Identify fraud from Enron email

Also mentioned in iPython Notebook Enron Data.ipynb

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"]

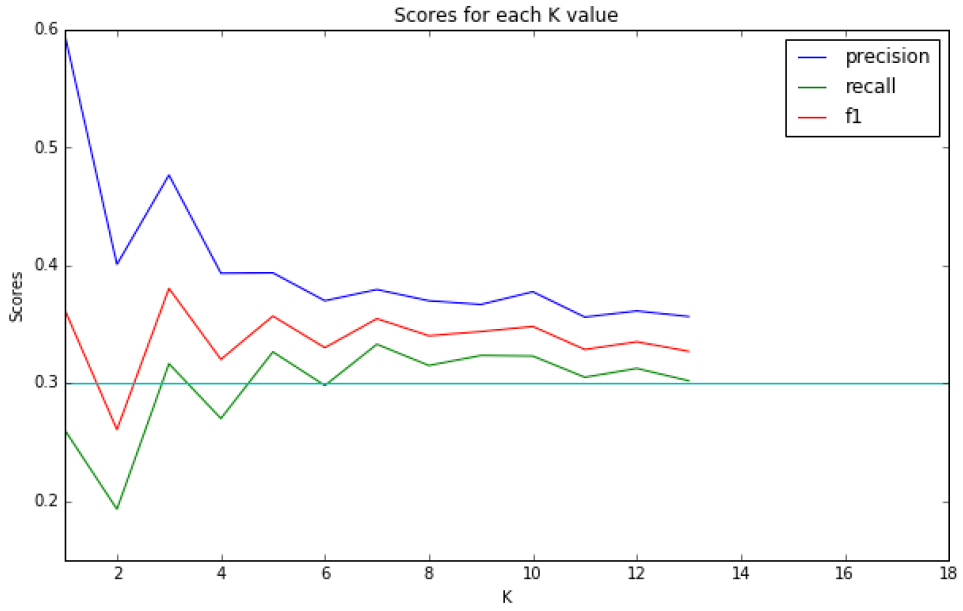
The goal of this project is to identify the person of interest, which means individuals who were indicted, reached a settlement or plea deal with the government, or testified in exchange for prosecution immunity in Enron scandal. Through machine learning we can find the person of interest based on financial and email data made public as a result of the Enron scandal.
As seen from the pdf of financial data the row 'TOTAL' is outlier. There are many entries in which the payments column is empty and only the stock values are filled, these can also be treated as outliers because here our concern is to identify the person of interest. However, if our concern wasn't person of interest this data can meaningful because these can be fake people added by person of interest intentionally. For this project I will be replacing these NaN values with zero.

Total Number of Insiders: 146
Number of POI's: 18
Fraction of examples that are POIs: 0.123287671233
Number of features: 21

As the number of POI's are less, I have used stratified shuffle split which maintains the ratio between POI and non POI while training as well as testing the model.
As there are more non POI's than POI accuracy will give wrong result because accuracy is number of samples which are predicted correctly divided by total number of samples, most of them are non POI's so it will be predicted correctly only. Therefore, evaluation matrix such as precision and recall will be better.

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 importance 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"]

```
In [87]: runfile('/Users/aishwaryapatil/Downloads/ud120-projects-master 2/
final_project/poi_id.py', wdir='/Users/aishwaryapatil/Downloads/ud120-projects-
master 2/final_project')
Reloaded modules: feature_format, tester
K: 1
        exercised_stock_options
K: 2
        total_stock_value
K: 3
        bonus
K: 4
        salary
K: 5
        fraction_of_messages_to_poi
K: 6
        deferred_income
K: 7
        total_payments
K: 8
        shared_receipt_with_poi
K: 9
K: 10
        loan_advances
K: 11
        director_fees
K: 12
        fraction_of_messages_from_poi
K: 13
        to_messages
K: 14
        deferral_payments
```




```
In [6]:
```

Used SelectKbest to calculate feature importance and ran the classifier on each value of K.
Plotted graph to check which value of K is best. Found k=3 to be the best.
(I have commented the code for calculating the best k and for plotting it)
For email features I engineered two new fetaures fraction_of_messages_to_poi, fraction_of_messages_from_poi , just using from_message and to_message didn't make sense because goal is to identify the person of interest , if the fraction number of messages from poi to a particular person(vice versa) is high then the chances are more, that the other person is person of interest.
Performed GridSearch CV and noted the value of best parameters. Applied Principal Component Analysis, as there might be some features which were related, in order to not miss out on collinearity.

Using fraction_of_messages_from_poi and fraction_of_messages_to_poi the precision and recall didn't change because according to feature importance using selectkbest exercised_stock_options , total_stock_value, bonus were of higher importance.

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

I tried using KNearestNeighbor first, which gave precision of 0.27766 and recall of 0.26774.
Then, performed Logistic Regression which gave a high precision 0.42534 and recall of 0.27021 .
Decision Tree Classifier gave better result with precision of 0.34023 and recall of 0.35350.
(I tried other classifiers also but they were taking too much time to run)

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"]

Classifiers have parameters which affect the performance of the model. To tune parameters means to adjust the parameters of the classifier in order to give the best performance.

The parameters involved with my final classifier, that is Decision Tree classifier, were min_samples_split, splitter and random state. Used GridSearchCV in order to find the best parameters.

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 item: "validation strategy"]

After training the model with a dataset, it is important to test the model with a different dataset, this is known as validation. Classic mistake one can make is using the training data for testing the model by doing so the model overfits the data(high variance) and if some other unseen data is used for testing, the error rate will be high. It is important to either spilt the data into different parts one for training and one for testing, or use a different dataset for both the task. Used StratifiedShuffleSplit for splitting my data with 1000 iterations.

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. [relevant rubric item: "usage of evaluation metrics"]

As there are more non POIs than POIs in the dataset, evaluating the model using accuracy will give wrong result as accuracy is number of samples(both POI and non POI) predicted correctly divided by total number of samples.

Precision and recall will be better metric here. Precision and recall give only the total number of POIs which were predicted correctly. Recall is proportion of samples that were actually POI and were predicted by model to be POI. Precision is proportion of samples that were predicted as POI and actually were POI.

With Decision Tree Classifier, got a precision of 0.34023 and recall of 0.35350

Type of validation used?

Used Stratified Shuffle split as the number of POIs are less, Stratified Shuffle maintains the ratio between POI and non-POI while training as well as testing the model.
Cross Validation just divides the samples into folds , there might be a possibility that first half has just Non POIs and no POIs , the model might predict the accuracy to be high due to this. It is important to shuffle as well as maintain the actual ratio between while training the model , hence using Stratified Shuffle Split makes sense.