

Udacity Data Analyst Nanodegree

Machine Learning Project

Identifying Enron's POIs' (Person of Interest)

by Mustafa Bilal Tahir

Introduction

The Enron scandal shocked the world in 2001 and has gone down in history as one of the most glaring examples of corporate greed and lax regulatory oversight. It has led to new laws coming into play and a new way of thinking about assessing a company's financial health. The funny thing about Enron was that they were never technically hiding anything. Most of the evidence collected against them came from documents *they* disclosed. The problem was that they buried all the critical information beneath layer after layer of complex regulations and accounting garble. Malcolm Gladwell captures this duplicity really well in one of his pieces titled "Open Secrets - The mystery of Enron"¹. Consequently, one of the few good things to come of this tragic affair where thousands of employees lost their life's savings was that the Enron scandal was heavily studied and its data very well documented by the investigators who looked into it. Our objective is to use Machine Learning to see if, looking at the dataset, we could have predicted if an employee in the database was a Person of Interest (POI). A POI is defined as an employee who was indicted for fraud in the forthcoming investigation or was at least called on to testify.

The Dataset

The Enron dataset has the following fields for each employee:

```
{'salary', 'to_messages', 'deferral_payments', 'total_payments',  
'exercised_stock_options', 'bonus', 'restricted_stock', 'shared_receipt_with_poi',  
'restricted_stock_deferred', 'total_stock_value', 'expenses', 'loan_advances',  
'from_messages', 'other', 'from_this_person_to_poi', 'poi',  
'director_fees', 'deferred_income', 'long_term_incentive', 'email_address',  
'from_poi_to_this_person':}
```

The 'poi' field is the indicator which identifies the people who were POIs. There were 146 data points or employees in total in the dataset. Of these, only 18 were POIs. Looking at these numbers off the bat we can tell that our Machine Learning algorithm has a strong chance of being biased because the majority of the people in the data are not POIs.

Other differentiation points that can lead to potential issues include, for example, the values for the total payments field. The percentage of the population that has an "NaN" value (No value) for their total payments is 14.4%. However, if you just look at the POIs there is no one who has this value. This could be due to the way the data points were added (Non POIs could be added from a spreadsheet while the POIs added manually and with more care to have all of their information). We should be careful in how our Machine Learning Algorithm interprets this. For example if we later on added more people to the dataset by hand and also added in their total

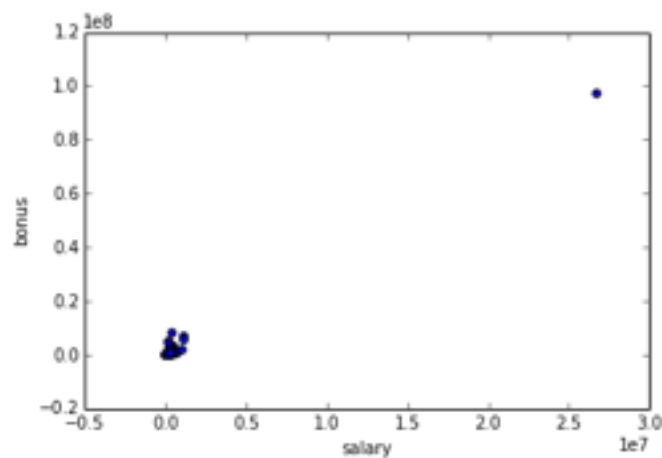
¹ <http://www.newyorker.com/magazine/2007/01/08/open-secrets-3>

payments information, our algorithm could falsely predict someone is a POI based on this information even though it is unlikely they are considering how small that population is compared to the whole.

