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 identify persons of interest (POI) from a list of Enron insiders. This investigation references a database of employee financial and email data that became part of the public domain as a result of the legal proceedings of the Enron fraud investigation. The dataset contains 146 entries with eighteen identified POIs (12% of entries). There are a total of 21 different features available. Only three features were used and they were chosen using the SelectKBest alogrithm. The financial information in the dataset is incomplete which is why there are multiple entires with 'NaN' values. Given that the main goal of this project is to classify individuals by using a multidimensional dataset, this is a good problem to solve with machine learning.

   There were three outliers in the Enron dataset: an entry for 'TOTAL' which had the total for each financial feature in the dataset, 'TRAVEL AGENCY IN THE PARK' which isn't a real person, and 'LOCKHART EUGENE E' which only had NaN values. The structure of the dataset was a dictionary, therefore these were easily removed from the dataset by using the .pop method.

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"]**

   I used GridSearchCV to test whether using univariate feature selections (SelectKBest), principal components, or a combination of the two was best to classify individuals. I used MinMaxScaler to scale the features in the dataset prior to passing them into the PCA algorithm given its reliance on variance. I found that selecting three of the original features yielded the best f1 scores. After making this determination, I went back and ran the SelectKBest algorithm on its own to find out which were the best three features in the dataset. I found that deferral payments, total payments, and exercised stock options had the highest feature scores. The scores for each of these was 9.819, 8.962, and 9.956 respectively. These were the three features that I used in the final classifier.

   I tried generating my own feature named compensation_ratio which is a function of dividing the total stock value of an individual by the combined value of their salary, bonus, and total stock value. I created this feature to explore if POI's were more likely to

have a higher compensation_ratio as compared to non-poi's. After running the SelectKBest algorithm, I found that this feature had a relatively low score (4.939) and should not be included in the classifier.

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 ended up using a decision tree as my final classifier because it had the highest recall and precision score. The scores for these were 0.317 and 0.319 respectively. The decision tree algorithm was the only one that I tested that was able to achieve a score of 0.3 for both metrics. I also tried using Naïve Bayes, Random Forest, and K-Nearest Neighbors (KNN). One notable result from these tests was the KNN classifier. It yielded the best precision with a score of 0.6 but it lacked the required recall.

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 the parameters of an algorithm means adjusting the settings in order to achieve the best performance possible. If you do not tune your algorithm well you could negatively affect the overall performance. I tuned every algorithm that I tested in this project by using the GridSearchCV functionality provided in Sklearn to ensure that I was achieving the best results possible. I tuned the decision tree that I choose as the classifier for this project by adjusting the calculation method for information gain and the min_samples_split which controls the minimum number of samples required to split a node. I also used GridSearchCV to find the optimal number of features that I should use from SelectKBest.

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"]**

   Validation is a way to estimate the performance of an algorithm on an independent dataset. It also serves as a check on overfitting. A classic mistake that you can make is allocating too many entries from your dataset to be used for training. In doing this you would achieve the best learning results but would your validation process would be comprised. In turn, you wouldn't know how well your algorithm would actually perform on an independent dataset. Given the size of the dataset and the relatively small number of POIs, stratified shuffle split was used as the validation method. This method is useful in this case because it randomizes folds of training and testing sets while preserving the percentage of samples for each class.

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 validated the POI decision tree classifier by using precision and recall scores. The precision metric measures how accurate the algorithm is at labeling true POIs. In other words, it measures the number of true positives divided by true positives and false positives. Recall measures the effectiveness of the algorithm to identify that someone is a POI. It is a measurement of all of the items that are truly positive divided by how many total POIs there are in the dataset.