Project: Identify Fraud From Enron by Yiyi Tang

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? [relevant rubric items: "data exploration", "outlier investigation"]*

- Goal

Based on Enron's financial and email data, the project aims to select and tune a machine learning algorithm based on the predictor features that can better predict whether a given person is a person of interest (POI), which means individuals who were indicted, reached a settlement or plea deal with the government, or testified in exchange for prosecution immunity.

- How machine learning is useful

Machine learning will be very useful toward this project. Because after cleaning the dataset, machine learning can help me select several highly related features using feature selection according to the goal of the project. Also, it allows me to try different algorithms on my data, and to test which one give the best prediction for identifying whether a given person is the person of interest.

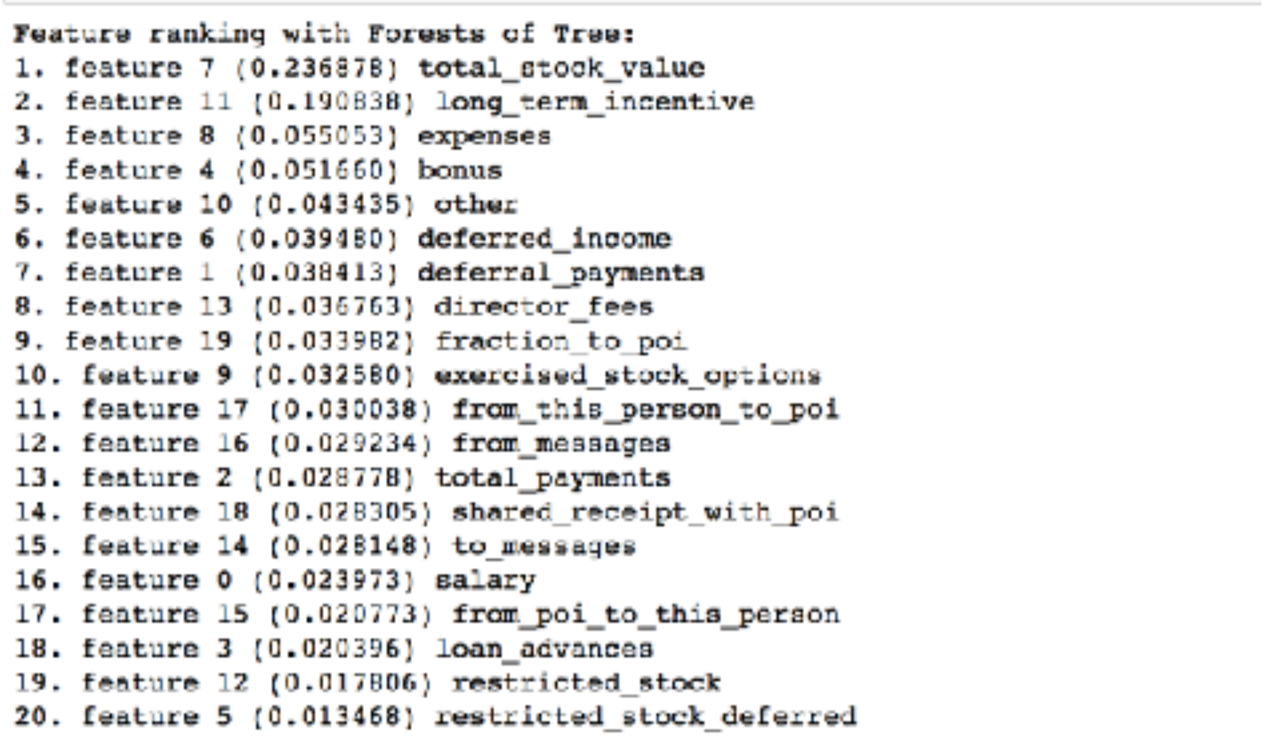
- Background of dataset

The data contains 146 Enron employees' financial and email information, including 21 features. All features are numeric, except for "email_address" (text string type) and "poi" (boolean, represented as interger). Also, the number of POIs is 18, and the number of non-POIs is 128 .

