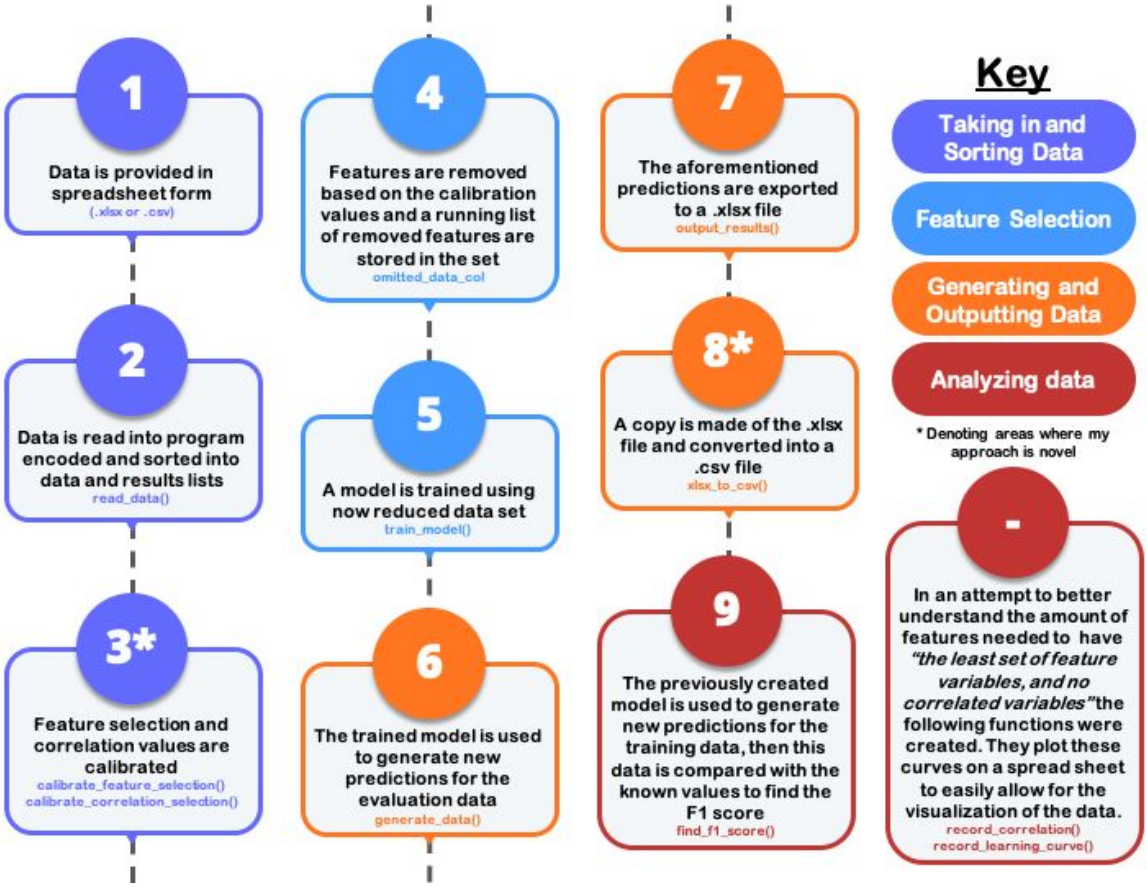
Wells Fargo Campus Analytics Challenge 2020 - My Methods
Daniel Ajagbusi
August 6th, 2020
University of Minnesota - Twin Cities

A visual description of the path of the data through



DISCLAIMER:
Though I refer to them as sets in the following passages, all "sets" other than ***omitted_data_col*** *are actually python lists. This choice was made to allow the passages to feel more fluid in their conceptions.*

1.) Data is provided in a spreadsheet

Because the data was provided in .xlsx format I decided to use the openpyxl python library to allow for simple read and write processes. However, upon reading the official rules I saw that we may have been expected to output the file as a .csv so I made note of this for later. After looking through the data I saw that each row was composed of 30 rational numbers and a letter, followed by a binary bit denoting the end result.

2.) Reading data into the program

I began by iterating over each row encoding the letter at the end of each row as its ASCII value minus 64, this is a form of ordinal encoding. (To result in A=1, B=2, … in order to make it easier

to interpret for anyone who was viewing the data once entered into the program) This was based on the assumption that this model would only be used for data in the provided format. The numbers did not need to be encoded, because machine learning processes work best with numbers. While this was happening the results (The final column) was also being recorded into its own set. These sets were made global to allow for interactions within other functions.

3.) *Feature selection and correlation value calibrations

Because one of the main points of the challenge was to create a model with *"the least set of feature variables, and no correlated variables in the set of predictors"* I devised a system to self calibrate the data that would be used to train the model. This system works by incrementing the percent of top variables / minimum correlation allowed for features (columns) to be used and comparing the resulting output from the newly trained model of the training data to the known values of the training data. The process works by having a user specify an accuracy percentage then the function will find the smallest number of features needed to reach said accuracy. Because I approach this in a linear fashion it can be very time consuming if the precision parameter is set especially low. However, I don't believe it would be too difficult to improve upon this in future iterations, because of the linear nature of logistic regression. But once a system is calibrated the values can be saved to be used instantly on the next run, hence why I set the variables of feature_variables_percentage_value = 50 and min_correlated_value_value = 5.64 respectively. Using 99% accuracy as a novel cut off point.

4.) Features deemed obsolete are omitted

Once we know the feature variable percentage and the minimum correlation value we would like to use to build our final model, through the process contained in the "run" function a set of omitted features are accumulated in the "omitted_data_col" set. These features are then removed from the working training data set.

5.) Training the model

After the undesirable features are removed from the training data set, this new data set in combination with the results data set acquired in step one are used to train the model. I made the choice to use logistic regression because of the binary nature of the results.

