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”].**

Enron Corporation was of of world's major electricity, Natural gas and services company founded in 1985 as a merger between two companies. Enron employed approximately 20,000 staff and with acclaimed revenues of nearly $101 billion during 2000 and named America's Most Innovative Company for six years in a row. How ever at the end of 2001, it was revealed that its reported financial condition was sustained by institutionalized, systemic and creatively planned fraud in modern history. Enron corpus is the larges email database in the open that consists over 600,000 emails generated by 158 employees of the Enron Corporation and acquired by the Federal Energy Regulation Commission investigation after the company's collapse (https://en.wikipedia.org/wiki/Enron_Corpus).

The dataset used for this analysis contains 146 records and a total of 21 features (with 1 labeled feature (POI), 14 financial features, 6 email feature). Within these record, 18 were labeled as a Person Of Interest, POI's (labeled as 1) and the rest as non-POI's(labeled as 0). The goal of this project is to use this dataset , select /engineer features , train and test, and tune identify Machine Learning to identify person of interest that has heavily involved in the enron corporate financial fraud. TOTAL row was removed as it was simply a record totaling all of the financial statistics from the financial data. In addition to ‘TOTAL’ row, the ‘LOCKHART EUGENE E’ and ‘ THE TRAVEL AGENCY IN THE PARK‘ rows were removed as the former contains no values for all features and the later is not a real name.   The dataset also contains a lot of NaN values for most features (as depicted in the table below)

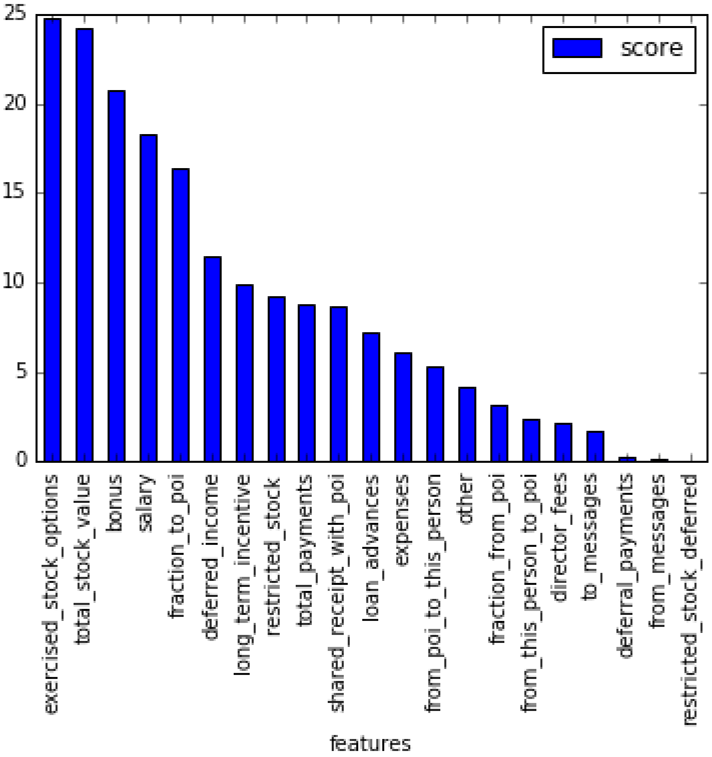
	Feature	**Number of NaNs**
0	bonus	64
1	deferral_payments	107
2	deferred_income	97
3	director_fees	129
4	email_address	35
5	exercised_stock_options	44
6	expenses	51
7	from_messages	60
8	from_poi_to_this_person	60

	Feature	**Number of NaNs**
9	from_this_person_to_poi	60
10	loan_advances	142
11	long_term_incentive	80
12	other	53
13	poi	0
14	restricted_stock	36
15	restricted_stock_deferred	128
16	salary	51
17	shared_receipt_with_poi	60
18	to_messages	60
19	total_payments	21
20	total_stock_value	20

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”, “properly scale features”, “intelligently select feature”].

A feature is a piece of information that might be useful for prediction quantity and quality of the features will have great influence on the performance of a model. Feature selection is one of the important tools in Machine Learning. In addition to the already available feature in the dataset, two feature were created , "fraction_to_poi’ and "fraction_from_poi’’ . This feature are so important in identifying person of interest as identify the fraction of emails send to and from poi’s as compared to total email messages, as they show who the poi was sending messages more frequently and who where sending more messages to the poi. The SelectKBest method was used to select feature for further use of feature in the machine learning process. This method gives score for features based on their relation which the target /response variable. The highest the score is the better the feature is to describe the response or target variable (the poi’s in these case). All the features were used in addition to the newly created features in the SelectKBest feature selection process except the ‘email address feature’. All the feature with the highest score are financial features, but fraction_to_poi which has the fifth highest score (shown below).