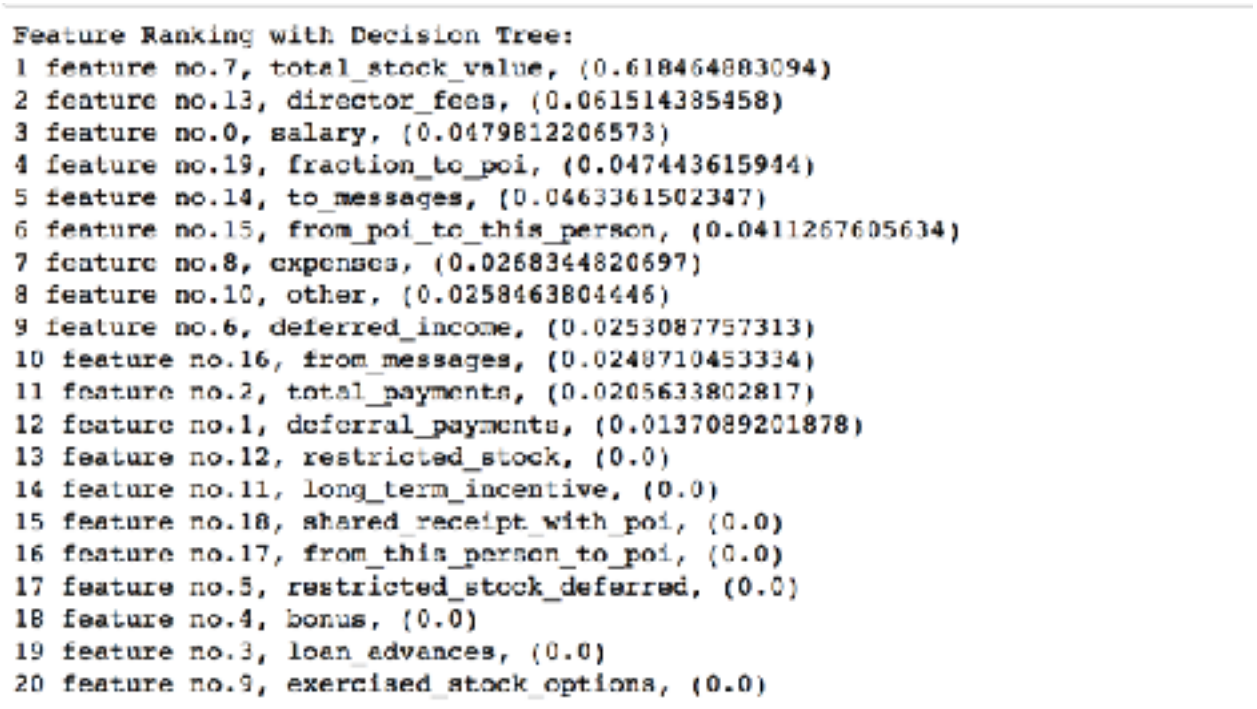
- Outliers

About outliers, when I plot scatter plot of "salary" and "bonus" variables, I found one outlier "TOTAL". It is not employee name, but a sub total in the original data. Also, I found two suspect persons when manually looking at the original data. One is THE TRAVEL AGENCY IN THE PARK. The other is LOCKHART EUGENE E, whose features' values are all "NaN". Both of them are non POI, and is lack of values, so I decide to remove these two outliers too.

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 ready-made 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. [relevant rubric items: "create new features", "intelligently select features", "properly scale features"]



```
Feature ranking with Forests of Tree:
1. feature 7 (0.236878) total_stock_value
2. feature 11 (0.190838) long_term_incentive
3. feature 8 (0.055053) expenses
4. feature 4 (0.051660) bonus
5. feature 10 (0.043435) other
6. feature 6 (0.039480) deferred_income
7. feature 1 (0.038413) deferral_payments
8. feature 13 (0.036763) director_fees
9. feature 19 (0.033982) fraction_to_poi
10. feature 9 (0.032580) exercised_stock_options
11. feature 17 (0.030038) from_this_person_to_poi
12. feature 16 (0.029234) from_messages
13. feature 2 (0.028778) total_payments
14. feature 18 (0.028305) shared_receipt_with_poi
15. feature 14 (0.028148) to_messages
16. feature 0 (0.023973) salary
17. feature 15 (0.020773) from_poi_to_this_person
18. feature 3 (0.020396) loan_advances
19. feature 12 (0.017806) restricted_stock
20. feature 5 (0.013468) restricted_stock_deferred
```



```
Feature Ranking with Decision Tree:
1 feature no.7, total_stock_value, (0.618464883094)
2 feature no.13, director_fees, (0.061514385458)
3 feature no.0, salary, (0.0479812206573)
4 feature no.19, fraction_to_poi, (0.047443615944)
5 feature no.14, to_messages, (0.0463361502347)
6 feature no.15, from_poi_to_this_person, (0.0411267605634)
7 feature no.8, expenses, (0.0268344820697)
8 feature no.10, other, (0.0258463804446)
9 feature no.6, deferred_income, (0.0253087757313)
10 feature no.16, from_messages, (0.0248710453334)
11 feature no.2, total_payments, (0.0205633802817)
12 feature no.1, deferral_payments, (0.0137089201878)
13 feature no.12, restricted_stock, (0.0)
14 feature no.11, long_term_incentive, (0.0)
15 feature no.18, shared_receipt_with_poi, (0.0)
16 feature no.17, from_this_person_to_poi, (0.0)
17 feature no.5, restricted_stock_deferred, (0.0)
18 feature no.4, bonus, (0.0)
19 feature no.3, loan_advances, (0.0)
20 feature no.9, exercised_stock_options, (0.0)
```

- Features used for POI identifier

Features I end up using: “poi”, “total_stock_value”, “long_term_incentive”, “expenses”, and “bonus”.

