- ***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 the project is to investigate the Enron data set and identify what Enron employees might have committed fraud based on the data. Machine learning allows to use/create algorithms, based on the findings found during the analysis of the data, to determine what characteristics makes someone a person of interest in the criminal case without the need to spend a significant amount of time investigating each person individually and with less chances of missing important information.
  For this dataset, some outliers were identified and removed such as some people missing all information in the financial features which doesn't help to investigate and find common characteristics among the personnel. There were several data points missing information which were set to zero.
  In total the data set contained information about 145 people plus a "Total" entry for each feature. From these 146 people, 18 were recognized as 'person of interest'.Each person had 21 features. The features with the most "NaN" or missing values were: loans advances(142), director fees(129) and restricted stock deferred(128).
- ***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"]***
  I ended up using the following features:
  - "Poi"
  - "Bonus"
  - 'Salary'
  - "Other"
  - "Exercised_stock_options"
  - "Restricted_stock"

  I removed these features from my original list:
  - 'Total_stock_value'
  - "Total_payments"
  - 'From_this_person_to_poi'
  - "To_poi_ratio"

- "From_poi_ratio"
- 'from_poi_to_this_person'

I used MinMaxScaler scaling in the fitting section in order to standardize features and chained it to pipeline. I created two new features: to_poi_ratio,from_poi_ratio to get a better idea on the ratio of the emails sent to/from a person of interest relative to the total number of messages someone sent/received. However, I decided not to include them since it didn't improve my algorithm and it might create data leakage. I used SelectKBest to select my features. I decided to use this feature selection as it seemed the most straightforward and the one that will provide me with enough information to in order to make my decision.
I started with all the original features and checked for precision and recall until it got to an acceptable level; however some of these features were included in another features, for example total_stock_value feature included exercised_stock_value and resctricted_stock. My algorithm performed better with exercised_stock_value and restricted_stock, and removing total_stock_value. Therefore I decided to remove total_stock_value. Below are the results for the final feature selection using SelectKBest.

Decision Tree Accuracy 0.825
precision 0.166666666667
recall 0.333333333333

SKbest features
1 feature poi (0.0022154421836)
2 feature bonus (1.41897215135)
3 feature salary (4.18698130863)
4 feature other (6.96579828364)
5 feature exercised_stock_options (8.99360333025)
6 feature restricted_stock (13.0844845791)

The performance of the original list of features including the new features:
Decision Tree Accuracy 0.813953488372
precision: 0.0
recall 0.0

The performance of all the original features without the new features:
Decision Tree Accuracy 0.790697674419
precision: 0.333333333333
recall 0.285714285714

- ***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 because it had better performance overall. I tried Naive Bayes and Support Vector Classification but Decision Tree gave me a better performance results. SVM accuracy is better but precision and recall were 0 so I didn't pursue this option.
GaussianNB:

training time: 0.002 s
GaussianNB()
testing time: 0.001 s
NB Accuracy 0.825
precision: 0.0
recall 0.0

Decission Tree:

training time: 0.004 s
testing time: 0.001 s
Decision Tree Accuracy 0.825
precision: 0.166666666667
recall 0.333333333333

Support Vector:

training time 0.011 s
testing time: 0.001 s
SVM Accuracy 0.925
precision: 0.0
recall 0.0

- ***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 parameters is the process of finding the best parameters to enable the algorithm to perform the best. If the parameters are not chosen wisely, then the algorithm would lead to poor performance.
  I used gridsearch for my algorithm and tweaked the dictionary passed in param_grid. I passed various options for these parameters and let gridsearch chose the best fit.

- ***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 splitting the data into 2 parts, one as a training set and the other as a testing set to estimate the performance of the algorithm in independent datasets.
  The training set is used to learn more about the model, tune parameters and select features.
  The testing set is used to test the algorithm set up in the training set to check if it works in an

independent data set. I used stratified shuffle split for cross validation since, per the definition in sklearn, it returns stratified randomized folds while still preserving the percentage of samples. Since this is a small dataset, using stratified shuffle split ensures that the same percentage of each class is drawn to accurately make a prediction. A common mistake is sending back the training set into the predict method since the algorithm remembers the prediction error *

- ***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 represents the percentage of the number of data points labeled correctly
  Accuracy: 0.85814
  Precision measures the ability to present only relevant items. Meaning, out of the data points that were marked as being relevant, what percentage is truly relevant.
  $Precision = truly\ relevant / (truly\ relevant + false\ relevant)$
  Precision: 0.50232
  Recall measures the ability to present all relevant items. Meaning, out of the data points that are relevant to the data, what percentage of these were marked as relevant.
  $Recall = truly\ relevant / (truly\ relevant + false\ irrelevant)$
  Recall: 0.40850
  Taking these definitions and results into account then the ability of the algorithm to correctly identify a person of interest (POI) with high probability is 0.50232. Out of the all POIs, the algorithm can identify most of the POIs with a probability of 0.40850.

*Stackoverflow