exercised_stock_options	25.097542
total_stock_value	24.467654
bonus	21.060002
salary	18.575703
fraction_to_poi	16.641707
deferred_income	11.595548
long_term_incentive	10.072455
restricted_stock	9.346701
total_payments	8.866722
shared_receipt_with_poi	8.746486
loan_advances	7.242730
expenses	6.234201
from_poi_to_this_person	5.344942
other	4.204971
fraction_from_poi	3.210762
from_this_person_to_poi	2.426508
director_fees	2.107656
to_messages	1.698824
deferral_payments	0.217059
from_messages	0.164164
restricted_stock_deferred	0.064984



From the score distribution graph of the features, there is relatively big difference in score between ‘fraction_to_poi’ and ‘deferred_income’ as compared to other score difference between successive feature scores. As a result I decided to use the features having score of 16 and above to be included in my final feature list. Accordingly the list of features I used for the machine learning process to identify poi’s are the list below. As these features contains ranges of values, the MinMaxScaler() function is sued to scale the feature values.

feature_list = ['poi','exercised_stock_options','total_stock_value','bonus','salary','fraction_to_poi']

The one of the new features have higher score(show above) than the features they are derived from, specially, ‘fraction_to_poi’ among the feature with high score (16.641707). The performance all the algorithms are assessed by including including the new feature and the original feature in the final feature_list by keeping other features the same. Compared to the original features the new feature have improved the model performance. For instance if with consider the Nave Bayes (GaussianNB) algorithm selected as best performing algorithm, with the original feature its accuracy, precision and recall are 0.85621, 0.49545 and 0.3265 respectively . When the new features is used its accuracy, precision and recall become 0.85629, 0.49545 and 0.3265 respectively.

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

The dataset was first split into train and test data where the test-size were set 0.3 , where 70 % of the dataset were in the training set and the remaining 30% of the dataset in the test set. Multiple machine learning algorithm have been trained and tested. The algorithms used are GaussianNB, LogisticRegression, SVC, LinearSVC, RandomForestClassifier, DecisionTreeClassifier, AdaBoostClassifier and SGDClassifier. The overall accuracy score of all the algorithm is relatively high over 0.8 except SGDClassifier and DecisionTreeClassifier with score of 0.33 and 0.71. Accuracy is not well suited for this for model evaluation.It assumes that labels varieties are equally distributed. Looking into precision and recall, most algorithm even with the highest score of accuracy didn't perform well. For some algorithms the precision and recall is so high as 1.0 for poi's and non-poi's and zero other times. Naive Bayes and AdaBoostClassifier algorithms show consistent and more reliable precision, recall and f1-score higher than 0.3 and with accuracy score of 0.880952380952 and 0.809523809524 respectively. Based on the scores of the two algorithm Naive Bayes performs better than AdaBoostClassifier and selected for further parameter tuning.

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? (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"]

In machine learning parameter tuning is using different parameter to train the algorithm for the objective of optimizing the performance of an algorithm. Tuning of parameters is important to ensure the model does not over-fit its training dataset. Parameter tunning was employed for SVC classifier, however this algorithm was not the one which exhibits best performance . GridSearchCV is a way of systematically working through multiple combinations of parameter tunes,cross-validating as it goes to determine which time gives the best performance.

```
param_grid = {
        'C': [1e3, 5e3, 1e4, 5e4, 1e5],
        'gamma': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1],
      }
```

5. 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 process in which a machine learning algorithm's performance that is trained in one dataset usually called the training dataset is test on separate dataset called test dataset.

It is mainly used in setting where the goal is prediction and one wants to estimate how accurately a model(an algorithm) performs in practice. In the first part of selecting an algorithm a 'train_test_split' function from sklearn is used to split the dataset into training sets (70%) and test sets(30%). Each algorithm is trained using the training set and validated using the test set. Aftrer selecting the best performing model (see question 3). the top performing model is again tuned and validated, now using a different function namely StratifiedShuffleSplit is used to split the dataset into training and testing sets. StratifiedShuffleSplit provides train/split indices to split data into train and test sets. This cross-validation object in this case/project stratified turns 1000 stratified randomized folds by preserving the percentage of samples for each case (eg., percentage of poi's – represented as 1's and non-poi's represented as 0's). Stratification helps during train/test split in such a way that the folds are selected so that each fold contains roughly the same proportions of class labels, that otherwise in the shuflsplit one class of label could be concentrated in one fold or the other, which could lead to over-fitting or under-fitting condition .

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.

Evaluation metrics assessed in this project are accuracy score, precision,recall and f1-score. Accuracy assumes that distribution of labels are 50:50 and do not work well on skewed labels. In such cases evaluation metrics such as precision, recall and becomes important. Precision is the fraction of retrieved instances that are relevant while recall is the fraction of relevant instances that are retrieved by the model or algorithm. Suppose that we have 100 poi's (1's in this case) in total and of these the algorithm identify 30 of the as poi's but only 10 of them are real poi's the precision is given as 10/30 (0.1) and recall is 10/100 (0.01). The f1-score is simple the weighted average of recall and precision (i.e, f1-score = 2*(recall*precision/recall+precision)). The precision value for the final model is `0.49545` which means that among the poi's predicted by the algorithm as poi's are 49.545 percent and the other false positives. And the recall is `0.32650` which means that among the total poi's in the dataset the algorithm was able to identify only 32.65 percent as true positives.