Credit EDA Assignment

DEEPAK ROUT

Data Cleaning

- Prepared a data frame of missing value

```
null_df = pd.DataFrame(list(zip(null_col.index, null_col.values)), columns=['Column Name', '% NULL'])
```

```
null_df[null_df['% NULL'] > 50].shape
```

```
(41, 2)
```

- Above snippet shows 41 column with more than 50% data missing. Which is huge. We can't draw insightful idea from these column.

```
# Dropping all columns with 50% data missing
for col in null_df_50['Column Name']:
    df_ad.drop(col, axis=1, inplace=True)
```

```
# final shape after deletion, 122- 41
df_ad.shape
```

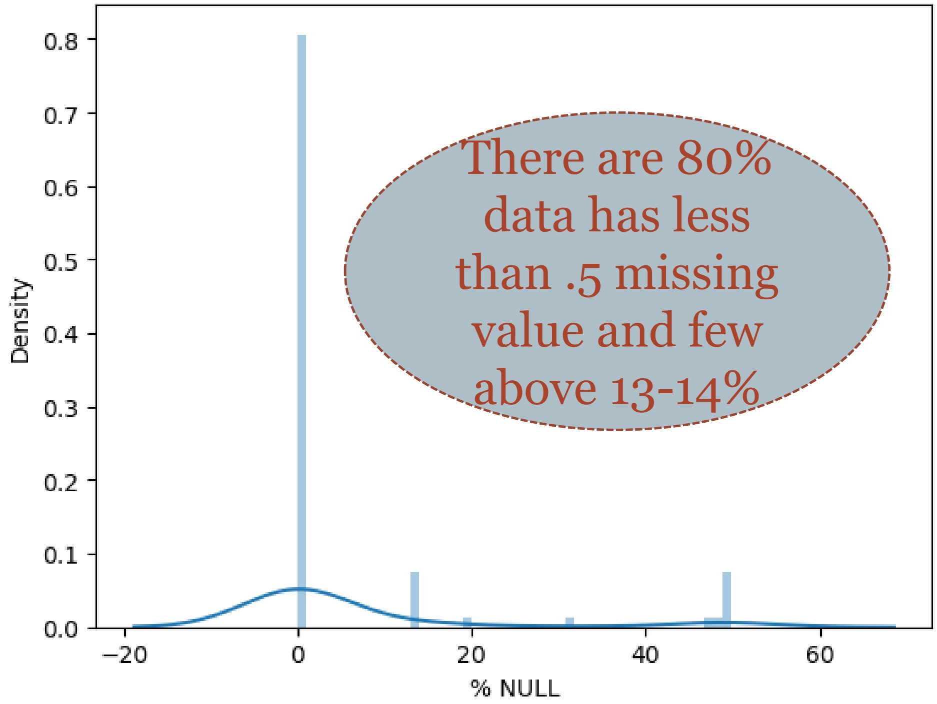
```
(307511, 81)
```

Visualizing % data missing

```
bins = [-0.5, 0.0,0.5,1.0,2.0,3.0,5.0,10.0, 20.0, 100.0,float('inf')]
labels = ['-1.0','0.0','0.5','1.0','2.0','3.0','5.0','10.0', '20.0', '100.0']
null_df['% NULL Group'] = pd.cut(null_df['% NULL'], bins, labels)
```

```
sns.distplot(null_df['% NULL'])
```

```
<AxesSubplot:xlabel='% NULL', ylabel='Density'>
```



Analysing missing Value below 13%

Only 10 columns with missing value 0.0 to 0.5, Analysis on missing data

- MCAR data need to be left out from analysis
- MAR data need a relation for missing

```
]: columns = null_df[(null_df['% NULL'] > 0.0) & (null_df['% NULL'] <= .5)]['Column Name']
for col in columns:
    print(col, df_ad[col].dtype)
```

```
AMT_ANNUITY float64
AMT_GOODS_PRICE float64
NAME_TYPE_SUITE object
CNT_FAM_MEMBERS float64
EXT_SOURCE_2 float64
OBS_30_CNT_SOCIAL_CIRCLE float64
DEF_30_CNT_SOCIAL_CIRCLE float64
OBS_60_CNT_SOCIAL_CIRCLE float64
DEF_60_CNT_SOCIAL_CIRCLE float64
DAYS_LAST_PHONE_CHANGE float64
```

missing Value analysis : NAME_TYPE_SUITE

```
df_ad.NAME_TYPE_SUITE.value_counts(normalize=True)
```

```
Unaccompanied      0.811596
Family             0.131112
Spouse, partner    0.037130
Children           0.010669
Other_B            0.005780
Other_A            0.002828
Group of people    0.000885
Name: NAME_TYPE_SUITE, dtype: float64
```

```
suite_mode = df_ad.NAME_TYPE_SUITE.mode()[0]
suite_mode
```

```
'Unaccompanied'
```

Most data are 'Unaccompanied', categorical data can be filled with mode and 81% are 'Unaccompanied'.

```
df_ad.NAME_TYPE_SUITE.fillna(suite_mode, inplace=True)
```

```
df_ad.NAME_TYPE_SUITE.value_counts(normalize=True)
```

```
Unaccompanied      0.812217
Family             0.130679
Spouse, partner    0.037008
Children           0.010634
Other_B            0.005761
Other_A            0.002819
Group of people    0.000882
Name: NAME_TYPE_SUITE, dtype: float64
```

Fixing invalid Value

Fixing Invalid Values

DAYS_EMPLOYED

days of employment can't be negative

```
df_ad.DAYS_EMPLOYED = df_ad.DAYS_EMPLOYED.apply(lambda x: abs(x))
```

DAYS_BIRTH

days of birth can't be negative

```
df_ad.DAYS_BIRTH = df_ad.DAYS_BIRTH.apply(lambda x: abs(x))
```

DAYS_REGISTRATION

```
df_ad.DAYS_REGISTRATION = df_ad.DAYS_REGISTRATION.apply(lambda x: abs(x))
```

DAYS_ID_PUBLISH

```
df_ad.DAYS_ID_PUBLISH = df_ad.DAYS_ID_PUBLISH.apply(lambda x: abs(x))
```

No Data type change is required.

all column are assigned with right data type.

Outlier

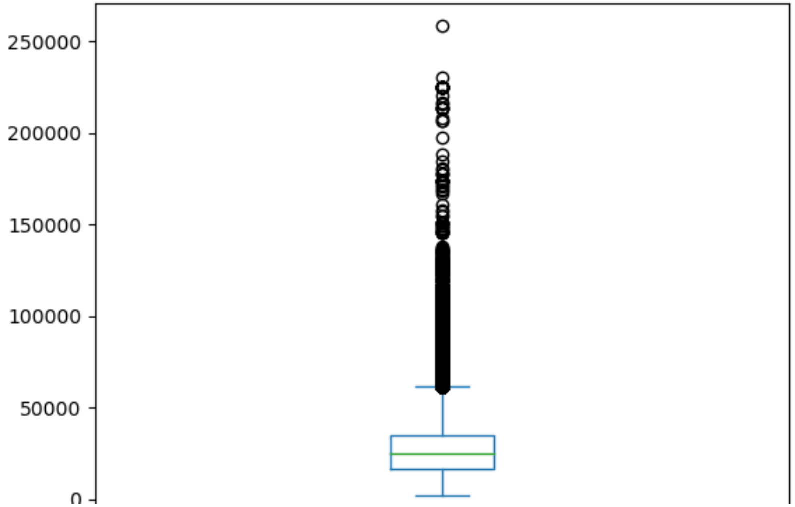
AMT_ANNUIT

```
df_ad.AMT_ANNUITY.describe()
```

```
count    307221.000000
mean      27120.452357
std       14492.106811
min        1615.500000
25%       16551.000000
50%       24916.500000
75%       34596.000000
max      258025.500000
Name: AMT_ANNUITY, dtype: float64
```

```
df_ad.AMT_ANNUITY.plot.box()
```

<AxesSubplot:>



We could see from the boxplot there are outlier.
Look at some percentile. It is evident.

```
p75 = df_ad.AMT_ANNUITY.quantile([.75]).values[0]
p25 = df_ad.AMT_ANNUITY.quantile([.25]).values[0]
p75 + 1.5*(p75-p25)
```

```
61663.5
```

