Project Overview

Home Credit Group

Home Credit B.V. is an international non-bank financial institution founded in 1997 in the Czech Republic.The company operates in 14 countries and focuses on lending primarily to people with little or no credit history. As of 2016 the company has over 15 million active customers, with two-thirds of them in Asia and 7.3 million in China. Major shareholder of company is PPF, a privately held international financial and investment group, which controls an 88.62% stake.

In 1999, Home Credit a.s. was founded in the Czech Republic and in 1999 company expanded to Slovakia. In 2000s company started to expand to Commonwealth of Independent States countries - Russia, Kazakhstan, Ukraine and Belarus.As of 2007 the company was the second largest consumer lender in Russia.In 2010s company expanded to Asia - China, India, Indonesia, Philippines and Vietnam.In 2010 the company was first foreign company to set up as a consumer finance lender in China. In 2015 company launched its operations in the United States of America through a partnership with Sprint Corporation.

Home Credit Group Loans:

Many people struggle to get loans due to insufficient or non-existent credit histories. And, unfortunately, this population is often taken advantage of by untrustworthy lenders.

Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience, Home Credit makes use of a variety of alternative data--including telco and transactional information--to predict their clients' repayment abilities.

While Home Credit is currently using various statistical and machine learning methods to make these predictions, they're challenging Kagglers to help them unlock the full potential of their data. Doing so will ensure that clients capable of repayment are not rejected and that loans are given with a principal, maturity, and repayment calendar that will empower their clients to be successful.

Home Credit Group

Project inspiration

This project was inspired by that fact that many people who deserves loan do not get it and ends up in the hands of untrustworthy lenders. This project is a competition from Kaggle. Below is the link: Kaggle | Home Credit Default Risk Competition.

Many people struggle to get loans due to insufficient or non-existent credit histories. And, unfortunately, this population is often taken advantage of by untrustworthy lenders.

Source : Kaggle

Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience, Home Credit makes use of variety of alternative data--including telco and transactional information--to predict their clients' repayment abilities.

While Home Credit is currently using various statistical and machine learning methods to make these predictions, they're challenging Kagglers to help them unlock the full potential of their data. Doing so will ensure that clients capable of repayment are not rejected and that loans are given with a principal, maturity, and repayment calendar that will empower their clients to be successful

Still on the project overview, here is the datasets and inputs.

Datasets and Inputs.

The datasets and inputs was provided by Kaggle. Data description is below : There are 7 different sources of data:

- **application_train/application_test**: the main training and testing data with information about each loan application at Home Credit. Every loan has its own row and is identified by the feature SK_ID_CURR. The training application data comes with the TARGET indicating **0** if the loan was repaid **Repayers** and **1** for default **Defaulters**

- **bureau**: data concerning client's previous credits from other financial institutions. Each previous credit has its own row in bureau, but one loan in the application data can have multiple previous credits.

- **bureau_balance**: monthly data about the previous credits in bureau. Each row is one month of a previous credit, and a single previous credit can have multiple rows, one for each month of the credit length.

