDER	F	M	XNA
TARGET			
0	188278	94404	4
1	14170	10655	0



Data Analysis

D. Bi/Multi-Variate Categorical - Categorical Analysis

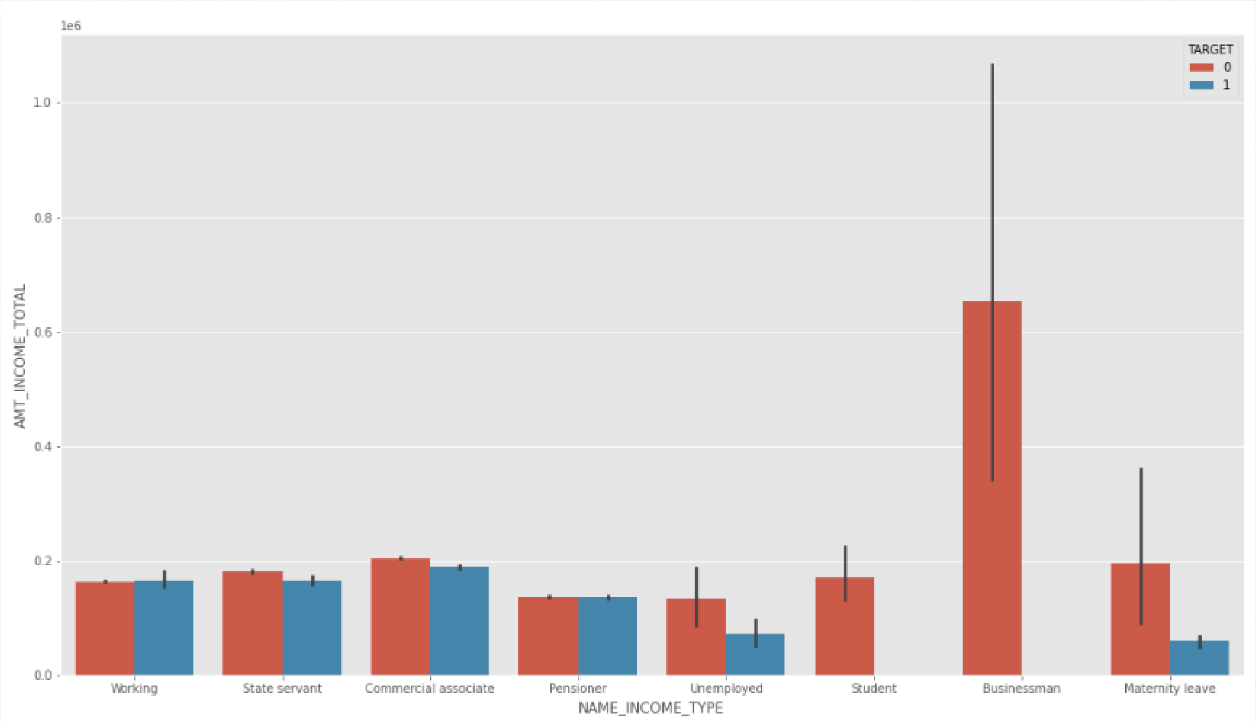
Observations from the above graphs - Married applicant in the age group 25-35 and 35-45 is the largest group of applicant with payment difficulties



