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?

- The main goal of the project was to create a machine learning model to classify people as possibly related to the fraud that bankrupt the Enron. On the dataset we have data related to payment, stock and email exchange plus metadata like the total number of emails sent by the person of interest (POIs), with these data we can identify people that tried or were linked to the criminal actions which resulted in the fraud. According to the PDF we had two obvious outliers, the "total" and "the travel agency in the park" that aren't possible POIs or have any information to consider, so they were deleted from the dataset.

1. 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.

- I used all the features available to aggregate information as much as possible, I just removed the emails accounts because for me it don't have significance to my machine learning model.

1. What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms?

- I end up using the `AdaBoostClassifier` because was the best result with the data used, it was compared to `GaussianNB`, `DecisionTreeClassifier` and `svm.SVC`.

1. 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).

- The parameter tuning is way to boost the machine learning algorithm to have the best perfomance, and it can be archived by testing and treating the results for each banch of parameter test. I just tested the number of estimator until the best results from AdaBoostClassifier, but I could have tried cross-validation to implement the settings with the best performance. It can also be done automatically by GridSearchCV using different settings combinations to return the best parameters to use.

1. What is validation, and what’s a classic mistake you can make if you do it wrong? How did you validate your analysis?

Validation is a way to confirm the results obteined by your model, and it prevets overfitting. My analysis was validate by using cross-validation, dividing the dataset into training data and testing

data. I also used the `test.py` script that uses `StratifiedKFolds` that is another option of cross-validator by splitting the dataset in many others.

1. 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.

algorithm	accuracy	recall
AdaBoostClassifier	0.85573	0.34050

An accuracy of 0.85573 means that around 85.6% of the person predicted as POI were right, it isn't high but is good enough for the project threshould. And a recall of 0.34050 means that the algorithm found 34.1% of the POIs present on the dataset.