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, we will build a person of interest (POI) identifier based on financial and email data made public as a result of the Enron scandal. We are given two kinds of features - financial and email. Each category has many subfeatures. Our job is using these features to identify whether a person is a POI or not. This is a classical classification problem in machine learning. We have two classes here, POI and non-POI. Because the features of POI and non-POI must be different, we can utilize those feature differences to separate POI and non-POI.

The total number of data points in the dataset is 146 with 18 POIs and 128 non-POIs. After exploring the dataset, I found many missing values and outliers in the dataset. I plot "salary" versus "bonus" and find obvious outliers from the plot. By looking at the plot, I find the outliers has property of "salary" greater than 1e7. So I simply remove records with "salary" greater than 1e7.

The variables with missing values are as follow:

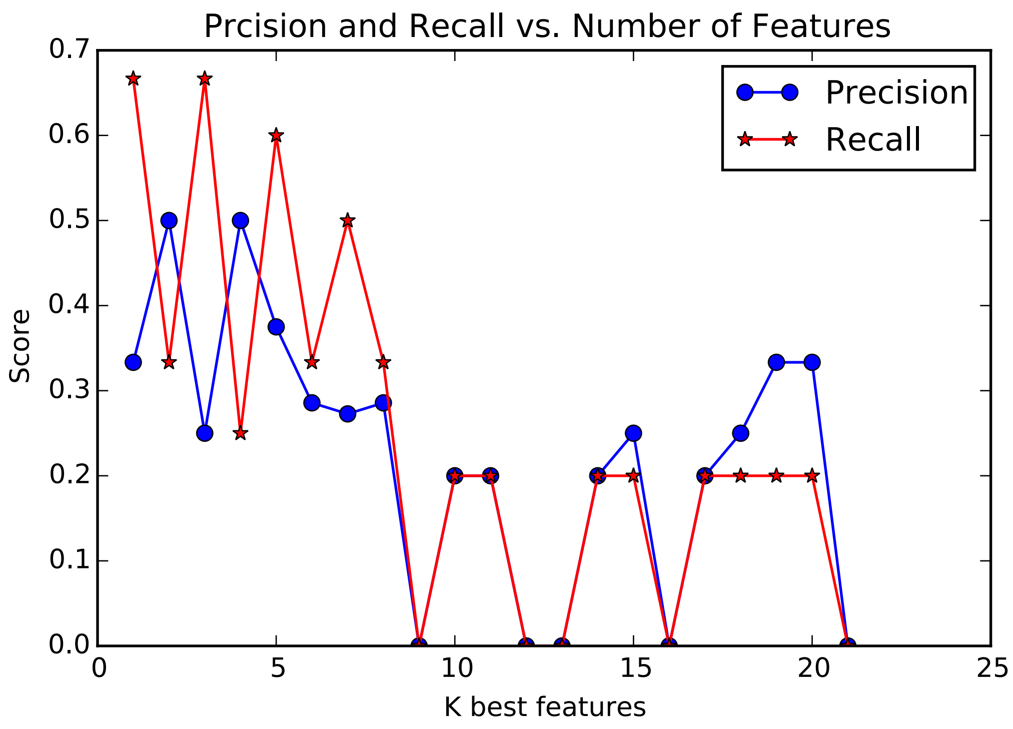
```
{'bonus': 64,
 'deferral_payments': 107,
 'deferred_income': 97,
 'director_fees': 129,
 'email_address': 35,
 'exercised_stock_options': 44,
 'expenses': 51,
 'from_messages': 60,
 'from_poi_to_this_person': 60,
 'from_this_person_to_poi': 60,
 'loan_advances': 142,
 'long_term_incentive': 80,
 'other': 53,
 'restricted_stock': 36,
 'restricted_stock_deferred': 128,
 'salary': 51,
 'shared_receipt_with_poi': 60,
 'to_messages': 60,
 'total_payments': 21,
 'total_stock_value': 20}
```

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", "properly scale features", "intelligently select feature"]

The features I used for POI identifier is as follow:
[u'salary', u'total_payments', u'exercised_stock_options', u'bonus',
    u'restricted_stock', u'shared_receipt_with_poi', u'total_stock_value',
    u'deferred_income', u'long_term_incentive', u'scaled_to_poi'].