Data Analysis

E. Bi/Multi-Variate Categorical - Numerical Analysis

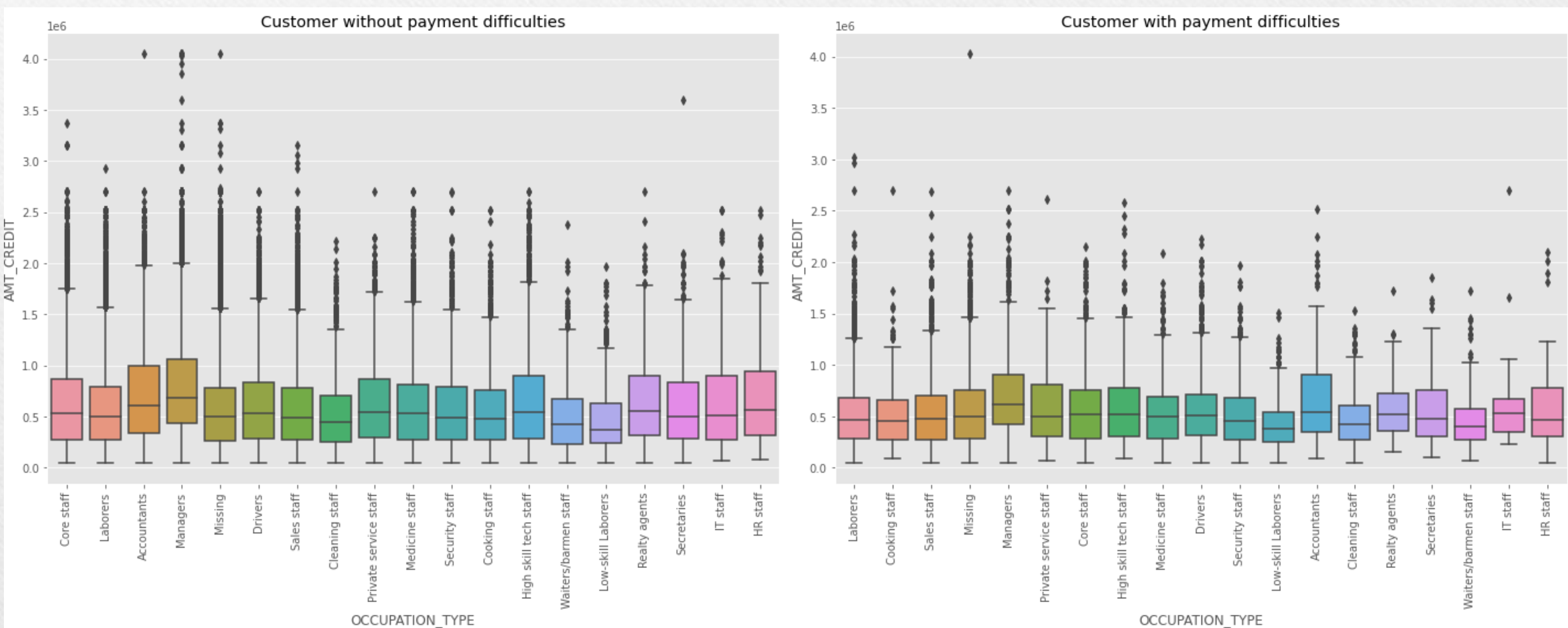
Observations: It can be seen that business man's income is the highest and they do not default. Commercial Associate default maximum in the income type category. We cannot infer much on prisoner and working income type as they appear to default and not default in equal proportions



Data Analysis

