# Machine Learning Engineer Nanodegree

## Capstone Project

Tim Latham
July 21st, 2017

## I. Definition

### Project Overview

Prior to beginning my education in artificial intelligence and machine learning I built an underwriting and pricing system for auto loans issued directly by car dealerships. In researching potential datasets to apply my new skills to I came across the Lending Club dataset on Kaggle and decided it would be a good test to see if I could apply my new skills in an area I was already familiar with. My goal for this project was to see if I could use machine learning on a subset of this data to accurately predict whether or not a loan would be successful based on the origination data.

Here is a description of the dataset provided by Wendy Kan on the Kaggle Lending Club data page that provides a good overview of the dataset this capstone project is analyzing:

- These files contain complete loan data for all loans issued through the 2007-2015, including the current loan status (Current, Late, Fully Paid, etc.) and latest payment information.
- The file containing loan data through the "present" contains complete loan data for all loans issued through the previous completed calendar quarter.
- Additional features include credit scores, number of finance inquiries, address including zip codes, and state, and collections among others.
- The file is a matrix of about 890 thousand observations and 75 variables.

### Problem Statement

My goal for this project was to see if I could use machine learning on a subset of this data to accurately predict whether or not a loan would be successful based on the origination data. The intended solution for this problem is a learner that could be used as an automated underwriting engine in the approval of new loans.

My strategy for solving this problem was as follows:

1. Review the data dictionary to understand the available fields
2. Based on past knowledge of loan underwriting remove fields not likely to be relevant to the results
3. Remove fields that duplicate similar information to avoid double counting
4. Remove Lending Club grading data to ensure results are independent of the existing engine

5. Explore the underlying dataset and determine which fields will require normalization/standardization or one-hot encoding.
6. Review data fields and identify records that may need to be removed.
    - Could be outliers that would overly impact the results
    - May also be null fields that would cause issues with learning
    - Loans that have not gone to a final status of paid in full or default are removed to ensure only full life-cycle loans have been included
7. Identify and separate the feature to be predicted from the target
8. Split the data into training & testing sets
9. Run various classifiers with different parameters, and identify the classifier and parameters that provide optimum performance in terms of time and accuracy

At the end of these steps a learner will exist that can be used with new origination data to accurately determine if a loan will pay in full or default.

## Metrics

The performance of this learner will be judged by time and accuracy. Speed measured by time is very important in making consumer lending decisions as the market and consumer preferences require virtually instant decisions. Accuracy is of utmost importance to the lender, as their goal is to ensure approval of all loans that will be successful, and minimize missing out on business they could have otherwise generated.

To judge time, two metrics will be used:

- Time to train the models in seconds
    - The model will need to be updated on an ongoing basis to ensure it reflects changes in market and other conditions that may be impacting results, therefore a faster time to train is desirable.
- Time to make predictions in seconds
    - Existing consumer loan underwriting engines deliver near instant decisions, so this model must be able to deliver results as fast as possible or it won't be a viable alternative.

To judge accuracy, one metric will be used:

- F1 score The F1 score incorporates both precision and recall, making it a more complete measure of model accuracy than many alternatives.
    - Details on F1 score and related data below are sourced via the links to the scikit-learn website
        - The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is:
            - F1 = 2 * (precision * recall) / (precision + recall)
        - Precision, Recall & F-measures
            - Precision is the ability of the classifier not to label as positive a sample that is negative, and recall is the ability of the classifier to find all the positive samples.

# II. Analysis

## Data Exploration

The Lending Club dataset is available from Kaggle under datasets, here is a link: https://www.kaggle.com/wendykan/lending-club-loan-data

The following table is the data dictionary provided at the link above, with columns I added to denote whether or not I included the field in the analysis, whether the data in the field required normalization or one-hot encoding, and notes providing more information justifying these decisions. The Excel version of this information is available in my GitHub site for this project.

