MET CS 521 Final Project

Predicting Credit Card Approval with Classification Models

Authors: Martin Bourbier, Aigerim Dussikenova & Jenny Hopkins

May 2022

INTRODUCTION

In this project, we used a dataset with 19 descriptors for each ID that represent factors which could contribute to whether a person would be accepted for a new credit card or not. The factor which we used as an approximation for accepting or denying the credit card application was from the "Status" column of data which described whether that person paid off their credit card for that month, or if they paid late, how late they were (within 1 month, 2 months, etc.). We simplified this to "accept" (status=0) if they paid off their credit card on time or didn't borrow any money that month, and "deny" (status=1) if they did not pay or had a late payment. Of course, in the real world it would not be as simple of a calculation.

We expected income to be a strong predictor of whether a particular person would have late payments (higher income, fewer late payments). Similarly, we expected to find a positive correlation with the number of family members and likelihood of having a late payment (more family members mean increased family expenses). However, surprisingly there was not one single factor that had a very strong correlation (either positive or negative) with the likelihood of having a late payment. This was true even after cleaning the data of variables that were clearly not useful or collinear with other variables - this will be discussed further in another section. The exploring_data.ipynb file explores possible correlations between the possible dependent variables and the output variable, and includes some visualizations.

We then used 3 machine learning models to predict the "accept" or "deny" status of the data's test set: logistic regression, decision tree classification, and random forest classification. Code for the 3 models and their corresponding visualizations can be found in the MLmodel_visualizations.ipynb file. (The code for the models is also included in the process_ml.py file).

DOWNLOADING THE DATASET

To reproduce the results in this report, follow these instructions:

- Run the get_dataset.py file, followed by extract_csv.py scripts locally. This will generate a file called concatenated.csv in the dataset folder, which will then be used to generate the reports and graphs in exploring_data.ipynb and MLmodel_visualizations. Each cell in the ipynb files should be run sequentially. To run get_dataset.py correctly, you need to create an account on kaggle and make sure to download your kaggle.json file. You can find resources online to do this.
- To have user interaction with the model (predict credit card approval results based on a series of questions), run main.py from your terminal and answer the questions.

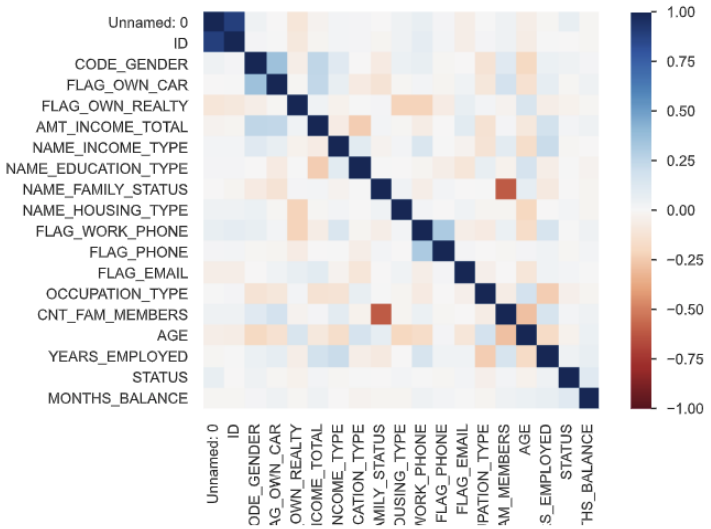
USER INTERACTION

To allow the user to interact with the model, our first idea was to gather all of the information that was used to train the model from the user and put it into a dictionary and then convert it to a data frame. But we realized that this did not work because the data had to be encoded exactly as it was when training the model. We then had to replicate each of the modification steps from extract_csv.py and then feed the data frame to the model. This process worked.

To interact with the program, simply launch main.py and you will be guided through the question set which will predict if your credit card application would be approved or denied.

DATA EXPLORATION

The original data was downloaded from Kaggle in 2 CSV files. We found that one of the original files listed the same ID number multiple times, with different values for months_balance (how many months the account has been open for) and status (whether it was paid off that month). This suggests time series data. Because each ID number was only listed once on the other CSV file, we needed to pick only one of the rows for each ID number. We chose to go with the highest "status" value, meaning that if the user had one or more late payments, they would be in our database as a late payment entry or credit card "deny". We then merged the two CSV files on the ID number. Then, after viewing visualizations and reports about the data, we eliminated variables that clearly had no effect on the outcome - for example, we eliminated flag_mobile since all values were 1. We also eliminated variables that had high collinearity with other variables - for example, we eliminated "count children" because count of family members was another variable. Finally, we dropped many of the no-late-payment rows so that the number of no-late-payment rows equaled the number of late-payment rows, to better assess the effectiveness of the machine learning models. Prior to dropping these values, the accuracy of the models was higher, but this was likely due to the fact that about 90% of the values were "accepts"/no late payments, so the models were accurately predicting results because that particular result was so common.



