Enron Fraud Detectors using Enron Emails and Financial Data

By: Rakpong Kittinaradorn

Date: 11 October 2016

Note: This report is in the form of questions and answers.

---

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?

This project is about Enron Fraud, one of the biggest financial crime in the American history. Enron Corporation was an American large energy company. It engaged in trading varieties of asset such as gas, electricity, and broadband services. Fortune Magazine rated this company to be the most innovative company many times during its prosperity. As the company expanded and its stock skyrocketed, it cannot maintain its financial stability. The company conducted sophisticated accounting to hide its financial incompetence. There are many people involved in committing this crime. The goal of this project is to look at financial and email data of Enron employee and use machine learning to identify person of interest (poi) who is involved in committing this crime.

This dataset have 146 data points in total. It comes with 14 readymade financial features and 5 readymade email features. With 19 total features, this dataset is very small. Another challenge of this dataset is its highly imbalance nature. Number of POI and non-POI are shown in the below table.

	POI	non-POI	Total
Number of data points	18	128	146

Because of the imbalance data, I will use precision and recall as evaluation metrics. Moreover there are missing values in the dataset. Table below shows the number of missing values for each feature.

Feature	Number of Missing values
loan_advances	142
director_fees	129
restricted_stock_deferred	128
deferral_payments	107
deferred_income	97
long_term_incentive	80
bonus	64
shared_receipt_with_poi	60
from_messages	60

from_poi_to_this_person	60
to_messages	60
from_this_person_to_poi	60
other	53
expenses	51
salary	51
exercised_stock_options	44
restricted_stock	36
total_payments	21
total_stock_value	20

The data have three outliers needed to take care with.

1. Total: This is the sum of all entries in the dataset.
2. The travel agency in the park: this entry is also not a person.
3. Lockhart Eugene E: this entry does not have any information apart from being a person of interest.

I removed these three entries as a preprocessing.

---

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 readymade 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.

Feature Score and Engineering

I started with all readymade features available. For financial features, I first check feature score from SelectKBest algorithm which measure the correlation between each feature and target label (poi). The result is in the table below.

Feature	Score
exercised_stock_options	24.815080
total_stock_value	24.182899
bonus	20.792252
salary	18.289684
deferred_income	11.458477
long_term_incentive	9.922186
restricted_stock	9.212811
total_payments	8.772778
loan_advances	7.184056
expenses	6.094173
other	4.187478
director_fees	2.126328
deferral_payments	0.224611
restricted_stock_deferred	0.065500

In addition to readymade feature I engineered one new feature.

1. ratio_bonus_salary = bonus / salary

This is included because typical employee should have similar bonus:salary ratio. But person involved in crime may have different ratio. Feature score for this new feature is 10.783585.

For email-related features, 5 readymade features are scored by SelectKBest.

Feature	Score
shared_receipt_with_poi	8.589421
from_poi_to_this_person	5.243450
from_this_person_to_poi	2.382612
to_messages	1.646341
from_messages	0.169701

I have engineered three new features as follow.

1. ratio_poi_message = shared_receipt_with_poi / (to_messages + from_messages)
2. ratio_to_poi = from_this_person_to_poi / to_messages
3. ratio_from_poi = from_poi_to_this_person / from_messages

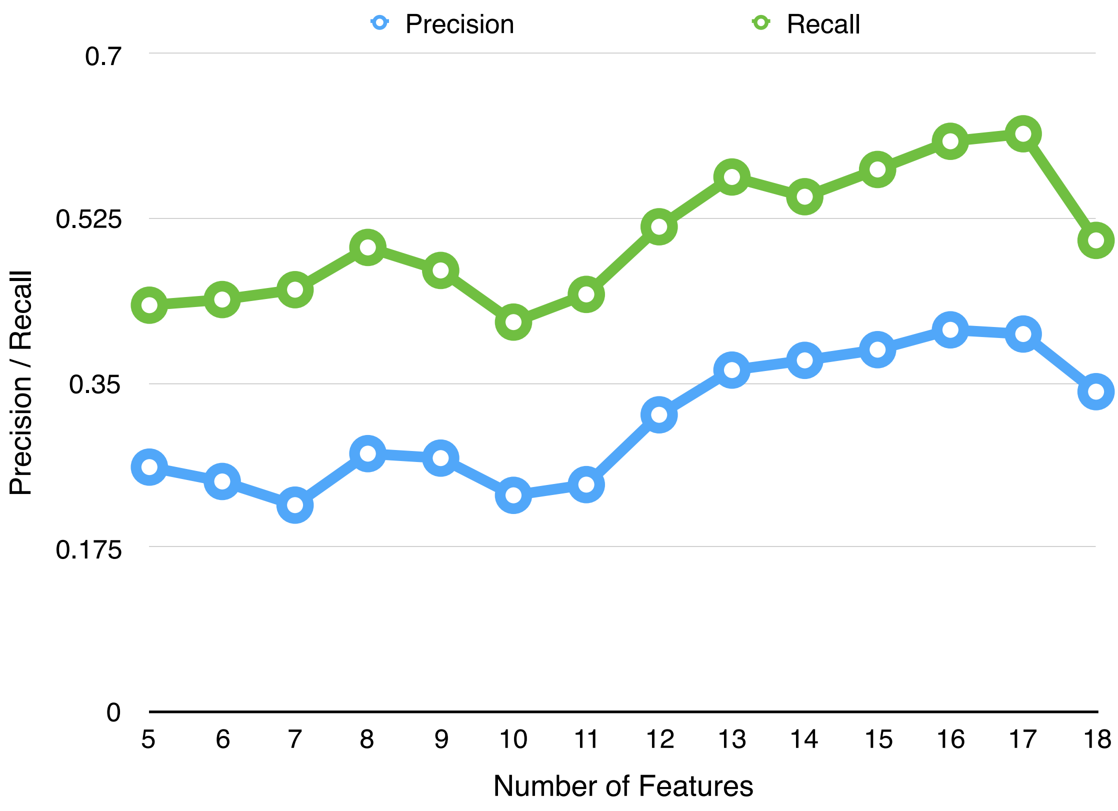
The rationale behind this came from the intuition that person of interest should have high proportion of email contacting each other. These three new features score well in SelectKBest algorithm.

