Mark Ciganovic
Project 6

Identify Fraud from Enron Email

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"]*

The goal of this project is predict who would be a person of interest (poi) from the both the financial and email data about this person. The dataset would constitute three types of information, which is the financial information about a person, the email information about a person, and whether or not the person is a person of interest (poi). It is clear as to what the independent variables are, and what the dependent variable is. The independent variables are the relevant financial and email information, and the dependent variable is whether or not the person is a "poi". For the original dataset, there was a total of 146 data points, which 18 were persons of interest and 128 were not. There were 21 features in the original dataset, which I was able to reduce to 6, with 2 of the final features being a mathematical relationship between 4 of the original features. For many of the features of the dataset, there were missing values. The way that I handled missing data was to place a "0" for missing financial data, and to use the median of columns for missing email data. Given that the total payments and total stock value required that missing values were "0" to add up correctly, it seemed necessary to use "0" for missing financial data. I found that I received both a higher precision and recall when I chose the median instead of "0" for missing data in the email features, so I used the median for email features. The goal of the predictive analysis was to find the best financial and email information about a person to determine whether or not, they were a "poi". Dealing with outliers were relatively easy, since the outliers were clearly conceptual outliers. Since the goal was to determine who was a person of interest, we needed to limit the data to persons. This is why I eliminated both "TOTAL", and "THE TRAVEL AGENCY IN THE PARK", since these were clearly not people, but a total and a sham account. Besides that, I thought it would be wise to include the rest of the data due to the small size of the dataset.

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"]

The features that I ended up using in my POI identifier were, "exercised_stock_options", "total_stock_value", "bonus", "shared_receipt_with_poi", "percent_emails_to_poi", and "percent_emails_from_poi". I used the selection process of using SelectKBest along with a Decision Tree through a Pipeline through a GridSearchCV to determine the best features, and then to test those features. Through a method of using insight and analysis, along with using feature selection algorithms, I found that I obtained good results when I combined the three best financial features selected by the Decision Tree and SelectKBest along with the three created email features. I then ran a Decision Tree through GridSearchCV to assess the six characteristics that I selected, I found that this did not help in selecting features, so I kept the six features that came about using a combination of the first selection process. I did not scale the data, since I used classifiers where feature scaling is unnecessary like Decision Trees, Logistic Regression, and a Naïve Bayesian Classifier. The new features that I created were "percent_emails_to_poi" and "percent_emails_from_poi". The reason that I did this is because I assessed that a percentage of emails that a person receives from a poi compared to the total that they received, and that a percentage they send to a poi compared to the total they send would be more informative than just total emails sent and received, and total emails sent and received to and from pois. These two new features show the relative intercommunication between pois compared to non-pois. As far as the financial feature weighting from the Decision Tree and SelectKBest, here are the results: exercised_stock_options at 25.10, total_stock_value at 24.47, and bonus at 21.06, and salary at 18.58. These were all the features listed by SelectKBest. The problem was that selecting only these features gave me both a precision and recall under .3, I decided through intuition and insight to add the two email features that I created, and the one created by Udacity to my dataset. I also decided to remove salary, since it was the lowest ranking financial feature, and by doing so, I would have an equal amount of both email and financial features, which were three each. I wanted a dataset that was both an equal combination of personal analysis and insight along with feature selection technology, and I believe I achieved that with the three financial features being selected through feature selection algorithms, and email features being selected through insight, analysis, and intuition. When I did this, I ended up getting a precision and recall both above .5. This was reason enough to justify using these features, which I then proceeded to analyze these features using a second Decision Tree. The second Decision Tree had these values: exercised_stock_options at .36, total_stock_value at .31, bonus at .25, shared_receipt_with_poi at .06, percent_emails_to_poi at .02, and percent_emails_from_poi at .0. When I eliminated percent_emails_from_poi in my dataset for analysis, both my precision and recall suffered, so I decided to keep it. Therefore these six features along with "poi" became my final dataset.

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 that I ended up choosing was a Decision Tree. I also tried a Naïve Bayesian Classifier, Logistic Regression, and a Random Forest, but a received the best results from a Decision Tree. It was only the Decision Tree that gave me both a recall and precision above a .3 on the first run, and so this is the algorithm that I decided to both tune, and use with the selection of the best features.

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"]*

To tune the parameters of an algorithm means to select the values for the different parameters which will give you a higher score when it comes to precision, recall, accuracy, or whatever metric you are using to score your algorithm. To tune my decision tree, I created a list of parameters for a decision tree including min_samples_split, range, criterion, and splitter. I then placed these parameters in a Grid Search which would select the parameters that gave the best result as far as precision and recall is concerned. My code printed out all of the values of these parameters for the Decision Tree.

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 the process by which you know that the scores you received from the algorithm are an accurate assessment of the rules of the algorithm. The classic mistake in validation is to depend heavily on chance, which means to randomly select one test set, and just one training set from the data, which means that the score heavily depends the which values have been randomly selected. This is a serious problem for the Enron data set considering its size, and number of pois. The validation was done through the test_classifier function in tester.py, which we were told by Udacity does all of the appropriate validation through sklearn.cross_validation.StratifiedShuffleSplit.

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"]*

The two evaluation metrics that I chose were the ones that were relevant for this project, which were both precision and recall. I obtained a precision of .52 and a recall of .54. The way to understand this as far as precision is concerned, that is the number of true positives divided by the sum of true positives and false positives. The true positives would be the number of

individuals that the algorithm correctly identified as a poi, and the false positives are the number of people that were identified as pois by the algorithm, which were not pois. What this means is that out of a hypothetical random sample of 100 people who are determined to be people of interests by the algorithm, only 52 are actually pois. As far as recall is concerned, that is the number of true positives divided by the sum of true positives and false negatives. Since the true positives were explained when articulating precision, the explanation has already been provided. The false negatives are the number of people not identified as pois that are actually pois. What this means is that out of a hypothetical random sample of 100 pois, the algorithm will only classify 54 as pois. I think ethically, the precision is a more important metric given this project, because an important principle is innocent until proven guilty, and if we only have a 52 percent chance of determining that someone is a poi from our algorithm, we need more evidence to conclude that they are a poi. Someone cannot be determined as guilty based upon what is close to a 1 in 2 chance, which is ultimately reducible to determining guilt on a coin flip.