Case Study

Venkat Garapati

1/12/2020

Table of Contents

Business Objectives and Constraints

Objective:

The purpose of the project is to gauge the effectiveness of Python or R (you choose) and in model development. The model will be evaluated using AUC.

Constraints:

1. Mandatory Constraints.

   The given data is to be analyzed in the period of 48 hours. The data lacks description as the project is based on an individual ability to draw insights and build models.

2. Interpretability.

   This is partially important in our case study. The notion of the case study is that, if the model predicts well on test data, we don’t have to worry about the subsequent consequences of deployment. The data is evaluated on the basis of AUC score.

3. Predicting the probability of the target variable.

   In the case study, the data is imbalanced. So, we can’t use accuracy as an error metric. When data is imbalanced, we can use Log loss, F1-score and AUC. Here, we are sticking with AUC as it used for model evaluation.

4. Cost of Misclassification:

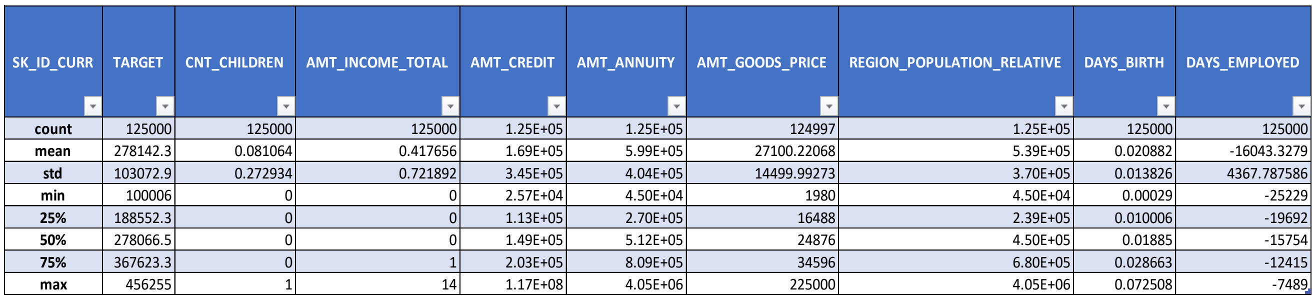
   For the following study, the cost of misclassification can be tolerated. As we do not have the data description, the misclassification is to be reduced as much as possible.

Data Description and Overview

The following Data is provided by Prestige financial team. The purpose of the project is to gauge one's effectiveness in Python or R (you choose) and in model development. The data lacks description. The work is to be approached through the methodology of Data Science. The data is given in structured form with the presence of NULL values. The dependent variable is the TARGET feature of the data.

Training set:

- RangeIndex: 125000 entries, 0 to 124999
- Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
- dtypes: float64(65), int64(41), object(16)
- Number of numerical columns: 106
- number of categorical columns: 16
- Basic glimpse/description of the data



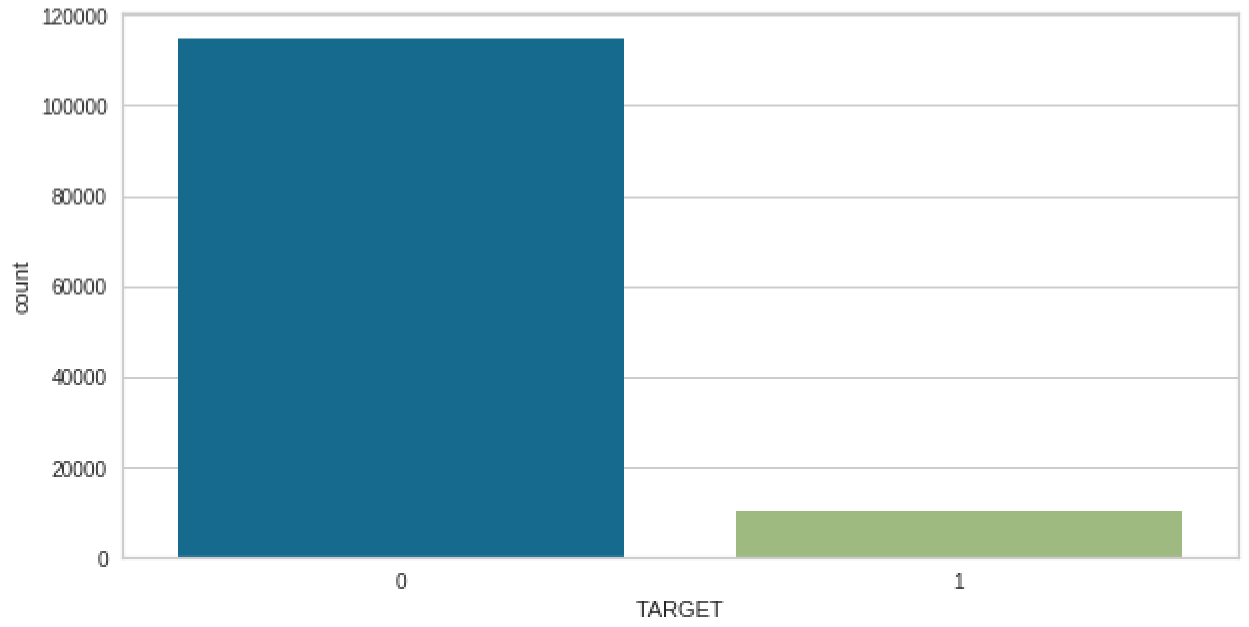
SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_POPULATION_RELATIVE	DAYS_BIRTH	DAYS_EMPLOYED
count	125000	125000	125000	1.25E+05	1.25E+05	124997	1.25E+05	125000	125000
mean	278142.3	0.081064	0.417656	1.69E+05	5.99E+05	27100.22068	5.39E+05	0.020882	-16043.3279
std	103072.9	0.272934	0.721892	3.45E+05	4.04E+05	14499.99273	3.70E+05	0.013826	4367.787586
min	100006	0	0	2.57E+04	4.50E+04	1980	4.50E+04	0.00029	-25229
25%	188552.3	0	0	1.13E+05	2.70E+05	16488	2.39E+05	0.010006	-19692
50%	278066.5	0	0	1.49E+05	5.12E+05	24876	4.50E+05	0.01885	-15754
75%	367623.3	0	1	2.03E+05	8.09E+05	34596	6.80E+05	0.028663	-12415
max	456255	1	14	1.17E+08	4.05E+06	225000	4.05E+06	0.072508	-7489