E. Bi/Multi-Variate Categorical - Numerical Analysis

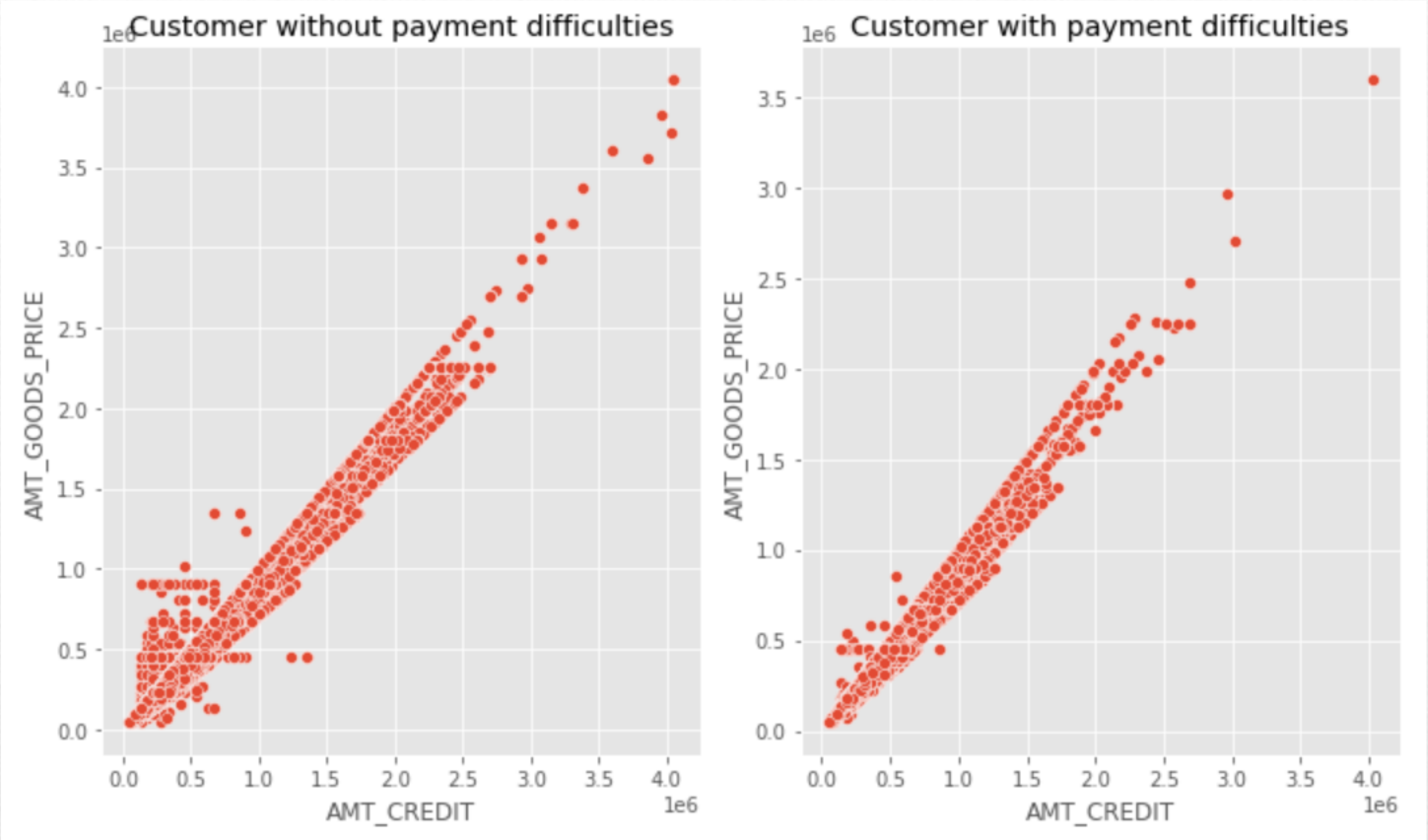
Observations : Here we can see that the range of the customers without payment difficulties (non defaulters) are more as compare to the customers with payment difficulties. We see that Occupation Type Accountants and Managers who have maximum Credit amount of the loan suffered from payment difficulties