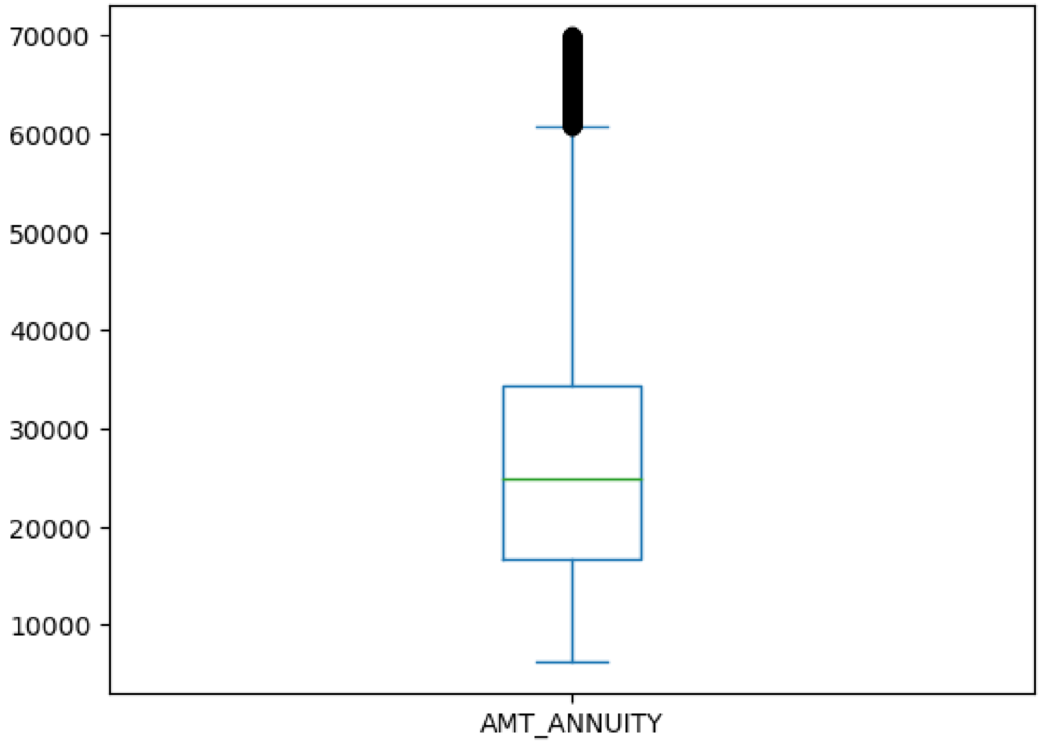
```
percentile = df_ad.AMT_ANNUITY.quantile([0.01, 0.05, .90, 0.9
percentile
```

```
0.01     6178.5
0.05     9000.0
0.90    45954.0
0.95    53325.0
0.99    70006.5
Name: AMT_ANNUITY, dtype: float64
```

If we treat AMT_ANNUITY

```
df_ad.AMT_ANNUITY[(df_ad.AMT_ANNUITY > percentile.values[0]) & (df_ad.AMT_ANNUITY < percentile.values[-1])].plot.box()
```

```
<AxesSubplot:>
```

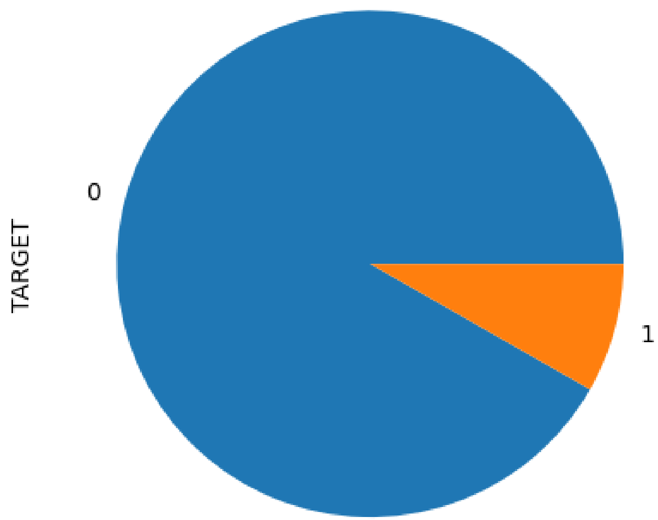


Other Outlier

- AMT_INCOME_TOTAL
- AMT_CREDIT
- DAYS_EMPLOYED
  - These data exceeds 1000yrs. Which can be treated as missing value. Days employed used to create years_employed.
  - Most people are employed less than 10yrs
- AGE
  - These data exceeds 1000yrs. Which can be treated as missing value. Days employed used to create years_employed
  - Most people are between 30-50.

Target Imbalance

```
df_ad.TARGET.value_counts(normalize=True).plot.pie()
plt.show()
```




```
df_ad.TARGET.value_counts(normalize=True)
```

```
0    0.917494
1    0.082506
Name: TARGET, dtype: float64
```

Creating two separate dataset for analysis

Defaulter vs Non-Defaulter

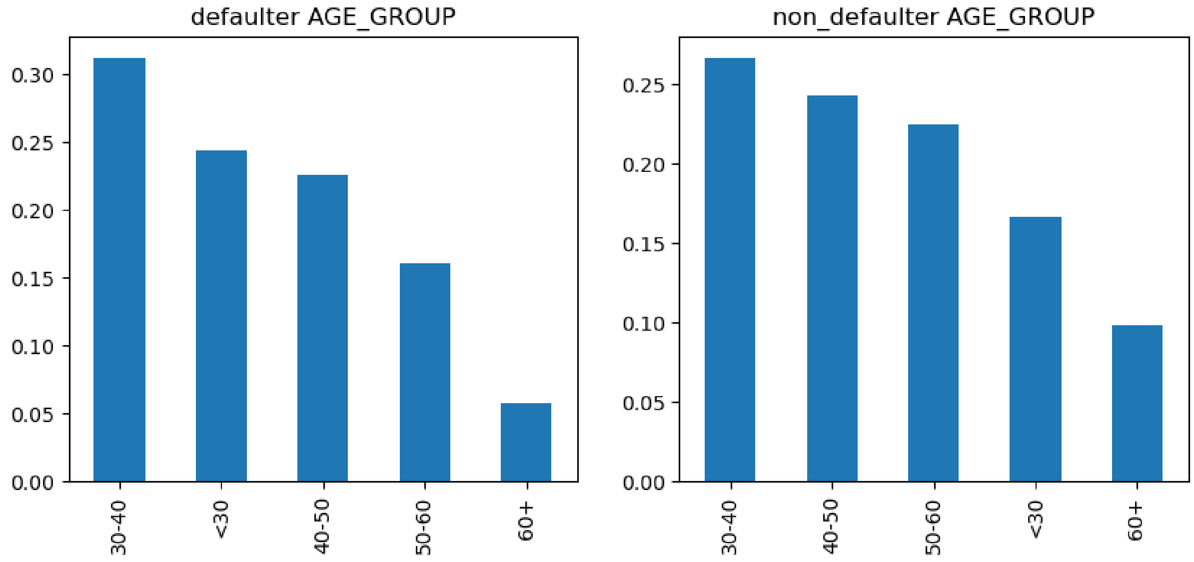
```
defaulter = df_ad[df_ad.TARGET == 1]
```

```
non_defaulter = df_ad[df_ad.TARGET == 0]
```

Univariate analysis

Univariate Analysis on both Defaulter and non defaulter Dataset

```
fig = plt.figure(figsize=[10,4])
ax1  = fig.add_subplot(121)
defaulter.AGE_GROUP.value_counts(normalize= True).plot.bar()
ax2 = fig.add_subplot(122)
non_defaulter.AGE_GROUP.value_counts(normalize= True).plot.bar()
ax1.title.set_text('defaulter AGE_GROUP')
ax2.title.set_text('non_defaulter AGE_GROUP')
plt.show()
```



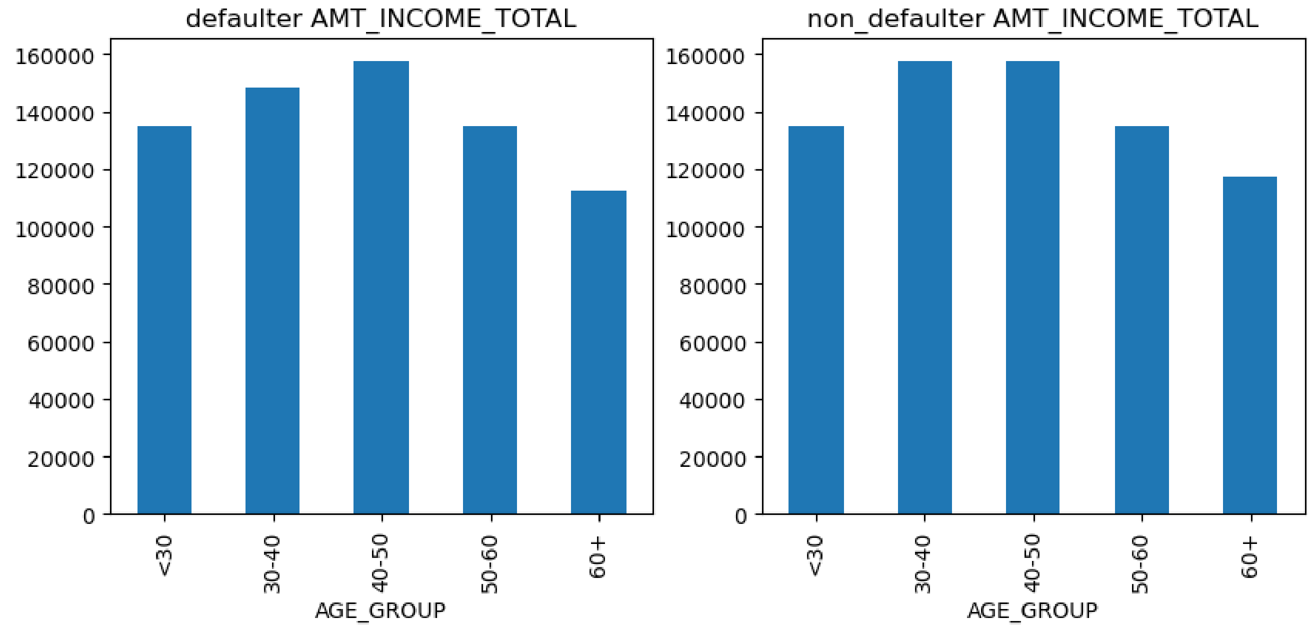
- as there more people around 30-40, it is in highest % in both category, deaulter and non-defaulter

