ENRON POI IDENTIFIER REPORT

Introduction

As one of the largest corporate fraud cases in American history, the downfall of Enron is both fascinating and uniquely well-documented (thanks to the ensuing federal investigation). In this project, I use email and financial data for 146 executives at Enron to identify persons of interest in the fraud case. A person of interest (POI) is someone who was indicted for fraud, settled with the government, or testified in exchange for immunity. This report documents the machine learning techniques used in building a POI identifier.

The Enron Data

The first step of this project is examining the data for any outliers or obvious mistakes. There was one significant outlier that arose from a spreadsheet summary line that got transcribed into the dataset; this outlier was removed by hand before proceeding. Two more financial outliers (Ken Lay and Jeff Skilling) were left in because they are clearly real Enron characters, and the fact that they made so much money off the company is not noise or a mistake--it's tightly entangled with the corporate fraud.

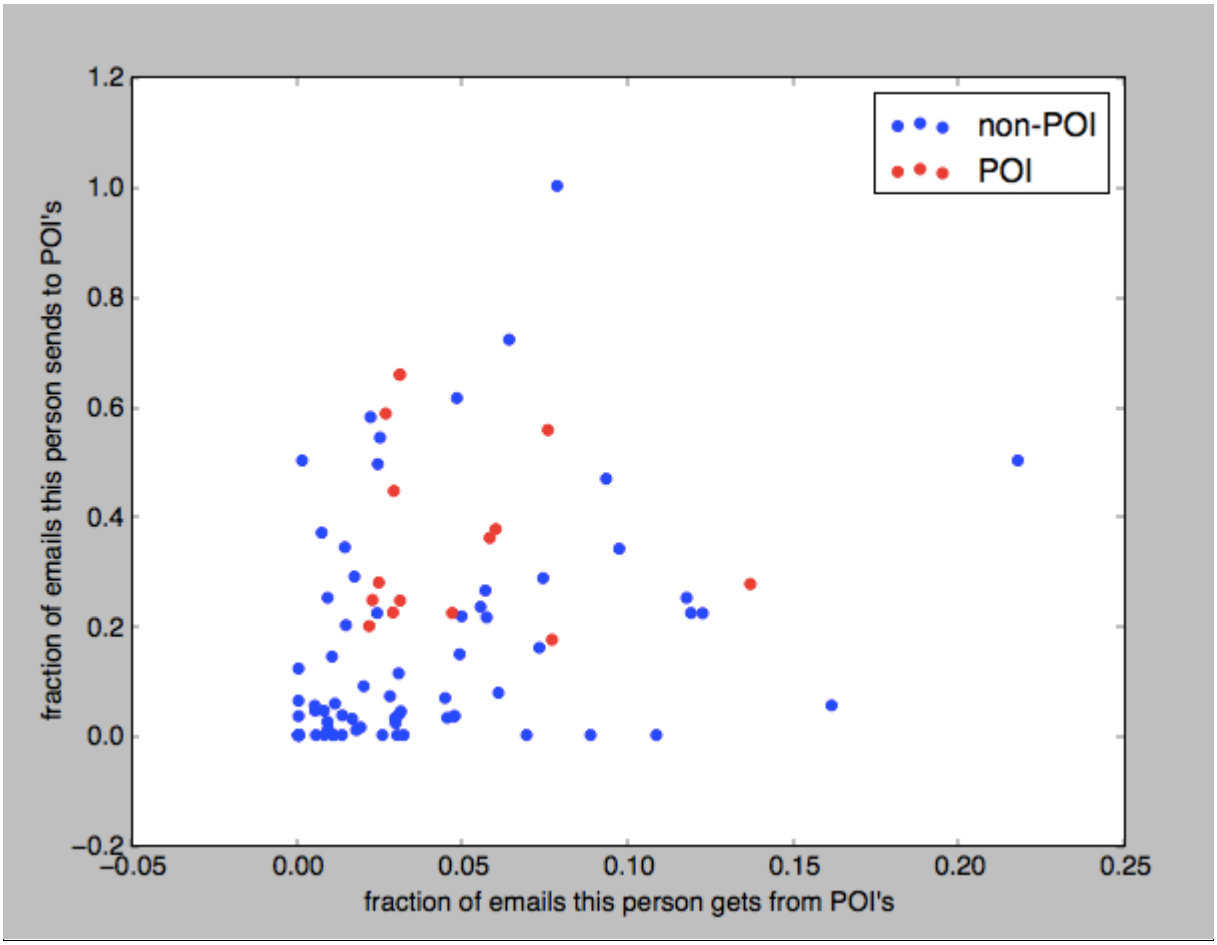
Total Person entries in the data set – 146
Total features for each person – 21
Count of Persons of Interest – 18
Count of people with salary entries - 95
Total salary paid for all the people combined - 53408458
Total salary paid only for the persons of interest - 6518563
Count of the people with bonus entries - 82
Total amount paid in terms of bonus for all the people combined -194687238
Total bonus paid only for the persons of interest - 33199999
Count of people who have email address - 111
Count of people who have value for total payments - 21
Count of people who have value for total stock value – 20
Total amount in terms of total stock value held by everyone - 853518639
Total amount in terms of total stock value held by only POI – 164982077
Percentage of people in the dataset having 'NaN' for their total payments – 14.4
Percentage of POIs in the dataset having 'NaN' for their total payments - 0