Feature	Score
ratio_poi_message	9.736968
ratio_to_poi	5.123946
ratio_from_poi	4.094653

Since these three new features score better than the readymade email features and they contain similar information, I will use only the new features. Note that total number of features is now 18.

Feature Selection

In this section, we select features by using score from SelectKBest algorithm. I test the performance of different set of features by evaluating precision and recall using final classifier and parameter in this report including all processes below this section. The result is shown in graph below. The best number of features is 17.



I also compare the effect of adding new ratio_bonus_salary feature, without this feature the best precision and recall are 0.37903 and 0.56400, respectively. With ratio_bonus_salary, the best precision and recall are 0.40229 and 0.6155, respectively. So it is good to add this new feature into further calculation.

Scaling and Feature Reduction

After this I created squared and logarithmic version of all selected features in order to capture nonlinear behaviour. Next step is to reduce the number of feature with PCA. In order to use PCA, I performed standard scaling. Without scaling, PCA will be biased toward features with high value. After scaling, I did GridSearchCV and found the optimal number of feature to be 15.

---

3. What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms?

I have tried 7 classifiers namely, Gaussian Naive Bayes, Logistic Regression, Decision Tree, Random Forest, Adaboost, K-nearest neighbour and Support Vector Machine. I use recall and precision as my criteria in choosing algorithm. Some give high recall and low precision while the others give low recall and high precision. The following table shows that performance of different models before feature engineering and parameter tuning.

Classifier	Precision	Recall	F1
Naive Bayes	0.255	0.428	0.320
Logistic Regression	0.163	0.175	0.172
Decision Tree	0.214	0.197	0.205
Random Forest	0.390	0.126	0.190
Adaboost (Tree)	0.210	0.192	0.200
Support Vector Machine	0.202	0.340	0.254
K nearest neighbor	0.655	0.199	0.305

I have tried tuning all classifier using GridSearchCV, I found Logistic Regression to be the best classifier.

---

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).

Most algorithms have many parameters needed to be tuned to the dataset. Without tuning, I never achieve the goal of getting both recall and precision higher than 0.3. Adjusting parameters is the way to make the algorithm learn the data efficiently. The most important parameter in this case is ‘class_weight’ because the dataset is highly imbalanced. Setting class_weight to ‘balanced’ help improve performance in most of the algorithm. Another important parameter is a regularisation factor “C”, this help balancing bias and overfit. I used GridSearchCV to help finding the best set of parameters for my algorithm.

---

5. What is validation, and what’s a classic mistake you can make if you do it wrong? How did you validate your analysis?

Validation is the way to measure performance of the algorithm. The classic mistake is when the model cannot generalise to data outside the training set. This is called overfitting. I validate my model by separating training and testing set. The Classifier is trained on the training set and measure the performance using testing set. Cross-validation and StratifiedShuffleSplit are used in my analysis. The StratifiedShuffleSplit is used because the data is imbalance.

---

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.

I use recall and precision as my metrics. Recall is the ratio between number of positive label model predict correctly versus total number of positive label in the dataset. In this case it is number of poi my model predict correctly versus total number of poi. Precision is the ratio between number of positive label model predict correctly versus total number label my model guess to be positive.

In this investigation, I am using machine learning as a preliminary poi identifier. The poi predicted by this algorithm have to be investigated further by human intelligence. With this goal in mind, I design the model to capture most of the poi available in the data. That is creating the model with as high recall as possible as long as precision is not too bad (> 0.3). Average recall and precision of my model is 0.61550 and 0.40229, respectively.