| LoanStatNew | Description | Include? | Normalization? | One-Hot? | Notes |
|---|---|---|---|---|---|
| home_ownership | The home ownership status provided by the borrower during registration. Our values are: RENT, OWN, MORTGAGE, OTHER. | Yes | No | Yes | Remove 185 records that aren't rent, own or mortgage , 3 different options to encode with one-hot |
| loan_status | Current status of the loan | Yes | No | Yes | This is the Y or target variable - convert to 0/1 format - with 1 = Charged Off |
| purpose | A category provided by the borrower for the loan request. | Yes | No | Yes | Unknown if relevant or not - 14 different options to encode with one-hot |
| term | The number of payments on the loan. Values are in months and can be either 36 or 60. | Yes | No | Yes | 60 month are almost 2x more likely to C-O. 2 options for one-hot, or convert to 0/1, 1 = 60 months |
| verification_status | Indicates if income was verified by LC, not verified, or if the income source was verified | Yes | No | Yes | 3 different options to encode with one-hot |
| annual_inc | The self-reported annual income provided by the borrower during registration. | Yes | Yes | No | C-O avg is significantly below PIF avg |
| collections_12_mths_ex_med | Number of collections in 12 months exduding medical collections | Yes | Yes | No | Remove 56 records that are blank for this item |
| delinq_2yrs | The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years | Yes | Yes | No | 0 blanks |
| dti | A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower's self-reported monthly income. | Yes | Yes | No | 0 blanks |
| emp_length | Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years. | Yes | Yes | No | Need to convert the text field to actual numbers, then normalize ("2 years", etc)<br>* -1 = n/a, 0 = < 1 year, etc |
| funded_amnt | The total amount committed to that loan at that point in time. | Yes | Yes | No | Is the absolute level relevant? C-O avg is meaningfully higher than PIF avg |
| inq_last_6mths | The number of inquiries in past 6 months (excluding auto and mortgage inquiries) | Yes | Yes | No | 0 blanks |
| installment | The monthly payment owed by the borrower if the loan originates. | Yes | Yes | No | Prob less relevant than DTI |
| mths_since_last_delinq | The number of months since the borrower's last delinquency. | Yes | Yes | No | If blank, none in history, set these items to = 180 |
| mths_since_last_major_derog | Months since most recent 90-day or worse rating | Yes | Yes | No | If blank, none in history, set these items to = 180 |
| mths_since_last_record | The number of months since the last public record. | Yes | Yes | No | If blank, none in history, set these items to = 180 |
| open_acc | The number of open credit lines in the borrower's credit file. | Yes | Yes | No | 0 blanks |
| pub_rec | Number of derogatory public records | Yes | Yes | No | 0 blanks |
| revol_util | Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit. | Yes | Yes | No | Remove 199 records where this is blank |
| acc_now_delinq | The number of accounts on which the borrower is now delinquent. | No | No | No | 99.7% of the records show 0, not a relevant feature for the analysis |