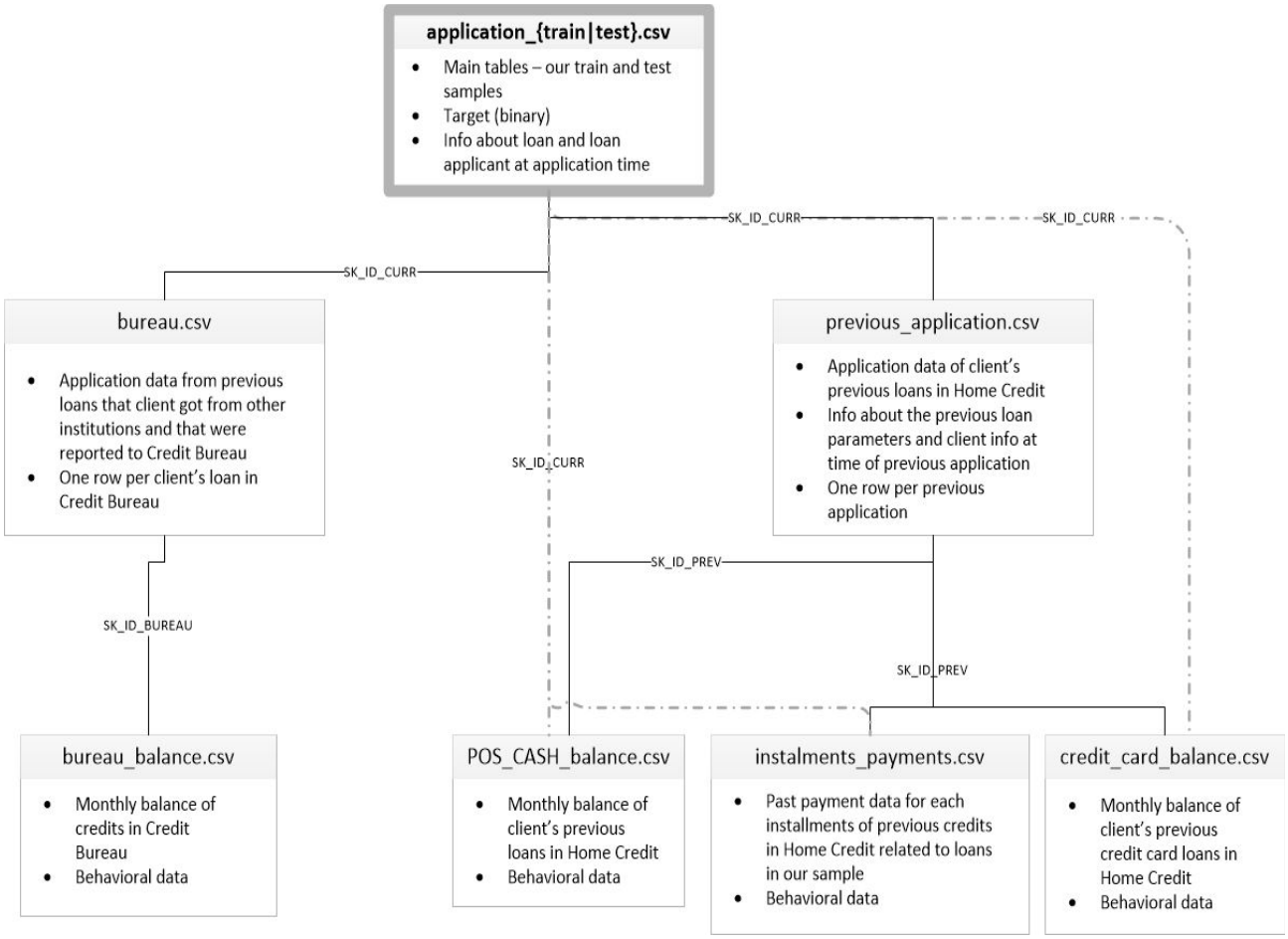
- **previous_application**: previous applications for loans at Home Credit of clients who have loans in the application data. Each current loan in the application data can have multiple previous loans. Each previous application has one row and is identified by the feature SK_ID_PREV.

- **POS_CASH_BALANCE:** monthly data about previous point of sale or cash loans clients have had with Home Credit. Each row is one month of a previous point of sale or cash loan, and a single previous loan can have many rows.

- **credit_card_balance**: monthly data about previous credit cards clients have had with Home Credit. Each row is one month of a credit card balance, and a single credit card can have many rows.

- **installments_payment**: payment history for previous loans at Home Credit. There is one row for every made payment and one row for every missed payment. For more information on what each data represents, please read the PROPOSAL, or Kaggle

- Below is a diagram of how the data are connected.



Problem Statement

Based on the purpose the dataset was given in the first place which is to predict the probability that an applicant is capable of repaying a loan. My problem statement revolves around this.