Testing Set:

- RangeIndex: 48744 entries, 0 to 48743
- Columns: 121 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
- dtypes: float64(65), int64(40), object(16)
- Number of numerical columns: 105
- number of categorical columns: 16

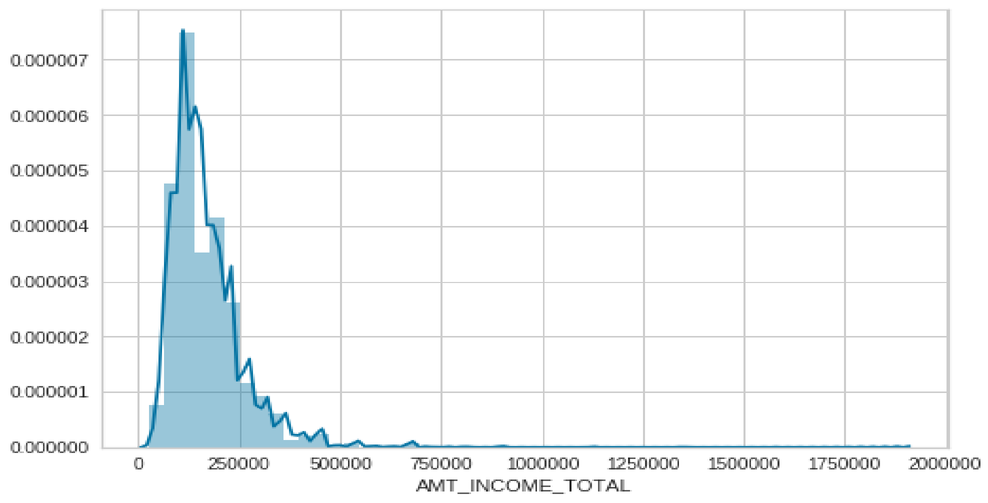
Exploratory Data Analysis

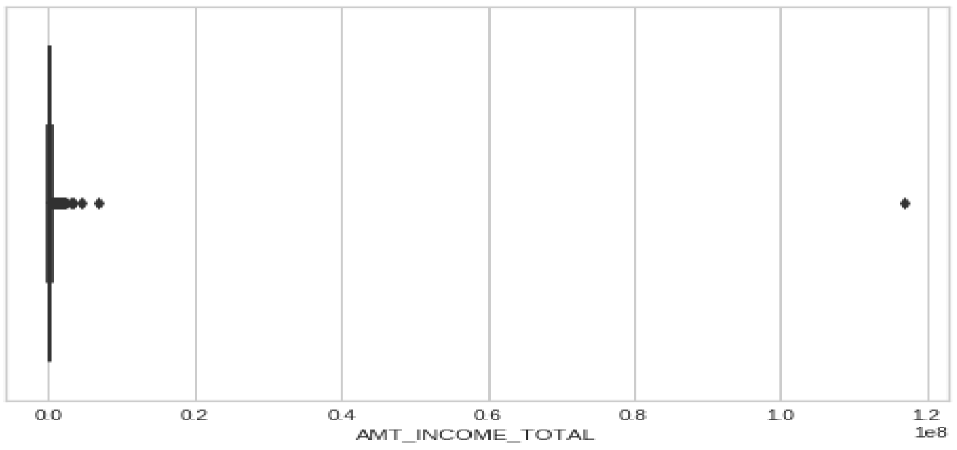
Let's check the distribution of data points among output class.



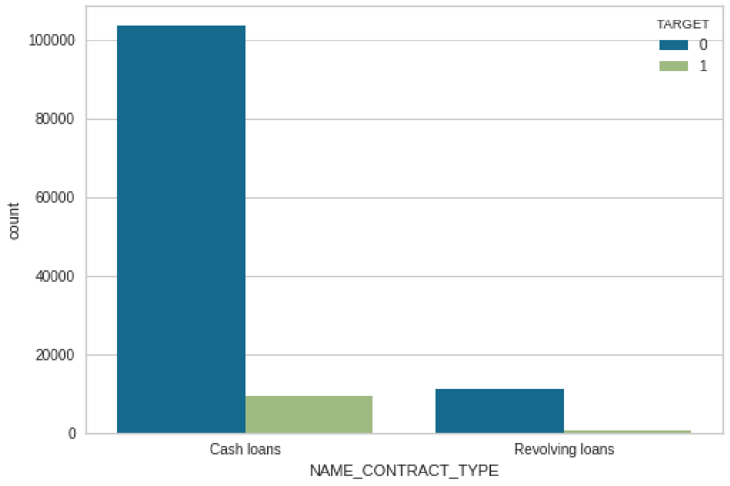
The data is imbalanced (91.9% (114867- 0) and 8.07%(10133-1)) and we need to handle this problem.

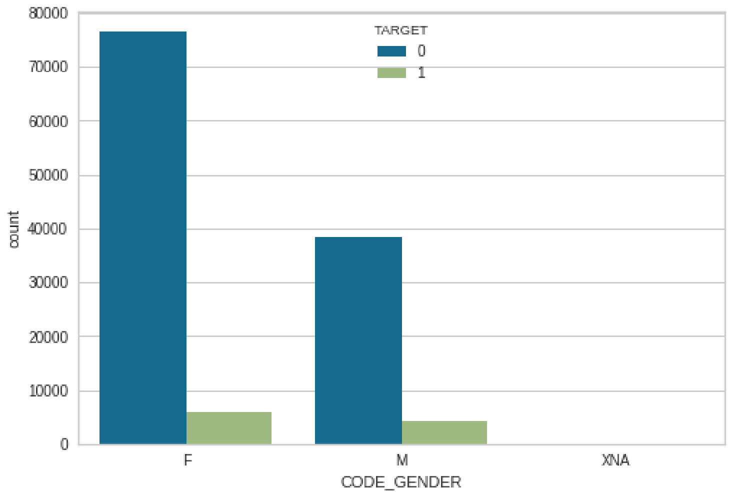
The following is the distribution and boxplot of the amt_income_total. We can observe that the most points lie below 2000000. The distribution is right skewed and there are extreme values, we can apply log distribution. People with high income tend to go for 1. (0 - 94.680851, 1 - 5.319149). Also it can be observed from the boxplot that there are few outliers which can be removed.