| LoanStatNew | Description | Include? | Normalization? | One-Hot? | Notes |
|---|---|---|---|---|---|
| addr_state | The state provided by the borrower in the loan application | No | No | No | Exclude geo impact, focus on pure credit related info |
| all_util | Balance to credit limit on all trades | No | No | No | 99.9% blanks |
| annual_inc_joint | The combined self-reported annual income provided by the co-borrowers during registration | No | No | No | Only one item with a joint borrower in the list |
| application_type | Indicates whether the loan is an individual application or a joint application with two co-borrowers | No | No | No | Only one item with a joint borrower in the list |
| collection_recovery_fee | post charge off collection fee | No | No | No | Not relevant to classifying PIF vs. C-O |
| desc | Loan description provided by the borrower | No | No | No | Focus on pure credit related info |
| dti_joint | A ratio calculated using the co-borrowers' total monthly payments on the total debt obligations, excluding mortgages and the requested LC loan, divided by the co-borrowers' combined self-reported monthly income | No | No | No | Only one item with a joint borrower in the list |
| earliest_cr_line | The month the borrower's earliest reported credit line was opened | No | No | No | Need to transform this to a time in years - date difference between this and the issue_d field - exclude for initial run |
| emp_title | The job title supplied by the Borrower when applying for the loan.* | No | No | No | Focus on pure credit related info |
| fico_range_high | The upper boundary range the borrower's FICO at loan origination belongs to. | No | No | No | Not in the csv data file |
| fico_range_low | The lower boundary range the borrower's FICO at loan origination belongs to. | No | No | No | Not in the csv data file |
| funded_amnt_inv | The total amount committed by investors for that loan at that point in time. | No | No | No | Average is 99%, not especially relevant |
| grade | LC assigned loan grade | No | No | No | Double counting - grade is driven by credit factors |
| id | A unique LC assigned ID for the loan listing. | No | No | No | Not relevant to classifying PIF vs. C-O |
| il_util | Ratio of total current balance to high credit/credit limit on all install acct | No | No | No | 99.9% blanks |
| initial_list_status | The initial listing status of the loan. Possible values are – W, F | No | No | No | Not relevant to classifying PIF vs. C-O |
| inq_fi | Number of personal finance inquiries | No | No | No | 99.9% blanks |
| inq_last_12m | Number of credit inquiries in past 12 months | No | No | No | 99.9% blanks |
| int_rate | Interest Rate on the loan | No | No | No | Double counting - rate is driven by grade |
| issue_d | The month which the loan was funded | No | No | No | Not relevant to classifying PIF vs. C-O |
| last_credit_pull_d | The most recent month LC pulled credit for this loan | No | No | No | Not relevant to classifying PIF vs. C-O |
| last_fico_range_high | The upper boundary range the borrower's last FICO pulled belongs to. | No | No | No | Not in the csv data file |
| last_fico_range_low | The lower boundary range the borrower's last FICO pulled belongs to. | No | No | No | Not in the csv data file |
| last_pymnt_amnt | Last total payment amount received | No | No | No | |
| last_pymnt_d | Last month payment was received | No | No | No | |

| LoanStatNew | Description | Include? | Normalization? | One-Hot? | Notes |
|---|---|---|---|---|---|
| loan_amnt | The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value. | No | No | No | |
| max_bal_bc | Maximum current balance owed on all revolving accounts | No | No | No | 99.9% blanks |
| member_id | A unique LC assigned Id for the borrower member. | No | No | No | |
| mths_since_rcnt_il | Months since most recent installment accounts opened | No | No | No | 99.9% blanks |
| next_pymnt_d | Next scheduled payment date | No | No | No | |
| open_acc_6m | Number of open trades in last 6 months | No | No | No | 99.9% blanks |
| open_il_12m | Number of installment accounts opened in past 12 months | No | No | No | 99.9% blanks |
| open_il_24m | Number of installment accounts opened in past 24 months | No | No | No | 99.9% blanks |
| open_il_6m | Number of currently active installment trades | No | No | No | 99.9% blanks |
| open_rv_12m | Number of revolving trades opened in past 12 months | No | No | No | 99.9% blanks |
| open_rv_24m | Number of revolving trades opened in past 24 months | No | No | No | 99.9% blanks |
| out_prncp | Remaining outstanding principal for total amount funded | No | No | No | |
| out_prncp_inv | Remaining outstanding principal for portion of total amount funded by investors | No | No | No | |
| policy_code | publicly available policy_code=1<br>new products not publicly available policy_code=2 | No | No | No | All items remaining are a 1 |
| pymnt_plan | Indicates if a payment plan has been put in place for the loan | No | No | No | |
| recoveries | post charge off gross recovery | No | No | No | |
| revol_bal | Total credit revolving balance | No | No | No | Does not appear to be relevant to outcome |
| sub_grade | LC assigned loan subgrade | No | No | No | Double counting - grade is driven by credit factors |
| title | The loan title provided by the borrower | No | No | No | |
| tot_coll_amt | Total collection amounts ever owed | No | No | No | 25% blanks, exclude field? |
| tot_cur_bal | Total current balance of all accounts | No | No | No | 25% blanks, exclude field? |
| total_acc | The total number of credit lines currently in the borrower's credit file | No | No | No | Does not appear to be relevant to outcome |
| total_bal_il | Total current balance of all installment accounts | No | No | No | 99.9% blanks |
| total_cu_tl | Number of finance trades | No | No | No | 99.9% blanks |
| total_pymnt | Payments received to date for total amount funded | No | No | No | |
| total_pymnt_inv | Payments received to date for portion of total amount funded by investors | No | No | No | |
| total_rec_int | Interest received to date | No | No | No | |
| total_rec_late_fee | Late fees received to date | No | No | No | |
| total_rec_prncp | Principal received to date | No | No | No | |
| total_rev_hi_lim | Total revolving high credit/credit limit | No | No | No | 25% blanks, exclude field? |
| url | URL for the LC page with listing data. | No | No | No | |
| verification_status_joint | Indicates if the co-borrowers' joint income was verified by LC, not verified, or if the income source was verified | No | No | No | Only one item with a joint borrower in the list |
| zip_code | The first 3 numbers of the zip code provided by the borrower in the loan application. | No | No | No | Exclude geo impact, focus on pure credit related info |

