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”]

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, I use email and financial data for executives at Enron to identify persons of interest in the fraud case. A person of interest (POI) is someone who was indicted for fraud, settled with the government, or testified in exchange for immunity. This report documents the machine learning techniques used in building a POI identifier.

When I explore the dataset, I find that there is one "person" who has very large values on each feature. After some investigations, I find there is a key "TOTAL" in the data disctionary. That's definite an outlier since we want to investigate the POI. So I use "data_dict.pop("TOTAL",0)" to remove that data. After removing that outlier, I find there are still some people who have very large salary but low bonus or have very large bonus but low salary. I would not remove those data since they may be POI.

After deleting "TOTAL", there are total 145 points. There are 18 POI points and 127 non-POI points. The percentage of POI is about 12.4%. So we can't use usual accuracy metric. I need to use precision/recall to measure the performance of the classifiers. Besides I find there are many "NaN". For examaple, The features which has "NaN" below 40% are just "total_payments","exercised_stock_options","restricted_stock","total_stock_value","other","expenses",and "email_address". The feature "email_address" is useless.

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, f 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. [relevant rubric items: “create new features”, “properly scale features”, “intelligently select feature”]

First, I use as many features as possible, features_list =
['poi','salary','deferral_payments','total_payments','loan_advances','bonus','restricted_stock_deferred','deferred_income','total_stock_value','expenses','exerci
overfit a decision tree, I find both the precision and recall are 0.0. Then I check the importances of each feature, I find that the importances for
'salary','deferral_payments','loan_advances','bonus','restricted_stock_deferred','deferred_income','long_term_incentive','director_fees','from_poi_to_this_pers
are all 0.0. So I try to delete these features and keep the features_list =
["poi","total_payments","exercised_stock_options","restricted_stock","total_stock_value","other","expenses"].Then I overfit a decision tree again,I find the importance for each feature
{"total_payments":0,"exercised_stock_options":0.2074,"restricted_stock":0.1331,"total_stock_value":0,"other":0.4926,"expenses":0.1667}. And the correspond precision is 0.25, recall is 0.25. Then I remove feature "total_payments",and do the classification again, I find the importance for each feature
{"exercised_stock_options":0.1148,"restricted_stock":0.1331,"total_stock_value":0.09,"other":0.4928,"expenses":0.1667}. And the corresponding precision is 0.375,recall is 0.33. Now we have better performance. But I notice that the importance for "total_stock_value" is still very low. So I try to remove "total_stock_value" and do the classification again. I find the importance for each feature
{"exercised_stock_options":0.1148,"restricted_stock":0.1331,"other":0.5854,"expenses":0.1667}. And the corresponding precision is 0.375,recall is 0.33. Even though removing "total_stock_value" neither increases nor hurts our performance, I prefer to remove it and just keep features_list =
["poi","exercised_stock_options","restricted_stock","other","expenses"].

I created two new features. One is the feature for the fraction of emails from POI to this person and the other is the feature for the fraction of emails from this person to POI. Because I think if one person contacts with POI very much, this person quite be POI too.But when I add these two new features to my features_list, I find without these two new features, precision = 0.375 and recall = 0.33, with these two new features, precision = 0.5,recall = 0.25. So I decide not to use these two new features in order to get both high precision and recall. I end up using features_list =
['poi',"exercised_stock_options","restricted_stock","other","expenses"].

I don't have to do feature scaling since I end up using Decision Tree, which will not be affected by feature rescaling. But I indeed do feature scaling before I compare different algorithms since some of them need feature scaling.

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”]

I end up using Decision Tree. I tried Naive Bayes classifier, Decision Tree classifier, Support vector machine classifier, K nearest neighbors classifier, AdaBoost classifier and Random Forest classifier. The models perform quite differently.

Algorithm	precision	recall
Naive Bayes	0.37	0.25
Decision Tree	0.375	0.33
Support vector machine	0	0
K nearest neighbors	0	0
AdaBoost classifier	0.75	0.25
Random Forest	0.5	0.25

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). [relevant rubric item: "tune the algorithm"]

Tuning the parameters of an algorithm means checking the performance of the algorithm by changing the its parameters. If we don't tune the parameters well, we may miss some well-performed algorithm. I use the GridCV method to tune the parameters. First I set parameters = {'criterion': ('gini','entropy'),'splitter':('best', 'random'), 'min_samples_split':[2,4,6,8,10,12,14,16,18,20,24,26,30]}, then by the GridCV method, I find the best estimaator is DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None, max_features=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, random_state=None, splitter='best')

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 item: "validation strategy"]

Validation is that we split our data into training data and testing data. We use the training data to train our model and use the testing data to test our model. A classic mistake is the we use the data in class A to train our model, while we use the data in class B(different from class A) to test our data. In this case, our model will perform very bad. I use cross_validtion method to set 80 percent of the data as training data and 20 percent of the data as testing data. In the process of selecting my algorithm, I also use stratified shuffle split method in cross_validation to validate my data because the dataset is reletively small and stratified shuffle split can try to use all the data to train and all the data to test. Besides, since the data contains only about 12% POI points, shuffling the data can try to avoid the case where we use different points to train and test.

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 use precision and recall as my evaluation metrics. Average precision of my classifier is 0.31. Average recall of my classifier is 0.35.

Whenever a POI gets flagged in my test set, I have about 31% confidence that it's very like to be a real POI.

Every time a POI shows up in my test set, I have about 35% confidence to identify him or her.