"I hereby confirm that this submission is my work. I have cited above the origins of any parts of the submission that were taken from Websites, books, forums, blog posts, github repositories, etc."

1. **Dataset univariate understanding:**
    - # of data points: 146
    - POI: 18
    - NON-POI: 128
    - # Features: 21
    - Features with missing values: 'from_poi_to_this_person', 'from_this_person_to_poi', 'to_messages', 'from_messages'

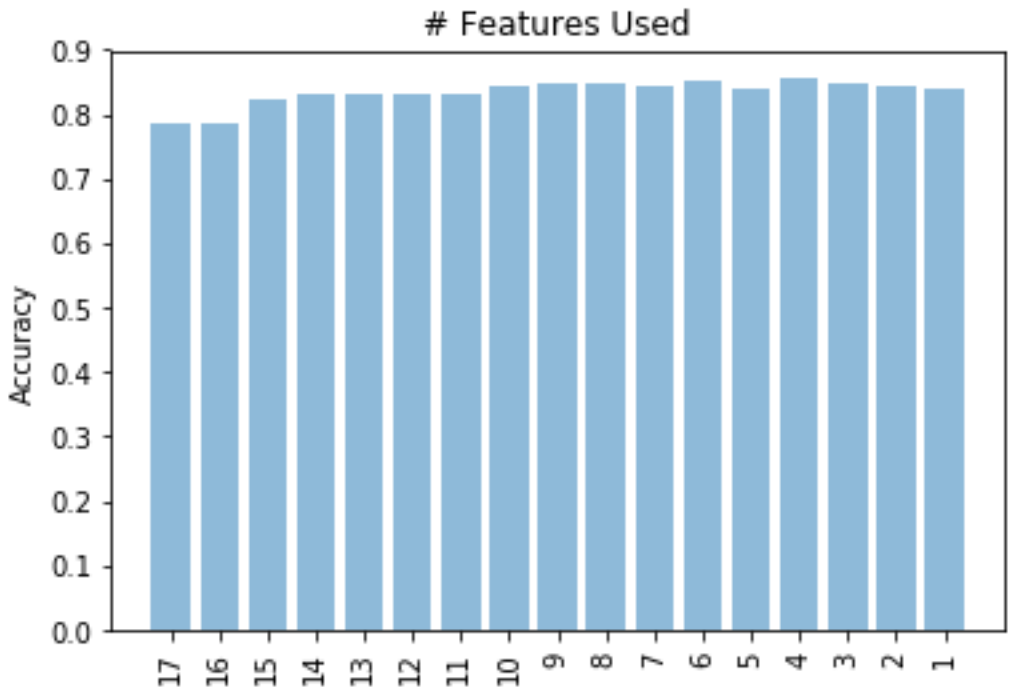
2. **Summarize for us the goal of this project and how machine learning is useful in trying to accomplish it. As part of your answer, give some background on the dataset and how it can be used to answer the project question. Were there any outliers in the data when you got it, and how did you handle those? [relevant rubric items: "data exploration", "outlier investigation"]**

   The goal from a superficial level was to determine if persons of interest could be determined/validated from an algorithmic perspective. I came to understand throughout the module & project that while machine learning can be used in applications of driving cars & predictive modeling, rather, it is an extension of the modeling and capabilities that a Data Scientist can use to better understand problems and what things may be most important. With regards to tackling the problem of 'identifying POI's' the use of machine learning is a helpful investigative tool that would further allow to determine someone's true innocence/guilt. The dataset, encompassing email & financial data allows a Data Scientist to examine some of the relationships that were derived from the legal process that followed Enron POI's and the accompanying data trail left.
   As it relates to outliers, the major one's corrected were a 'Totals' row and varying persons in the dataset that didn't have fully completed information.

2. **What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what feature you tried to make, and the rationale behind it. (You do not necessarily have to use it in the final analysis, only engineer and test it.) In your feature selection step, if you used an algorithm like a decision tree, please also give the feature importances of the features that you use, and if you used an automated feature selection function like SelectKBest, please report the feature scores and reasons for your choice of parameter values. [relevant rubric items: "create new features", "intelligently select features", "properly scale features"]**

   Features used: POI, Excercised_stock_options, total_stock_value, bonus, salary and fraction_to_poi.