I used both decision tree and forests of tree for feature selections. I chose features whose rankings are above 0.05 at least. So there are actually 5 features above 0.05: total stock value, long term incentive, expenses, bonus and director fees. The reason I excluded director fees from my final feature list is because the number of its “NaN’ values is 129 (around 86.6% values is “NaN”). This may result in bad influence on my model. So I exclude “director_fees” from my list.

- Didn’t scale my features

I used both decision trees and forests of tree for my feature selections. Their splits are based on each feature, and so they are independent from one another. Thus, feature scaling is not necessary.

- Create new features

I created new feature called “fraction_to_poi”:

fraction_to_poi = from_this_person_to_poi / to_messages

It tells a given person’s fraction of messages which are sent to poi. In decision tree’s features ranking, the importance score of “fracition_to_poi” is 0.047. While in forest tree, the importance score of it is 0.034. Although “fraction_to_poi” is not top 3 important features according to its importance score, it is a bit useful for identifying POI.

3. What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms? [relevant rubric item: “pick an algorithm”]

Using testing script I got the performance (precision score, recall score and F1 score) of different algorithms I tried.

- Performance of Naive Bayes

The precision score is 0.45052, the recall score is 0.32550, and F1 is 0.37794.

- Performance of Decision Tree

The precision score is 0.36272, the recall score is 0.36000, and F1 is 0.36136.

- Performance of tuned Decision Tree

I tried to tune parameters like "min_samples_split", "max_depth", "class_weight", "splitter" and etc. The overall precision score are okay (around 0.38204), but the recall score are not as good as precision score, around 0.29150.

I ended up using Naive Bayes. I also tried decision tree. Both the precision and recall scores of Naive Bayes are higher than decision trees, so does the F1 scores. This means Naive Bayes does a better job identify POI.

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? What parameters did you tune? (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). [relevant rubric items: "discuss parameter tuning", "tune the algorithm"]

Tuning parameters of an algorithm helps to get a better performance, such as high F1 scores and both high and balanced scores of precision and recall. Algorithms have their default parameters for performance. If you decide to tune the parameters but don't tune them well, the performance will actually be worse compared to using default parameters. Because bad tuning will negatively influence the "learning process" from variable data. As a result, you will have a model returning false data.

However, some algorithms do not have parameters to tune, like naive bayes. I actually tried naive bayes for this project, and the precision and recall scores were good(precision=0.4500, recall=0.2925). However, I want to tune decision tree to see if it could be improved and be better than Naive Bayes. Thus, I start tuning decision tree's parameters.

I used GridSearchCV to decide the optimal parameters for my algorithm. The attribute "best_params_" will return the optimal values or options of particular parameters for the best performance. I tuned "min_samples_split", "max_depth", "class_weight", and "splitter" parameters. The tuning result of decision tree was not good, I could not improve its recall score. The performance of Naive Bayes is better than decision tree. Therefore, I ended up using Naive Bayes.

5. What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis? [relevant rubric items: "discuss validation", "validation strategy"]

When performing a machine learning experiment, it is common to split your data into two groups: training data and testing data. Validation is the process checking your model's prediction (which was predicted from learning training data) against testing data. The classic mistake is training and testing on the same data. This results in a perfect score, but is useless when predicting other unseen data in the future. This classic mistake is called "Overfitting".

I used cross validation to split my data into two groups: training data and testing data. My model learnt from training data, and I will check my model's performance using the testing data.

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. [relevant rubric item: "usage of evaluation metrics"]

I used precision score, recall score and F1 score to evaluate my model's performance: how good my identifier works to predict POI. The precision is 0.45052. This means, people who are classified as POI by my model, 45.052% of them are actual POI. The recall score is 0.32550, meaning from the number of actual POI, my model correctly identifies around 33% of them.

Reference:

https://discussions.udacity.com/t/starting-on-the-project/30096/3
https://discussions.udacity.com/t/final-project-feature-selection/331644/6
https://datascience.stackexchange.com/questions/10773/how-does-selectkbest-work
https://discussions.udacity.com/t/missing-values-in-enron-data-set/281235
https://discussions.udacity.com/t/error-of-decisiontrees-featureimportances-attribute/174240/2
https://public.tableau.com/profile/diego2420#!/vizhome/Udacity/UdacityDashboard
http://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html
https://discussions.udacity.com/t/final-project-how-to-add-the-new-created-features-to-the-dataset/250408/2
https://discussions.udacity.com/t/when-would-we-not-use-feature-scaling/290523/2

https://stackoverflow.com/questions/31161637/grid-search-cross-validation-in-sklearn
http://scikit-learn.org/stable/modules/cross_validation.html

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.