Identify Fraud from Enron Email

--- Hang Zhu

Q: 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?

Project Background

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, a significant amount of typically confidential information entered into the public record, including tens of thousands of emails and detailed financial data for top executives.

In this project, the goal is to build a classifier to predict if a person is Person of Interest in the fraud case. Machine Learning could be help to accomplish the goal since it can learn the pattern from training data given and with the pattern it would predict Person of Interest. Features are required for Machine Learning. In this project, features involve Enron email information and financial data. I used all 23 features at the beginning as exploration.

Data Exploration:

Total data points: 146

Poi: 18

Non-poi: 127

Number of features(final): 19

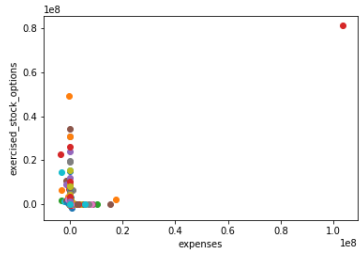
Features with most missing values:

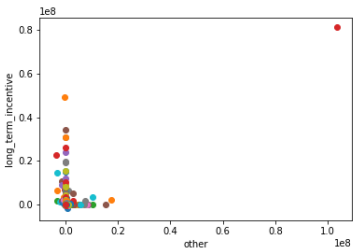
loan_advances(142), director_fees(129),

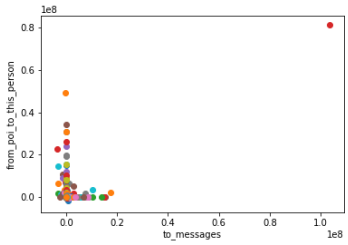
restricted_stock_deferred(128), deferral_payments(108), deferred_income(97)

Check outliers.

I plotted some pairs of features in a scatter plot and found out that there is always a point which is far away from other points in each plot. Thus, this point should be regarded as outlier. After checking given data description and data, I found out it is 'Total'. This point was removed. After removing this point, other points are pretty normal even if there are a couple of points with high value.







I checked the number of NaN value for each person and there is one person without non-NaN value(LOCKHART EUGENE E), so I also removed this point.

Q: 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.

Feature Engineering:

There are 5 features which include at least 60% NaN values, so I removed them. Another feature 'email' is string feature, so I also remove it.

Then I used all remaining features to start my feature selection:

'salary', 'total_payments', 'bonus', 'restricted_stock_deferred', 'total_stock_value', 'expenses' , 'restricted_stock', 'exercised_stock_options', 'other', 'long_term_incentive', 'to_messages', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi'

Feature scaling:

I didn`t scale the features because scaling would mask the information from the features.

Features Selection:

1. Feature scores from SelectKBest (f_classif) for selection.

('bonus', 0.017226936204361963), ('exercised_stock_options', 556.77730806873853),
('expenses', 16.643636), ('from_messages', 253.7096405), ('from_poi_to_this_person', 73.48004838395714),
('from_this_person_to_poi', 22.088154496033482), ('long_term_incentive', 21.379903333455612),
('other', 52.56178797), ('restricted_stock', 1.3179737674, ('restricted_stock_deferred', 2.47656886),
('salary', 24.1767139733), ('to_messages', 43.56982723), ('total_payments', 772.433411), ('total_stock_value', 1.712424636015998)

4 features have scores less than 10: 'bonus', 'restricted_stock', 'restricted_stock_deferred' and 'total_stock_value'. Therefore, I will select all features except the four.

2. Feature importance from Decision Tree for feature selection. Feature importance is below.

('bonus', 0.023814773980154354), ('exercised_stock_options', 0.22173212435424228),
('expenses', 0.10958968225136767), ('from_messages', 0.036515986769570012),
('from_poi_to_this_person', 0.050275633958103645), ('from_this_person_to_poi', 0.031753031973539139),
('long_term_incentive', 0.043849425106315953), ('other', 0.056097023153252483),
('restricted_stock', 0.08732083792723265), ('restricted_stock_deferred', 0.072675561955763551),
('salary', 0.039691289966923927), ('to_messages', 0.080325333464903906),
('total_payments', 0.074914973198167409), ('total_stock_value', 0.071444321940463074)

Select features with importance over 0.03, then 'bonus' is removed and remaining features are selected.

Performance comparison

I used Logistic Regression as model and run cross validation with performance as precision/recall.

Precision: the proportion of people who are predicted as POI correctly in all people predicted as POI.

Recall: the proportion of people who are predicted as POI correctly in all POIs actually.