Data Analysis

F. Bi/Multi-Variate Numerical - Numerical Analysis

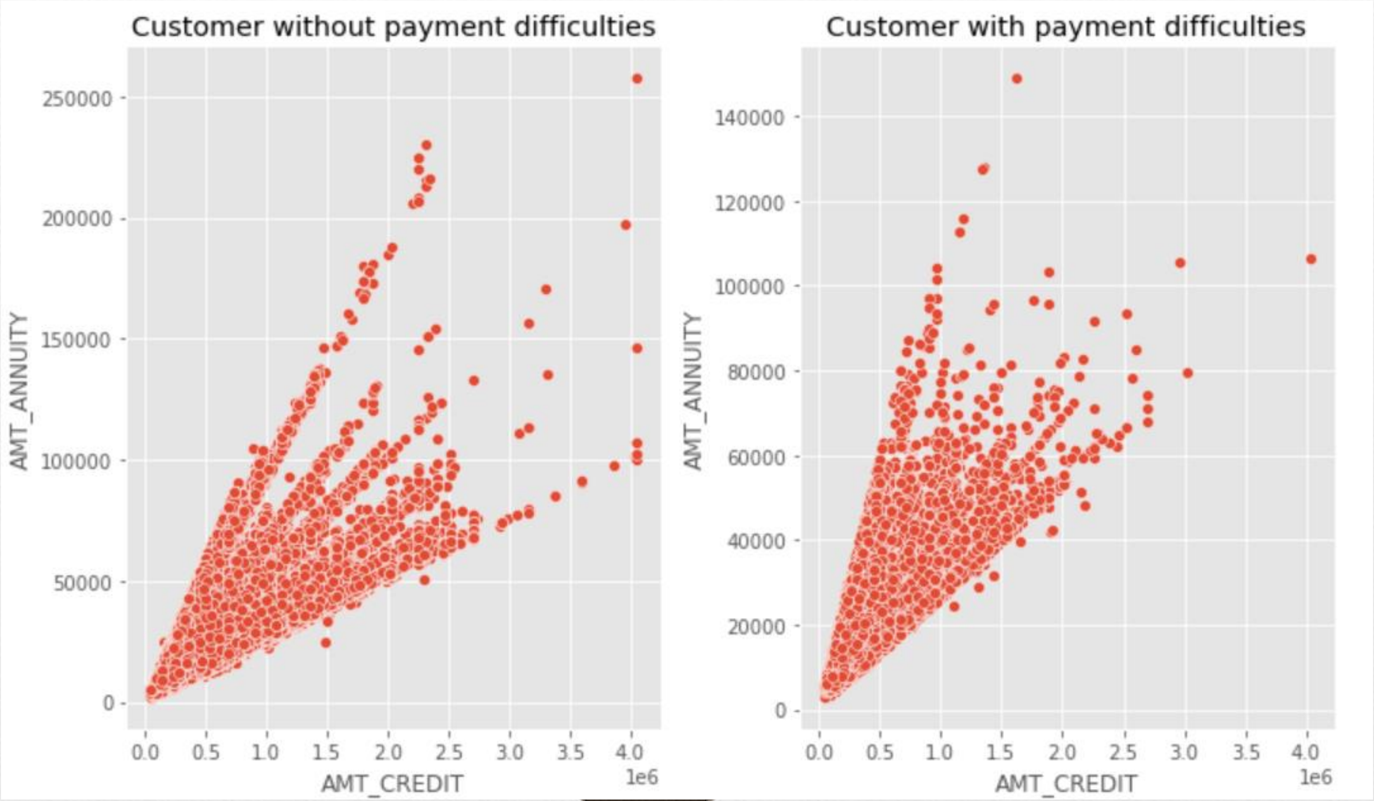
Observations : Here we can see that, positively correlated(goods price is positive correlated to credit amount)



