1- The goal of this project is to build a machine learning agorithm that identifies Enron employees who might have commited fraud, using Enron's financial and email datasets which give related information on people somehow involved with the company. The Financial dataset include 12 features such as salary and bonus and 146 datapoints. The email dataset includes information on the number of email transactions between POI and each person and the total number of email transactions. There are 6 features there. The datasets can be used to train a machine learning algorithm to indentify individuals that might be involved in fraud as it provides also labels for individuals, which it make it also suitable for supervised machine learning. The datasets used have many missing features for individuals and a few outliers. I took out three features: **1- directors_fee 2- loan_advances 3- restricted stock deferred. only 16 non_pois received directors_fee and non of the pois. only 2 of the non_pois get "loan advances" and 1 of pois. this feature clearly does not give enough information and should be removed. deferral_payment has been eliminated because of lack of info for many entries. there is no poi with "restricted stock differed" and only 17 non_pois have values for that feature, so this feature was removed as well my feature selection algorithm chose among the remainder. the outliers I removed include : "TOTAL" was removed because it is sum of all values and obviously an outlier. "THE TRAVEL AGENCY ON THE PARK" was removed because it is not a person and "LAY KENNETH L" was removed because of all his feature values being way out of range. I used pop method to remove outliers. These were outliers compared to other non_pois. I would like to add that standard deviation from the median was only used to give me an idea about which data entries might be outliers, I also visually inspected all data points.** . I used intuition, a code for scatter plot and codes for finding out liers based on standard deviation. The file including the code for sifting through dataset is included in the submission package.There are only 18 pois out of 146 enteries and the rest are non_poi.

2- Through visual inspection and the code I ran, I could identify 3 features that has very few features and did not give enough information and therefore had to be removed. SelectKBest was used to select features in a loop which iterated over the number of the features. Although, the two algorithms that I used, decision tree and Naive Bayes, did not need feature scaling, I performed it anyways. Two new features were created from email features indicating the fraction of the emails recieved from POIs and sent to POIs. It makes sense to add these features as a POI might be interacting more with other POIs. The algorithm looped over the number of features and selected k best featured, k being incremented in every iteration and the features being tried on the machine learning algorithm of choice. The features selected for decision tree were:

```
Precision:  0.562242374279  Recall:  0.341

Number of features:  4

features:  ['bonus', 'deferred_income', 'shared_receipt_with_poi',
'fraction_to_poi']

Feature importances(scores):  [ 14.02443762  12.12703855  13.54943277
16.32528383]
```

3- I ended up choosing decision tree algorithm because of it's better results. I used Naive Bayes as the first algorithm and I got precision of 3.5 and recall of 3.5 with it. With the latter no parameter tuning is needed but with decision trees, I had to tune min_sample_split, trying different values and 10 seems to be giving the best results. I tried Kmeans as well and the results for precision and recall were just bellow 0.3. The decision tree seem to be the fastest algorithm to run either.

4- Tuning the parameter of a machine learning algorithm means how do we want to fit it to the training data. It is finding a ballance between variance and bias, which means having the parameters that enable our program to make the most accurate predictions with keeping its ablility to generalize to new datasets. If the parameters are not tuned well we run into either overfitting or underfitting. I had to use different min_sample_split for my decision tree to get the best results. I also tried GridSearchCv for SVR algorithm.

5- accuracy, precision and recall were used as metrics for validation. The latter two are the more reliable ones. precision means the fraction of the predictions that were correct. recall is all targets that were predicted right. in the case of this project, precision means the proportion of individuals that were predicted as pois that were actually pois and recall means the fraction of predictions that were rightly labeled as pois. In this project, StratifiedShuffleSplit was used to split the data set into training and testing sets. I made a small change to the tester.py file so that it returns values ,so that I could use it for validation, with out changing it's primary functionality. It's important to validate the algorithm in orther to avoid overfitting and also fully evaluate the performance of the algorithm. using StratifiedShuffleSplit is particularly important for this project because of the small number of the datapoints.

6- I ended up choosing decision tree for algorithm. It gave precision of : 0.562242374279 and recall of 0.341. it means that out of all test datapoints that were predicted as pois : 0.562242374279% of them were correctly predicted by the algorithm and out of all the datapoints that were actually pois 0.341% of them were labeled as pois.