# Exploratory Visualization

## Summary of labels:

- Number of loans paid in full: 45161.0
- Number of charge-offs: 207374.0
- Success rate of loans: 17.88%

The following data summary includes a summary of the remaining features of the dataset provided using the .describe() function. This function summarizes continuous data providing the mean, standard deviation, minimum, maximum, and 25%/50%/75% levels.

I used this during the data scrubbing process to identify outliers for removal or modification, as items that are significantly (3+ standard deviations for example) above or below the mean can overly impact results.

**Summary of features using .describe()**

| Field | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| funded_amnt | 252,535 | 13,527.64 | 8,108.76 | 500.00 | 7,200.00 | 12,000.00 | 18,125.00 | 35,000.00 |
| term | 252,535 | 0.22 | 0.42 | - | - | - | - | 1.00 |
| installment | 252,535 | 418.25 | 244.90 | 15.69 | 239.56 | 365.23 | 547.55 | 1,424.57 |
| emp_length | 252,535 | 5.56 | 3.79 | (1.00) | 2.00 | 6.00 | 10.00 | 10.00 |
| annual_inc | 252,535 | 72,533.06 | 58,792.48 | 3,000.00 | 45,000.00 | 62,000.00 | 87,000.00 | 8,706,582.00 |
| dti | 252,535 | 16.55 | 7.79 | - | 10.76 | 16.20 | 22.00 | 57.14 |
| delinq_2yrs | 252,535 | 0.25 | 0.74 | - | - | - | - | 29.00 |
| inq_last_6mths | 252,535 | 0.85 | 1.07 | - | - | 1.00 | 1.00 | 8.00 |
| mths_since_last_delinq | 252,535 | 115.82 | 73.38 | - | 37.00 | 180.00 | 180.00 | 180.00 |
| mths_since_last_record | 252,535 | 166.99 | 35.98 | - | 180.00 | 180.00 | 180.00 | 180.00 |
| open_acc | 252,535 | 10.93 | 4.81 | 1.00 | 7.00 | 10.00 | 14.00 | 30.00 |
| pub_rec | 252,535 | 0.14 | 0.43 | - | - | - | - | 7.00 |
| revol_util | 252,535 | 54.31 | 24.73 | - | 36.30 | 55.80 | 73.90 | 150.00 |
| collections_12_mths_ex_mec | 252,535 | 0.01 | 0.09 | - | - | - | - | 6.00 |
| mths_since_last_major_dero | 252,535 | 154.31 | 54.19 | - | 180.00 | 180.00 | 180.00 | 180.00 |

# Algorithms and Techniques

I intend to develop a custom program to scrub the data and provide a starting file for the learner to be built from. There is no standard algorithm that completes all the file cleanup processes I am contemplating.

I intend to use the pandas get_dummies() algorithm to generate one-hot encoding for fields I have identified as needing that. This automates a process I would otherwise have to complete manually.

I intend to use the sklearn train_test_split() algorithm to generate my training and testing data from the full data set. This has the advantage of quickly completing what would otherwise be a convoluted process, and ensures that my test data is never seen during training.