Data Analysis

F. Bi/Multi-Variate Numerical - Numerical Analysis

Observations: Here, We can conclude that, people with out payment difficulties take more credit for the annuity



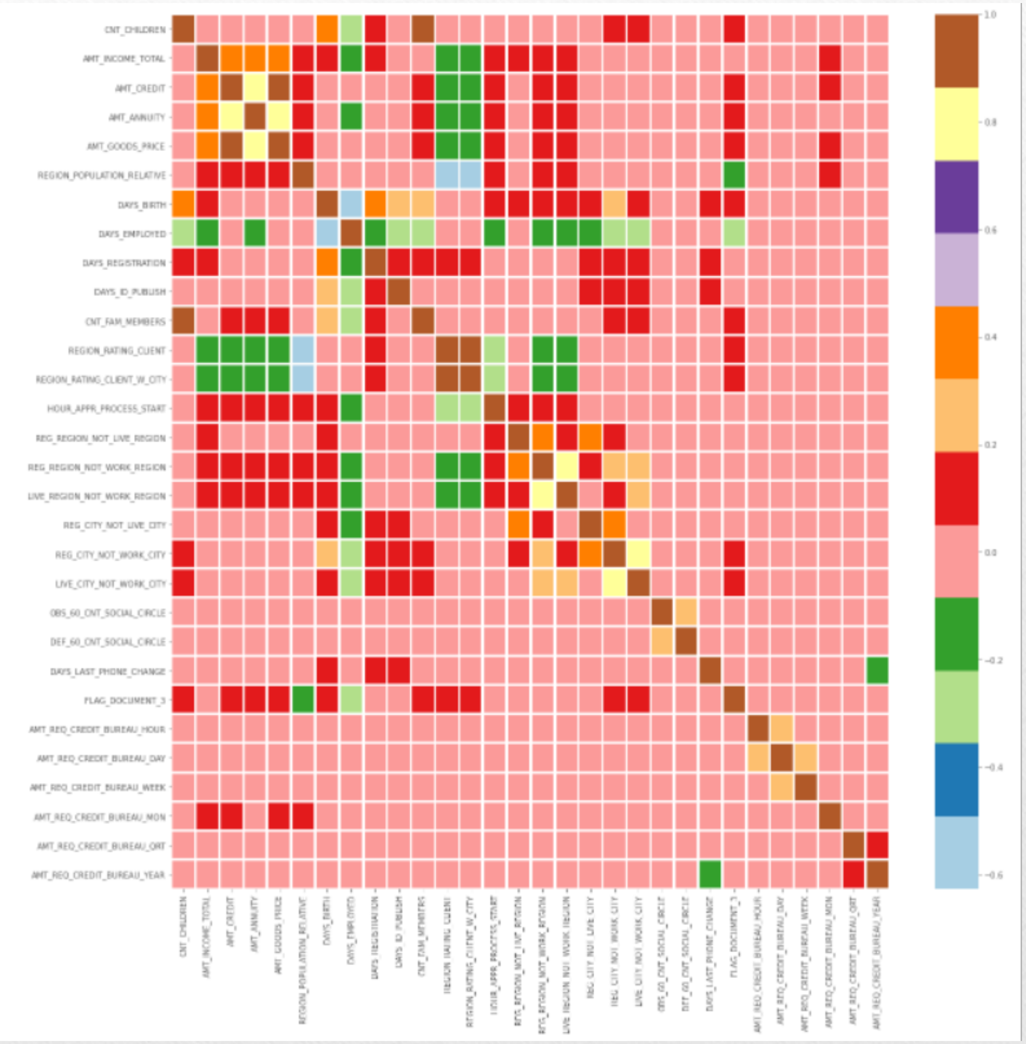
Data Analysis

G. Correlational Analysis

Non Defaulter Heat Map

Observations : Credit amount is highly correlated with amount of goods price which is same as repayers, But the loan annuity correlation with credit amount has slightly reduced in defaulters(0.75) when compared to repayers(0.77). We can also see that repayers have high correlation in number of days employed(0.62) when compared to defaulters(0.58).

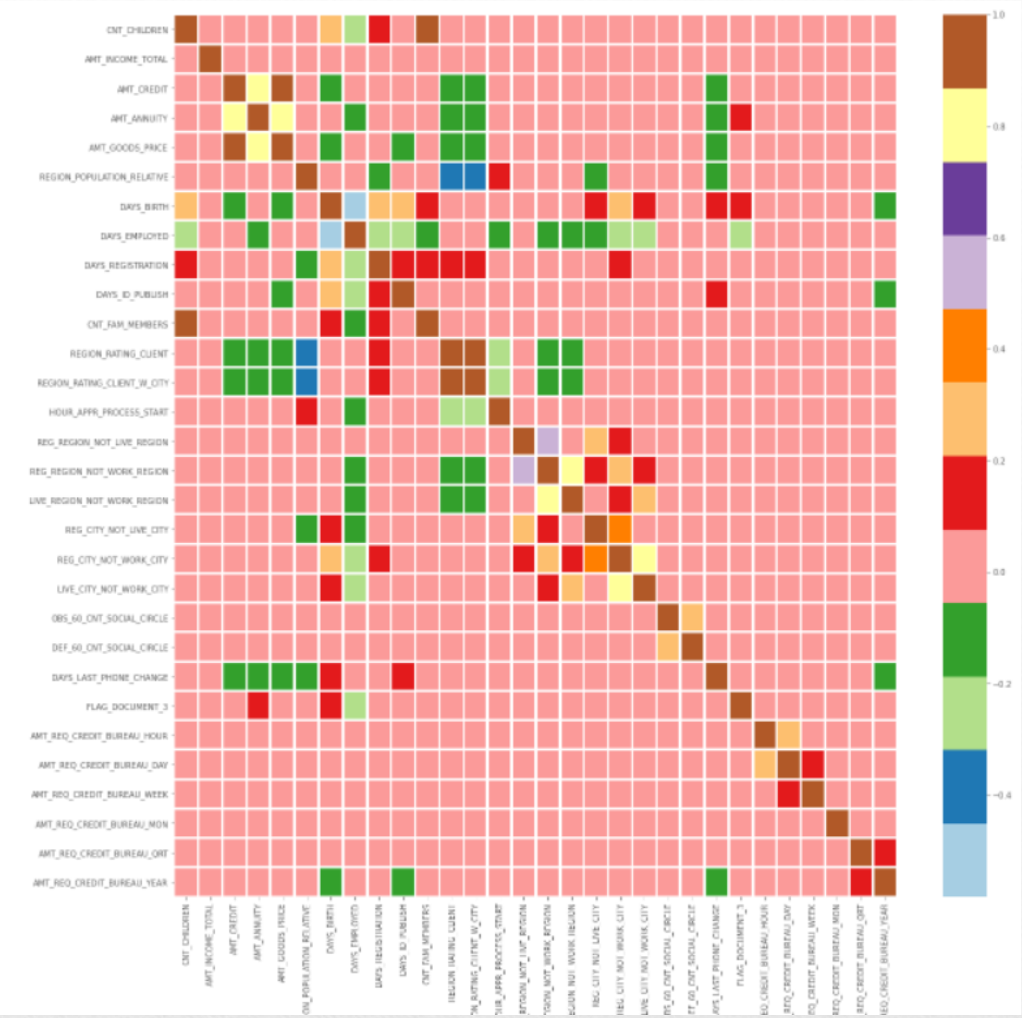
Learning : If we use a pallets which have different colours, it helps determine the value of the correlation even without having the value mention on the plot.



