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"]

The goal of this project is to predict people who might be involved in Enron fraud using email data and financial data of the employees. This is achieved in this project through the feature 'poi'. The algorithm tries to predict the class (POI/Non-POI) for each data point based on the features selected. Supervised classification machine learning is very useful for such cases. It learns the patterns from training set and uses the information gained to predict the test set. Although, the dataset is limited, i.e. there are only certain number of employees, the machine learning algorithm does a good job of predicting the class for each employee.

First part of this project involved collecting background information for the fraud. I watched the documentary suggested in the course. Only then did I understand the magnitude of this fraud. I also became familiar with a lot of names involved in the fraud. This information also helped me narrow down my features. There were totally 146 employees in the data set with 18 people classified as POI. There were 21 features provided for each employee. I also created a new feature later. Therefore, totally there were 22 features under consideration. A lot of these features had missing values. I counted the NaNs in each feature and printed it to get an overview.

Feature	**Number of NaNs**
bonus	64
deferral_payments	107
deferred_income	97
director_fees	129
email_address	34
exercised_stock_options	44
expenses	51
from_messages	59
from_poi_to_this_person	59
from_this_person_to_poi	59
loan_advances	142
long_term_incentive	80
other	53
poi	0
restricted_stock	36
restricted_stock_deferred	128
salary	51
shared_receipt_with_poi	59
to_messages	59

The outliers were determined by plotting histograms for 14 of the 18 features. The omitted features were those that contained a lot of Nans (> 100). The plots showed that almost all features had outliers. Printing the top 6 values showed that they were genuine employees and mostly POIs. Data point 'TOTAL' was also removed as learned from the lectures. Checking for similar wrong entries by scanning key names, I found another data point 'THE TRAVEL AGENCY IN THE PARK'. I removed it before proceeding with search for negative values found in histograms. The negative values found in 'restricted_stock' and 'total_stock_value' are compared with the pdf data and errors are corrected before proceeding further.

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"

Before going into the selection process, I created feature 'fraction_from_poi' which determined the fraction of messages received from POIs in the dataset. I created this with the intuition that POIs connected in the fraud must have communicated a lot frequent than other non-POIs. I further though that 'fraction_from_poi' must be a better indicator than the fraction sent to a POI, since POI sending emails frequently to an employee must be a rare event than employees sending emails to POIs, who happen to be mainly top management. Thus usual monthly reports sent by non-POIs might also be accounted for in fraction sent to poi.

The selection of parameters was a difficult task. I first gathered background knowledge by watching the documentary about Enron fraud. Then temporarily selected 8 features and omitted those which contained a lot of NaNs (above 100). I also included the new feature I created.

Then I programmatically evaluated all the possible combinations of these 8 features (256 combinations) using "combinations" procedure from "itertools" library. The evaluation metrics for each combination of features were stored in a dictionary. I selected the best features by sorting the dictionary values for highest recall. The top 5 combinations of features and their metrics are given in the table below sorted according to recall using Decision trees classifier.

Features	**Accuracy**	**Precision**	**Recall**
Bonus, long_term_incentive, from_poi_to_this_person	0.788	0.413	0.396
fraction_from_poi	0.785	0.936	0.382
Bonus, exercised_stock_options	0.777	0.347	0.382
Salary, exercised_stock_options, fraction_from_poi	0.807	0.376	0.381
Salary, restricted_stock, exercised_stock_options, fraction_from_poi	0.821	0.374	0.378

The selected features were 'bonus', 'long_term_incentive', 'from_poi_to_this_person'. I also tried the same iterative process with Naïve Bayes classifier. But did not use it finally, since the maximum recall value obtained for it was 0.34. The reason for selecting to maximize recall score was that I wanted to capture as many POIs as possible even though it came at the expense of catching a few innocent employees. The aim of this project is to find the employees who were part of the fraud using email and financial data. Thus, even though a few innocent employees were misclassified, they could be cleared after further investigation. This means we would be able to catch all people involved in the fraud.

I did not use any scaling because the algorithms (Decision Tree and Naïve Bayes) that I chose were not affected by the distance of the points to the boundary. After finding that decision tree gave the best recall score for the three selected features, I calculated the importance for features and they had an average of 0.33, 0.25 and 0.40 respectively.

The newly engineered feature was also not selected for the final features to be used in the classifier as it had a negative impact on the precision and recall values. Average values of metrics with and without new feature are tabled below.

