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, by using the Enron financial and email dataset, build a person of interest (POI) identifier. In other words, identify Enron employees who may have committed fraud. The dataset has financial and email data that were collected and released as part of the US federal investigation into financial fraud. It is important to note that Person of interest means a person who is charged by the law for committing a crime.

This work was done by exploring different machine learning algorithms and address different feature selection methods. By exploring the data it revealed some outliers, that were eliminated from the data set. It was removed because it would negatively affect the results of the algorithms.

An overview of the dataset:

- Dataset length - 146
- 21 features
- Number of POIs - 18

Additionally, I've removed the feature '*loan_advances'* as this feature was missing values.

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 SelectKBest to choose which features I would include in the final list:

['salary', 'total_payments', 'bonus', 'total_stock_value', 'shared_receipt_with_poi', 'exercised_stock_options', 'expenses', 'deferred_income', 'long_term_incentive', 'from_poi_to_this_person']

I created five features for this project after performing an analysis with SelectKBest, that I've used the number 10 for k that was decided to deliver the best mix of performance (timing), precision and recall. Three of them are financial features, where I used to get the fraction portion from the type of financial incentives received, in this case:

- Salary
- Bonus
- Stock

It is possible to check below the features selected by SelecKBest:

So, with this features it is possible to understand if a higher fraction of payments in certain modalities led to POI.

Additionally, I've used more two features here, one of the features, in this case 'to_poi', I used to check the fraction of messages that were sent to POIs of the total sent messages. Additionally, I created another one feature, 'from_poi', that is a fraction of messages that were received from a POIs out of the total messages received.

So, if a has been some communication between a person and a POI, it could mean that this person is also a POI. If we get some smaller value of one of these features, ‘to_poi’ and ‘from_poi’, it would mean more communication and hence higher probability of this person being a POI.

If a person has been communicating with POI a lot, it could mean he/she is also a POI. Smaller value of to_poi and from_poi would mean more communication and hence higher probability of being a POI himself/herself. Next, lets evaluate if including these two parameters improve the accuracy of our predictions

Additionally, during the GridSearchCV, all features were scaled using a MinMaxScaler so the model would perform optimally with scaled features.

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”]

By checking the scikit learn documentation I ended using the following algorithms:

- GaussianNB
- DecisionTree
- SVC

With the following results:

Algorithm	Accuracy	Precision	Recall
GaussianNB	0.3023255813953488	0.125	0.6666666666666666
SVC	0.5116279069767442	0.17391304347826086	0.6666666666666666
DecisionTreeClassifier	0.5116279069767442	0.4	0.3333333333333333

It is important to note that precision and recall are two extremely important model evaluation metrics. While precision refers to the percentage of your results which are relevant, recall refers to the percentage of total relevant results correctly classified by the algorithm. Unfortunately, it is not possible to maximize both these metrics at the same time, as one comes at the cost of another.

In other words, precision measures how many positive predictions were actual positive observations, so, the proportion of predicted POIs that were actually POIs, and recall measures how many positive observations were predicted correctly, so, the proportion of actual pOIs that were predicted correctly.

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"]

Machine learning models are parameterized so that their behavior can be tuned for a given problem. A tuning parameter is parameter used in statistics algorithm in order to

control their behaviour. The algorithm creates normally for each value of the tuning parameter a different model. The algorithms, in general, are very powerful, however, they can be fragile in some aspects due to how they are applied to solve general purpose problems.

I optimized the GaussianNB process with a focus on the parameters, as Pipeline allow a more customizable way to do this. I used a specific param_grid and by doing this was able to get the following results:

- Accuracy: 0.30232 -> 0.82427
- Precision: 0.125 -> 0.32717
- Recall: 0.666 -> 0.30100

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 process where you can check how your model performs on unseen data. One thing, that is called Overfitting, is when you tune your model to be able to predict your training data with a good performance, however, it performs poorly on unseen out-of-sample testing data. So, one of the goals in validation is to avoid overfitting, which can be accomplished through a process called cross-validation.

To validate my analysis, I used the method of the tester.py that utilized the StratifiedShuffleSplit, folds = 1000 evaluator to provide metrics on the classifier. I used the StratifiedShuffleSplit for validation because this cross validation object returns stratified randomized folds, and fits the structure for the unbalanced data.