1. how many people pay loans exploratorily
2. What income class and family type default the most ?
3. What family status default the most ?
4. And of course what is the probability that an applicant will default ?
5. What categories of goods the applicants use their loan to purchase ?

It is of utmost important to note that the problem we have here is a ***classification problem***. That is given the dataset can we classify the applicant or predict the probability of default. This is in discrete and not continuous time. If it was continuous time then we will consider it to be a ***regression problem.***

Metrics

The section discusses the metrics used during the project. Evaluation metrics are those metrics that tell us how our model is performing and how to improve it. At least it pinpoint where to go next.

There are various evaluation metrics that can be used depending on the characteristics of the problem and problem domain. Metrics measure the distance between the model and the data. Since we are faced with a classification problem, here are some metrics for the evaluating the performance of the model. They are

- Precision
- Recall
- F1 score
- Confusion matrix
- Accuracy
- ROC_AUC

There are other metrics i did not use by i recommend always to check the scikit learn documentation page. Model evaluation: quantifying the quality of predictions like precision_recall curve, roc curve. This metrics are used to measure classification performance. The aforementioned metrics apply to binary cases where our target is binary. That is, either true or false , 1 or 0.

Before i talk about what these terms mean, i will define some important concepts to help us understand the aforementioned terminology.
HO is null hypothesis.
H1 is alternate hypothesis.

True Positive (TP) : is an outcome where the model correctly predicts the positive class. That is to say the model's prediction is equal to the actual class or outcome.

True Negative (TN) : is an outcome where the model correctly predicts the negative class. That is to say the model's prediction that the event is actually not there and it is truly not there.

False Positive (FP) : is an outcome where the model is predicting that an outcome is there but it is apparently not there meaning the model incorrectly predicts the positive class. This is known as Type 1 error. That is the model is rejecting the actual situation that is true. I.e HO is true but rejected.

False Negative : is an outcome where the model is prediction that an outcome is not there not it is actually there. Model is incorrectly predicting the negative class. That is failing to believe a true alternative hypothesis or failing to reject a false H0.

Now i will define the classification metrics :
Precision : is the proportion of positive identification that are actually correct. It is calculated as TP/(TP+FP). A model that produces no FP has a precision of 1.0
Recall : is the proportion of actual positives that was correctly classified. It is calculated as TP/(TP+FN). A model that produces no FP has a recall of 1.0

F1 score : can be in interpreted as a weighted average of precision and recall, where 1 is the best value and 0 for the worst.
Confusion matrix : it is an N*N table that summarizes how successful a classification model's predictions were.That is the correlation between the label and the model's classification.

Accuracy : is said to be defined as the fraction predictions our model got right. That is : number of correct predictions over the total of predictions.

ROC_AUC : An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds.
Further reading

Data exploration

The dataset are in csv format. The dataset contains 7 .csv files indicating different description. The data was imported from Kaggle api into Jupyter Notebook which the standard notebook for data science. The data contains training set (*application_train.csv*) and the testing set (*application_test*).

- The training set contains 307511 rows (*labels*) and 122 columns (*features*). The target value or what i am trying to predict is '**TARGET'** so after formatting the dataset into a pandas dataframe it can be accessed with *app_train.***TARGET.**
  y is assigned as the *app_train.***TARGET.**
- The testing set contains 48744 rows (*labels*) and 121 columns (*features*).
- HomeCredit description file gives us more information about each columns in the data set.
- The data has 3 data-types :
  - Float (*numeric*) : there are 65 columns with float types .
  - Integer (*numeric*) : there are 41 columns with integer types.
  - Object (*non-numeric*): there are 16 columns with object types.

Numeric can be of discrete time or continuous time horizon. Non-numeric are variables contain labels values rather numeric values. They are sometimes called nominal . Non-numeric data-types can be of text etc. These are also called categorical variables.

- Missing values are values with Nan. There are 67 columns with missing values.
- The shape of the test set and training set are not the same.
- The sample submission.csv file was removed since it is not needed for the analysis but it serves as a guide to what the submission should look like.