- But population under 30 is less , defaulter is at highest

Age Group vs Income

age_group VS income

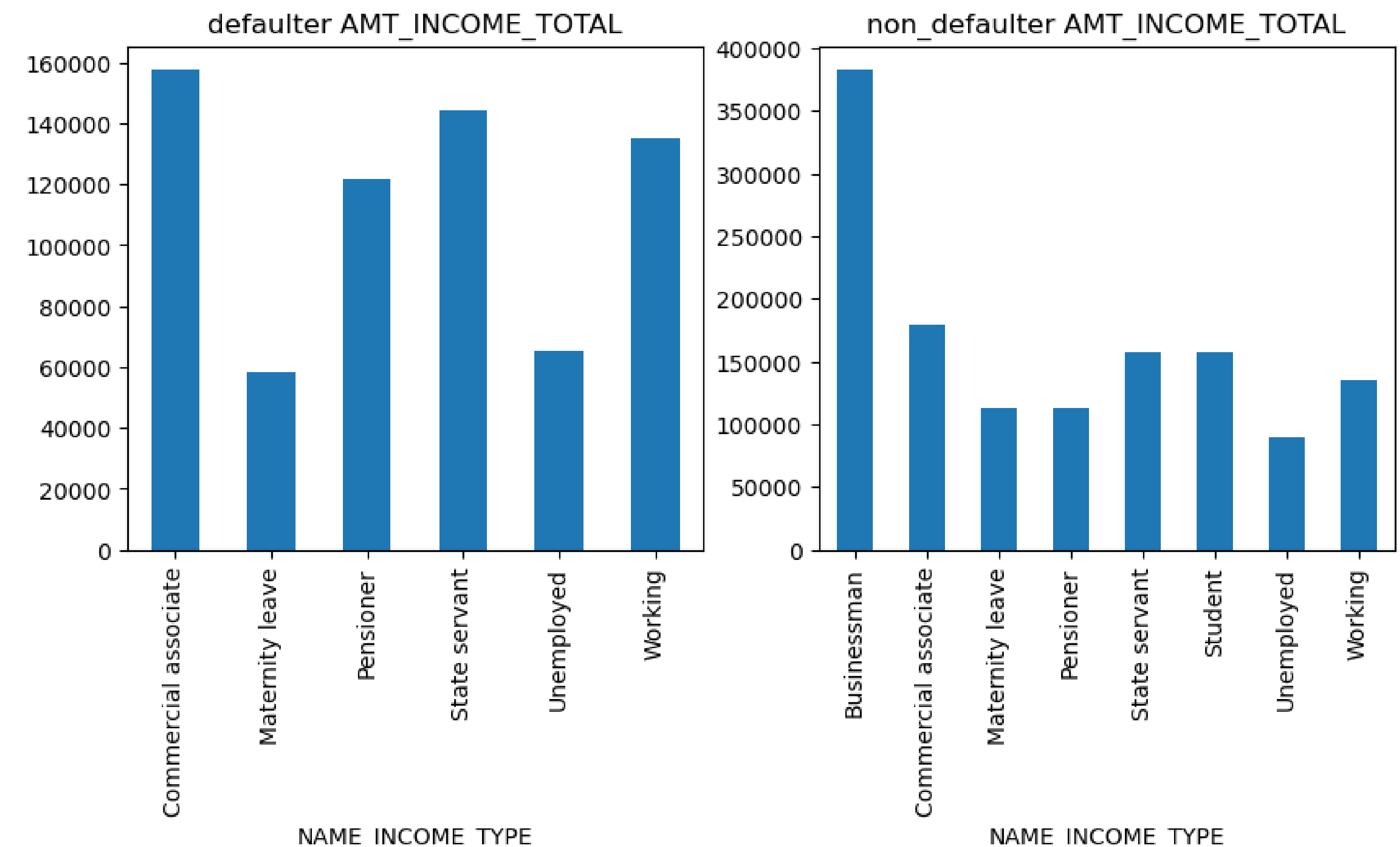
```
fig = plt.figure(figsize=[10,4])
ax1  = fig.add_subplot(121)
defaulter.groupby('AGE_GROUP')['AMT_INCOME_TOTAL'].median().plot.bar()
ax2 = fig.add_subplot(122)
non_defaulter.groupby('AGE_GROUP')['AMT_INCOME_TOTAL'].median().plot.bar()
ax1.title.set_text('defaulter AMT_INCOME_TOTAL')
ax2.title.set_text('non_defaulter AMT_INCOME_TOTAL')
plt.show()
```



Few age group was more defaulter, income ain't the Reason. 30-50 earns Good.

NAME_INCOME_TYPE vs AMT_INCOME_TOTAL

```
fig = plt.figure(figsize=[10,4])
ax1  = fig.add_subplot(121)
defaulter.groupby('NAME_INCOME_TYPE')['AMT_INCOME_TOTAL'].median().plot.bar()
ax2 = fig.add_subplot(122)
non_defaulter.groupby('NAME_INCOME_TYPE')['AMT_INCOME_TOTAL'].median().plot.bar()
ax1.title.set_text('defaulter AMT_INCOME_TOTAL')
ax2.title.set_text('non_defaulter AMT_INCOME_TOTAL')
plt.show()
```



Business man are not Defaulter

High Earners

- We can’t say high earners are defaulter.
- Commercial Associate are the defaulter.

Female Vs Male

```
df_ad.CODE_GENDER.value_counts()
```

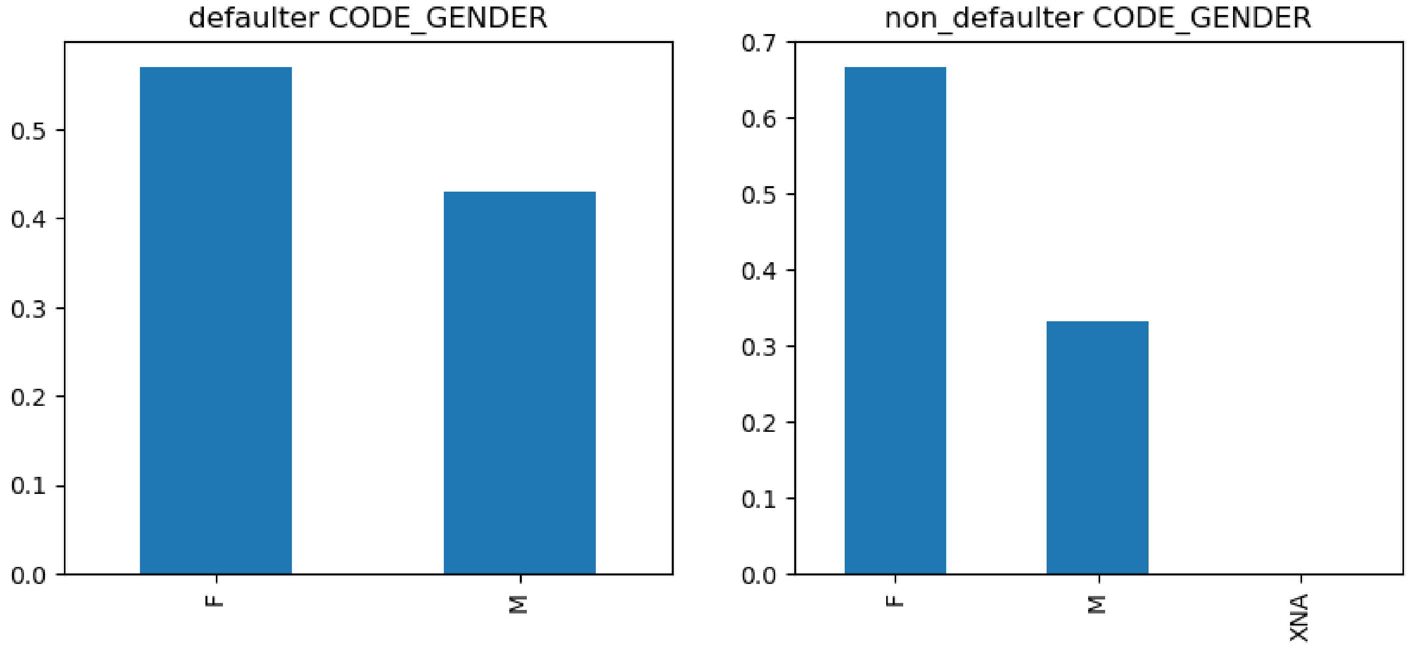
```
F      193274
M       99805
XNA         4
Name: CODE_GENDER, dtype: int64
```

not much to deduce from above, higher female number leads higher %.

