Enron Fraud Detection Project

By Bhavin V. Choksi

Project Goal

In 2000, Enron was one of the largest companies in the US with revenues of over US$100 billion. However, due to widespread corporate fraud, it filed for bankruptcy at the end of 2001 - the largest in US history at the time.

Thirty four individuals, mainly Enron executives, were either indicted or convicted for committing fraud such as hiding debt and overstating revenue. The goal of this project is to use publicly available financial and email data of top Enron executives to build a machine learning alogorithm that can predict persons of interest (POI), i.e., individuals who were indicted, reached a settlement or plea deal, or testified in exchange for prosecution immunity.

Data Exploration

The financial data is composed of remuneration that each Enron insider received as payment or stock award. Payments are composed of items such as salary, bonus, expenses, etc. Stock awards comprise of information about stock options exercised, restricted stock, and so on. At an aggregate level, financial data is available for almost all individuals.

The email data includes information such as number of messages an individual sent or received in total, and how many of those were sent to or received from POIs. Such data is missing for over 40% of individuals. As a result, remuneration information may play a bigger role in predicting POIs than communication information. But, that will be confirmed by more formal selection methods.

Outliers

There were two individuals who received outsize pay as compared to other top executives, as shown by the two points that stand out on the plot below. One of them turned out to be Ken Lay – Enron's chairman, while the other was just a total of the numbers. The total was removed as it is not an individual.

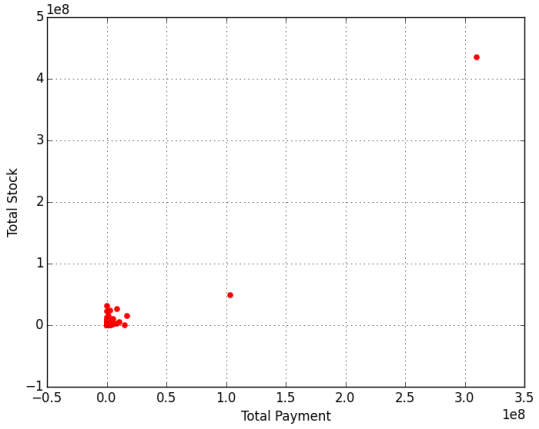

Fig. 1

Feature Scaling

As only two discrete outcomes are possible in predicting POIs, classification algorithms such as AdaBoost, Decision Tree and Random Forests were used.

Missing values were replaced by 0s. Possibly as a result, the classifier Support Vector Machines (SVM) did not work. The range of values for remuneration are quite large, scaling them on a range of 0 to 1 could have been useful. Unfortunately, feature scaling only works with SVM, not with the other classifiers that were used.

Feature Selection

Selecting features, i.e., elements of data, that can best make predictions in a classification algorithm required several steps:

1. SelectKBest was used including all features except a few such as email_address, from_messages, to_messages, because logically they cannot be factors in predicting POIs. The 3 best features were all remuneration related - bonus, exercised_stock_options and total_stock_value.

2. The first plot below (Fig. 2) and mean and median values of the 3 best features suggest that POIs exercised stock options and were awarded stock to a greater degree than non-POIs. The plot (Fig. 3) indicates that barring a few, all POIs received a bonus, while many non-POIs did not.

	POI Median	Non-POI Median
Bonus	$1,225,000	$100,000
Ex. Stock Options	$1,288,766	$591,250
Stock	$2,206,835	$877,611

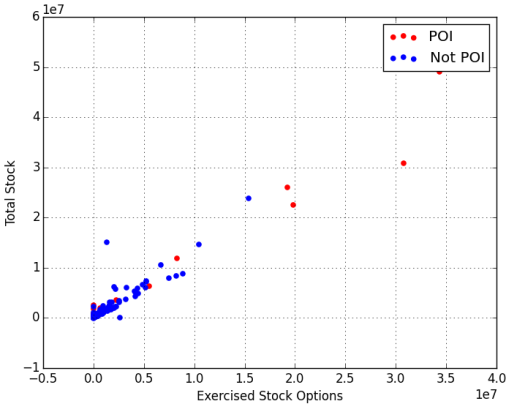

Fig. 2

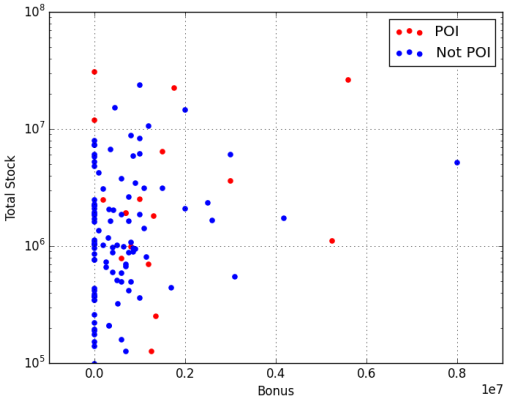

Fig. 3

3. Communication features such as messages sent to and received from POIs were plotted and summarized before ruling them out. The plot (Fig. 4) of total messages sent to and received from POIs did not indicate a clear trend. The average and median number of messages sent and received indicated that communication between POIs was more frequent.