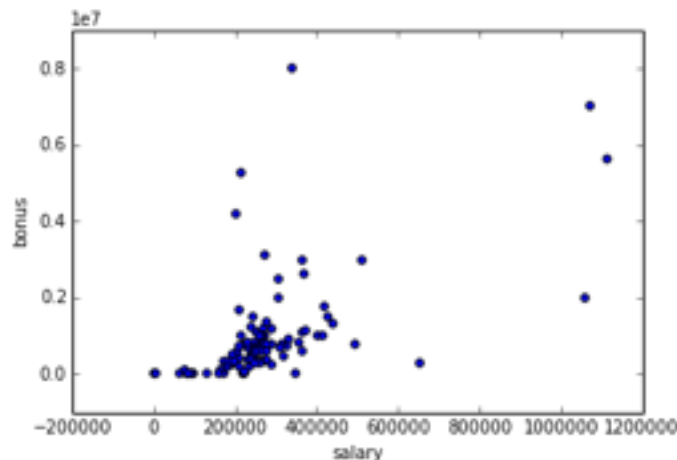
Outliers

Plotting the data using the Salary and bonus fields gives us a good idea of the spread and allows us to identify an outlier immediately. considering just how far away it is, we were able to conclude that it was actually the 'Total' field mistakenly left in. We removed this data point but kept other data points which appeared to be outliers because of their high bonus and salary (Kenneth Lay and Jeff Skilling) which were legitimate data points.

Data Plot with Outlier



Data Plot without Outlier



Using a

As we can see the data looks much better in the second plot once the outlier is removed.

Features Selection

The features were all ranked for importance using a Decision Tree Classifier to fit the model (the selection of which is explained below). Here is the list:

```
('from_messages', 0.3715676466061259)
('other', 0.23924335886780096)
('total_stock_value', 0.18186401613367895)
('director_fees', 0.096994141937962117)
('poi', 0.063046192259675379)
('shared_receipt_with_poi', 0.047284644194756538)
('salary', 0.0)
('to_messages', 0.0)
('deferral_payments', 0.0)
('total_payments', 0.0)
('exercised_stock_options', 0.0)
('bonus', 0.0)
('restricted_stock', 0.0)
('restricted_stock_deferred', 0.0)
('expenses', 0.0)
('loan_advances', 0.0)
('from_this_person_to_poi', 0.0)
('deferred_income', 0.0)
('long_term_incentive', 0.0)
('from_poi_to_this_person', 0.0)
```

Using this, we can eliminate the all the features which have a 0 value. The poi is the label so we have to keep that. We also replace 'shared_receipt_with_poi' with 'expenses' based on trial and error as the fit is much better without the inclusion of this feature.

Two new features were created and tested as well:

1. The fraction of all emails to a person that were sent from a person of interest
2. The fraction of all emails that a person sent that were addressed to persons of interest

The logic for using these metrics was that employees with a larger proportion of their email correspondence to and from with a POI would themselves likely be a POI as well. Reading in these new features and re-testing, our list of features that had a non-zero importance becomes:

```
('from_poi_to_this_person_frac', 0.22412918115876462)
('from_this_person_to_poi_frac', 0.22412918115876462)
('total_stock_value', 0.2030169268933314)
('restricted_stock_deferred', 0.15131086142322089)
('from_messages', 0.12207361729122496)
('salary', 0.10807918673087208)
('shared_receipt_with_poi', 0.063046192259675379)
('bonus', 0.047284644194756538)
('other', 0.047284644194756538)
('total_payments', 0.033774745853397542)
```

The two new features show up in our list of most important features so it does appear that these features add value to the algorithm. We did not do any feature scaling as it was not necessary with our algorithm (see below).

Model Selection

We compared 3 models initially without any parameter tuning: Gaussian Naive Bayes, Decision Tree, Adaboost (SVM was not included because it was taking too long to train the data). The Decision Tree and Adaboost handily won out against the Naive Bayes model and the decision Tree classifier had the highest precision and recall and was therefore ultimately selected.

The precision and recall were further improved after tuning the parameters (particularly the criterion parameter which was switched to entropy from the default gini method). GridSearchCV was used to tune the parameters of the classifier but the recommended settings seemed to optimize precision over recall and so ultimately the parameters chosen were further refined using trial and error.

The results themselves were validated using the Stratified ShuffleSplit cross validation iterator from sklearn. This cross validator is a mix of Kfold cross validation with stratified random folds so the data is not being cut in one fashion only. Our final average metrics were:

R-Squared Fit: 0.81040

Precision: 0.39641

Recall: 0.37500

Conclusion

Although the dataset was challenging (a small group of POIs in a large population of Non-POIs), we were able to get some decent performance out of our algorithm. We could potentially make further improvements by looking at the data in a more detailed fashion and trying to come up with more features.