Female Vs Male

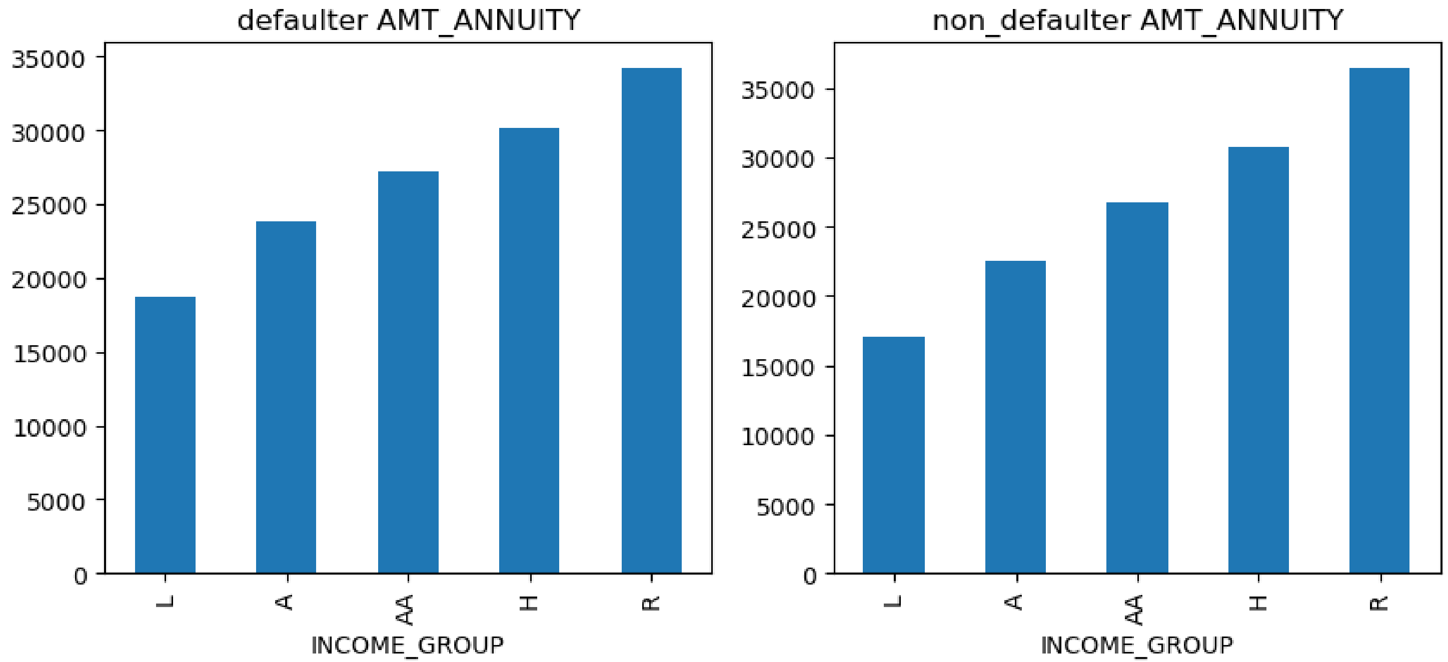
```
fig = plt.figure(figsize=[10,4])
ax1  = fig.add_subplot(121)
defaulter.CODE_GENDER.value_counts(normalize=True).plot.bar()
ax2 = fig.add_subplot(122)
non_defaulter.CODE_GENDER.value_counts(normalize=True).plot.bar()
ax1.title.set_text('defaulter CODE_GENDER')
ax2.title.set_text('non_defaulter CODE_GENDER')
plt.show()
```

Female to male ratio is high. So both defaulter and non defaulter ratio is high.



INCOME_GROUP TYPE vs AMT_ANNUITY

```
fig = plt.figure(figsize=[10,4])
ax1  = fig.add_subplot(121)
defaulter.groupby('INCOME_GROUP')['AMT_ANNUITY'].median().plot.bar()
ax2 = fig.add_subplot(122)
non_defaulter.groupby('INCOME_GROUP')['AMT_ANNUITY'].median().plot.bar()
ax1.title.set_text('defaulter AMT_ANNUITY')
ax2.title.set_text('non_defaulter AMT_ANNUITY')
plt.show()
```



Higher the income higher annuity is being issued

TARGET, AMT_ANNUITY, AMT_INCOME_TOTAL

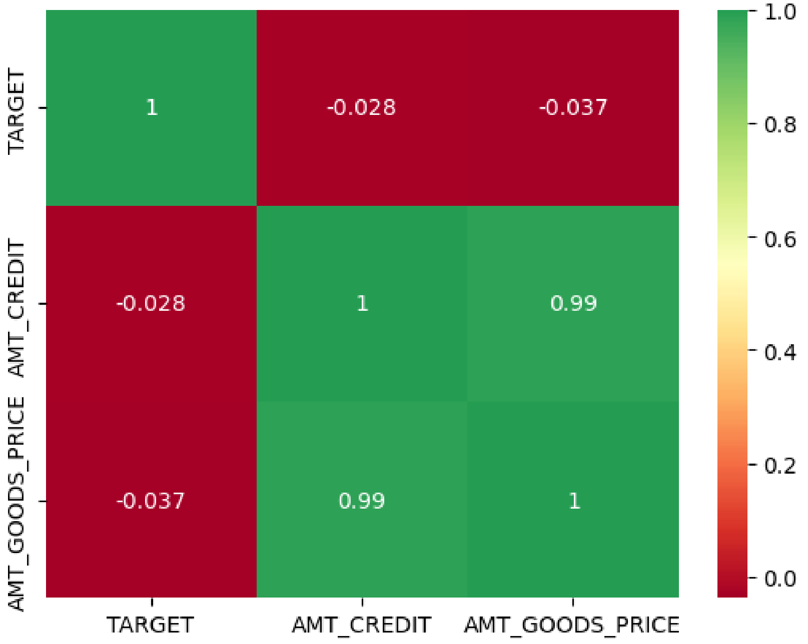
```
sns.heatmap(df_ad[['TARGET', 'AMT_ANNUITY', 'AMT_INCOME_TOTAL']].corr(), annot= True, cmap= "RdYlGn", center=.3)
plt.show()
```

There is +ve relation between Income, and anuity but there is negative relation between target vs anuity and income

- higher the income lower the defaulter
- higher the anuity lower the defaulter

'TARGET, AMT_CREDIT, AMT_GOODS_PRICE

```
sns.heatmap(df_ad[['TARGET', 'AMT_CREDIT', 'AMT_GOODS_PRICE']].corr(), annot= True, cmap= "RdYlGn", center=.55)
plt.show()
```



There is +ve relation between AMT_CREDIT, and AMT_GOODS_PRICE but there is negative relation between target vs AMT_GOODS_PRICE and AMT_CREDIT

- higher AMT_CREDIT lower the defaulter
- higher AMT_GOODS_PRICE lower the defaulter

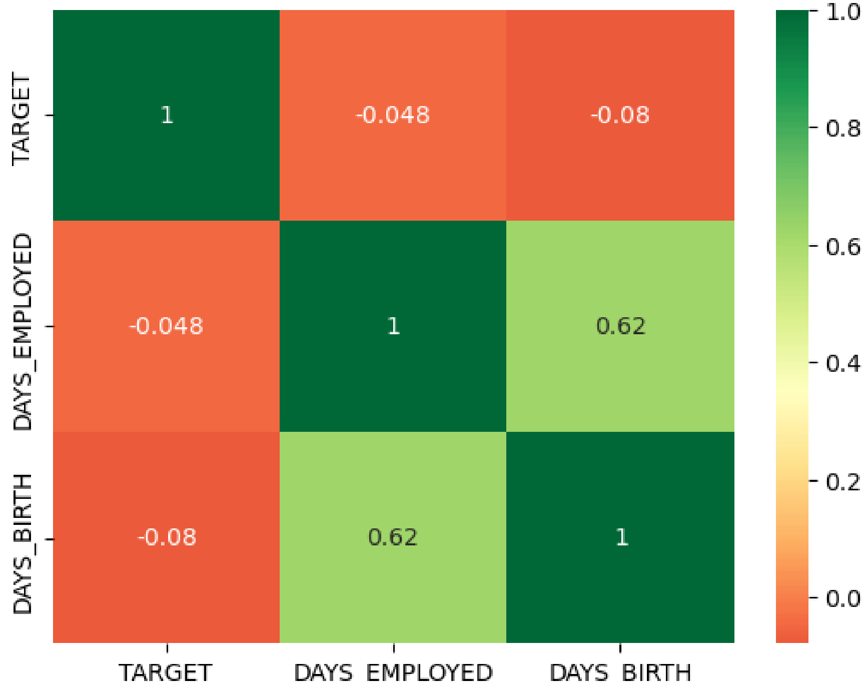
Good price might be depicting quality product

```
sns.heatmap(df_ad[['TARGET', 'REGION_POPULATION_RELATIVE', 'AMT_INCOME_TOTAL']].corr(), annot= True, cmap= "RdYlGn", center=.35)
plt.show()
```

- Highly polulated region must be among with city, where it has less defaulter.
- Highly populated regio are high earning place

TARGET, DAYS_EMPLOYED, DAYS_BIRTH

```
sns.heatmap(df_ad[['TARGET', 'DAYS_EMPLOYED', 'DAYS_BIRTH']].corr(), annot= True, cmap= "RdYlGn", center=.35)
plt.show()
```



- elder person with more work experience are tends to default less

Previous Application Data Set

```
df_prev_ds.shape
```

```
(1670214, 37)
```

```
prev_col = df_prev_ds.isna().sum()*100/df_prev_ds.shape[0]
```

```
prev_null_df = pd.DataFrame(list(zip(prev_col.index, prev_col.values)), columns=['Column Name', '% NULL'])
```

```
prev_null_40 = prev_null_df[prev_null_df['% NULL'] > 49]
prev_null_40
```