	Avg.		Median	
	POI	Non-POI	POI	Non-POI
Messages From POI	76	33	52.5	0
Messages To POI	51	20	14.5	0

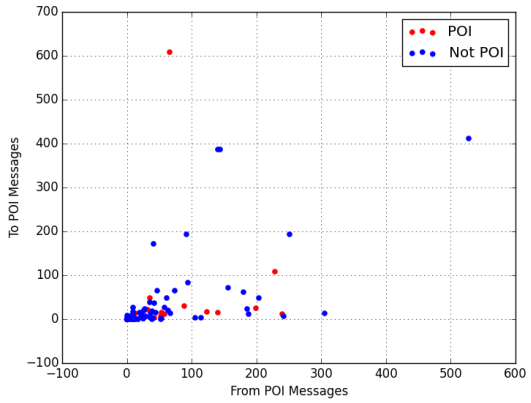

Fig. 4

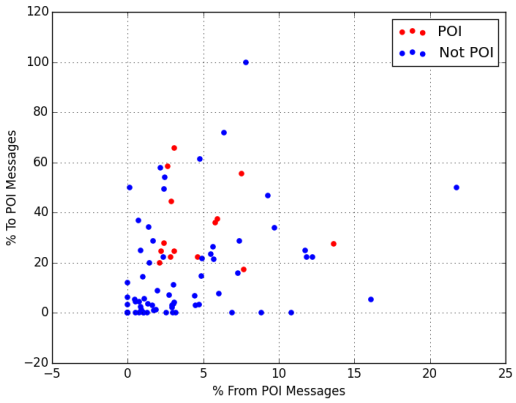

Fig. 5

Feature Creation

A new feature was created to test if the proportion of messages sent to and received from POIs could be important. The resulting median and plot (Fig. 5) indicated that POIs sent a far greater proportion (over 20%) of their messages to other POIs, while receiving messages from other POIs at a similar rate as non-POIs.

	POI Median	Non-POI Median
Messages From POI	3.06%	2.27%
Messages To POI	27.63%	5.37%

SelectKBest was used to pit the newly created feature - from_this_person_to_poi_percent - against the previously selected 3 best features. However, the new feature ranked last and scored very poorly against the other three.

Cross-Validating, Tuning and Evaluating Classifiers

In order to assess how well a prediction model will perform in practice, it would be a mistake to use all available data to train a model. It is important to hold back some data to test how accurately a model predicts outcomes with yet unseen information. Such a model validation technique is called cross-validation.

The implemented classifiers were cross-validated using the K-Folds iterator. The data was divided into twenty equal parts, rotating each part to test the model, while using the remaining 95% to train the model. The predictions were then compared to actual POI values to evaluate performance using precision and recall metrics.

A binary classification model's performance can be evaluated by it's precision and recall. Precision is the fraction of retrieved instances that are relevant, while recall is the fraction of relevant instances that are retrieved. In the context of this project, recall would measure the proportion of actual POIs that are identified. Precision would measure how many identified POIs are truly POIs. The goal is to achieve precision and recall of 0.3 or better.

At first, AdaBoost, Decision Tree and Random Forests classifiers were tested using the 3 best selected features. Decision Tree and Random Forests surpassed 0.3 precision and recall, while AdaBoost narrowly missed the mark for precision and recall.

Machine learning models take parameters so that their performance can be tuned to achieve optimal results. Finding the best combination of parameters is a search problem. GridSearchCV does an exhaustive search over specified parameter values for an estimator.

The Random Forests classifier builds multiple decision trees as specified by its n_estimators parameter, which was set to 5. The n_estimators parameter was tuned using GridSearchCV by providing options for 5, 10 and 15 decision trees. The best parameter value varied with each run – either 5 or 10. With tuning, precision degraded slightly while recall improved.

	Precision	Recall
AdaBoost	0.25	0.28
Decision Tree	0.35	0.39
Random Forest	0.64	0.39
Random Forest w/GridSearchCV	0.62	0.44

Final Thoughts

If casting a wider net to identify all possible POIs is the goal, then Decision Tree may be the better option due to its better recall rate observed over multiple runs of each classifier.

However, owing to its precision rate being far better than other classifiers, Random Forests may be the way to go to avoid casting doubt on people who may not be involved in fraud, and to ensure effort is not wasted investigating them.