I intend to use a variety of sklearn classification algorithms to train the model and determine which is optimal. I have set up the dataset to be predicting a binary 1/0 pass/fail outcome, therefore it is a classification task. I will use multiple classification algorithms tested against the time and accuracy metrics defined earlier rather than hoping I just happen to guess the best one to use.

# Benchmark

I have chosen a benchmark of 0.90 or greater for both training and testing as determining the success of a model. Additionally, the gap between training and testing results should be less than 0.025. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. Ensuring that the model meets these two benchmarks will ensure that the model is performing at an optimal level with minimal risk of overfitting.

# III. Methodology

## Data Preprocessing

The data dictionary Excel file documented and linked in section II. identify which fields are included, normalized/standardize, and one-hot encoded.

The loanFileCleanup.py code performs the following steps using Python with the pandas module imported:

1. Reads the loan.csv file provided in the Kaggle Lending Club zip
2. Removes the appropriate fields from the dataset
3. Removes records as identified in my modified Excel data dictionary file
4. Converts data fields in the manner described in the modified data dictionary file
5. Exports the modified dataset to a new file, loan_data.csv, to be used in the loanAnalysis.ipynb for building the model

The loanAnalysis.ipynb is the master file for the project, but does also include some data preprocessing steps:

1. Read the new loan_data.csv file
2. Store the 'loan_status' feature in a new variable and remove it from the dataset
3. Scaling of target variables. That is, we'll shift and scale the variables such that they have zero mean and a standard deviation of 1.
4. Complete the train-test data split

## Implementation

The algorithms and techniques for this model were implemented using the Jupyter (iPython) notebook program provided via an installation of Anaconda Navigator.

Libraries utilized in the loanAnalysis.ipynb file included numpy, pandas, time, scikit-learn, and matplotlib.

The implementation went through the following steps:

1. Run the loanFileCleanup.py program to generate the loan_data.csv file for use in the loanAnalysis.ipynb Jupyter notebook.
2. Import the relevant libraries
3. Read the data file
4. Provide a quick view of the data to ensure import was successful
5. Provide brief summary of imported data
6. Store the 'loan_status' feature in a new variable and remove it from the dataset
7. Calculate and display summary statistics
8. Perform one-hot encoding for 3 fields requiring it

9. Scale the target variables that have continuous data
10. Complete the train-test split
11. Bring in the code from the student_intervention project and modify for this project for training and predictions
12. Import classifiers, initialize, train, make predictions, evaluate results
13. Evaluate alternative learning rate parameters and optimize
14. Implement neural networks at default
15. Enhance neural network via parameter optimization

## Refinement

In the initial grouping of classifiers the AdaBoost classifier performed best (and was the only one to meet all requirements defined in metrics above), I therefore selected that for refinement via parameter optimization. I tweaked a number of the parameters (not shown in final product) to determine that learning rate provided the most benefit. I then ran the AdaBoost classifier at a variety of learning rates (not shown) and left the final at what I determined to be the optimal setting based on the previously established time and accuracy metrics.

I also wanted to explore a basic neural network for comparison, so I ran the scikit-learn multi-layer perceptron (MLPClassifier). For the initial run I left default settings (with the exception of a random_state of 42). This provided comparable performance to the AdaBoost classifier, although training time was much slower, prediction time was much faster and accuracy was comparable.

I then tuned the MLPClassifier to see if similar accuracy results could be achieved while bring training time closer to the levels used in AdaBoost. Changing the activation to 'identity' significantly cut the training and prediction time with minimal impacts to accuracy.

- Activation function for the hidden layer.
    - ‘identity’, no-op activation, useful to implement linear bottleneck, returns f(x) = x

Finally, I changed early_stopping to be true, which further cut the training time, while having a minimal impact of prediction time and accuracy.

- early_stopping : bool, default False
    - Whether to use early stopping to terminate training when validation score is not improving. If set to true, it will automatically set aside 10% of training data as validation and terminate training when validation score is not improving by at least tol for two consecutive epochs.