	Accuracy	**Precision**	**Recall**
With newly engineered feature	0.76	0.35	0.33
Without new feature	0.78	0.41	0.39

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 two algorithms: Decision tree and Naïve Bayes. The selection of the algorithm came down to aim of this project. I wanted to maximize recall score as much as possible. When Naïve Bayes was used, I got high precision score, but low recall score. However, when I used Decision tree, I was able to obtain higher recall value and comparable accuracy for the same set of features selected. Hence I finally picked Decision Tree to use for this project. The table below shows the metrics for the same set of features using the two classifiers.

Features	**Decision Tree**			**Naïve Bayes**		
	Accuracy	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
bonus, long_term_incentive, from_poi_to_this_person	0.79	0.41	0.39	0.81	0.49	0.24

This was also evident, when I displayed different combinations of feature selection with precision and recall greater than 0.3. There were lot less combinations with high scores for Naïve bayes than Decision Tree. Hence I picked Decision Tree to use in this project.

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 helps to squeeze out extra performance from the selected algorithm and improve the machine learning results. One way of tuning an algorithm is to tune the parameters associated with it. However, the tuning must be done with confidence in the test results. This means that techniques that reduce the variance of the algorithm performance must be used. One method is to use cross validation with large number of folds.

I tuned the algorithm using GridSearchCV to select the best parameters for the classifier. I chose to vary parameters 'criterion', 'splitter' and 'min_samples_split'. During tuning, I used recall as the method to score the classifier rather than accuracy, which is present as the default argument. If the tuning is not done properly the algorithm might be too sensitive to some outliers or wrong entries. This will lead to very complex boundaries which perform very well on the test set, but are very bad on the train set.

I mainly focused on parameter min_samples_split. I wanted to test it for three values: 2, 5, 10. I also varied the criterion between the two possible inputs 'gini' and 'entropy'. Splitter input was selected

between ‘best’ and ‘random’. After using the grid search, I found that the best values for ‘min_samples_split’ was 2, criterion was ‘gini’ and splitter was ‘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 a process where the data set is split into train and test data. The train data is used to fit the classifier and train it. The test data is used to predict the classes for the test data set. This method is used to avoid overfitting of data. This helps the user to check the performance of the model. If entire data is used instead of train and test data, there would be no chance to see how the model works for new data. While using validation strategy, if the model performs poorly on test data, the user can infer that he has over fitted the dataset.

I evaluated by using StratifiedShuffleSplit present in the test_classifier() function in the file tester.py. The advantage of using StratifiedShuffleSplit comes from the main dataset which has unbalanced classes i.e. a lot of Non-Pois than Pois. By using this StratifiedShuffleSplit cross validation method, we make sure that the percentage of samples of each class are relatively same as the entire dataset for each fold. Also, the samples are shuffled which increases the possibility of random folds.

It provides train/test indices for the number of folds specified as a parameter. In this project the number of folds (n_iter) were 1000. That means the data was split into test and trainset 1000 times and the average scores are returned. Each time a random test data is used for the prediction.

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”]

Accuracy

Accuracy is defined as the ratio of number of correct predictions made across all classes to the total number of predictions made. In context of this project, it refers to the fact that how many employees were correctly predicted as either a POI or non-POI. The score for the model generated in this project was approximately 79%. That means around 80% of the employees were predicted correctly either as a POI or Non-POI. However, one shortcoming of this metric is that it is not ideal for skewed classes. We have a lot more Non-POIs than POIs. The model can learn well to predict Non-POIs than POIs. Hence a metric like precision or recall can be used to suit the needs.

Recall

Recall refers to the correct positive predictions made out of the total actual positives in the data set. It is difficult to understand about recall. Hence I will explain about recall using this project dataset. The recall score defines the number of POIs identified out of the total number of POIs in the dataset. Ideally this value must be equal to 1 so that we identify all the POIs correctly. However, the model developed in this project has a value around 0.39. The feature selection and grid search for best parameters was done with an aim to maximize recall score. The idea was that most POIs in the dataset must be identified. Even though some innocent employees might be captured due to low values of precision or accuracy, they could be cleared after further investigation. Precision was given second highest importance after recall

Precision

Precision refers to the total correct positive predictions made out of all the positive predictions. In simple terms, it indicates how likely a person identified as a POI is a POI. The model has a prediction score of about 0.41. During feature selection, a higher score was also possible with other features. Since the aim was to capture as many POIs as possible, the accuracy and precision were given second best preference and made sure that these values did not drop below 0.7 and 0.3 respectively, which was considered as a benchmark.