Data Analysis

G. Correlational Analysis

Observations : There is a severe drop in the correlation between total income of the client and the credit amount(0.038) amongst defaulters whereas it is 0.342 among repayers. Days_birth and number of children correlation has reduced to 0.259 in defaulters when compared to 0.337 in repayers. There is a slight increase in defaulted to observed count in social circle among defaulters(0.264) when compared to repayers(0.254)



Data Analysis

F. Join Previous Application Data & Current Application data

```
Join = pd.merge(left=New_app,right=New_papp,how='inner',on='SK_ID_CURR')
```

```
Join.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	F
0	100002	1	Cash loans	M	N	
1	100003	0	Cash loans	F	N	
2	100003	0	Cash loans	F	N	
3	100003	0	Cash loans	F	N	
4	100004	0	Revolving loans	M	Y	

5 rows × 107 columns

Data Analysis

G. Analysis on Joined Data

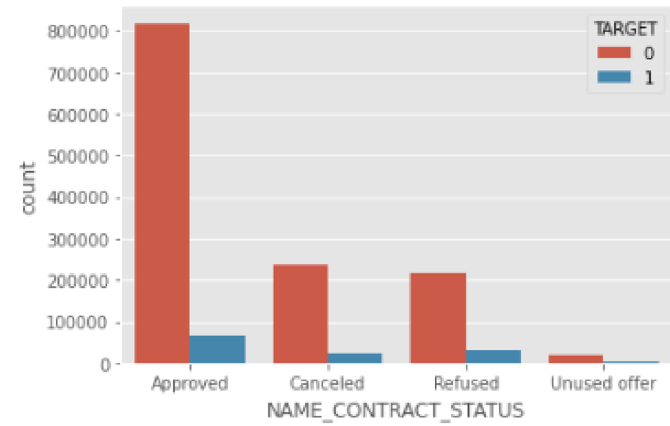
Observations : 90% of the previously cancelled client have actually repayed the loan. Revisiting the interest rates would increase business opportunity for these clients.88% of the clients who have been previously refused a loan has payed back the loan in current case. Refusal reason should be recorded for further analysis as these clients would turn into potential repaying customer

```
Join['NAME_CONTRACT_STATUS'].value_counts(normalize=True)*100
```

```
Approved        62.679378
Canceled        18.351900
Refused         17.357984
Unused offer     1.610737
Name: NAME_CONTRACT_STATUS, dtype: float64
```

```
sns.countplot(x='NAME_CONTRACT_STATUS', data=Join,hue='TARGET',
```

```
<AxesSubplot:xlabel='NAME_CONTRACT_STATUS', ylabel='count'>
```




```
group = Join.groupby("NAME_CONTRACT_STATUS")["TARGET"]
Values = pd.concat([group.value_counts(),round(group.value_coun
Values['Percentage'] = Values['Percentage'].astype(str) +"%" # 
print (Values)
```

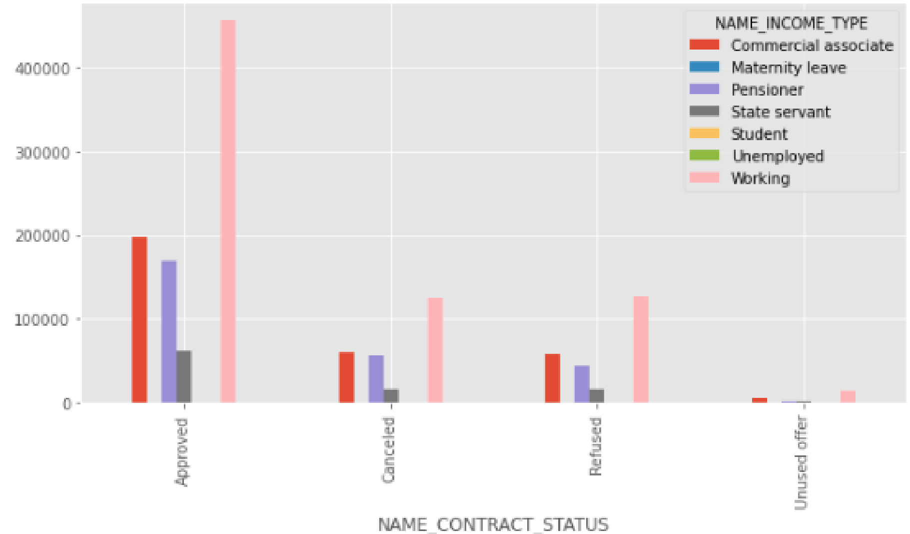
```
                             Counts Percentage
NAME_CONTRACT_STATUS TARGET
Approved             0       818856     92.41%
                     1        67243      7.59%
Canceled             0       235641     90.83%
                     1        23800      9.17%
Refused              0       215952      88.0%
                     1        29438      12.0%
Unused offer         0        20892     91.75%
                     1         1879      8.25%
```

Data Analysis

G. Analysis on Joined Data

Observations : Highest number of approvals for working applicant followed by state servant, But Working professional still remain high as there is a big difference. Maximum Refusal are for the unemployed.