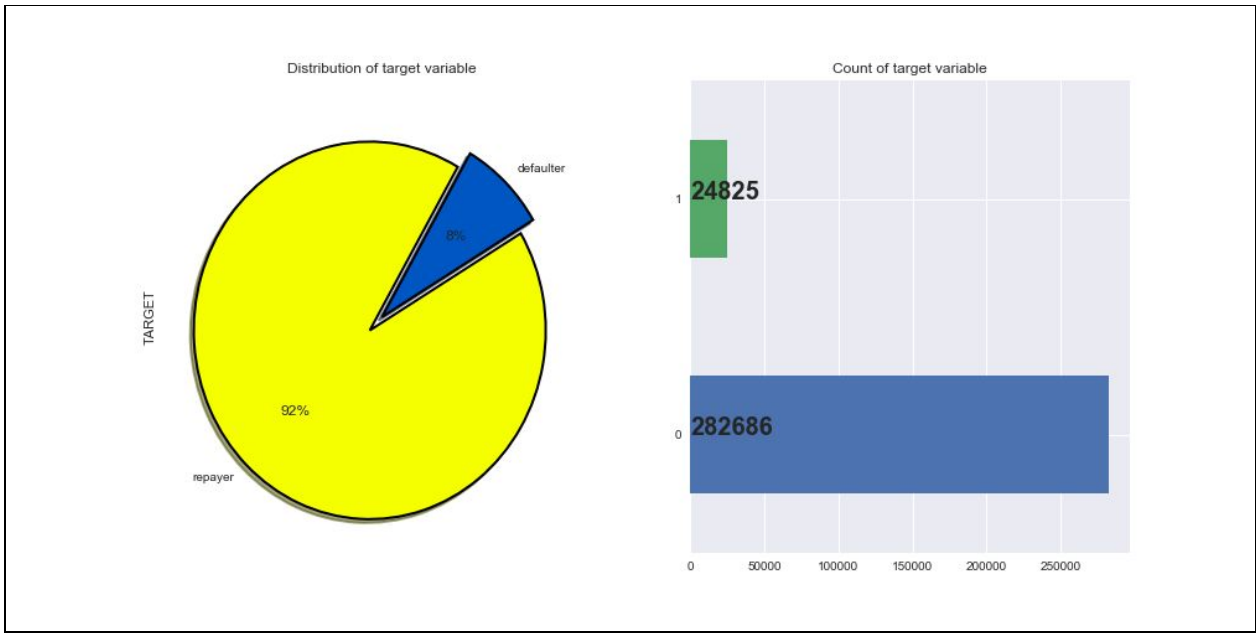
From data description, we can see the basic statistics of all the columns.
The basic statistics derived from the target columns are :

- the mean of the training set : 0.080729
- Standard deviation : 0.272419
- Min : 0.0 and max : 1.0

- In our dataset we have a situation of imbalance class problem.
- There seems to be some kind of anomalies on the DAYS_EMPLOYED columns where we have 1000 years in terms of days employed.

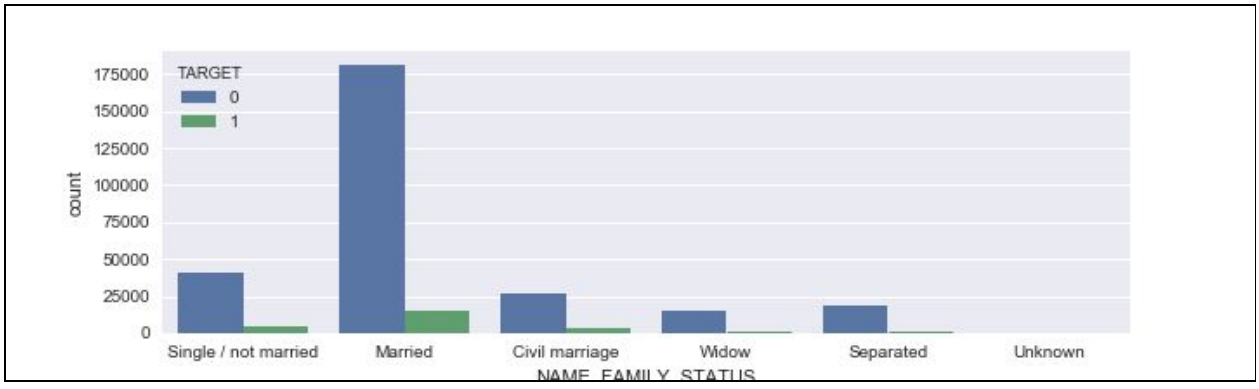
Exploratory Visualization :
In this section i will be showing some relevant visualization about the data.

