Identifying Fraud from Enron Emails and Financial Data

INTRODUCTION:

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, there was a significant amount of typically confidential information entered into the public record, including tens of thousands of emails and detailed financial data form its executives.

In this project I will try to use machine learning with sklearn to train my model to find if a person might be involved in the fraud or not. Our sample data is collection of data from the email and financial information of the people who has committed the fraud and people who did not. The sample will be put into various machine algorithms and the results are interpreted and the best model is chosen.

Summarize:

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 goal of the project is to use the information from the financial and email data from the sample of both POI and non-POI to build a predictive model which is trained using Sklearn machine learning algorithms to find if a new person is really a person who might be involved in the fraud. The POI (Person of Interest) is a feature in the data set which identifies if a person convicted or not. This feature is the primary key or dependent variable I will be using to train my model by trying to predict if a person is a person of interest or not. The data set has 146 person and each person has 21 different features from financial and email together. There are a total of 18 POI in the data set.

   Not all features for all the person has value there are a lot of missing value. This missing value (NaN) are transformed into 0 for convenience. I was able to find outliers one such outlier is the “TOTAL”. The “TOTAL” was a spreadsheet error it didn’t have to say anything about the person instead it was just a sum total of different financial feature. I was able to spot it visualizing the financial data on scatter-plots and removed it.

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?

   Two new feature was add, ‘fraction_to_poi’ and ‘fraction_from_poi’ these feature are the fraction of amount mail from this person to the total mail this person has to everyone and amount of mail send a poi sent to a person to the total mail this person has received from everyone respectively. The intuition is that a person and the poi have communicated a lot then this person might also be a poi. I was able to do feature selection using `feature_importances_` attribute in the decision tree and able to rank the feature which are most useful.

The table has the different feature and its importance

Feature	importance
Expenses	0.236583012
Bonus	0.232361309
exercised_stock_options	0.16875873
from_messages	0.10971461
fraction_to_poi	0.09825626
restricted_stock_deferred	0.083011583
Other	0.071314496
to_messages	0
deferral_payments	0
deferred_income	0

All the other feature which are not shown in the table have importance close to zero as such there are not important features. I have tried at least 4 model and I think scaling is very important since we have two different units one financial as dollars and email as count thus feature scaling was done using `StandardScaler`. There is exception too in case of tree based algorithm like decisions tree where scaling is of no use I haven't rescaled my features. The finial feature I will be using are "other", "expenses","fraction_to_poi",'shared_receipt_with_poi',"exercised_stock_options" these are handpicked to get more precision and recall.

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

   The following algorithms was used in the project Decision-Tree, PCA with SVC, Random-Forest, Voting classifier with decision tree and random-forest.

Algorithm	Precision	Recall
Decision-Tree	0.45	0.42
PCA with SVC	0.6	0.5
Random-Forest	0.46	0.21
Voting classifier	0.4	0.37

   Initially PCA with SVC seem a good model but applying PCA to a very small data size may over fit our model and a detail look after transforming with PCA using visualization (graph in notebook) PCA is not doing a good job in fitting, as more POI and non-POI are together thus high precision and recall from the SVC might be due to overfitting. The Decision Tree is more opt algorithm from the precision and recall as I end up using it.

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?

   In case of SVC the turning parameters are gama and c varying these parameters can have impact on the model. It is not always the default value or any specific values for this parameter will result a good model. The `gridsearchcv` was used in this case and the range of number was plugged into the model to get the best fit. If you don't turn this parameter you might compromising yourself from having a best model for an average model.

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"]
   Validation is method to generalize our machine learning algorithm. The general principle of validation is to develop a model without bias so that when the model is introduce with unseen data it pretty much preform the same way as it does with our train data. The reason for doing splitting or k-fold the data is, we can validate our metrics such as "accuracy" from testing our test data which was spited from the data-set as that this test data is new to the model. I feel a classic mistake is over-fitting this might happen if you train all your data points and try to test with the same data sets. This mistake will give you good accuracy for your model but when a new data is introduce then the performance will be pretty bad. I validate using cross validation with 8 fold in `gridsearchCV` and train_test_split to split the data set to train and test dataset with test_size=0.4.

6. Evaluation metrics:
   Accuracy is not a good measure when you have a skewed classes and in enron we have just 18 poi when compared with 146 poi. The metric I prefer is precision and recall. Precision is the ration of true positive (poi) to the total positive (poi) prediction. If your model has high precision and it predict someone as poi then the chances of that person being poi is very high. Recall is the ratio of true positive to total positive records (total number of poi labeled in the data set). Recall is related to sensitivity. If your recall is high you're able to predict more number of poi but chances are the person might not be really a poi.

   Conclusion:
   The Decision Tree we have used have a very decent precision and recall. I prefer precision over recall because if our model make some false prediction it is very easy for a person using our model to stop believing and even stop using my model. So I believe to have model with high precision at it initial testing.

   Reference:

   1. Sklearn documentation.
   2. Y-hat .com
   3. http://sebastianraschka.com.
   4. Udacity.com