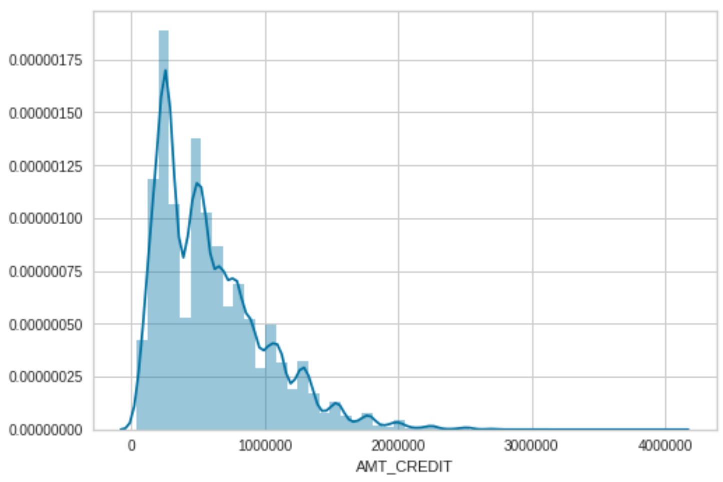


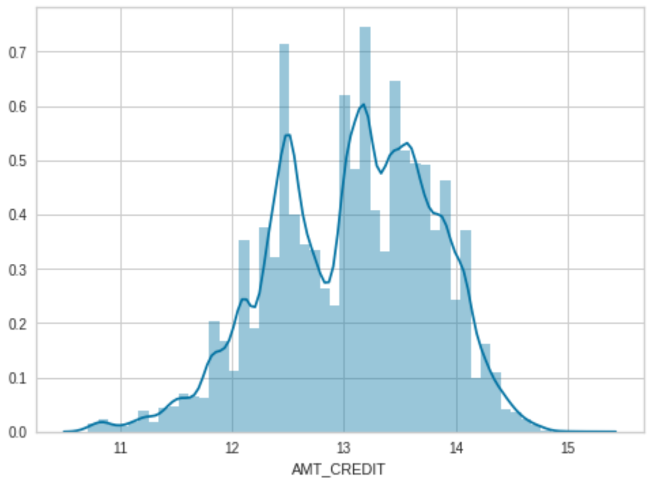
Many people are willing to take cash loan than revolving loan. There are more female subscribers.



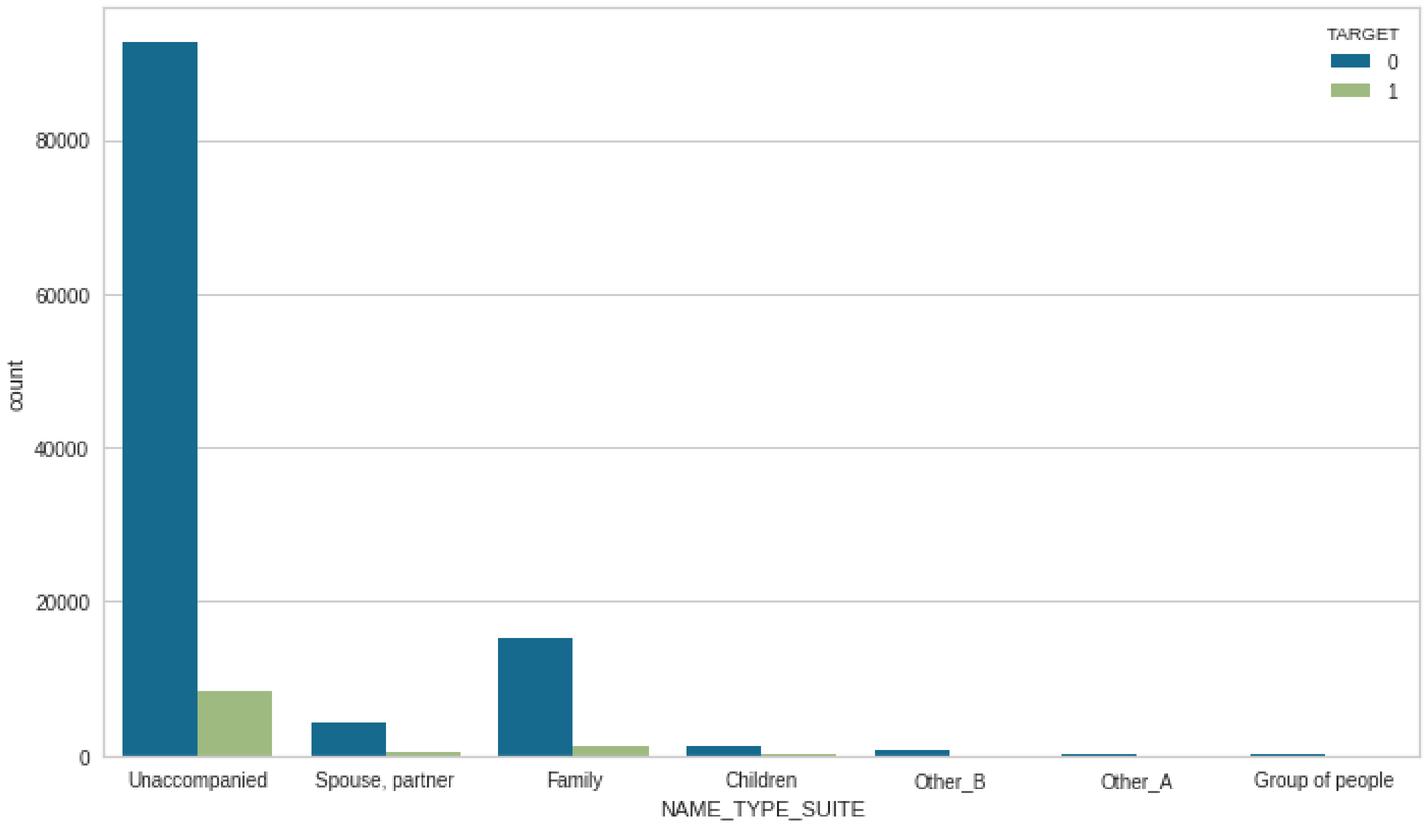


Distribution of amount of credit. Originally the distribution is right skewed, we used log transformation to make it normal distributed.

