**Enron Submission Free-Response Questions**

A critical part of machine learning is making sense of your analysis process and communicating it to others. The questions below will help us understand your decision-making process and allow us to give feedback on your project. Please answer each question; your answers should be about 1-2 paragraphs per question. If you find yourself writing much more than that, take a step back and see if you can simplify your response!

When your evaluator looks at your responses, he or she will use a specific list of rubric items to assess your answers. Here is the link to that rubric: [[**Link**](https://www.google.com/url?q=https://review.udacity.com/%23!/projects/3174288624/rubric&sa=D&ust=1521827604802000&usg=AFQjCNGdLZNqHrwoSLO2RdSlNyf0HFW7Uw)] Each question has one or more specific rubric items associated with it, so before you submit an answer, take a look at that part of the rubric. If your response does not meet expectations for all rubric points, you will be asked to revise and resubmit your project. Make sure that your responses are detailed enough that the evaluator will be able to understand the steps you took and your thought processes as you went through the data analysis.

Once you’ve submitted your responses, your coach will take a look and may ask a few more focused follow-up questions on one or more of your answers.

We can’t wait to see what you’ve put together for this project!

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”]

Machine learning in this project can help to do prediction on classification of person of interest (POI) who committed Enron fraud, so we can identify who may be the POI by looking at financial features and email features provided by the dataset. The dataset contained financial dataset which includes the Payments and Stock Compensations to each individual. The email features contain the number of email sent to, sent from, and copied to each individual. And the poi column indicates the individual that were convicted for fraud, which serve as a label for our algorithm training.

My EDA shows there are originally 146 data point (record of individual name), and 22 features (columns). Out of the total 146 individuals, there are 18 individual are convicted as POI. The features that have majority value as NaN values are “loan advances” (only 4 record have number), “restricted\_stock\_deferred” (18), “deferral\_payments” (39), “deferred\_income” (49), “director\_fees” (17), “long\_term\_incentive” (66).

I looks for whether any pattern for the POI, and I see the POI has higher than average bonus, this makes sense because they have the largest incentive to commit fraud and also imply they are in higher management position to commit and hide the fraud.

The outlier is the Total row, which I removed it from dataset to avoid distort the algorithm training, and it makes the total record to be 145 afterward.

1. 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”]

I ended up with 9 features: 'exercised\_stock\_options', 'other', 'expenses', 'ratio email to poi','shared\_receipt\_with\_poi','total\_stock\_value', total\_payments', 'bonus', 'restricted\_stock'. My selection process is test the features\_importance by the DecisionTreeClassifier as well as by the AdaBoostClassifier, and remove the features that give 0 importance. And I use the tester.py to evaluate for F1 score, and the features importance provided by DecisionTreeClassifier gives 0.37 F1 score, higher than the features provided by AdaBoostClassifier. I also use SelectKBest together with GridSearchCV, intending to find the features gives best optimal F1 score, but the features indicated only give me less than 0.3 F1 score, so I decided to use the 9 features derived by the features importance from DecisionTreeClassifier.

I didn’t need to do scaling because Decision Tree and Ada Boost doesn’t require it.

I create 4 new features, and the rationale is per below:

* 1. **'ratio email from poi':** for that individual total received email, what is the % from a poi, this can be better indication of how involved that person is with POI to collaborate for fraud.
  2. **'ratio email to poi':** for that individual total sent email, what is the % sent to a poi.
  3. **'ratio of total payment vs bonus':** the % of bonus versus total payment received. I am thinking that bonus is considered a way to obtain large amount of money in short amount of time, and a larger proportion of bonus over total payment may signal a special type of favor that are granted due to collaboration of fraud
  4. **'ratio of total stock vs exercised stock':** the % of exercised stock versus total stock. I am thinking larger proportion of exercised stock option over total stock may signal a fraud insider who may be granted larger stock option as an incentive, but are aware of the fraud activity so are less confident to company's future and think of a way to exit, so exercise larger share the stock option.

By introducing the above email ratio features, my F1 score for DecisionTree increase from 0.21 to 0.31, which is good improvement.

As requested by the question: below is the features importance derived from DecisionTreeClassifier:

('exercised\_stock\_options', 0.21677011118291087),

('other', 0.18994891263592051),

('expenses', 0.16151489835700364),

('ratio email to poi', 0.13649901633967093),

('shared\_receipt\_with\_poi', 0.1185191078088922),

('total\_stock\_value', 0.055523084124288379),

('total\_payments', 0.050743657042869622),

('bonus', 0.042286380869058032),

('restricted\_stock', 0.028194831639385791),

1. 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”]

I ended up using DecisionTreeClassifier. I did an initial assessment on Logistic Regression, Decision Tree, Random Forest and AdaBoostClassifier using tester.py, but then I decide to not use Logistic Regression and Random Forest soon after because the initial assessment for these 2 only give F1 score around 0.17, versus Decision Tree 0.21 and Ada Boost 0.34. I later do more features selection and parameter tuning on DecisionTree and AdaBoost, and is able to get to F1 score 0.50 with Decision Tree, while AdaBoost only reach about 0.37, and AdaBoost is much slower than DecisionTree, so I decided to use DecisionTree to conclude.

1. 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”]

Tune the parameters is to adjust the way the algorithm assess what is considered the best fitted label. If I don’t do well, the algorithm’s prediction accuracy can be severely impacted. A well tuned parameter supposed to overcome overfitting problem (did well on training set, but did poorly on testing set). I manually did the parameter tuning by changing each parameter one at a time and test to get to the highest F1 score. For example, for Decision tree, I change the criterion between ‘gini’ (for Gini impurity) and ‘entropy’ (for information gain) which is to control how the decision tree decides where to split the data. I also change min\_samples\_split, splitter, max\_depth and min\_samples\_leaf, aiming to get to the highest F1 score.

1. 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”]

Validation is a way to measure how well the algorithm can predict the result correctly. Sometimes an algorithm can perform extremely well on the training dataset, but did poorly on the test dataset or new dataset, which could be a classic mistake of overfitting. To avoid this, we need a way to evaluate the algorithm before putting it in use. I did the cross- validation which is to split the dataset into training and testing data using train\_test\_split to use 25% of dataset as test group, and asses its precision score, accuracy score, recall score, and F1 score.

1. 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”]

Accuracy score: Out of the total prediction, what proportion do you predict correctly. (my score at 88.2%)

Precision score: Out of the total data predicted as POI, what proportion do you predict correctly. (my score at 57%)

Recall score: Out of the total real POI, how many you predicted as POI correctly. (my score at 47.5%)

F1 score: The weighted average of Precision score and Recall score. (my score at 51.8%)