Project Five
Machine Learning

1 - 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”]

This project involved examining financial and email data from a set of Enron employees. The goal of this project was to use the supplied data to create an algorithm capable of determining whether a given actor could be classified as a person of interest.

The dataset contained 146 data points, each with 21 features. These included 14 financial features, 6 email features, and one boolean feature indicating whether the data point referred to a poi. In the case of this feature, the dataset was heavily skewed toward non-POIs. Only 18 of the elements are POIs, which would have an impact on my validation procedures (addressed below). Additionally, I created two more email features.

In my analysis, I initially included ten features, and then used SelectKBest to narrow it down to the seven most relevant (‘salary', 'total_stock_value', 'bonus', 'total_payments', 'long_term_incentive', 'exercised_stock_options', ‘to_poi_rate’).

In my outlier investigation, I noticed that one row was merely an aggregation and removed it. I also noticed that some entries had no values entered for any of the variables I was investigating. I removed them as well.

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”, “properly scale features”, “intelligently select feature”]

I created two additional features for my investigation. These features concerned the ratio of emails sent to pois out of total emails sent, and the ratio of emails received from pois out of total emails received. My thinking for implementing these was that these was that they would be a better judge of involvement with other pois than just the raw number of poi emails sent and received respectively.

Once the features were created, I used the min/max scaler and then ran selectkbest to choose the seven most meaningful features. They were ‘salary', 'total_stock_value', 'bonus', 'total_payments', 'long_term_incentive', 'exercised_stock_options', ‘to_poi_rate’ The number of features was set at seven because that was deemed the optimal value during Gridsearch.

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”]

My final algorithm used DecisionTree. I also tried a KNearest Neighbors and a gaussian algorithm, and while both performed decently in Gridsearch when using f1 score as a metric, neither was able to achieve the level of precision I did with the DecisionTree Classifier (See below).

After initial testing of each algorithm, I had the following f1 scores for each:

Knearest: .262
Gaussian: .307
DecisionTree: .317

My basic workflow for the final algorithm was to run selectKbest to get the seven most important features, then scale those features with the minimal scaler. After that, I ran Kfold cross validation on a decision tree classifier with the following parameters: min_samples_split = 3, splitter = “random", criterion = 'entropy'

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? (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 item: “tune the algorithm”]

Tuning an algorithm is what makes it more or less appropriate for your data set. It can also greatly affect the bias and variance of your algorithm. Parameters are tuned by experimenting with different values for the parameters, or by using functions such as GridSearch to find the optimal values (In many cases, the best approach is to use both methods.

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 item: “validation strategy”]

Validation is the process of splitting data into sets, and using some data to train and some to test. It is important, because we need a way of seeing the effects of our algorithm on data separate from what it was trained on. Without this process, we would only be able to see the ability of the classifier to summarize the data used to train it.

A classic mistake is to judge the value of the algorithm based on the overall accuracy in cases where the data outcomes may be very imbalanced.

In this case, I had such a data set, and decided to validate it with a combination of accuracy, precision, and recall scores, to get a good idea of how effective it was overall.

Another common mistake is to only split the sets once, sometimes leading to sets that are very dissimilar or non-representative of the dataset as a whole. In these cases, your

validation process may not produce results representative of the algorithm's overall abilities.

I attempted to avoid this by implementing Kfold Cross Validation, which separated the dataset into several folds and trained & tested many times. This allowed me to gain a broader view of the overall algorithm quality. I also used the tester.py file which validated the algorithm after 15000 predictions.

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"]

On my testing data, my evaluations yielded the following metrics:

Accuracy Score - .828

This above probability reflects the rate at which data points in the testing set were labeled correctly.

Recall Score - .328

This reflects that POIs in the testing set were correctly identified 40% of the time.

Precision Score - .345

This indicates that when my algorithm predicted that a data point in the testing set represented a POI, it was correct two thirds of the time.

Several other versions of the algorithm yielded higher accuracy scores (sometimes higher than .90). However, since the goal of this analysis was specifically to identify POIs, I placed a higher premium on recall and precision scores. Using the tester.py file, this algorithm routinely provided scores near the above.