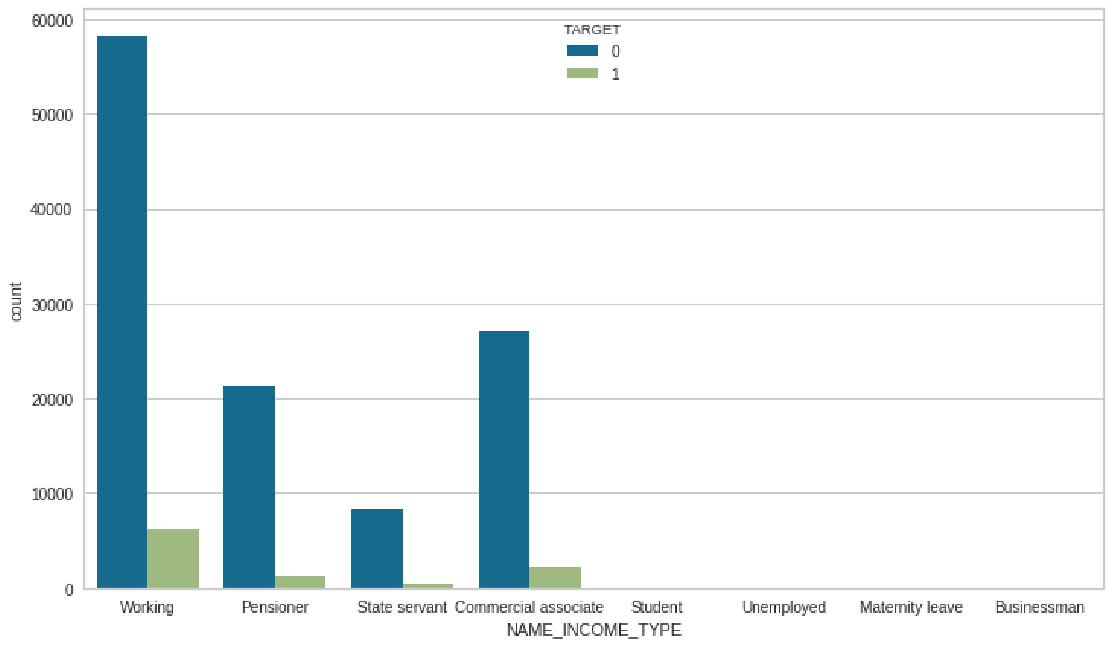


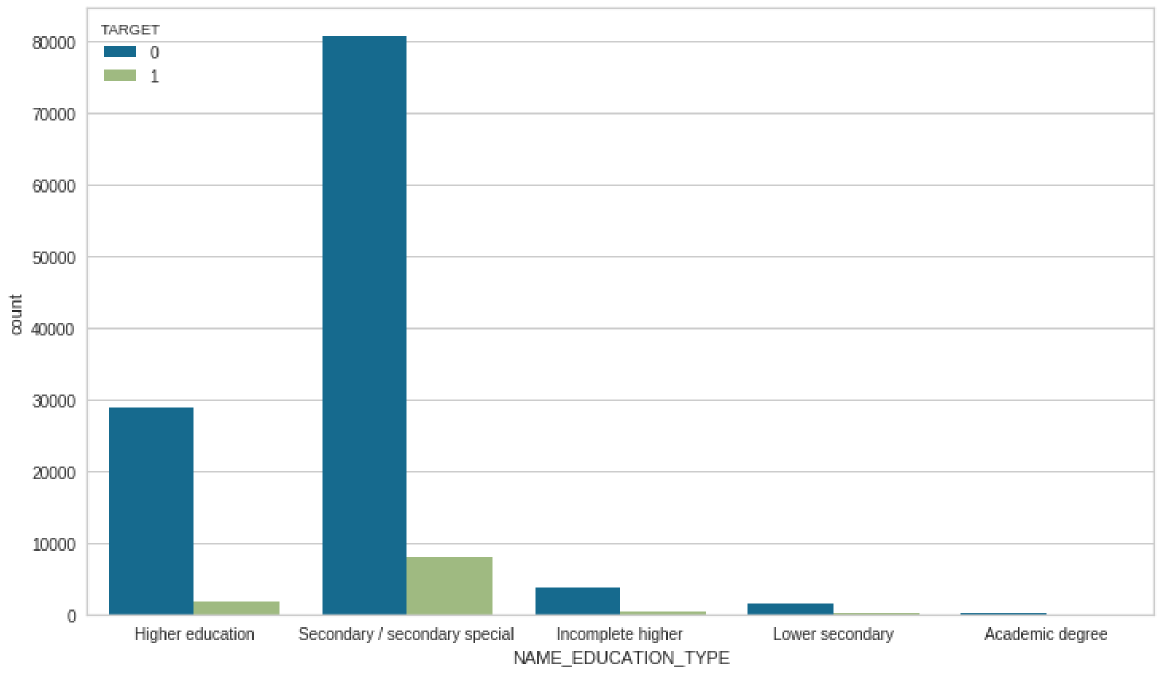


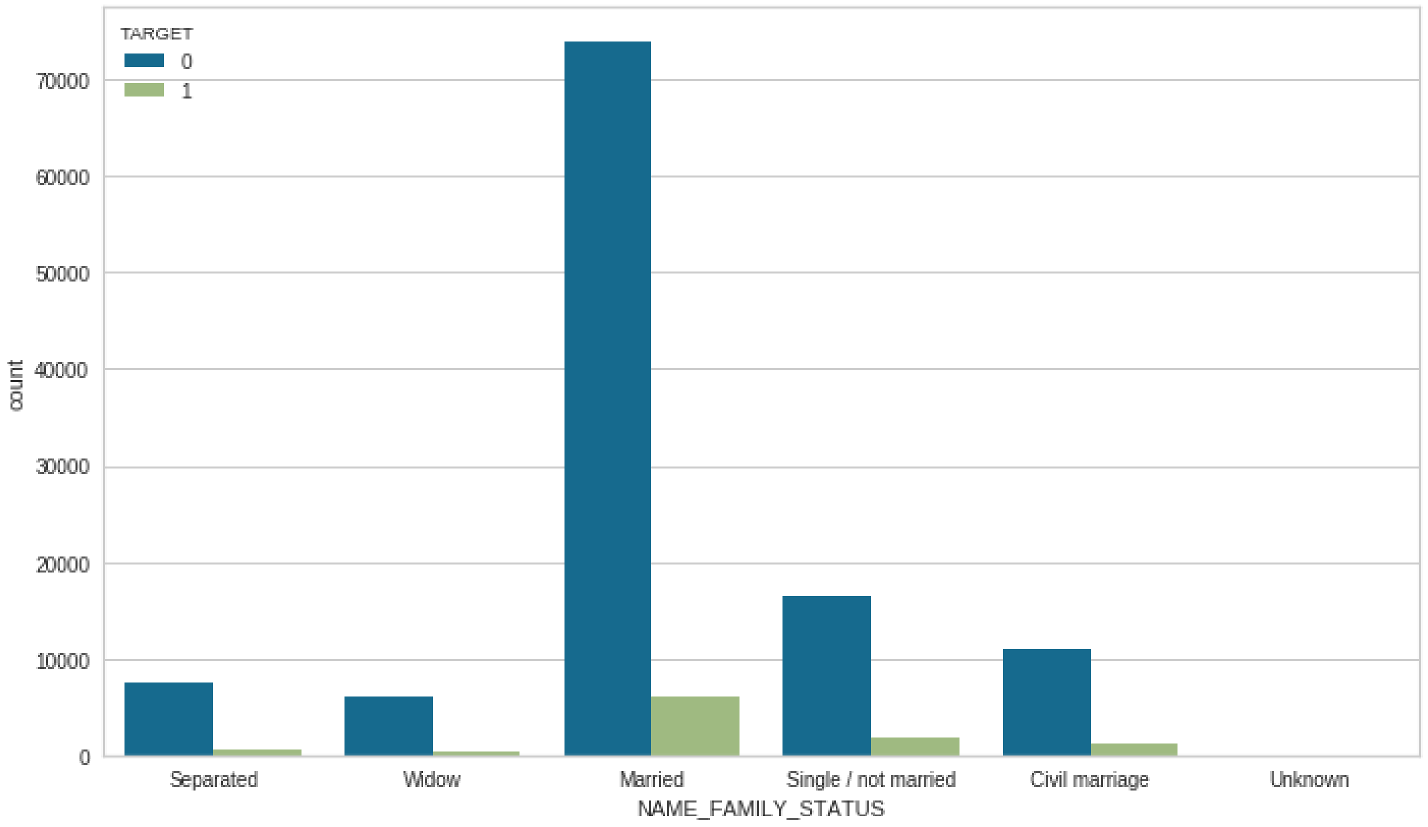
Distribution of Name of type of the Suite in terms of target variable.

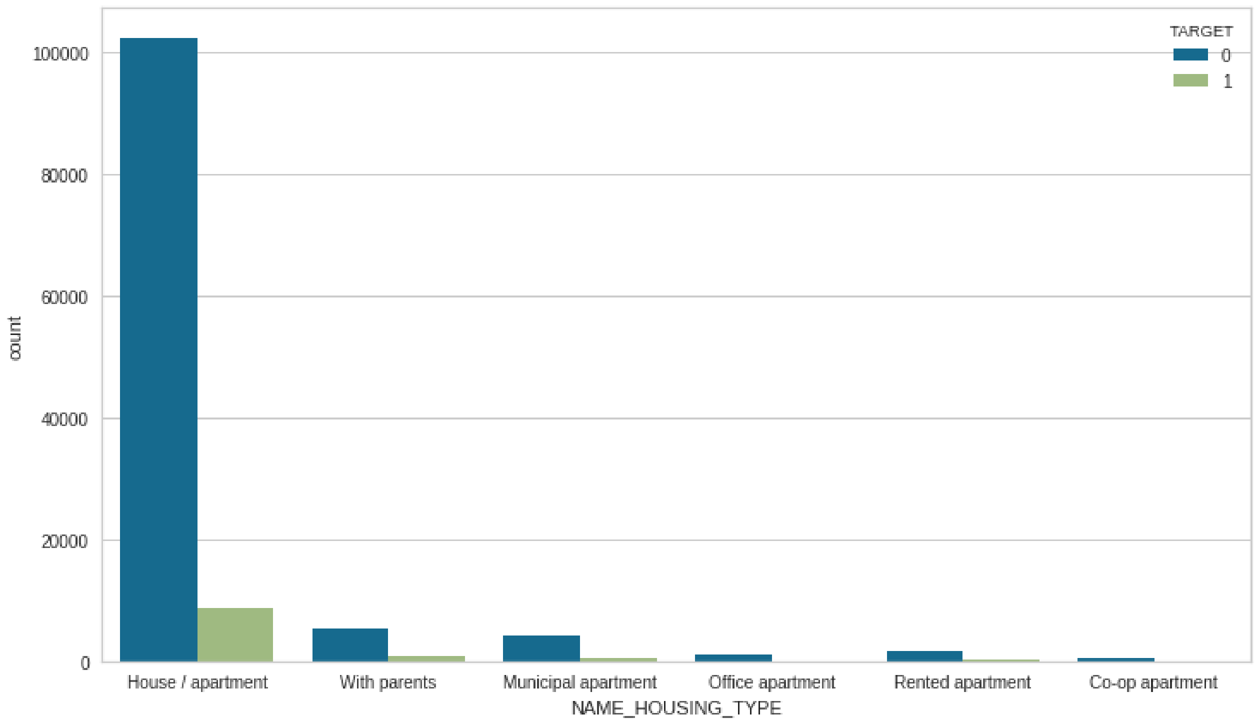


Distribution of name of income, education, family status and income type based on the target variable. It can be observed that someone belonging to working, some sort of education, married and apartment would be subscribed to the target variable 1.