Features	Precision(1)	Recall(1)
Original	0.20000000000000004	0.22222222222222221
Select K best(11)	0.52380952380952384	0.27777777777777773
Selected from Decision Tree	0.48148148148148145	0.27777777777777773

Therefore, I chose features from SelectKBest(11)
'salary', 'total_payments', 'expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'to_messages', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi'

Adding Features

After feature selection, I got 11 features from original feature set. Then I would try to add some new features
I created 3 features: 'fraction_stock_incentive', 'ratio_salary_exercised_stock_options', 'ratio_salary_total_payments'

'fraction_stock_incentive': the division of 'exercised_stock_options' and 'long_term_incentive'.
This feature gives the ratio between excercised stock and long term incentive. Both of them measure the treasure except salary and bonus and both of them would be relatively large, so I would like to see the ratio

'ratio_salary_exercised_stock_options': the division of 'exercised_stock_options' and 'salary'.
'ratio_salary_total_payments': the division of ' total_payments ' and 'salary'.
For some POI, even if they have not high salary, they still have very high exercised_stock_options or total_payments

Performance comparison

Conducted 3-fold cross validation with precision/recall on the data.

	Precision	Recall
Select K best(11)	0.52380952380952384	0.27777777777777773
Add 'fraction_stock_incentive'	0.547619047619	0.333333333333
Add 'ratio_salary_exercised_stock_options'	0.436507936508	0.166666666667
Add 'ratio_salary_total_payments'	0.47619047619	0.222222222222

After comparing the performance, I decided to add only 'fraction_stock_incentive' in my final feature set.

Final Features used:

'salary', 'total_payments', 'expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'to_messages', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi', 'fraction_stock_incentive'

Q: 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 vitally important. I used K fold cross validation for feature selection and parameter tuning. It is to split data into K folds and in the validation process, it would put the Kth fold as test data with others training data. Finally, it will give K training test processes and accuracies, averaging the accuracies gives the final validation result. On the other hand, I split data into training and test data, after selecting the best set of parameters using cross validation on training data, I compared the best performances of all algorithms on the test data.

Through validation, we can also know if our model is overfitting by compare performance on validation data and training data. If I didn`t do validation and used performance on training data as criteria to select features or models, then we can`t handle overfitting problem since the training accuracy would much higher than final test performance and the comparison would be hard.

Q: 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?

Algorithms and parameter tuning

Parameter tuning is to manage to find a set of parameters for the algorithm in order to achieve the highest performance. The purpose may not be achieved if tuning process is misconducted. For some complex algorithm with many parameters like random forest, grid search would be very slow if tuning many parameters with multiple values. On the other hand, optimal performance of the algorithm would not be achieved if tuning too few parameters or using too few values.

I used multiple algorithms and conducted grid search for parameter tuning so that I can get a set of parameters which could make the performance best for this algorithm on the split test data set. I used F1 score as criteria for parameter tuning and precision/recall as final performance measure.

Logistic Regression
Parameters:
'C': [5, 10, 20, 40 , 80, 160, 320, 5000, 1000, 5000],
'penalty': ['l1', 'l2']
Best parameter: {'penalty': 'l1', 'C': 160}
Performance on test data: Precision: 1.00 Recall: 0.20

Decision Tree
Parameters:
'max_depth': [5, 8, 10, 12, 14, 15],
'min_samples_split': [2,4, 6, 8],
'min_samples_leaf': [1, 2]
Best parameter: {'min_samples_split': 2, 'max_depth': 8, 'min_samples_leaf': 1}
Performance on test data: Precision: 0.67 Recall: 0.40

Naïve Bays
Performance on test data: Precision: 0.50 Recall: 0.20

Random Forest
'n_estimators': [100],
'max_depth': [5, 10],
'min_samples_split': [2,4, 6],
'min_samples_leaf': [1, 2, 4],
'n_jobs': [-1]
Best parameters:
{'min_samples_split': 6, 'n_estimators': 100, 'n_jobs': -1, 'max_depth': 10, 'min_samples_leaf': 1}
Performance on test data: Precision: 1.00 Recall: 0.20

For comparison, I use precision/recall as performance measure

Algorithm	Precision(1)	Recall(1)
Logistic Regression	1.00	0.20
Decision Tree	0.67	0.40

Naïve Bayes	0.50	0.20
Random Forest	1.00	0.20

After I used the four algorithms and tuned the parameters to get the best performance, I would choose Logistic Regression as my final model. 1 Precision means that 100% of people are actually POI among all people who are predicted as POI. 0.2 Recall means that 20% of people are predicted as POI among all people who are actually POI.