	Column Name	% NULL
6	AMT_DOWN_PAYMENT	53.636480
12	RATE_DOWN_PAYMENT	53.636480
13	RATE_INTEREST_PRIMARY	99.643698
14	RATE_INTEREST_PRIVILEGED	99.643698
20	NAME_TYPE_SUITE	49.119754

```
# Dropping all columns with 50% data missing
for col in prev_null_40['Column Name']:
    df_prev_ds.drop(col, axis=1, inplace=True)
```

```
df_prev_ds.shape
```

```
(1670214, 32)
```

Treating Missing Type

```
df_prev_ds.AMT_CREDIT.dtype # data type suppose to float but it shows object
```

```
dtype('O')
```

```
df_prev_ds[df_prev_ds.AMT_CREDIT.apply(lambda x:  isinstance(x,float)) != True]
```

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_GOODS_PRICE	WEEKDAY_APPR_PROCESS_START	HOUR_APPR_PROCESS_START	FLAG_LAST_APPL_PER
1127152	2204450	438387	Revolving loans	0.0	0.0	Cash	Cash	FRIDAY	10	

Remove Cash from AMT_CREDIT

```
# data type suppose to float but it shows object
df_prev_ds = df_prev_ds[df_prev_ds.AMT_CREDIT.apply(lambda x:  isinstance(x,float)) == True]
df_prev_ds.shape
```

```
(1670213, 32)
```

AMT_GOODS_PRICE

```
# data type suppose to float but it shows object
df_prev_ds[df_prev_ds.AMT_GOODS_PRICE.apply(lambda x:  isinstance(x,float)) != True].shape[0]/df_prev_ds.shape[0]
```

```
0.23081726701923647
```

Invalid values that provide no information regarding true value, better to treat them as missing value

missing value is nearly 23%. better to leave it as it is

```
df_prev_ds[df_prev_ds.AMT_ANNUITY.apply(lambda x:  isinstance(x,float)) != True].shape[0]/df_prev_ds.shape[0]
```

```
0.22286678405688376
```

```
df_prev_ds[df_prev_ds.AMT_ANNUITY.apply(lambda x:  isinstance(x,float)) != True].shape[0]/df_prev_ds.shape[0]
```

```
0.22286678405688376
```

Invalid values that provide no information regarding true value, better to treat them as missing value

missing value is nearly 22%. better to leave it as it is

Invalid Data

DAYS_DECISION

```
#correcting negative value
df_prev_ds.DAYS_DECISION = df_prev_ds.DAYS_DECISION.apply(lambda x: abs(x))
```

DAYS_FIRST_DUE

```
#correcting negative value
df_prev_ds.DAYS_FIRST_DUE.fillna(0, inplace=True)
df_prev_ds[df_prev_ds.DAYS_FIRST_DUE == 'Cash'].DAYS_FIRST_DUE.shape[0]/df_prev_ds.shape[0]
```

```
0.4029809371619069
```

nearly 40% data entered wrong, better to leave as it. Not to include in our analysis

```
df_prev_ds.DAYS_LAST_DUE.fillna(0, inplace=True)
df_prev_ds[df_prev_ds.DAYS_LAST_DUE == 'Cash'].DAYS_LAST_DUE.shape[0]/df_prev_ds.shape[0]
```

```
0.4029809371619069
```

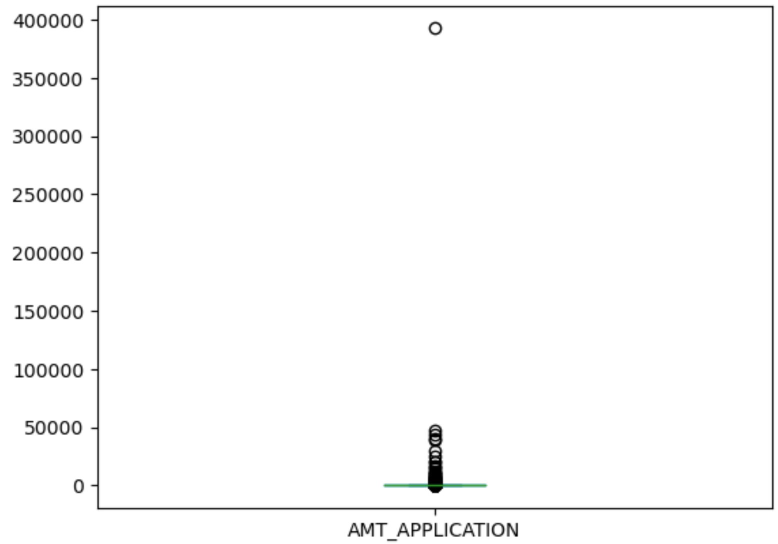
nearly 40% data entered wrong, better to leave as it.

Missing value marked as Cash.

Outlier

```
df_prev_ds.AMT_APPLICATION.value_counts().plot.box()
```

```
<AxesSubplot:>
```




```
df_prev_ds.AMT_APPLICATION.quantile([.5,.7,.8,.9,.95,.99])
```

```
0.50      71046.0
0.70     144769.5
0.80     228937.5
0.90     450000.0
0.95     787500.0
0.99    1350000.0
Name: AMT_APPLICATION, dtype: float64
```

```
df_prev_ds[df_prev_ds.AMT_APPLICATION < 450000].AMT_APPLICATION.plot.box()
```

AMT_APPLICATION continue

```
df_prev_ds[df_prev_ds.AMT_APPLICATION < 450000].AMT_APPLICATION.describe()
```

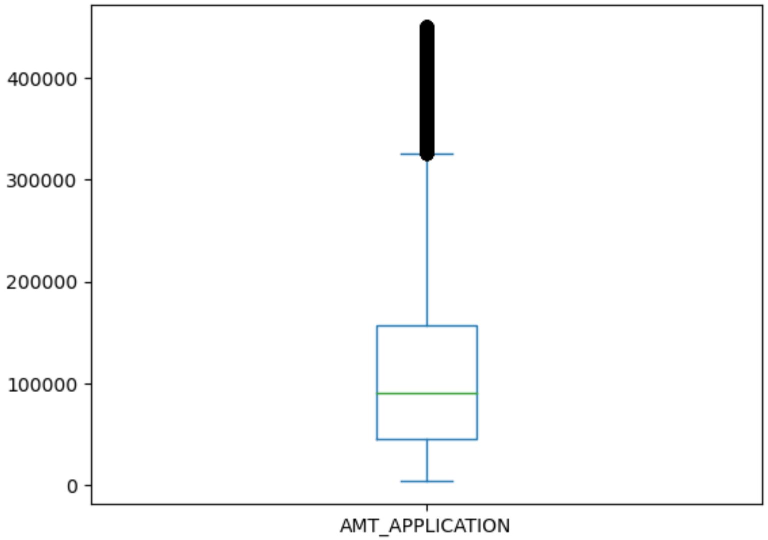
```
count    1.466539e+06
mean     8.567685e+04
std      9.347516e+04
min      0.000000e+00
25%      0.000000e+00
50%      5.526000e+04
75%      1.319895e+05
max      4.499955e+05
Name: AMT_APPLICATION, dtype: float64
```

There is lot many missing value in AMT_APPLICATION

- Nearly 25% data are 0

```
df_prev_ds[(df_prev_ds.AMT_APPLICATION < 450000) & (df_prev_ds.AMT_APPLICATION >0)].AMT_APPLICATION.plot.box()
```

```
<AxesSubplot:>
```



We spot two issues with AMT_APPLICATION

- 25% data are missing
- We need to cap the higher amount too

SELLERPLACE_AREA

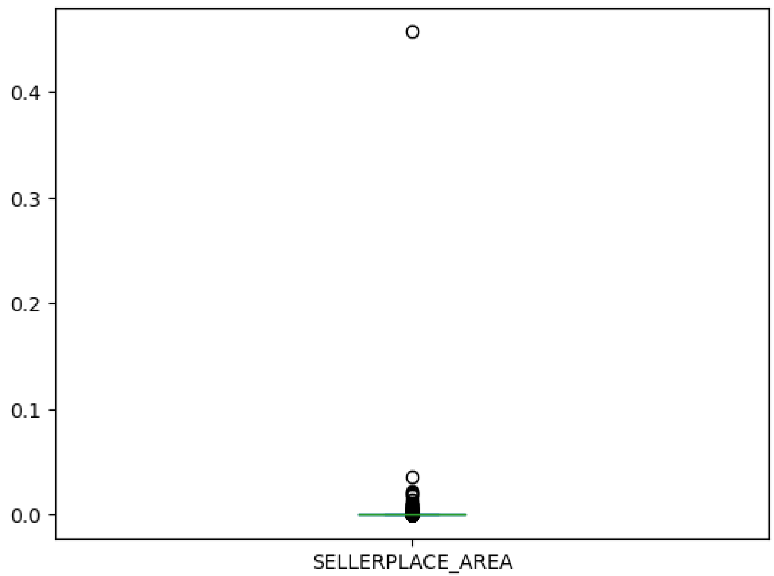
SELLERPLACE_AREA

```
df_prev_ds.SELLERPLACE_AREA.describe()
```

```
count    1.670213e+06
mean     3.139513e+02
std      7.127446e+03
min     -1.000000e+00
25%     -1.000000e+00
50%      3.000000e+00
75%      8.200000e+01
max      4.000000e+06
Name: SELLERPLACE_AREA, dtype: float64
```

```
df_prev_ds.SELLERPLACE_AREA.value_counts(normalize= True).plot.box()
```

```
<AxesSubplot:>
```