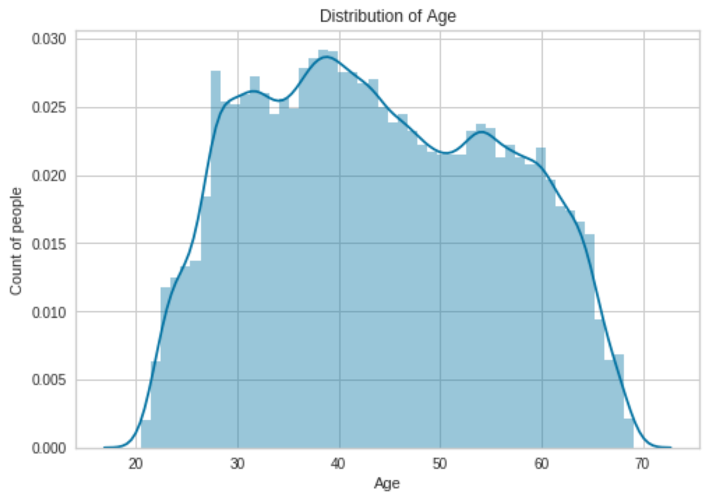


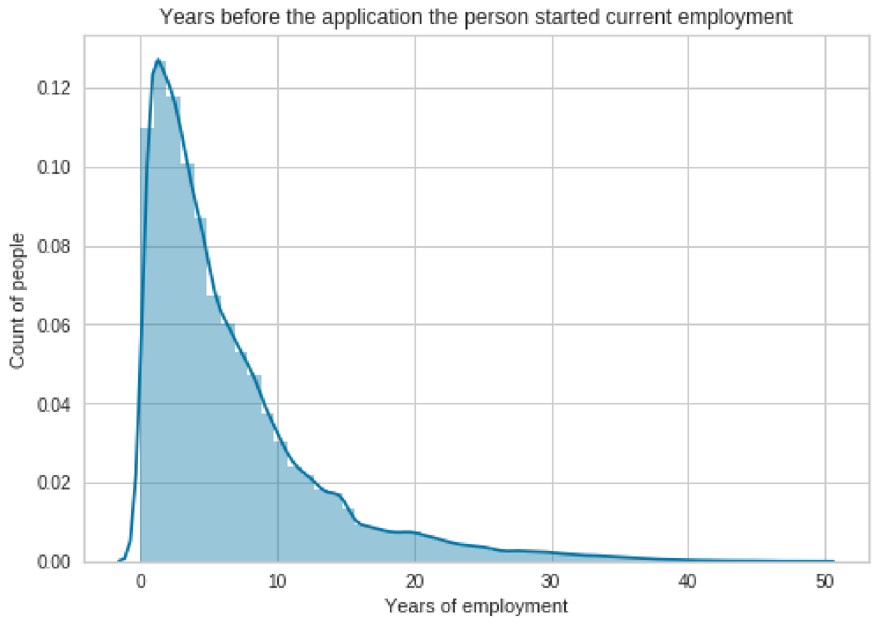



TARGET
0
1
count
70000
60000
50000
40000
30000
20000
10000
0
Separated
Widow
Married
Single / not married
Civil marriage
Unknown
NAME_FAMILY_STATUS


TARGET
0
1
count
100000
80000
60000
40000
20000
0
House / apartment
With parents
Municipal apartment
Office apartment
Rented apartment
Co-op apartment
NAME_HOUSING_TYPE

Distribution of the population age and years of employment. People with less than 2 years of employment are less likely to subscribe for the target variable-1.





Data Preparation

Imputing NULL values:

The training and testing data consist NULL values for significant number of features. The NULLS have been imputed through mean, median and mode. Further, techniques like kNN and MICE imputation were carried out utilizing the imblearn and fancyimpute python packages. After, careful evaluation of the final AUC scores for various imputation. Median was preferred as it had slightly a better AUC score for the model than the other imputation techniques. KNN and MICE were not used due to time consumption and computer limitations. Additionally, the NULL values in the category columns were labelled unknown as it was harder to classify due to lack of data description.

Feature Engineering:

Initially, the categorical data points were dummy coded by dropping the first rows. New features were added to the training and test data based on the understanding of the data. The following added features are:

First, the approach was to understand some common features such as annuity, credit, age, days of employment, revenue earned, number of adults. After extensive approach from the boxplots and bar plots. The following features were added.