```
Ct= pd.crosstab(index=Join['NAME_CONTRACT_STATUS'],columns=Join['NAME_INCOME_TYPE'])
Ct.plot(kind="bar", figsize=(10,5),stacked=False)
plt.show()
print(Ct)
```



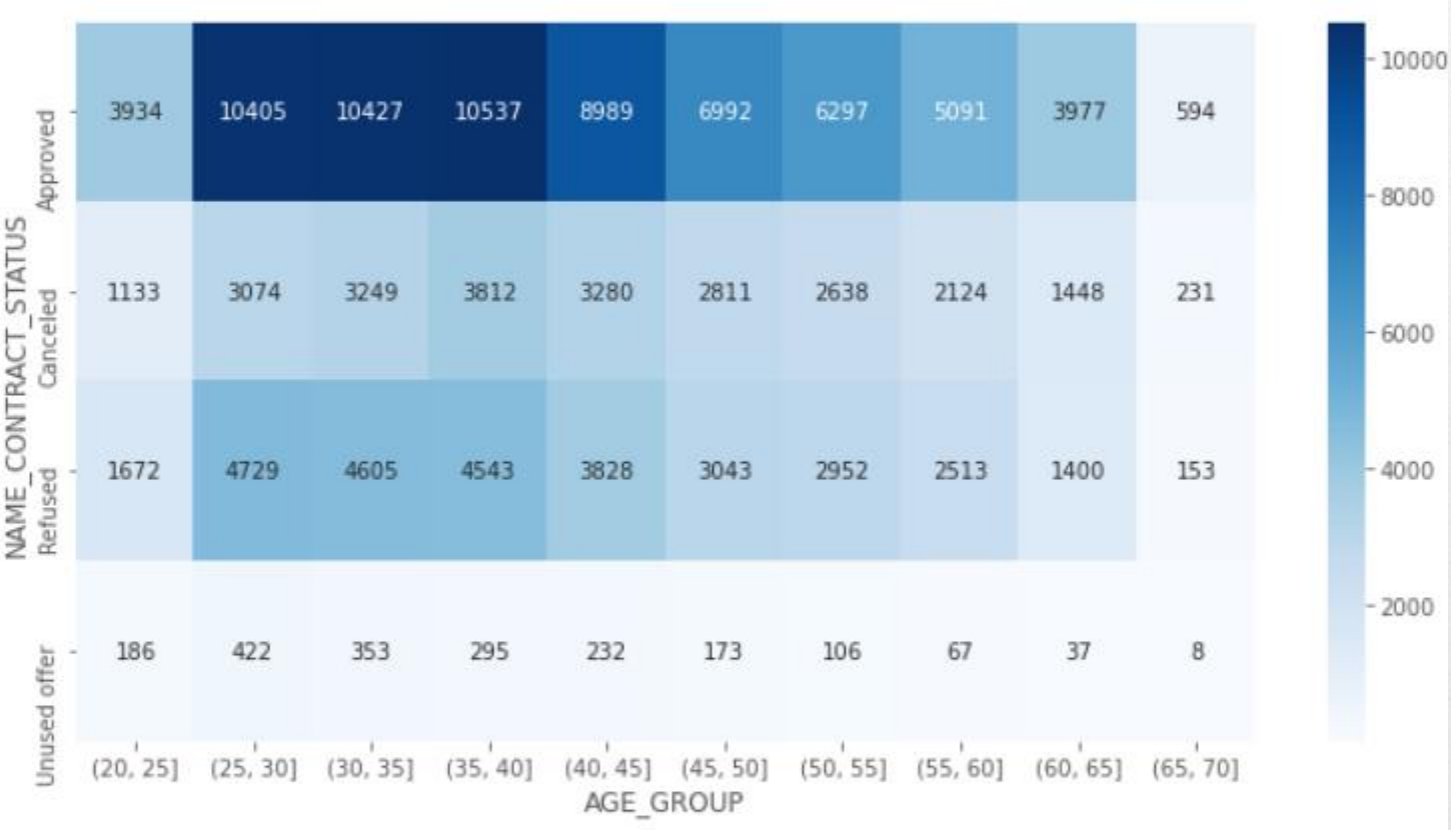