Feature Processing

Once the data was cleaned of outliers, the next step was selecting features to use. This was an iterative process, where a handful of features were selected for a first pass based on visualizations of the data that suggested that they might have some discriminatory power. These features were then used in the next steps of the process, selecting and evaluating a classification algorithm. Once a decision tree was selected as the algorithm for the POI identifier (further details on that process are given in the

next section), the features were revisited for optimization. No feature scaling was deployed, as it's not necessary when using a decision tree.

Two new features were created and tested for this project. These were



- the fraction of all emails to a person that were sent from a person of interest
- the fraction of all emails that a person sent that were addressed to persons of interest

The hypothesis behind these features was that there might be stronger email connections between POIs than between POIs and non-POIs, and a scatterplot of these two features suggests that there might be some truth to that hypothesis.

In order to find the most effective features for classification, feature selection was deployed to rank the features.

- The top 5 features were then selected for use in the final classifier.
- Trying both smaller and larger numbers of features negatively impacted the precision and recall.

ENRON POI IDENTIFIER REPORT

- The selected features were "fraction_to_poi", "bonus", "salary", "total_stock_value" and “exercised_stock_options”
- These were features selected using the SelectKBest feature selection process
- As can be seen from the list of features used in the classifier, out of the two new fractional email features only "fraction_to_poi" was used in the final version because it helped the precision and recall.

Algorithm Selection and Tuning:

Below three algorithms were tried for classifying the POI.

- Naïve Bayes
- Decision tree
- Support Vector Machine

Steps following for selecting the algorithm and tuning the parameters:

- Support Vector Machine was performing well in terms of accuracy but in terms other evaluation metrics it was performing very badly.
- Manually tried various combinations of the parameters for SVM but still nothing improved.
- Naïve Bayes was performing better than the decision tree in terms of metrics, so decided to tune the decision tree parameters to get the best possible values.
- Manually tried various combinations of the parameters for decision tree classifier and was finding it difficult to find which value was the giving the good scores.
- When it was difficult to find the best values for a set of parameters, decided to try the GridSearchCV
- Using the GridSearchCV selected the best possible values for the parameters like splitter, criterion, min_samples_split and min_samples_leaf
- Performance of the Decision tree improved and was looking slightly better than Naïve Bayes but the results were not consistent.
- Tired setting the random_state parameter for Decision Tree.
- Even though got the consistent results after setting random state parameter, performance was not as good as Naïve Bayes

7-fold Cross Validation was used along with the GridSearch for selecting the best performing algorithm.

ENRON POI IDENTIFIER REPORT

Below were the results for Naïve Bayes for 7-fold:

Fold	Precision	Recall	F1 Score	Accuracy
1	0.5	0.5	0.5	0.894736842
2	0	0	0	0.894736842
3	0.33	0.5	0.4	0.842105263
4	1	0.67	0.8	0.947368421
5	1	0.5	0.67	0.894736842
6	0.5	1	0.67	0.944444444
7	0.25	0.25	0.25	0.666666667
Mean	0.511428571	0.488571429	0.47	0.869256475

Below were the results for Decision Tree before using GridSearch:

Fold	Precision	Recall	F1 Score	Accuracy
1	0.166666667	0.5	0.25	0.684210526
2	0	0	0	0.789473684
3	0	0	0	0.842105263
4	1	0.666666667	0.8	0.947368421
5	1	0.5	0.666666667	0.894736842
6	0	0	0	0.944444444
7	0.5	0.25	0.333333333	0.777777778
mean	0.380952381	0.273809524	0.292857143	0.840016708