One question i asked myself was that: how many people pay loans exploratorily ?

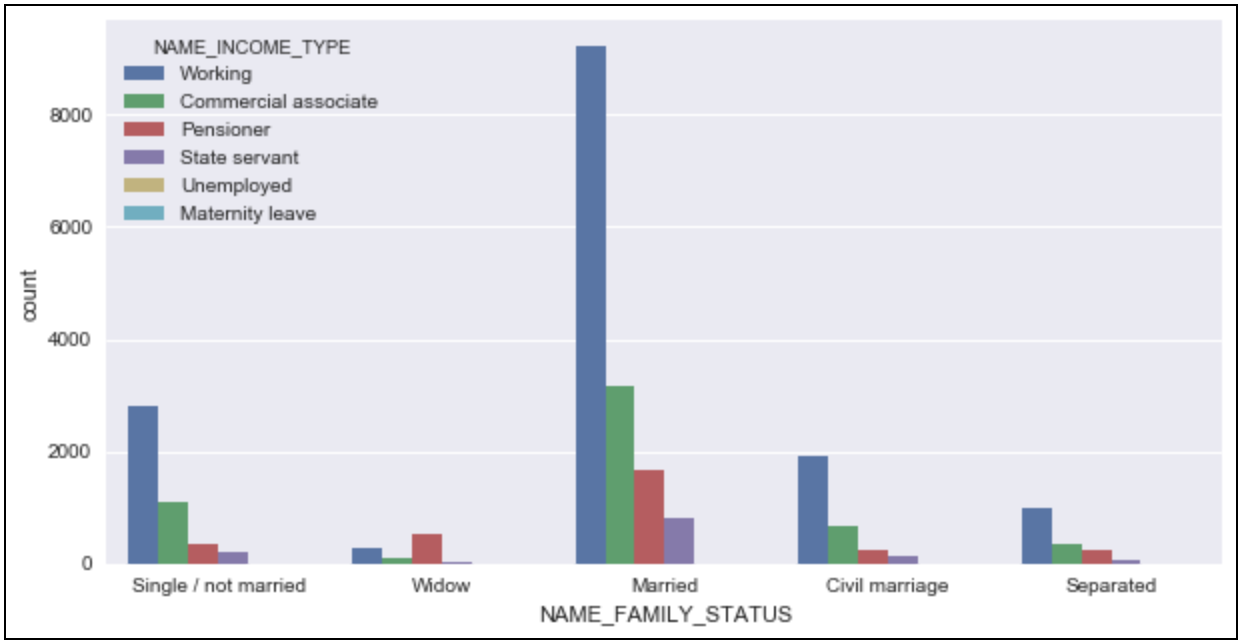


Certainly it is obvious that there is imbalance class problem here.
The difference between the two class is pretty too far. According to the analysis, it shows that the most of the applicant are able to pay for the loan.

