BFSI CASE STUDY

Submitted by

Chitra D Nair

DS C60 Batch

Stages of the case study

This case study progressed through the following stages to build a machine learning model.

1. Problem Statement
2. Data Understanding- inspection and preparation of data
3. Exploratory Data Analysis
4. Feature Engineering
5. Handling Imbalanced Classes
6. Evaluation Metric
7. Conclusion

Problem Statement

- The primary objective of this study is to build a bank's internal end-to-end customer scoring mechanism for loans.
- The scoring mechanism will help in deciding which loans need to be disbursed and which ones to reject, based on the past behavior of the applicant and application information.

Data Understanding

The data frame for analysis was created by merging the application data and bureau data files.

The data was then prepared for analysis through the following steps.

1. Checking the data types of attributes in the dataset

2.Missing value treatment

3.Dropping redundant columns in the dataset

4.Converting binary variables to 0's and 1's

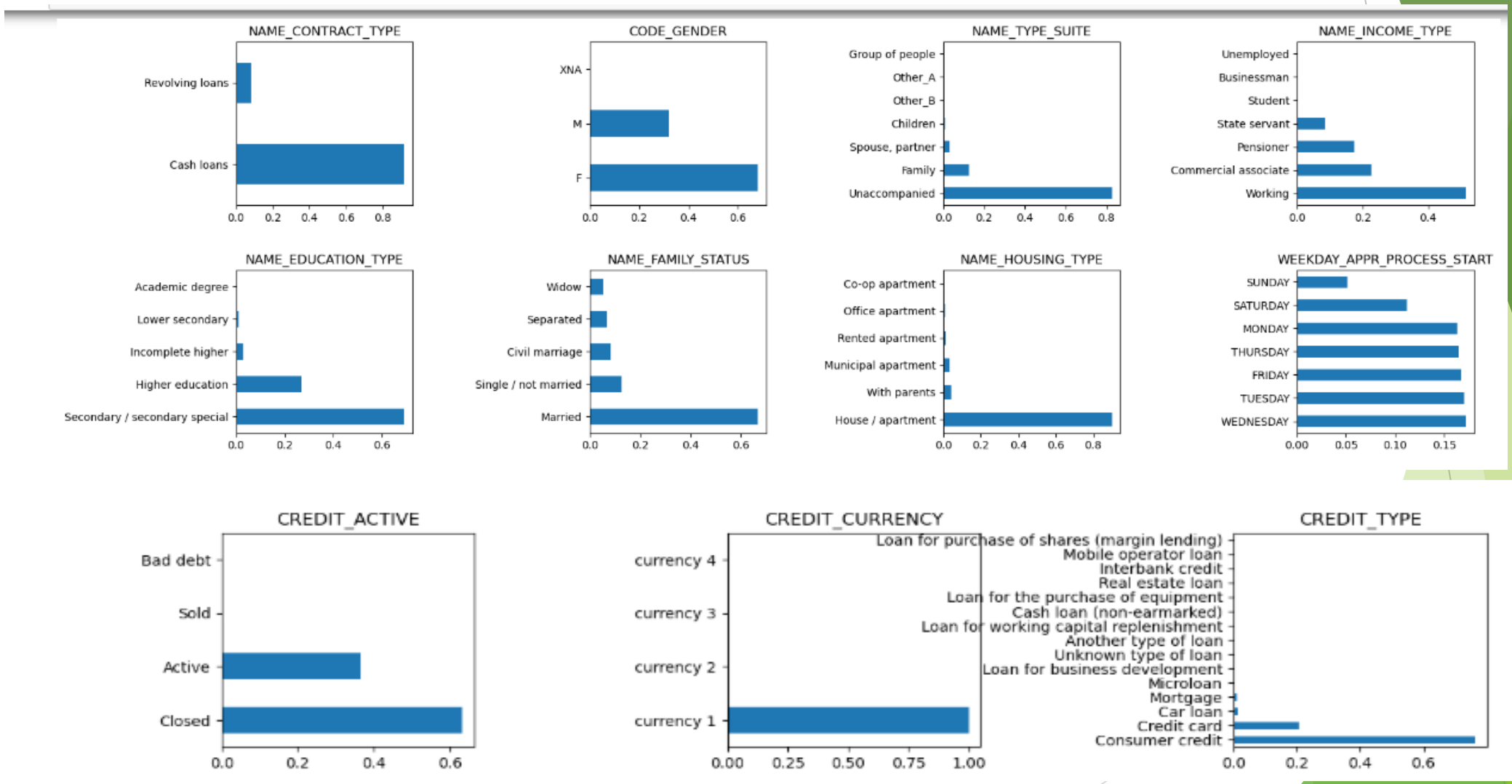
5. Identifying the categorical columns in the dataframe