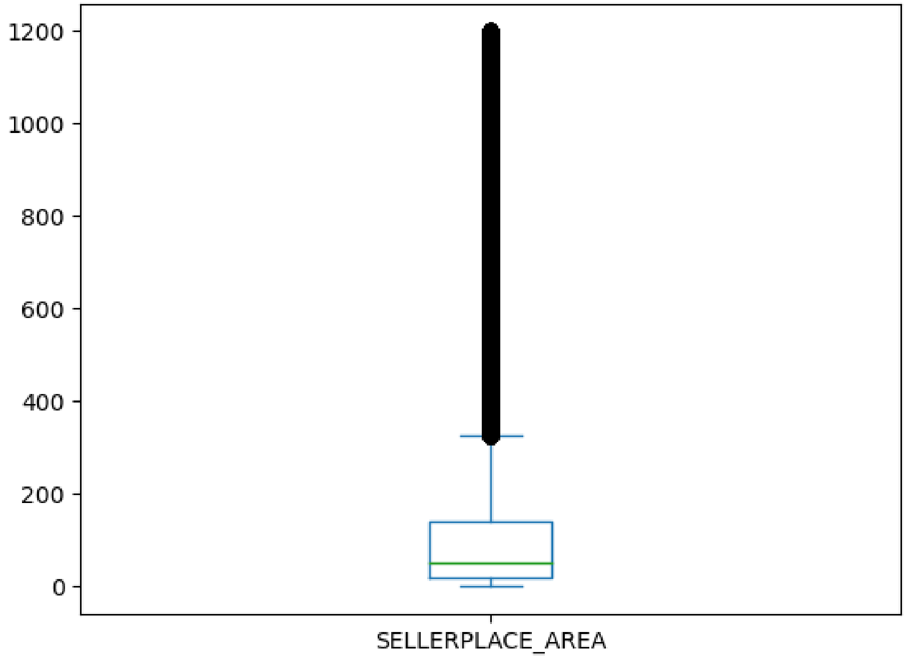
We can spot similar issue with SELLERPLACE_AREA

- There is missing value marked as -1.

SELLERPLACE_AREA

```
df_prev_ds.SELLERPLACE_AREA[(df_prev_ds.SELLERPLACE_AREA >-1) & (df_prev_ds.SELLERPLACE_AREA < 1200)].plot.box()
```

```
<AxesSubplot:>
```



We need to treat both upper fence outlier and missing value

There is lot many missing value in SELLERPLACE_AREA, which marked as -1. They should not be taken into analysis

- Nearly 25% data are -1

Merge Data

Merging two data set

```
[ ]: final = pd.merge(left=df_ad,right=df_prev_ds, how='inner', left_on='SK_ID_CURR', right_on='SK_ID_CURR')
     final.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT_x	AMT_ANNUITY_x	AMT_GOODS_PRICE_x	NAME_TYPE_SUITE	NAM
0	100002	1	Cash loans	M	N	Y	0	202500.0	406597.5	24700.5	351000.0	Unaccompanied	
1	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	35698.5	1129500.0	Family	
2	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	35698.5	1129500.0	Family	
3	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	35698.5	1129500.0	Family	
4	100004	0	Revolving loans	M	Y	Y	0	67500.0	135000.0	6750.0	135000.0	Unaccompanied	

```
[ ]: final.shape
```

```
[ ]: (1350541, 117)
```

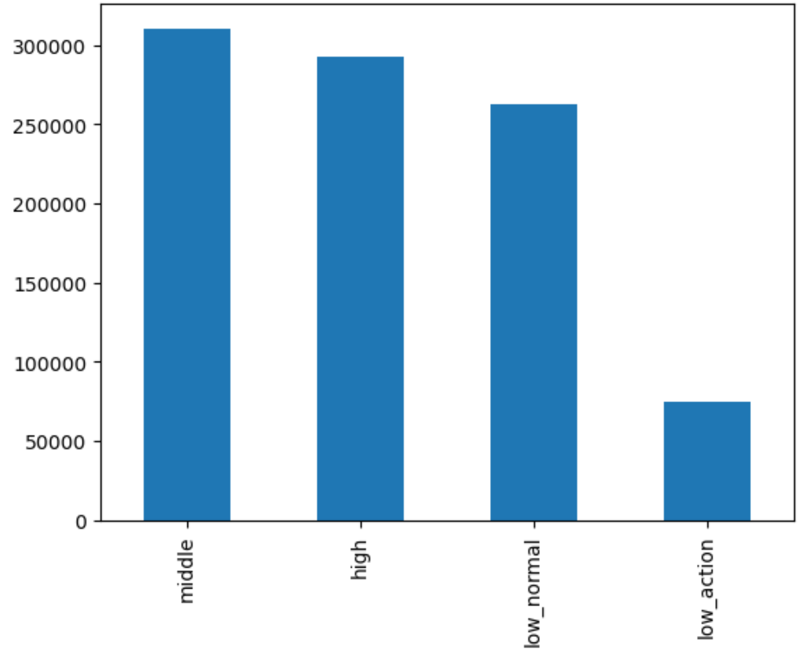
NAME_YIELD_GROUP

```
final.NAME_YIELD_GROUP.isna().sum()
```

```
0
```

```
final[final['NAME_YIELD_GROUP'] != 'XNA']['NAME_YIELD_GROUP'].value_counts().plot.bar()
```

```
<AxesSubplot:>
```



Middle and high interest rate sold more

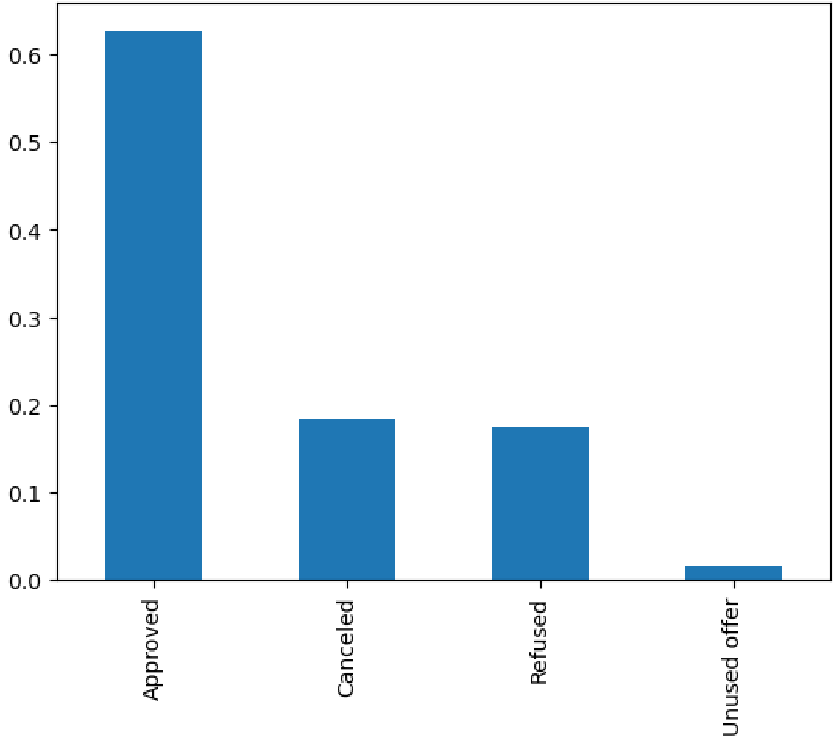
NAME_CONTRACT_STATUS Vs TARGET

```
contract_status = final['NAME_CONTRACT_STATUS'].unique()
contract_status
```

```
array(['Approved', 'Canceled', 'Refused', 'Unused offer'], dtype=object)
```

```
final['NAME_CONTRACT_STATUS'].value_counts(normalize=True).plot.bar()
```

```
<AxesSubplot:>
```

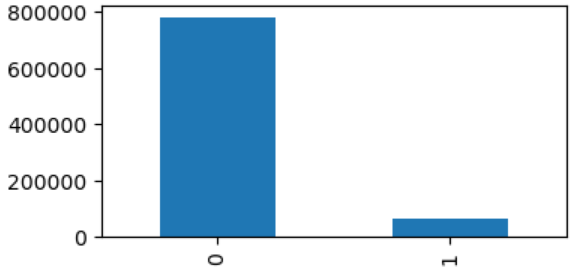


Major Contract status is Approved

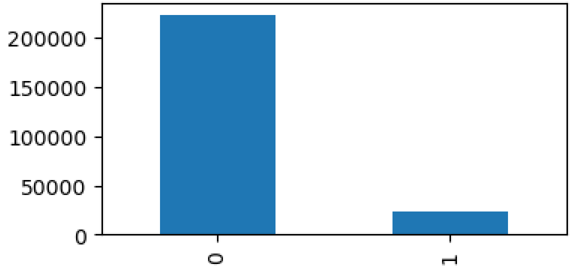
Continue

```
for status in contract_status:
    plt.figure(figsize=[4, 2])
    print(status, final[final['NAME_CONTRACT_STATUS'] == status].TARGET.value_counts(normalize=True))
    final[final['NAME_CONTRACT_STATUS'] == status].TARGET.value_counts().plot.bar()
    plt.show()
```