List of new features based on the insights of the data
df['AMT_LEFT_EXP'] = df['AMT_INCOME_TOTAL'] - df['AMT_GOODS_PRICE'] - df['AMT_ANNUITY']
df['AMT_INCOME_TOTAL - AMT_GOODS_PRICE'] = df['AMT_INCOME_TOTAL'] - df['AMT_GOODS_PRICE']
df['CNT_adults'] = df['CNT_FAM_MEMBERS'] - df['CNT_CHILDREN']
df['income_contrib'] = df['AMT_INCOME_TOTAL'] / df['CNT_adults']
df['days_employed_perc'] = df['DAYS_EMPLOYED'] / df['DAYS_BIRTH']
df['days_education'] = df['DAYS_EMPLOYED'] - df['DAYS_BIRTH']
df['credit_minus_goods'] = df['AMT_CREDIT'] - df['AMT_GOODS_PRICE']
df['credit_div_goods'] = df['AMT_CREDIT'] / df['AMT_GOODS_PRICE']
df['annuity_length'] = df['AMT_CREDIT'] / df['AMT_ANNUITY']
df['AMT_CREDIT / AMT_INCOME_TOTAL'] = df['AMT_CREDIT'] / df['AMT_INCOME_TOTAL']
df['income_credit_perc'] = df['AMT_INCOME_TOTAL'] / df['AMT_CREDIT']
df['income_per_person'] = df['AMT_INCOME_TOTAL'] / df['CNT_FAM_MEMBERS']
df['AMT_INCOME_TOTAL / 12 - AMT_ANNUITY'] = df['AMT_INCOME_TOTAL'] / 12. - df['AMT_ANNUITY']
df['AMT_INCOME_TOTAL / AMT_ANNUITY'] = df['AMT_INCOME_TOTAL'] / df['AMT_ANNUITY']

df['annuity_income_perc'] = df['AMT_ANNUITY'] / (1 + df['AMT_INCOME_TOTAL'])
df['DAYS_REGISTRATION / DAYS_ID_PUBLISH'] = df['DAYS_REGISTRATION'] / df['DAYS_ID_PUBLISH']
df['ann_len_emply_ratio'] = df['annuity_length'] / df['DAYS_EMPLOYED']
df['children_ratio'] = df['CNT_CHILDREN'] / df['CNT_FAM_MEMBERS']

Secondly, another set of features were added in order to improve the AUC score after initial runs. These features were mostly based on the days of registration, flagged documents, own car and telecommunication characteristics.

df['reg_div_publish'] = df['DAYS_REGISTRATION'] / df['DAYS_ID_PUBLISH']
df['birth_div_reg'] = df['DAYS_BIRTH'] / df['DAYS_REGISTRATION']
df['flag_document_sum'] = df['FLAG_DOCUMENT_2'] + df['FLAG_DOCUMENT_3'] + df['FLAG_DOCUMENT_4'] + df['FLAG_DOCUMENT_5'] + df['FLAG_DOCUMENT_6'] + df['FLAG_DOCUMENT_7'] + df['FLAG_DOCUMENT_8'] + df['FLAG_DOCUMENT_9'] + df['FLAG_DOCUMENT_10'] + df['FLAG_DOCUMENT_11'] + df['FLAG_DOCUMENT_12'] + df['FLAG_DOCUMENT_13'] + df['FLAG_DOCUMENT_14'] + df['FLAG_DOCUMENT_15'] + df['FLAG_DOCUMENT_16'] + df['FLAG_DOCUMENT_17'] + df['FLAG_DOCUMENT_18'] + df['FLAG_DOCUMENT_19'] + df['FLAG_DOCUMENT_20'] + df['FLAG_DOCUMENT_21']
df['is_na_amt_annuity'] = 1.0*np.isnan(df['AMT_ANNUITY'])
df['age_finish'] = df['DAYS_BIRTH']*(-1.0/365) + (df['AMT_CREDIT']/df['AMT_ANNUITY']) *(1.0/12)
df['new_inc_per_chld'] = df['AMT_INCOME_TOTAL'] / (1 + df['CNT_CHILDREN'])
df['new_car_to_birth_ratio'] = df['OWN_CAR_AGE'] / df['DAYS_BIRTH']
df['new_car_to_employ_ratio'] = df['OWN_CAR_AGE'] / df['DAYS_EMPLOYED']
df['new_phone_to_birth_ratio'] = df['DAYS_LAST_PHONE_CHANGE'] / df['DAYS_BIRTH']
df['new_phone_to_birth_ratio_employer'] = df['DAYS_LAST_PHONE_CHANGE'] / df['DAYS_EMPLOYED']

Due to lack of description most of these features were added intuition and heuristic approach. For example, we do not know what do flagged documents and what documents it is referring to. Its harder to interpret. Therefore, we tried to derive some sort of information from lack of data description. New features were added based on future instances such new child in the household, new car, new phone, how old when credit and annuity finish.

Thirdly, some flagged features were dropped due to presence of empty values. Additionally, new features were added after closer examination of the outlier plots. This was an extensive approach and only few variables were added to the datasets due to time constraints. The following features are:

Features based on the outliers and boxplots
df['DAYS_EMPLOYED'].replace(365243, np.nan, inplace = True)
df.loc[df['OWN_CAR_AGE'] > 80, 'OWN_CAR_AGE'] = np.nan

```
df.loc[df['REGION_RATING_CLIENT_W_CITY'] < 0, 'REGION_RATING_CLIENT_W_CITY'] = np.nan
df.loc[df['AMT_INCOME_TOTAL'] > 1e8, 'AMT_INCOME_TOTAL'] = np.nan
```

Lastly, additional features were added mostly based on the ext source. Given the lack of data description. It was harder to interpret what the variables described. Time was limited to perform data aggregation for other set of features. These additional features have contributed to an improvement for the model AUC score.

Additional features by data aggregation
`df['EXT_SOURCE_mean'] = df[['EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3']].mean(axis = 1)`
`df['EXT_SOURCE_std'] = df[['EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3']].std(axis = 1)`
`df['EXT_SOURCE_prod'] = df['EXT_SOURCE_1'] * df['EXT_SOURCE_2'] * df['EXT_SOURCE_3']`
`df['EXT_SOURCE_1 * EXT_SOURCE_2'] = df['EXT_SOURCE_1'] * df['EXT_SOURCE_2']`
`df['EXT_SOURCE_1 * EXT_SOURCE_3'] = df['EXT_SOURCE_1'] * df['EXT_SOURCE_3']`
`df['EXT_SOURCE_2 * EXT_SOURCE_3'] = df['EXT_SOURCE_2'] * df['EXT_SOURCE_3']`
`df['EXT_SOURCE_1 * DAYS_EMPLOYED'] = df['EXT_SOURCE_1'] * df['DAYS_EMPLOYED']`
`df['EXT_SOURCE_2 * DAYS_EMPLOYED'] = df['EXT_SOURCE_2'] * df['DAYS_EMPLOYED']`
`df['EXT_SOURCE_3 * DAYS_EMPLOYED'] = df['EXT_SOURCE_3'] * df['DAYS_EMPLOYED']`
`df['EXT_SOURCE_1 / DAYS_BIRTH'] = df['EXT_SOURCE_1'] / df['DAYS_BIRTH']`
`df['EXT_SOURCE_2 / DAYS_BIRTH'] = df['EXT_SOURCE_2'] / df['DAYS_BIRTH']`
`df['EXT_SOURCE_3 / DAYS_BIRTH'] = df['EXT_SOURCE_3'] / df['DAYS_BIRTH']`