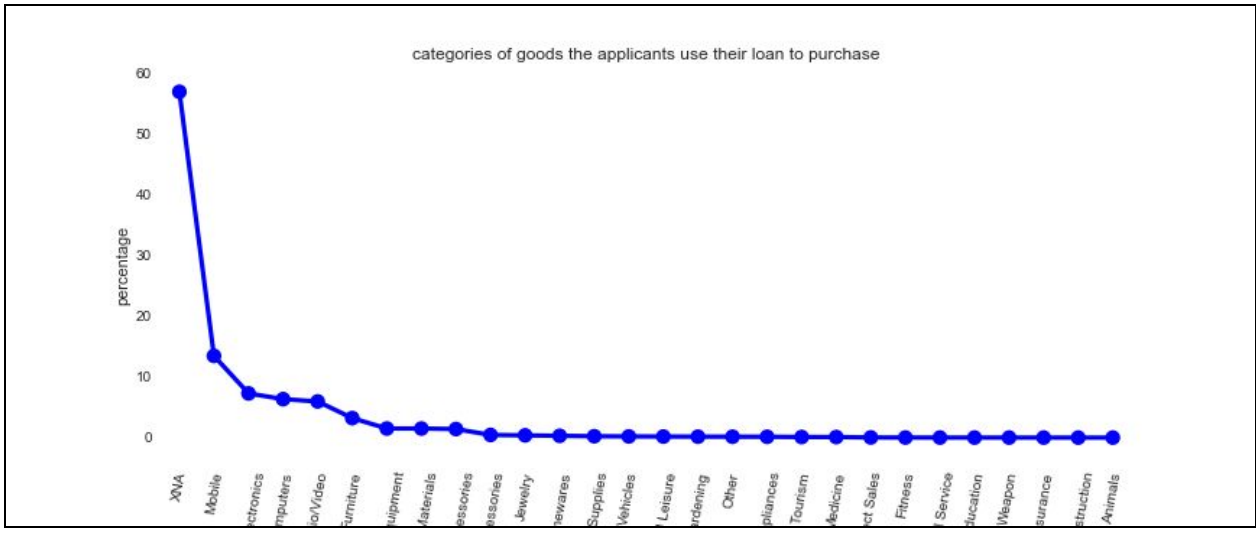
Another question is: What is the family status of the applicants ?



Another question: what income class and family type default the most



What categories of goods the applicants use their loan to purchase ?



Algorithms and Techniques.

In any machine learning project, the algorithms to be used depends on the problem at hand as all algorithms and techniques has their own strengths and weaknesses.
As i have stated before, this is a classification problem with an intention to predict the probability of applicants defaulting.

- **Classification model**

Classification depends on whether the variables we are trying to predict are Binary or Non-Binary.

- Binary variables are those variables where the outcome we are looking are either 1 or 0, True or False.
- Non-Binary variables are those variables where the outcome we are looking are categorical. for example looking at the dataset and predicting where the color of the dress of a person will be Yellow, Brown or Blue

- Binary Classification Model :
  - Logistic regression
  - Decision Trees
  - Support Vector Machine (SVM) : good for anomaly detection especially in large feature sets but the model is slow.

- Non- Binary Classification Model:
  - Adaboost
    Random Forest
    Decision Tree
    Neural Networks

- Considering choosing an algorithm :
  - Take note of the accuracy
  - Training time
  - Linearity
  - Number of parameters
  - Number of features

The Machine Learning Algorithm Cheat Sheet

Logistic Regression Model

LR is statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

Logistic regression generates a probability value between 0 and 1. An example is when you are considering a logistic regression for spam detection. Logistic regression is a very efficient technique when calculating probabilities. Logistic Model out of the box algorithm can be found on the scikit learn documentation page.

Pros are:

- Low variance
- provides probabilities for outcomes
- works well with diagonal (feature) decision boundaries

Cons:

- High bais

Further reading

Decision Tree Classifier

Decisions tree is a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Decision Tree classifier algorithm can be found on the scikit learn documentation page.

Pros:

- They are easy to interpret.
- Requires little data preparation
- Can easily handle qualitative (categorical) features

Cons:

- The classifier is prone to overfitting. Decision-tree learners can create over-complex trees that do not generalise the data well. Setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

Further reading, Classification Model Pros and Cons

Random Forest

Random forest is a type of supervised learning algorithm. This is an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

Pros:

- reduced variance (relative to regular decision trees).
- important when dealing with multiple features which may be correlated

Cons:

- Not as easy to visually interpret

Additional reading, Classification Model Pros and Cons

BENCHMARK

Benchmark as the word itself is a standard against the solution you are making or coming up with. The whole point of it is to get a feel if the solutions are better or worse. Now let's put it in context of machine learning. Benchmarking here means, a standard solution which already performs well.
This nice article helps clarify things in terms of what benchmarking means.
Questions to ask here we will be :

What are the current benchmarks in practice ?
What factors am i going to use to compare my solution against the benchmark. E.g is it accuracy , amount of dataset etc.