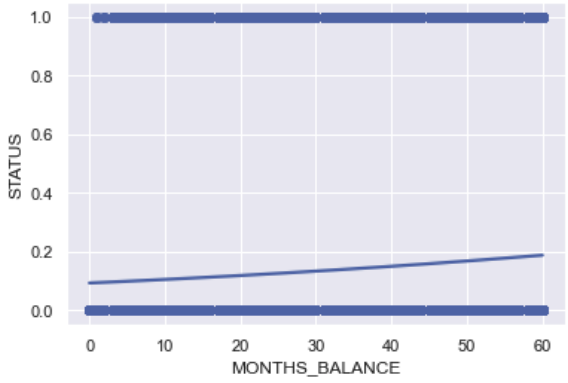
Correlation matrix of factors contributing to the model. Note "Status" does not have a high correlation with any other variable.

MACHINE LEARNING MODELS

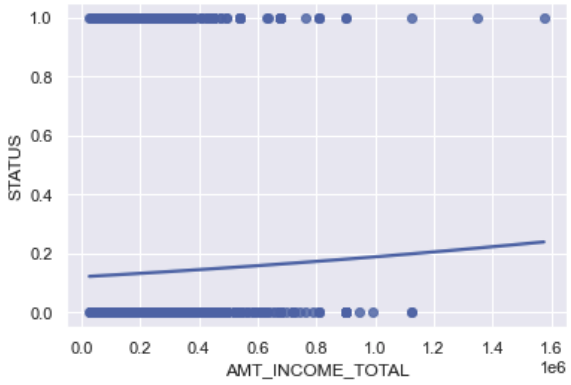
LOGISTIC REGRESSION

The logistic regression model is a basic model for binary classification problems. In our case, this was the first model we picked because we knew we had a binary classification problem (we want to know if the credit card application should be approved or denied). This model falls under the Supervised Learning category. We give the model a set of input and the corresponding output. Based on that, the model will adapt to make sure it gets the right output based on a given input as often as it can.

From the logistic regression correlation graphs we generated, we are including here two graphs that show perhaps surprising results: 1) There is a positive correlation between the number of months an account is open, and the chance that it has a late payment (one would think that the longer a person has an open account, the more likely they are to be responsible and pay on time - however, more months also means more opportunities for late payments.) 2) Higher income is positively correlated with late payments.



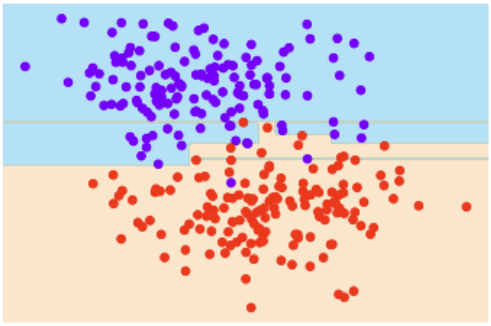
Number of months account is open vs. Late payments (positive correlation)



Income vs. Late payments (positive correlation)

DECISION TREE CLASSIFIER

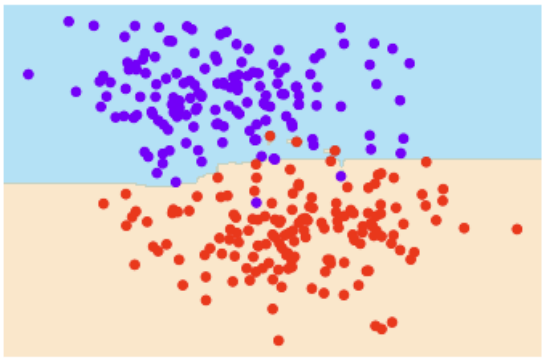
The Decision Tree Classifier is a model that is made to perform well on both regression and classification problems by learning simple decision rules based on the input it is given. From these rules, it creates a decision tree. This model works for our problem since it can learn to recognize the different information that we gathered from the prospective applicant and predict whether their card should be approved or not.