```
NAME_INCOME_TYPE      Commercial associate  Maternity leave  Pensioner  \
NAME_CONTRACT_STATUS
Approved                            198507               10     170144
Canceled                             59785                2      57674
Refused                              58117                3      43802
Unused offer                          5072                1       1924

NAME_INCOME_TYPE      State servant  Student  Unemployed  Working
NAME_CONTRACT_STATUS
Approved                      61630       20          68   455720
Canceled                      15679        3          16   126282
Refused                       15597        1          38   127832
Unused offer                   1518        0           1    14255
```

Data Analysis

G. Analysis on Joined Data

Observations : Maximum no. of approval are for age group 35-40. While most of the approved application are of people between Age 25 - 45.



Data Analysis

Inference

Decisive Factor whether an applicant will be Repayer:

- **NAME_EDUCATION_TYPE:** Academic degree has less defaults.
- **NAME_INCOME_TYPE:** Student and Businessmen have no defaults.
- **ORGANIZATION_TYPE**: Clients with Trade Type 4 and 5 and Industry type 8 have defaulted less than 3%
- **DAYS_BIRTH:** People above age of 50 have low probability of defaulting
- **DAYS_EMPLOYED**: Clients with 40+ year experience having less than 1% default rate
- **AMT_INCOME_TOTAL**: Applicant with Income more than 700,000 are less likely to default

Decisive Factor whether an applicant will be Defaulter:

- **CODE_GENDER**: Males are at relatively higher default rate
- **NAME_FAMILY_STATUS** : People who have civil marriage or who are single default a lot.
- **NAME_EDUCATION_TYPE**: People with Lower Secondary & Secondary education
- **NAME_INCOME_TYPE**: Clients who are either at Maternity leave OR Unemployed default a lot.
- **OCCUPATION_TYPE**: Avoid Low-skill Laborers, Drivers and Waiters/barmen staff, Security staff, Laborers and Cooking staff as the default rate is huge.
- **ORGANIZATION_TYPE**: Organizations with highest percent of loans not repaid are Transport: type 3 (16%), Industry: type 13 (13.5%), Industry: type 8 (12.5%) and Restaurant (less than 12%). Self-employed people have relative high defaulting rate, and thus should be avoided to be approved for loan or provide loan with higher interest rate to mitigate the risk of defaulting.
- **DAYS_BIRTH**: Avoid young people who are in age group of 20-40 as they have higher probability of defaulting
- **DAYS_EMPLOYED**: People who have less than 5 years of employment have high default rate.
- **CNT_FAM_MEMBERS**: Client who have children equal to or more than 9 default 100% and hence their applications are to be rejected.
- **AMT_GOODS_PRICE**: When the credit amount goes beyond 3M, there is an increase in defaulters.

Data Analysis

Inference

Target/focused variable for Application dataset - TARGET
Target/focused variable for Previous dataset - NAME_CONTRACT_STATUS

Top Major variables to consider for loan prediction:

- NAME_EDUCATION_TYPE
- AMT_INCOME_TOTAL
- DAYS_BIRTH
- AMT_CREDIT
- DAYS_EMPLOYED
- AMT_ANNUITY
- NAME_INCOME_TYPE
- CODE_GENDER
- NAME_HOUSING_TYP

Data Analysis

Inference

Defaulters' Characteristics :

All the below variables were established in analysis of Application data frame as leading to default.
Checked these against the Approved loans which have defaults, and it proves to be correct

- Medium income
- 25-35 years olds, followed by 35-45 years age group
- Male
- Unemployed
- Labourers, Salesman, Drivers
- Own House - No

Other IMPORTANT Factors to be considered :

- No of Bureau Hits in last week. Month etc – zero hits is good
- Amount income not correspondingly equivalent to Good Bought – Income low and good value high is a concern
- Previous applications with Refused, Cancelled, Unused loans also have default which is a matter of concern. This indicates that the financial company had Refused/Cancelled previous application but has approved the current and is facing default on these.
- Credible Applications refused
- Unused applications have lower loan amount. Is this the reason for no usage?
- Female applicants should be given extra weightage as defaults are lesser.
- 60% of defaulters are Working applicants. This does not mean working applicants must be refused. Proper scrutiny of other parameters needed
- Previous applications with Refused, Cancelled, Unused loans also have cases where payments are coming on time in current application. This indicates that possibly wrong decisions were done in those cases.