Data Cleaning:

After addition of the new features. We have to understand the impact of the features. In the sense that, we have an imbalanced training set and do not want our predictions to be biased more towards 0's than 1's. Initially, the data is rid of empty features that do not hold any information value like SK_ID_CURR. Then features comprising of similar distribution to the target variable binary outcomes are removed as they do not hold enough information for the distribution of the classes. The following process is carried out based on statistical theory such as correlations and p-values. All the insignificant p-values features are dropped due to statistical insignificance (p-value > 0.05). further, features consisting of non-similar distributions both on the training and testing data sets were dropped. Having similar feature's distributions on both the data sets will be help improve the model AUC score. Lastly, lightgbm classifier was used to remove the unwanted features for the lightgbm classifier. This process can also be carried out using random forest or XG-boosting but decided to use lightgbm to keep track of time consumption.

Machine Learning Models

Algorithms:

Logistic regression, random forests, light-GBM and XG-boosting were used to train the models.

Approach:

Initially logistic regression was used to train the models but were later discarded as they yielded lower AUC scores than the boosting models. Stratified k-fold was used for the cross-validation purposes. Initially, the hyperparameters were tuned by using Bayesian optimization approach utilizing the bayes opt python packages. The data, objective function and initializer was wrapped inside a custom-made function to yield the best parameters. Using the obtained parameters, models were trained using the boosting algorithms. Further, manual hypertunning was carried out for lightgbm and stacking approach was also experimented to check if there was an improvement of the AUC score.

Parameters used for lightGBM hyper tuning: • 'num_leaves': (30, 45), • 'colsample_bytree': (0.1, 1) • 'subsample': (0.1, 1) • 'max_depth': (5, 15) • 'reg_alpha': (0, 10) • 'reg_lambda': (0, 10) • 'min_split_gain': (0, 1) • 'min_child_weight': (30, 45)	Parameters used for XGboosting hyper tuning: • 'max_depth': (2, 40), • 'gamma': (0.001, 10.0) • 'min_child_weight': (0, 20) • 'max_delta_step': (0, 10) • 'subsample': (0.4, 1.0) • 'colsample_bytree' :(0.4, 1.0)
Parameters obtained after hyper tuning: • 'num_leaves': 45 • 'colsample_bytree': 0.9497036 • 'subsample': 0.5 • 'max_depth': 6 • 'reg_alpha': 0.09 • 'reg_lambda': 0.34 • 'min_split_gain': 0.56 • 'min_child_weight': 30	Parameters obtained after hyper tuning: • 'colsample_bytree': 0.7 • 'silent': 1 • ‘gamma’: 0.235 • 'subsample': 0.7 • 'colsample_bytree': 0.956 • 'learning_rate': 0.075 • 'objective': 'binary:logistic' • 'max_depth': 4 • 'num_parallel_tree': 1 • 'min_child_weight': 1 • 'nrounds': 200

Results

The following were the AUC scores of the trained models

Model	AUC score (values rounded to 3 decimal points)
Light GBM	0.756, 0.779(tried manual hypertunning)
XG boost	0.754 (time consuming and therefore discarded)
Stacking	0.757

Discussion

The time constraint had a major impact on the project. Great deal of time was spent on understanding the behavior of the features. Plots were extensively plotted to understand the behavior of the data. Lack of data description has limited interpretability of the features. Python was used as the programming tool to complete the case study. The feature engineering is the tricky part as it hard to interpret what the variables described. Intuitive approach was carried out to understand and creation of new features. Missing data imputation was mostly based on mean or median or mode. In our case median was used. KNN and MICE were computationally slow and therefore were discarded. Categorical imputation was carried out imputing the unknown label. The categorical variables were transformed using the pandas dummy function by dropping first rows to avoid multicollinearity Data aggregation of was carried on few variables like EXT source. The Data sets were cleaned using the outlier approach by removing outliers and treating few outliers as NaN values, considering similar distributions of the features on both data sets, removal of features having same distribution of classes and LGBM classifier for important features . Several custom-made functions were used to train the model, obtain scores, learning curves and ROC curves. Stacking was also experimented but due to lack of knowledge and experience could not effectively build a stacking model. The biggest hurdle to the case was the time factor. The project is to be completed within a time frame of 48 hrs. Generally, for a data science project more time is consumed on the feature engineering part. For the following the case study an excess of 20 hours was consumed for solving the puzzle of feature engineering. Further, stacking was built to improve the model AUC.

References

1. Course notes of DR. Eakin (inferential statistics), DR. Nerur (Data Science) and DR. Vagefi (Data Science) from University of Texas at Arlington.
2. https://stats.stackexchange.com/questions/243207/what-is-the-proper-usage-of-scale-pos-weight-in-xgboost-for-imbalanced-datasets
3. https://datascience.stackexchange.com/questions/12984/list-of-feature-engineering-techniques
4. https://medium.com/@abbdar/first-steps-in-machine-learning-predicting-subscription-for-bank-deposits-866516b90e4
5. https://dnc1994.com/2016/05/rank-10-percent-in-first-kaggle-competition-en/

Acknowledgment

I would like to thank the Steve Warnick (Chief Credit & Analytics Officer) and Prestige financial for the opportunity.

Work Matrix

Total time for Case Study	48 hrs
Data Understanding	4 hrs
Feature Engineering	20 hrs
Model building	10 hrs
Total hours worked to solve the case study	34 hrs