# IV. Results

## Model Evaluation and Validation

The Results tab of the provided Excel data dictionary file shows a visual summary in tabular format of the results for each of the model runs summarized in the Refinement section above. In the end, the MLP classifier with the activation and early_stopping parameters modified as discussed met all the required metrics and delivered the optimal performance. The data set is of significant size, and testing indicates the model generalizes well to unseen data as shown by the minimal gap between training and testing scores. The model is robust, and can be quickly and easily be updated as new data comes in to ensure it is continuously accounting for the latest results. The F1 score of over .90 for both training and testing exhibits a high degree of precision and recall, indicating that the results are trustworthy for a production operation.

## Justification

I chose a benchmark of 0.90 or greater for both training and testing as determining the success of a model. Additionally, the gap between training and testing results should be less than 0.025. As shown in the following table, the MLPClassifier with activation and early_stopping optimized exceeds these levels and provides the best overall performance. The final model is accurate and fast enough to be used in a production environment.

| Model | Train time | Prediction time | F1 Training | F1 Testing | Training - Testing |
|---|---|---|---|---|---|
| GaussianNB | 0.1306 | 0.0441 | 0.8657 | 0.8650 | 0.0007 |
| AdaBoostClassifier | 9.2355 | 0.2538 | 0.9013 | 0.9019 | (0.0006) |
| RandomForestClassifier | 4.3735 | 0.2354 | 0.9953 | 0.8847 | 0.1106 |
| AdaBoostClassifier (learning_rate) | 9.1149 | 0.2455 | 0.9016 | 0.9027 | (0.0011) |
| MLPClassifier | 30.4920 | 0.0685 | 0.9019 | 0.9005 | 0.0014 |
| MLPClassifier (activation) | 3.9776 | 0.0398 | 0.9013 | 0.9022 | (0.0009) |
| MLPClassifier (activation & early stop) | 2.3122 | 0.0373 | 0.9010 | 0.9014 | (0.0004) |
| | | | | | |
| Green Pass/Red Fail | | | X | X | X |
| Green Best/Red Worst (model too) | X | X | | | |