6. Addressing the negative values in certain columns in the data set

7. Converting the DAYS_BIRTH column to age of applicant in years and changing the data type

Exploratory Data Analysis(EDA)

- Univariate Analysis of categorical variables



Inferences : Univariate Analysis of categorical variables

Majority of the loans are cash loans

Majority of the loan applicants are females

Majority of the applicants belongs to working group

Majority of the applicants have secondary/secondary special education

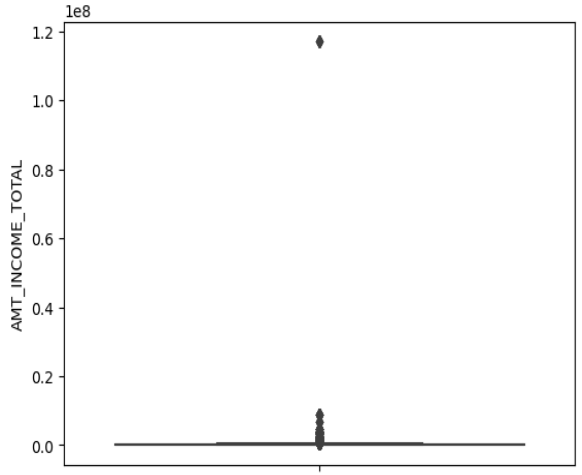
5. Most of the applicants belonged to married category

6. Majority of the credit bureau reported credits have closed status

7. Consumer credit is the majority credit type of credit bureau credit

Univariate Analysis of Numeric variables

1.AMT INCOME TOTAL

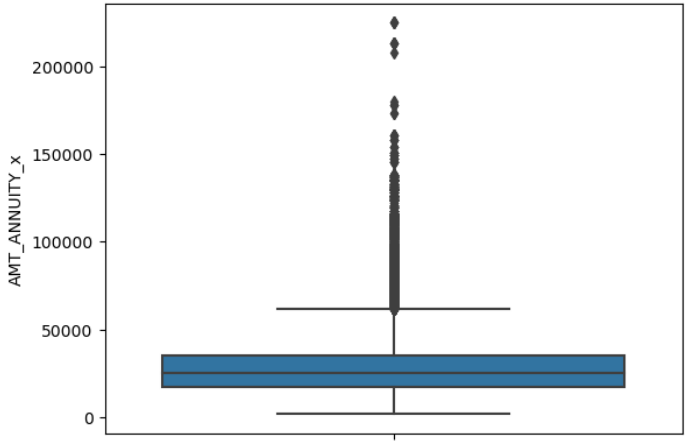


We observe that the 50th percentile or median income amount is 1.575 lakhs and 99th percentile is 5.4 lakhs whereas the max income is 11.7Crore in the defaulter dataset which is the outlier and a point of interest here because why such a high-income applicant becomes a defaulter

The outliers were fixed by caping the values.

2.AMT_CREDIT variable- Credit amount of the current loan---There are no outliers in the column

3.AMT_ANNUITY_x Variable- Annuity payment towards the current loan



Clearly there are outliers in the variable. The outliers were fixed by caping the values.