6.) Generating predictions

Reading in the data from the evaluation spreadsheet, similar to step one then removing the features denoted in step 4 and using the model trained in step 5, I was able to generate predictions and subsequently store said predictions of each of the 7000 rows denoted in the evaluation set provided.

7.) Exporting the predicted data to a .xlsx file

Using the data generated in step 6 I cycled through the predictions along with a counter variable to export the data in the format denoted in the official rules.

8.) *Creating a copy as a .csv file

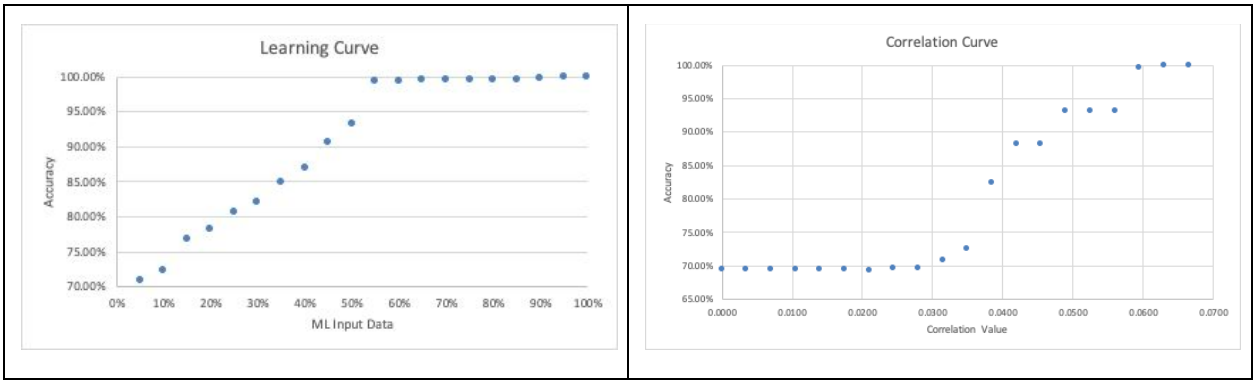
As was denoted in step 1, I was unsure if we were expected to have our final results as a .xlsx or a .csv so I created a function to make a copy of the .xlsx generated in step 7, but export it as a .csv to ensure all rules were followed.

9.) Finding the F1 score

Because an additional part of the challenge was to calculate the F1 score. Using the sklearn.metrics library I compared the provided results set with a set created by having the model made in step 5, generate a result set using the training data. This resulted in an F1 score of **0.5614035087719299**

-.) Plotting the Curves

Because this is my 1st machine learning project and I have yet to take a class on it, I was unsure how to go about picking the bounds of my feature selection process. After reading a couple of articles on the topic I found that there was no one correct way to do so. So logically I created a system to increment the top feature selection percentage and the correlation value. Exporting this data into an excel spreadsheet, then manually comparing the cells using some excel techniques. (I later automated this and included it in the final program, though the code utilizing it is commented out) I was able to visually see the learning curve (See below graphs) of the machine learning model. From this idea, I thought of the idea to have the model be able to calibrate itself, which ended up working pretty well if I do say so myself. Because the data from the calibration process matched up with what I saw on the curves I knew my values were valid.



Summary

In short, I built a machine learning model that is able to calibrate itself to a user's specified accuracy percentage, by excluding the lower half of the predetermined effective features and correlation values. This mode has the added benefit of never having to guess if 1, or 2 more features could have removed a large portion of errors, however, due to the linear nature of the calibration system the amount of time it takes is a heavy drawback. It is important to note that this is a one time cost if you choose to save the values it arrives at. Because the model I created has a F1 score > .5 mark (0.5614035087719299) I believe it is suitable for experimental use, however it may require some tweaking before it is ready for commercial use. With quickly

self-calibrating machine learning models we may be able to determine relevant data and subsequently arrive at solutions much faster than we would have otherwise. Overall I believe that the process I began here can be utilized to help improve machine learning models already in use.

Environment Configuration:

```
python==3.7.4(conda)
numpy==1.14.5
openpyxl==3.0.4
pandas==0.23.1
scikit-learn==0.22.1
```

Sources

"1.13. Feature Selection¶." *Scikit*, scikit-learn.org/stable/modules/feature_selection.html.

Brownlee, Jason. "How to Choose a Feature Selection Method For Machine Learning." *Machine Learning Mastery*, 30 June 2020, machinelearningmastery.com/feature-selection-with-real-and-categorical-data/.

Brownlee, Jason. "Ordinal and One-Hot Encodings for Categorical Data." *Machine Learning Mastery*, 29 June 2020, machinelearningmastery.com/one-hot-encoding-for-categorical-data/.

Detective, The Data. "A Look into Feature Importance in Logistic Regression Models." *Medium*, Towards Data Science, 14 Nov. 2019, towardsdatascience.com/a-look-into-feature-importance-in-logistic-regression-models-a4aa970f9b0f.

Malik, Usman. "Applying Filter Methods in Python for Feature Selection." *Stack Abuse*, Stack Abuse, stackabuse.com/applying-filter-methods-in-python-for-feature-selection/.

Sunil RayI am a Business Analytics and Intelligence professional with deep experience in the Indian Insurance industry. I have worked for various multi-national Insurance companies in last 7 years. "Regression Techniques in Machine Learning." *Analytics Vidhya*, 15 Apr. 2020, www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/.

Vickery, Rebecca. "Optimising a Machine Learning Model with the Confusion Matrix." *Medium*, Towards Data Science, 27 Sept. 2019, towardsdatascience.com/understanding-the-confusion-matrix-and-its-business-applications-c4e8aaf37f42.