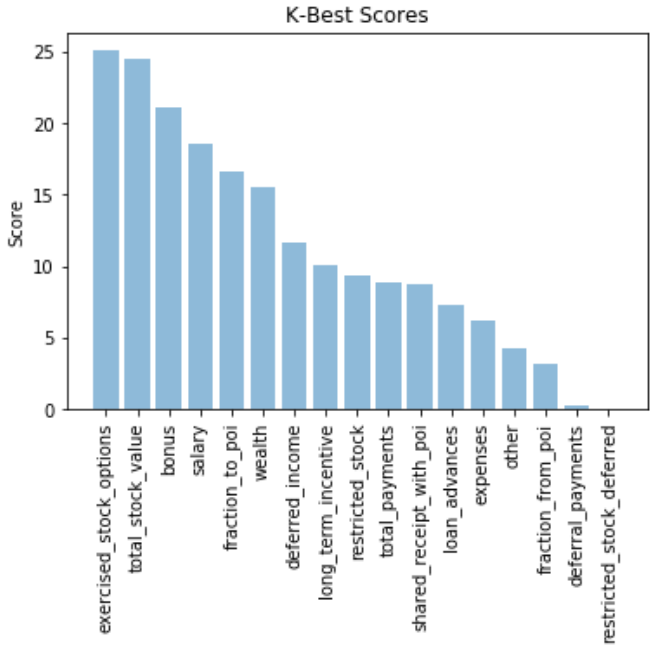
   Using K-Best selection. See figure below. I decidedly went with 5 features. Unfortunately, my created feature 'wealth' didn't make this selection.



After looping through all possible #'s of features it became evident that utilizing 4 features would be optimal.

A MinMaxScaler was used, reason being, there were many features that had data with great range and transforming into a scale from 0 to 1 made most sense.

Feature made was the 'wealth' feature. The idea behind this feature was to better summarize on the monetary gains received at Enron. While this feature was well intended in practice (and didn't receive much of a drop off from K-Best) this only negatively impacted the results by ~1% accuracy, ~8% precision and ~4% recall.



Results of SelectKBest by feature score:

- 'exercised_stock_options', 25.09
- 'total_stock_value', 24.46
- 'bonus', 21.06
- 'salary', 18.57
- 'fraction_to_poi', 16.64
- 'wealth', 15.55

3. What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms? [relevant rubric item: "pick an algorithm"]

The algorithm I choose was Guassian Naïve Bayes. I also evaluated both Adaboost and a Decision Tree Classfier. In terms of performance it resulted as so: Guassian, Adaboost and then Decision Tree.

4. What does it mean to tune the parameters of an algorithm, and what can happen if you don't do this well? How did you tune the parameters of your particular algorithm? What parameters did you tune? (Some algorithms do not have parameters that you need to tune -- if this is the case for the one you picked, identify and briefly explain how you would have done it for the model that was not your final choice or a different model that does utilize parameter tuning, e.g. a decision tree classifier). [relevant rubric items: "discuss parameter tuning", "tune the algorithm"]

What is meant by tuning the parameters of an algorithm is to optimize these values that allow for a machine learning algorithm to learn in the best way possible. If this can't be done well, the results of the algorithm may prove to be dissatisfactory for the intended model. As I used Gaussian there was no need for parameter tuning; however, in the event of using Adaboost this is how I went about tuning: One, of the most important parameters is the n_estimators, which is the # of iterations the algorithm goes through before terminating (can be less if perfect fit is achieved). Another important parameter is the learning rate, which is used in conjunction with n_estimators and shrinks the contribution of each classifier. A balance between these 2 determines how fast the algorithm is able to learn and converge on a model. From a high

perspective, this algorithm attempts to give proper weighting (pull) to the features integrated and re-running over the same data.

5. **What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis? [relevant rubric items: "discuss validation", "validation strategy"]**

   Validation is an attempting to discover the balance between testing data and learning data, where one can avoid overfitting. A classic mistake in validation is throwing all your data into the training set and running against some testing set. Because we cannot afford to reserve a fraction of the dataset for testing, I used a shuffle split validation (at 100 iterations) which preserves the ratio between classes. Preserving the ratio is important because of the possibility of poor distribution between classes.

6. **Give at least 2 evaluation metrics and your average performance for each of them. Explain an interpretation of your metrics that says something human-understandable about your algorithm's performance. [relevant rubric item: "usage of evaluation metrics"]**

Accuracy:	84%
Precision:	.44
Recall:	.358
F1	.394

   In summary, my algorithm performed at 84% accuracy. This is to say that, given any random person from the Enron dataset we could assume that 84% of the time they would be classified correctly as POI or not. In understanding precision and reclass, this starts with understanding the question: "Is this person a POI?". From a model perspective, how do the resulting positives (where model predicts person is POI) compare to that which truly belongs to the POI set. The ratio between these 2 is precision. Recall differs, in that, of persons that are truly POI, what ratio were correctly classified as POI? This is recall. The 2 terms are inversely proportional to each other.