Below were the results for Decision Tree after using GridSearch without random_state parameter:

Fold	Precision	Recall	F1 Score	Accuracy
1	0.25	0.5	0.571428571	0.789473684
2	0	0	0	0.842105263
3	1	0.5	0.666666667	0.947368421
4	0.666666667	0.666666667	0.666666667	0.894736842
5	1	0.75	0.857142857	0.947368421
6	0	0	0	0.888888889
7	0.666666667	0.5	0.571428571	0.833333333
mean	0.511904762	0.416666667	0.476190476	0.877610693

Based on the above result even though Decision tree looked to be performing better than Naïve Bayes, the results were not consistent when tried running multiple times and often it was degrading performance.

ENRON POI IDENTIFIER REPORT

Tried Decision Tree classifier by setting the random_state parameter to 29 and below is the results:

Fold	Precision	Recall	F1 Score	Accuracy
1	0.33	0.5	0.4	0.842105263
2	0.33	0.5	0.4	0.842105263
3	0	0	0	0.789473684
4	0.4	0.66666667	0.5	0.789473684
5	1	0.5	0.66666667	0.894736842
6	0	0	0	0.888888889
7	0.5	0.5	0.5	0.777777778
mean	0.365714286	0.38095238	0.35238095	0.832080201

Results of the decision tree for random_state parameter setting was consistent but it was not performing as well as Naïve Bayes.

As per the test_classifier the results were

GaussianNB:
Accuracy: 0.84879 Precision: 0.45558 Recall: 0.30000
F1: 0.36177 F2: 0.32199
Total predictions: 14000 True positives: 600
False positives: 717 False negatives: 1400 True negatives: 11283

DecisionTreeClassifier:

Accuracy: 0.80500 Precision: 0.32636 Recall: 0.34300
F1: 0.33447 F2: 0.33954
Total predictions: 14000 True positives: 686
False positives: 1416 False negatives: 1314 True negatives: 10584

Decided to go with **Naïve Bayes** based on the precision, recall, f1 and accuracy scores.

Analysis Validation and Performance

This process was validated using 7-fold cross-validation. The precision and recall are somewhat different for each of the 7 folds, so these numbers were averaged and the average precision and recall were used as the final metric to quantify the performance of the algorithm.

As listed in Table above, the average precision was 0.51 and the average recall was 0.49

ENRON POI IDENTIFIER REPORT

Discussion and Conclusions

The precision can be interpreted as the likelihood that a person who is identified as a POI is actually a true POI; the fact that this is 0.51 means that using this identifier to flag POI's would result in 49% of the positive flags being false alarms. Recall measures how likely it is that, given that there's a POI in the test set, this identifier would flag him or her; 49% of the time it would catch that person, and 51% of the time it wouldn't.

While these numbers are reasonably good, certainly better than guessing or a very unoptimized classification process, it would still be challenging to rely exclusively on this POI identifier alone to flag POIs. If either precision or recall were close to 1.0 then it would be easier to interpret as either

(1) A very conservative algorithm that sometimes misses POI's but is rarely wrong when it does find one,

(2) A trigger-happy algorithm that rarely misses POI's but is prone to false alarms.

Biggest problem with the dataset:

- There were only 18 examples of POIs in the dataset.
- There were 35 people who were POIs in "real life", but for various reasons, half of those are not present in this dataset.
- Data set is not complete

Paths for Improvement:

- "More data is better than a finely-tuned algorithm" maxim of machine learning, having a larger dataset is likely the single biggest way to improve the performance of this analysis.
- Dig in more into the contents of the emails.
- Extract more data from the contents of the emails.
- Get into the details of the meetings and the telephonic calls between the people in the dataset
- Also try to get more data about the meetings of the people in the dataset with the people in powerful positions in the government and its agencies during the period of the fraud.

ENRON POI IDENTIFIER REPORT

References:

http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier

http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html#sklearn.feature_selection.SelectKBest

http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.GridSearchCV.html

http://stackoverflow.com/questions/5843817/programmatically-install-nltk-corpora-models-i-e-without-the-gui-downloader

http://machinelearningmastery.com/blog/

https://www.coursera.org/learn/machine-learning

https://weka.waikato.ac.nz/

http://stackoverflow.com/questions/21391429/classification-tree-in-sklearn-giving-inconsistent-answers

“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.”