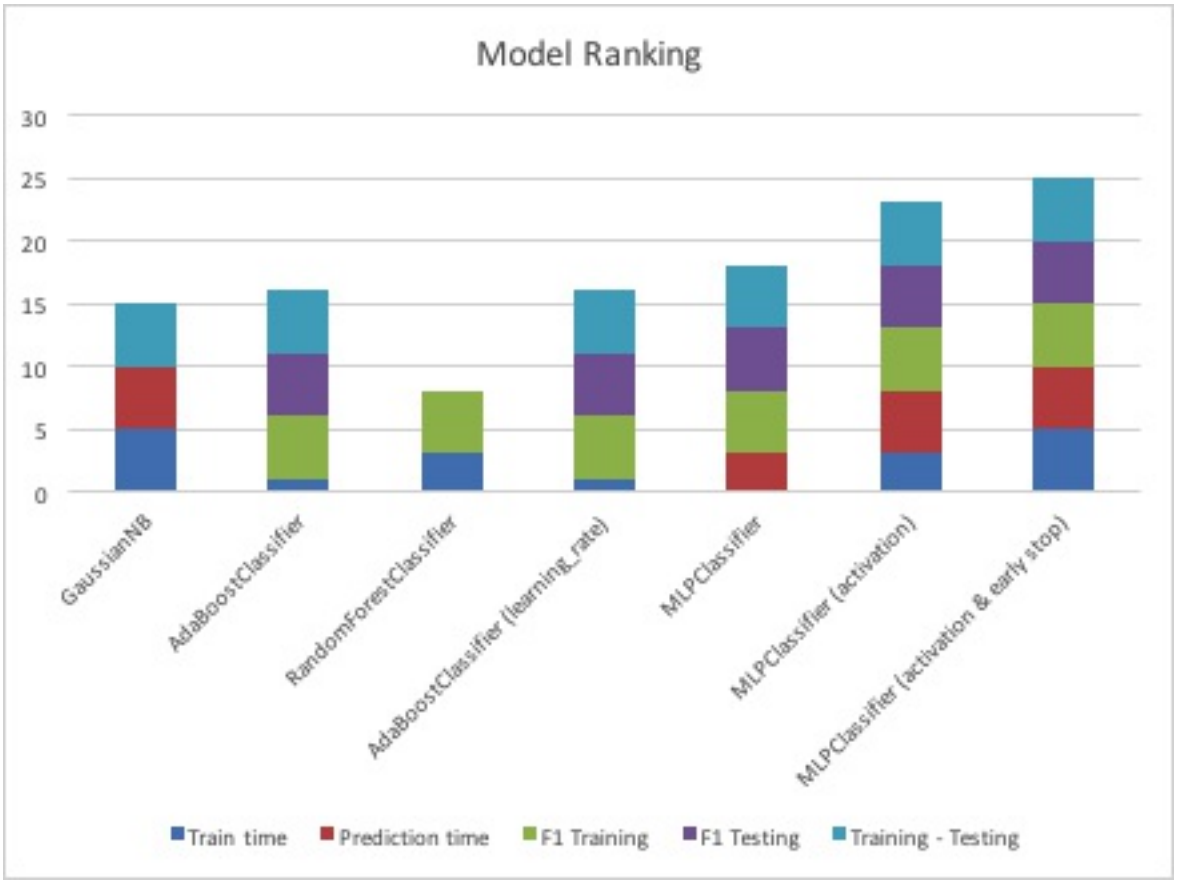
# V. Conclusion

## Free-Form Visualization

I created a scoring system to provide a heuristic for relatively ranking the various models. The following table shows the scoring system and the outcomes:

| Scoring the Models | | | | | | |
|---|---|---|---|---|---|---|
| Model | Train time | Prediction time | F1 Training | F1 Testing | Training - Testing | Total Points |
| GaussianNB | 5 | 5 | 0 | 0 | 5 | 15 |
| AdaBoostClassifier | 1 | 0 | 5 | 5 | 5 | 16 |
| RandomForestClassifier | 3 | 0 | 5 | 0 | 0 | 8 |
| AdaBoostClassifier (learning_rate) | 1 | 0 | 5 | 5 | 5 | 16 |
| MLPClassifier | 0 | 3 | 5 | 5 | 5 | 18 |
| MLPClassifier (activation) | 3 | 5 | 5 | 5 | 5 | 23 |
| MLPClassifier (activation & early stop) | 5 | 5 | 5 | 5 | 5 | 25 |
| | | | | | | |
| Points Possible | 5 | 5 | 5 | 5 | 5 | |
| Max Points | <3 | <.05 | >.90 | >.90 | <.025 | |
| Min Points | >10 | >.2 | <.90 | <.90 | >.025 | |
| | | | | | | |
| Green Best/Red Worst | | | | | | |

The following graphic provides a clean visual representation of the relative performance of each model, as well as highlighting the contribution of each metric to the total score:



The heat mapped table and stacked column graphic clearly show why the MLP classifier was chosen as the best of the possible options. It had the highest overall score, which also happened to be a perfect score based on the scoring methodology applied.

## Reflection

Overall, this was a very interesting project as it utilized my existing knowledge of consumer loan underwriting and allowed me to apply my new skills in machine learning. It was interesting to me to rediscover that machine learning is not a magic bullet; there is still significant human involvement required in the domain of expert knowledge and feature selection as well as other parts of the process. The most difficult portion of the project for me was presenting the model and my findings in a clear and concise manner at the end so that others can follow my logic and see the value I have created. The final model met my expectations for the problem, I wish I had this skillset when I was building my pricing and underwriting engine for consumer auto loans! I would definitely use this model in a general setting to solve similar types of problems.

## Improvement

This implementation was built specifically for loans in the realm that Lending Club underwrites and originates. The performance data is necessarily impacted by their customer base, target market, etc. It would be beneficial to broaden it to a general consumer loan underwriting engine, or enable to build upon the Lending Club data for originator specific underwriting in a similar marketspace. Broadening the dataset with a particular institutions historical dataset and training the model accordingly would allow for more accurate and tailored results for their risk profile.