```
Approved 0    0.92263
1    0.07737
Name: TARGET, dtype: float64
```

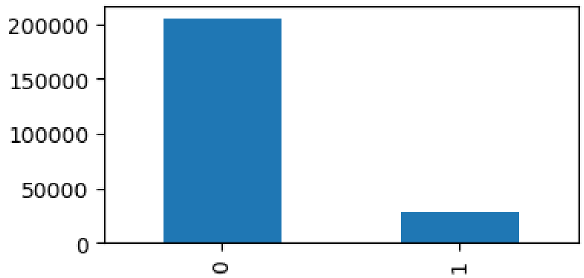



```
Canceled 0    0.906181
1    0.093819
Name: TARGET, dtype: float64
```

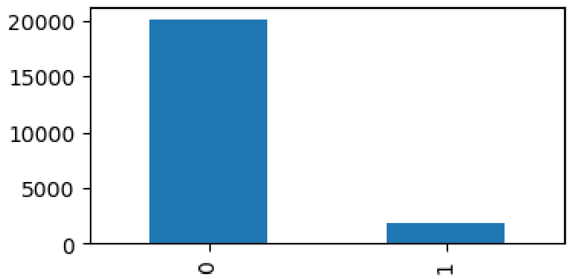


Continue

```
Refused 0    0.877459
1    0.122541
Name: TARGET, dtype: float64
```




```
Unused offer 0    0.916284
1    0.083716
Name: TARGET, dtype: float64
```

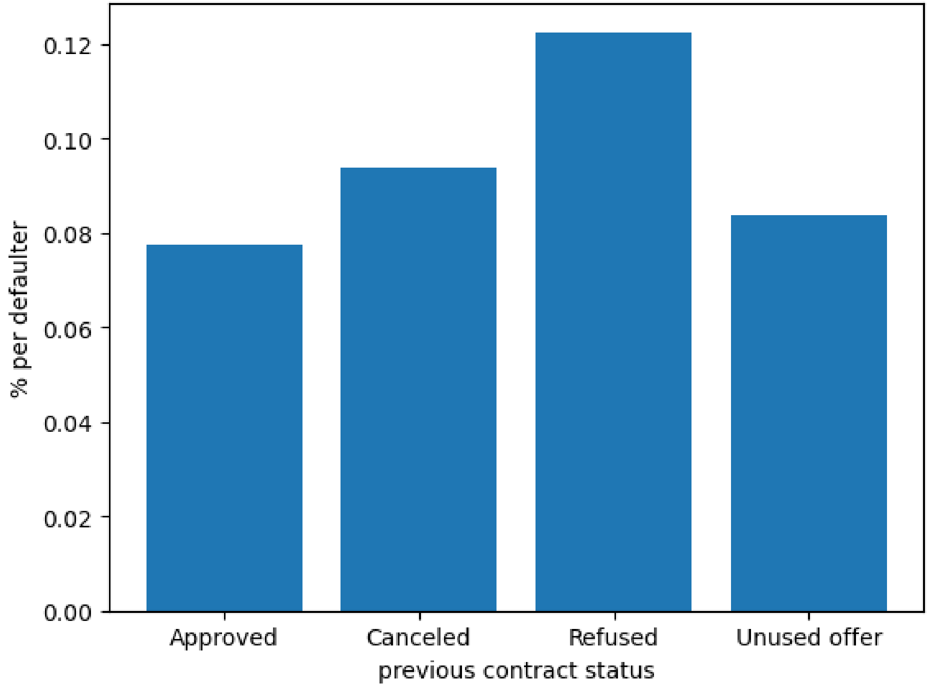


7.7 % defaulter from Approved loans, but again there is majority contract status is approved
Simillar trends in other group too. These are lower in number compare to approved.
Major concern why cancelled, refused and unused person given loan current time.

Continue with Bivariate

```
total = final['NAME_CONTRACT_STATUS'].value_counts().sort_index()
target_count = final.groupby('NAME_CONTRACT_STATUS')['TARGET'].sum().sort_index()
plt.bar(x=total.index, height= target_count.values/total.values)
plt.xlabel("previous contract status")
plt.ylabel("% per defaulter")
```

```
Text(0, 0.5, '% per defaulter')
```

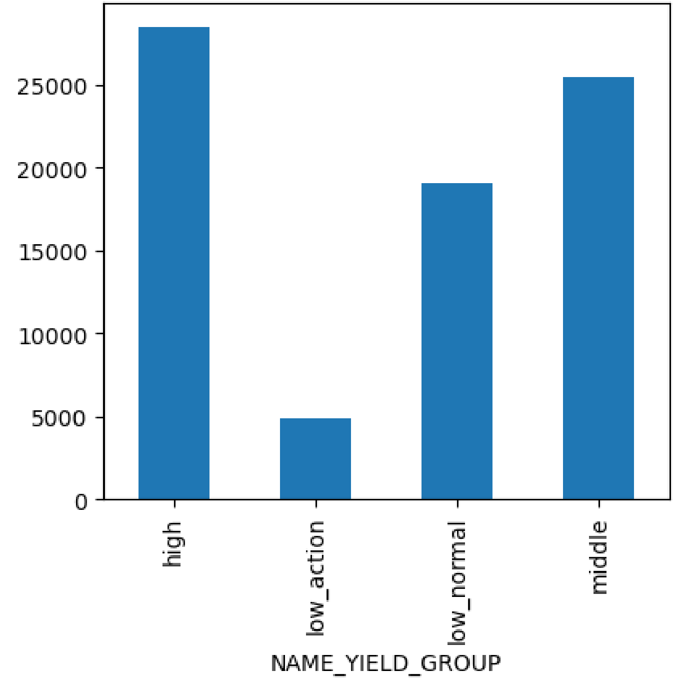


If we look at normalize data, it shows Refused for loan in previous application are most defaulter. A strong scrutiny required on approving loan.

Bivariate

```
fig = plt.figure(figsize=[10,4])
ax1  = fig.add_subplot(121)
final[final['NAME_YIELD_GROUP'] != 'XNA'].groupby('NAME_YIELD_GROUP')['TARGET'].sum().plot.bar()

plt.show()
```



High and middle yield group are higest defaulter

Correlation of OWN_CAR, OWN_REALTY

```
import numpy as np
final["OWN_CAR"]=np.where(final.FLAG_OWN_CAR=="Y", 1, 0)
```

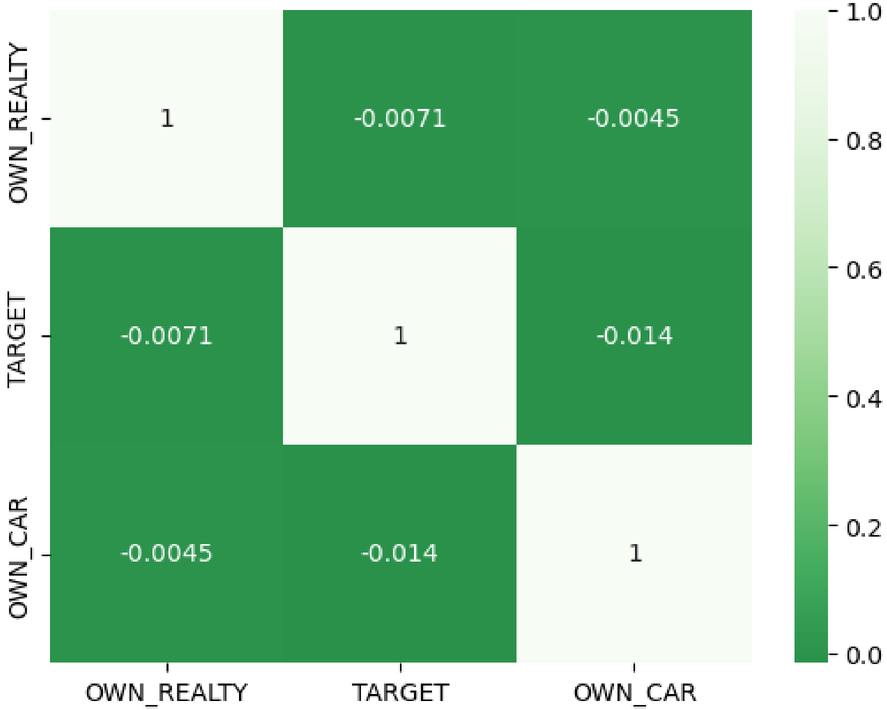
```
final.OWN_CAR.value_counts()
```

```
0    894998
1    455543
Name: OWN_CAR, dtype: int64
```

```
final["OWN_REALTY"]=np.where(final.FLAG_OWN_REALTY=="Y", 1, 0)
final.OWN_REALTY.value_counts()
```