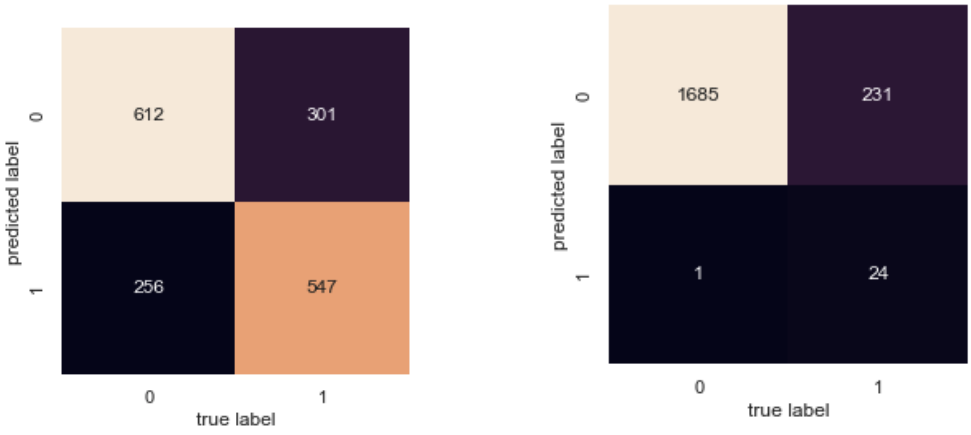
Decision Tree Classifier blob graph - determining the line between accept and deny

RANDOM FOREST

Of the three models, Random Forest Classifier performs the best, with the highest accuracy rate. It uses more estimators and fixes the overfitting problem of the Decision Tree Classifier, so the accuracy for predicting the outcome of new data is higher. By changing the number of estimators we can get a more robust model from it. We did not change the max_depth parameter, and it uses more memory, but it gives more accurate predictions for each estimator.



Random Forest Classifier blob graph - determining the line between accept and deny



Random Forest model Confusion Matrix - shows number of values where predicted label and true label match and mismatch (based on test results) with balanced "deny" and "accept" values in data. (left)

Random Forest model Confusion Matrix with unbalanced (more "accepts"/status 0) data. Note it predicts almost all "accept". (right)

PROJECT EXTENSIONS

USER INPUT INTERACTION WITH MODEL

To allow the user to interact with the model, we need to get all the information used to train the model from the user. This task is done by asking a series of questions from which we gather the information. It is then stored and modified to fit our model. Then we use the model to make a prediction. The output will either be 0 (denied) or 1 (approved) which allows us to give a message back to the user telling them whether or not their credit card application was approved or denied.

COMPARING THE 3 ML MODELS

For all models, we decided to train on 80% of the data and test on 20% of the data after looking at industry norms, and knowing that our sample size was neither too large or small (9702 values). Train-test splitting is important because it allows us to estimate the performance of our machine learning models. The model is trained on the train data, and then makes predictions on the test data – since we have the outputs of the test data, we can therefore find out approximately what percentage of the time the model will correctly predict the outcome.

The decision tree classifier model performs the worst, with an accuracy rate of 0.78; the random forest model is a large improvement over the decision tree classifier model, with an accuracy rate of 0.88. Logistic regression performs about the same as random forest, with an accuracy rate of 0.87. However, we would like to acknowledge that our data is very skewed, with 87% of the data points being "accepts"/no late payments. When we ran an experiment by dropping the number of "accepts" randomly so that the data would have 50% accepts and 50% denies, we found that the accuracy of all 3 models dropped to 0.52 - possibly because of the small number of data that remained (2,556 values total).

REAL-WORLD PROBLEMS (EXTENDING ANALYSIS)

There are some possible problems with our data and models. We oversimplified the classification problem by deciding to deny anyone who has one or more late payments. But late payments sometimes occur and people who make the occasional late payment are still able to get credit cards, so if we were able to build a more sophisticated model we could determine another more nuanced way of categorizing people into “accept” or “deny”.

Secondly, from looking closely at the data, it seems likely that the same individuals are listed multiple times with different ID numbers, months_balance, and status values but with the same data in all other columns. The result is that this may have negatively affected the results of our machine learning models, since it was “learning” from different rows of data that may actually have repeated information. If we were to do the project again, we could potentially find all of the same “people” through unique values for age and years employed (which are very specific float values), and perhaps occupation, and take an average value for months on the account and status (no late payment = 0 and late = 1), and accept if the average for status is below a certain threshold, for example 0.1. However, doing this consolidation may also diminish our data by quite a large amount, leaving us with a comparatively small sample size.

Our data does not appear to reveal any correlations between defaulting on payments and having a certain income level, or being of a certain gender, having a certain family size, marriage status, etc. The project would need to be repeated on further data sets to determine conclusively whether there could be such correlations.

CONCLUSION

This project allowed us to get a better look at a real world application of data science and machine learning. The thinking process about data cleaning was very interesting as we had to discuss which values to drop and how to convert or interpret the values, how to determine collinearity, what values matter most to the model, etc. A good example of data conversion is from DAYS_BIRTH to AGE. The machine learning part allowed us to expand our knowledge of the field by introducing 3 models and analyzing how they work to find the one that has the best fit for our data. We were also able to explore data visualization. All in all, we appreciated the opportunity to have some hands-on experience with data science and machine learning, and we learned a lot from this project.