Baseline

Baseline is like a model we are trying to beat. Base model should be either the results of a previous model or could be based on a research paper upon which you are looking to improve.

Suppose we have a model that is always predicting 1 (***Defaulters***), what would that model' accuracy and F-score be on this dataset ? This is a Naive Predictor that is showing what a baseline model without any intelligence. When there is no benchmark model set, getting a result better than random choice is a place you could start from.
Baseline prediction will be on random guess with 0.5. So let see if the model performs better than random guessing.

Data preprocessing.

Data preprocessing is a data mining technique used in cleaning, normalization, feature transformation and extraction on a data set. This is done because , in a real world there are lots of data inconsistency, incompleteness, lack of trend so data preprocessing helps address this issue. Note we have

- Data cleansing
- Data editing
- Data reduction
- Data wrangling

I used a MinMaxScaler which is used for scaling feature to range. During this analysis, it was scaled to 0 and 1.

One reason i am doing preprocessing here is because there are a lot of missing data point in the data set.

Going back to our anomalies in the analysis, where we seems to have a huge anomaly (max year is 1000 when converted to years). So i did a percentage comparison using its mean in relation to our target so see how they are affected. The total number of anomalies are 55374 values.

In order to fix this, i replaced those values with np.NaN just in case there is something connecting them. Afterwards, i removed the missing value and imputed some. This was done on both data set. I.e training and test set.

Implementation.

This section talks about the implementation of the algorithm.
I scaled the features into a range of 0 and 1.

I split the dataset (the training set that was given) into a training and test set using the train_test_split function from the scikit learn module.
C is a parameter used in Logistic regression which is for Inverse of regularization strength; must be a positive float. The smaller the value the better.

Refinement
Here is what i noticed when changing the values of C. There seems to be some kind of trade between accuracy and roc_auc score. The score is kind off of increasing and the accuracy of the model does not change a lot. To me this is because the logistic regression is linear, this algorithm can't change much the classification model as the data is not linearly separated.
Also i used a penalty of l2.

Model Evaluation
I tried 10 fold-cross validation to see if the accuracy goes up.
There was not much of a different when i evaluated the model using 10 fold cross validation for logistic.

For random forest algorithm, the model had 0.99 on training accuracy and then when test on the test set, it dropped to 0.92. Then used the 10 fold cross validation , the score mean is around 0.9192
Justification
Looking the current analysis, the model is performing better than the benchmark of 0.5 for random guess. It is also above the naive predictor.
I will not pretend like this cannot be improved because there is obvious room for improvement which requires some tuning and my learning curve and experience keeps increasing i can only improve the result. One thing i am thinking to use deep learning using neural network for the binary classification from keras module.

Reflection
Honestly the whole project was quite tedious. There is a lot to know and there is a lot i do not know. The most tedious part of this project was the data mining part of the project and implementing the algorithm. Fine tuning the parameters is very important which i think requires some level of maths and statistics. I think people should stop saying Machine learning does not require a lot of maths and statistics but certainly it does if you want to understand the algorithm well.
Also the visualization part. I think next time it is better to write a python file that i can just call when i want a certain visualization.

Improvement
For improvement i will try the deep learning classification algorithm from keras. Since i am more interesting in deep learning for artificial intelligence than machine learning itself but i have seen that taking machine learning is helping me to understanding how data can be

used. There are a lot of algorithms out there and i think it is important to choose few to keep researching on them depending on size of the data, training time, speed of prediction, accuracy as well. These are the things that matters in the real world.