```
1    977370
0    373171
Name: OWN_REALTY, dtype: int64
```

```
sns.heatmap( final[["OWN_REALTY","TARGET", "OWN_CAR"]].corr(), annot= True, cmap= "Greens_r", center=.3)
plt.show()
```

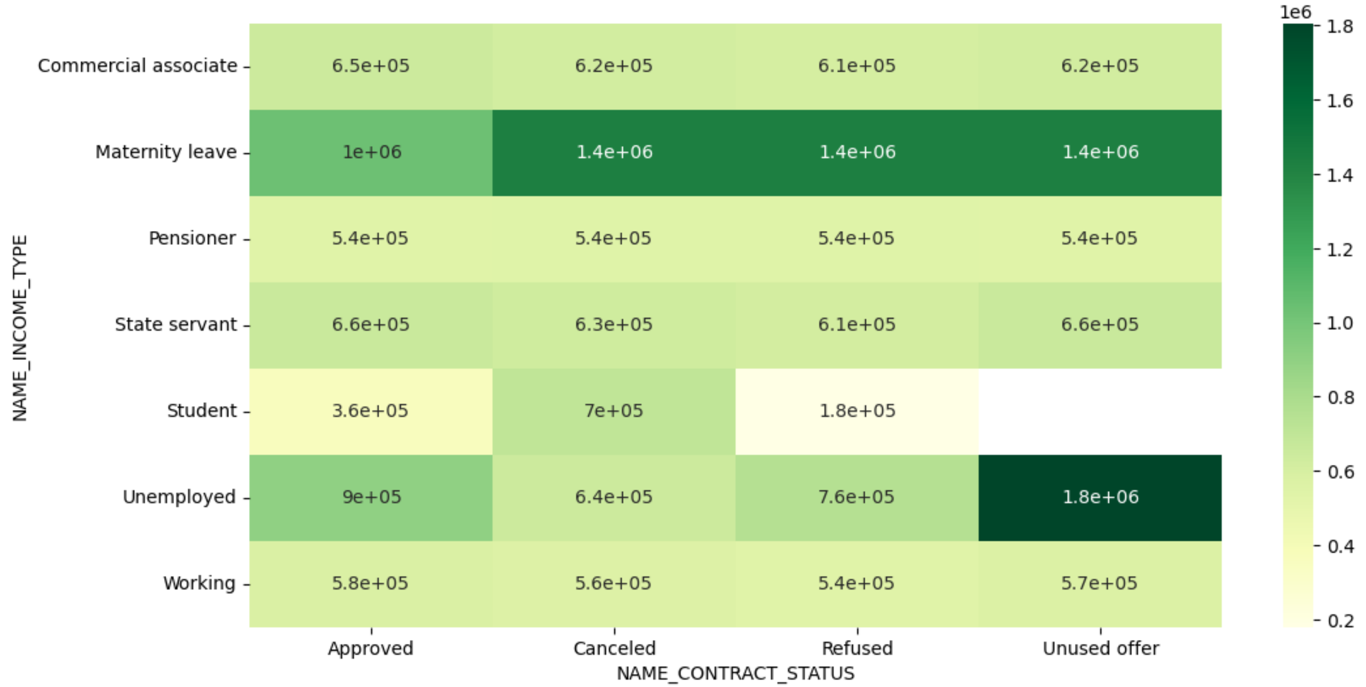


Owning Car and Realty shows financial security.

Owning car and realty has negative relation. those who own car and realty less like to default.

NAME_INCOME_TYPE VS NAME_CONTRACT_STATUS VS AMT_CREDIT

```
pvt_tbl = final.pivot_table(index='NAME_INCOME_TYPE', columns='NAME_CONTRACT_STATUS', values='AMT_CREDIT_x')
plt.figure(figsize=(12,6))
sns.heatmap(pvt_tbl, annot = True, cmap='YlGn')
plt.show()
```



Higher credit given to Unemployed and maternity leave

Working category was refused earlier

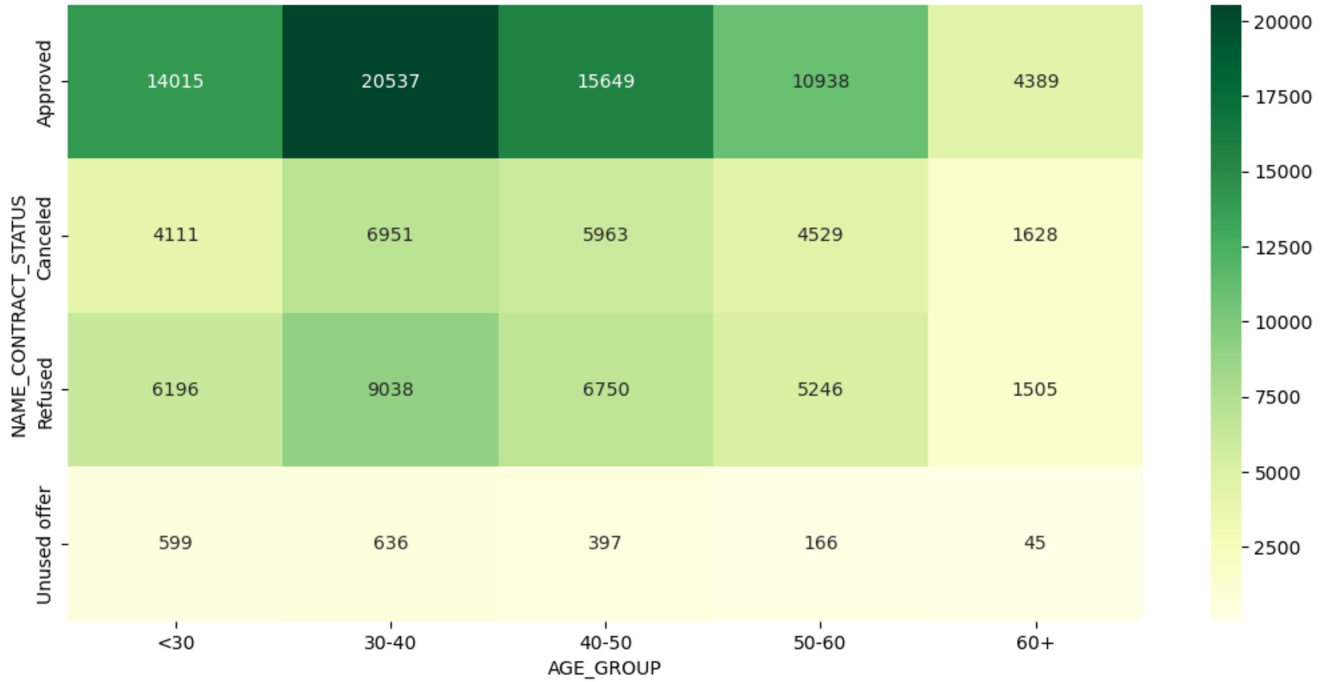
NAME_CONTRACT_STATUS VS NAME_INCOME_TYPE VS TARGET

```python
pvt_tbl=final.pivot_table(index="NAME_CONTRACT_STATUS",columns="NAME_INCOME_TYPE",values='TARGET', aggfunc="sum")
plt.figure(figsize=(14,6))
sns.heatmap(pvt_tbl, annot=True, cmap='YlGn', fmt="g")
plt.show()
```


```
#### most working  and approved defaulted. Working and refused also defaulted.</br>
#### Those data might be because there are many working category
```

NAME_CONTRACT_STATUS VS AGE_GROUP VS TARGET

```
pvt_tbl=final.pivot_table(index="NAME_CONTRACT_STATUS",columns="AGE_GROUP",values='TARGET', aggfunc="sum")
plt.figure(figsize=(13,6))
sns.heatmap(pvt_tbl, annot=True,cmap='YlGn', fmt="g")
plt.show()
```



30-50 group previously approved defaulted most

Refused at age 30-40 group defaulted. There might be reason why middle age group refused at first.

Further analysis on defaulted got approved

```python
app_dft = final[(final['NAME_CONTRACT_STATUS']=="Approved") & (final['TARGET']==1)]
```

```python
columns=['INCOME_GROUP','OWN_CAR','OWN_REALTY', 'AGE_GROUP','YEAR_EMPLOYED_GROUP', 'CODE_GENDER',"ORGANIZATION_TYPE", 'NAME_INCOME_TYPE','OCCUPATION_TYPE' ]
```

```python
for col in columns:
    print(app_dft[col].value_counts(normalize=True))
    print('*'*60)
```

```
A      0.313285
AA     0.230898
L      0.175609
H      0.169983
R      0.110224
Name: INCOME_GROUP, dtype: float64
************************************************************
0    0.684257
1    0.315743
Name: OWN_CAR, dtype: float64
************************************************************
1    0.706339
0    0.293661
Name: OWN_REALTY, dtype: float64
************************************************************
30-40    0.313408
40-50    0.238814
<30      0.213878
50-60    0.166921
```

Summary

- INCOME_GROUP most are average earner
- more defaulter not owning car
- more defaulter not owning realty
- Age group 30-40 most defaulted
- Female defaulted most
- Yeas employed less than 10 defaulted most
- income type working defaulted most
- laborers defaulted most

Female applied mostly for loan, female get special interest rate. Female defaulting can't be taken into account

Final Summary

- more defaulter not owning car
- more defaulter not owning realty
- elder person with more work experience are tends to default less
- higher AMT_CREDIT lower the defaulter
- higher AMT_GOODS_PRICE lower the defaulter.
- higher the income lower the defaulter
- higher the annuity lower the defaulter
- Female defaulted most, but they applied the most.
- Years employed less than 10 defaulted most
- income type working defaulted most.
- laborers defaulted most.
- 30-50 group previously approved defaulted most

Continue...

- Card portfolio has most defaulter
- High and middle yield interest rate group has highest defaulter. That might because it sold more.
- Those who refused earlier has defaulted current time.
- Higher credit given to Unemployed and maternity leave
- Highly populated region must be among with city, where it has less defaulter. Highly populated region might be high earning place
- Refused at age 30-40 group are defaulted. There might be reason why middle age group refused at first.