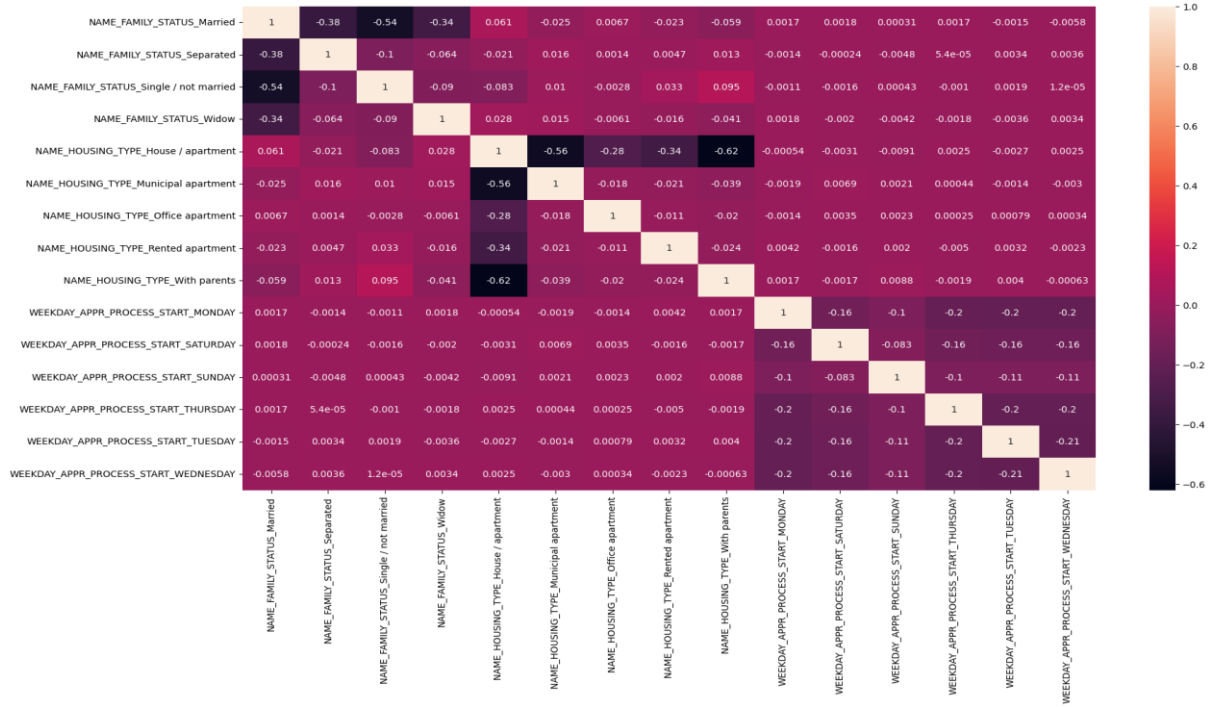
Dummy variables were created for categorical variables with multiple levels and repeated variables were dropped.

Test -Train split was performed on the dataframe--features were assigned to X and response variable ‘ TARGET’ was assigned to y.

Feature Scaling : Feature scaling was performed using MinMaxScaler()

The default rate was evaluated to approx. 8 percentage.

Plotting Correlation matrix of dummy variables: **The highly correlated variables were dropped**



Model Building

- To begin with a Logistic Regression model was built. The accuracy score obtained on train set and test set was 92%.
- Secondly a Random Forest model was built, and the accuracy score obtained was 99.9% on train set and 97.6% on test set.

- Both models were subject to evaluation by cross validation

	precision	recall	f1-score	support
0	0.92	1.00	0.96	717370
1	0.52	0.01	0.02	61204
accuracy			0.92	778574
macro avg	0.72	0.50	0.49	778574
weighted avg	0.89	0.92	0.89	778574

Logistic Regression model classification report

Feature Selection

- Feature selection was done using random forest classifier and top 19 features were selected.

```
for feature in zip(feats, rf.feature_importances_):
    print(feature)
```

```
('FLAG_OWN_CAR', 0.006465236832691794)
('FLAG_OWN_REALTY', 0.007447481881375911)
('CNT_CHILDREN', 0.009904063004414506)
('AMT_INCOME_TOTAL', 0.035825596163943645)
('AMT_CREDIT', 0.03966831266854324)
('AMT_ANNUITY_x', 0.04290948007735493)
('AMT_GOODS_PRICE', 0.03352773976813082)
('REGION_POPULATION_RELATIVE', 0.03495588994195136)
('DAYS_BIRTH', 0.03494954606129697)
('DAYS_EMPLOYED', 0.04181074286816888)
('DAYS_REGISTRATION', 0.046987940889300056)
('DAYS_ID_PUBLISH', 0.04528713486482347)
('FLAG_MOBIL', 0.0)
('FLAG_EMP_PHONE', 0.0010764055215996362)
('FLAG_WORK_PHONE', 0.006137375316203192)
('FLAG_CONT_MOBILE', 0.0002749606923410284)
('FLAG_EMAIL', 0.0036300652001148874)
('CNT_FAM_MEMBERS', 0.013540246309694257)
('REGION_RATING_CLIENT', 0.006016873791021417)
```