First, because missing values can give us trouble during building a model and imputation would introduce new errors, I simply exclude features that have more than 55% missing values. I choose 55%, because it corresponds with only 5 features and features with missing values greater than 50% are hard to use for building a model.

Then, the univariate selection method (SelectKBest) is used to choose the features. Univariate feature selection works by selecting the best features based on univariate statistical tests. It can be seen as a preprocessing step to an estimator. I use the Decision Tree model to iteratively try each best feature number, and get the best feature number to be 5. Because 5 gives the relative high score for both precision and recall as showing in the following figure. For the same value of score, I choose the least feature number, because more features need more data to train the model.



I scaled the "from_this_person_to_poi" and "from_poi_to_this_person" using "from_messages" and "to_messages", respectively, and create two new features "scaled_to_poi" and "scaled_from_poi". Because a person sending many emails to the POI may happen to be sending emails to everyone, we want a relative value here. Similar argument can be applied to people who received many emails.

Score generated by SelectKBest:
25.3801052998 expenses
24.7525230203 from_this_person_to_poi
21.327890414 exercised_stock_options
18.8617953165 salary
9.48074320348 other
8.96781934768 total_stock_value
8.90382155717 long_term_incentive
6.3746144902 shared_receipt_with_poi
1.75169427903 bonus
0.20970584227 total_payments
0.0644770280387 from_poi_to_this_person

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 tried Gaussian Naive Bayes and Decision Tree. I ended up with using Decision Tree, which gives better prediction. The performance metrics I used are precision, recall, and F1 score. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned. A high area under the curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. High scores for both show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall). The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal.

For GuassianNB, the precision is 0.25, recall is 0.20, and F1 score is 0.22.
For Decision Tree, the precision is 0.38, recall is 0.60, and F1 score is 0.46. Therefore, Decision Tree seems more powerful in this problem.

	Precision	Recall	F1 score
GaussianNB	0.25	0.20	0.22
Decision Tree	0.38	0.60	0.46

Reference:http://scikitlearn.org/stable/auto_examples/model_selection/plot_precision_recall.html

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"]

For some complex models like Decision Tree, there are some hyperparameters that need to set. These parameters can be very important for the problem that trying to solve. For the Decision Tree, there are different information gain functions you can choose. Different information gain functions represent different splitting criteria, which can generate different results and performance. If we fail to choose proper parameters, the model may give us bad results, and the prediction may not be right. In other words, the model may fail to have generalization.

Machine learning algorithms are parameterized and modification of those parameters can influence the outcome of the learning process. Think of each algorithm parameter as a dimension on a graph with the values of a given parameter as a point along the axis. Three parameters would be a cube of possible configurations for the algorithm, and n-parameters would be an n-dimensional hypercube of possible configurations for the algorithm. The objective of parameter tuning is to find the best point or points in that hypercube for your problem. You will be optimizing against your test harness, so again you cannot underestimate the importance of spending the time to build a trusted test harness.

For the parameter tuning, I use the grid search method. Basically, grid search method can try each parameter combination given a certain set of possible parameter values, and choose the one with the best performance through cross-validation.

Reference: http://machinelearningmastery.com/how-to-improve-machine-learning-results/

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 to ensure the model works well on data that not appear in the training dataset. A classical mistake is overfitting. Overfitting means the model may work great on the training data, but when apply it to other data (not in training dataset), the model gives very bad performance.

The dataset is split into training and testing. I use the training to generate the model, and use the testing dataset to evaluate the model performance.

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"]

For the model performance, precision, recall, and F1 score are used. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned. A high area under the curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. High scores for both show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall). The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal.

The precision (0.33), recall (0.33), and F1 score (0.33) are all greater than 0.3 for the testing dataset.