Handled imbalanced classes

- We can see that there is class imbalance- not a defaulter is at 92% and defaulter is only 7.8%

```
print(data.TARGET.value_counts())
data.TARGET.value_counts(normalize = True).reset_index()
```

```
0    1024712
1      87537
Name: TARGET, dtype: int64
```

	index	TARGET
0	0	0.921297
1	1	0.078703

- The class imbalance in the data set was handled using random under sampling and random oversampling techniques.

Evaluation metrics

In this problem **recall** was used as the evaluation metric since we attempt to capture the performance where we will be rightly predicting the loan defaulter. Following are the metrics generated.

- A)The logistic regression model- random under sampling has a recall score of 0.68

```
Accuracy:  0.6892005694163482
F1 score:  0.2562679288582903
Recall:  0.6785022595222724
Precision:  0.15796546632834396

 clasification report:
              precision    recall  f1-score   support

           0       0.96      0.69      0.80    307342
           1       0.16      0.68      0.26     26333

    accuracy                           0.69    333675
   macro avg       0.56      0.68      0.53    333675
weighted avg       0.90      0.69      0.76    333675


 confussion matrix:
 [[212102  95240]
 [  8466  17867]]
```

B)Logistic Regression model- Random Oversampling has a recall score of 0.68

```
print ('Recall: ', recall_score(y_test, y_pred_os))
print ('Precision: ', precision_score(y_test, y_pred_os))
print ('\n clasification report:\n', classification_report(y_test,y_pred_os))
print ('\n confussion matrix:\n',confusion_matrix(y_test, y_pred_os))
```

```
Accuracy:  0.6897699857645913
F1 score:  0.2568310718644554
Recall:  0.6792617628071241
Precision:  0.15835229335056702

 clasification report:
              precision    recall  f1-score   support

           0       0.96      0.69      0.80    307342
           1       0.16      0.68      0.26     26333

    accuracy                           0.69    333675
   macro avg       0.56      0.68      0.53    333675
weighted avg       0.90      0.69      0.76    333675


 confussion matrix:
 [[212272  95070]
 [  8446  17887]]
```

C) Random Forest - Random Under Sampling has a recall score of 0.92

```
Accuracy:  0.8910017232336854
F1 score:  0.5722384267971395
Recall:  0.9238218205293738
Precision:  0.4144928523964492

 clasification report:
              precision    recall  f1-score   support

           0       0.99      0.89      0.94    307342
           1       0.41      0.92      0.57     26333

    accuracy                           0.89    333675
   macro avg       0.70      0.91      0.75    333675
weighted avg       0.95      0.89      0.91    333675


 confusion matrix:
 [[272978  34364]
 [  2006  24327]]
```

C) Random Forest - Random Over Sampling has a recall score of 0.81

```
Accuracy:  0.9852041657301266
F1 score:  0.896605164506063
Recall:  0.812896365776782
Precision:  0.9995330593948449

 clasification report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99    307342
           1       1.00      0.81      0.90     26333

    accuracy                           0.99    333675
   macro avg       0.99      0.91      0.94    333675
weighted avg       0.99      0.99      0.98    333675


 confussion matrix:
 [[307332     10]
 [  4927  21406]]
```

Conclusion:

1.The logistic regression model- random under sampling has a recall score of 0.68

2.The Random forest model with random under sampling has a recall score of 0.92

3.The logistic regression model- random oversampling has a recall score of 0.68

The Random Forest model with - Random Over-Sampling has a recall score of 0.81

So, we have recall value highest for Random Forest model with random under sampling. Hence, we can select the Random Forest model with random under sampling as the best performing model .