Machine Learning project:

Introduction:

In the machine-learning project, we have used Enron data set to identify fraud from Enron emails. Enron is a big energy company in the states. It went through bankruptcy and it ends by a big scandal by fraud. Further, Many employees have sentenced to Jail.

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:

The goal of this project is to use machine learning to build predictive model to identify the person of interest in this case who might be involved based on different features like emails, bonus and other features.

Dataset:

The dataset contains 146 poi and more than 18 features including: 'salary','deferral_payments', 'total_payments', 'loan_advances' and other features . Further, the poi, 18 of them are person of interest and 128 are not poi.

Outlier:

We visualize the dataset by using scatter plot, and we found there is an outlier called ‘total’ and we removed it from our dataset. So , the total now is 145. Moreover, there are many missing values which I handle before using my classifiers because I could not use them with removing Nan from the dataset.

Feature:Nan

poi 0
salary 51
deferral_payments 107
total_payments 21
loan_advances 142
bonus 64
restricted_stock_deferred 128
deferred_income 97
total_stock_value 20
exercised_stock_options 44
long_term_incentive 80
restricted_stock 36
director_fees 129
to_messages 60
from_poi_to_this_person 60
from_messages 60
from_this_person_to_poi 60
shared_receipt_with_poi 60

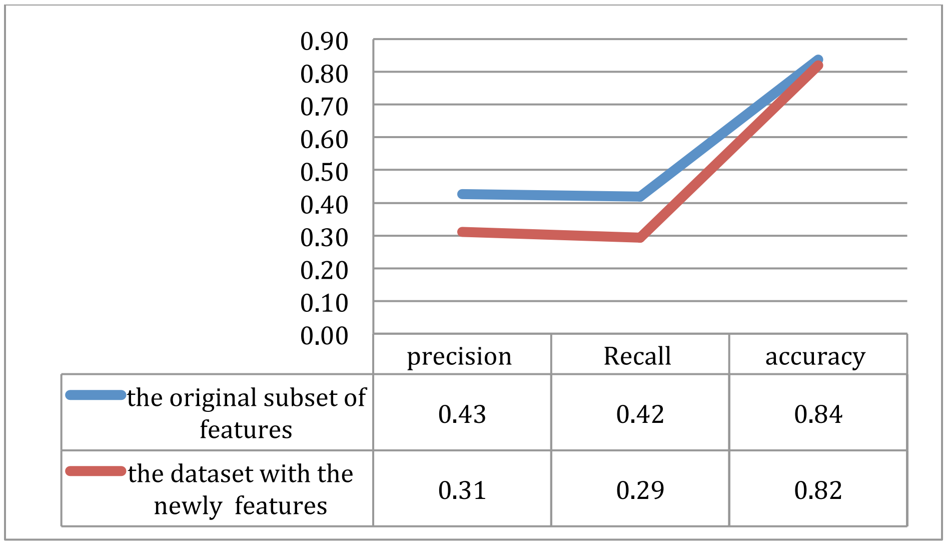
2. What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what feature 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"]

Features selection: we have used the selectKbest module and the manual selection from scikit-learn in the selection process. Furthermore, we have sort the features list to get its score to select the most high scores from all the below features if want to select manually. Nevertheless, we have excluded the email_address feature our list.

In addition , I did not use any scaling because I used Random Forest algorithm and GaussianNB algorithm, which do not any scaling for the data.

For the feature Engineering, I have created two features to calculate the ratio of emails that sent from poi(`'from_poi_ratio')` and the other to calculate the ratio emails that were sent to poi `('to_poi_ratio').` However , we created these two new features the score of importance of these features not very high like other features especially when we have used two feature lists one with new features and the other without feature list. We found out the performance and recall was very high without the new features that were created.

We train a classifier with original subset features and another with dataset the new features that were created and the results were the original features dataset has performed better than the dataset with new features as can be seen in the below graph:



	precision	Recall	accuracy
🟦 the original subset of features	0.43	0.42	0.84
🟥 the dataset with the newly features	0.31	0.29	0.82

The feature list score with new features:

exercised_stock_options	**25.09754153**
total_stock_value	24.46765405
bonus	21.06000171
salary	18.57570327
to_poi_ratio	16.64170707
deferred_income	11.59554766
long_term_incentive	10.07245453
restricted_stock	9.346700791
total_payments	8.866721537
shared_receipt_with_poi	8.746485532
loan_advances	7.242730397
from_poi_to_this_person	5.344941523
from_poi_ratio	3.210761917
from_this_person_to_poi	2.426508127
director_fees	2.107655943
to_messages	1.698824349
deferral_payments	0.21705893
from_messages	0.164164498
restricted_stock_deferred	0.064984312

The original dataset features with score:

exercised_stock_options	25.097541528735491
total_stock_value	24.467654047526398
bonus	21.060001707536571
salary	18.575703268041785
deferred_income	11.595547659730601
long_term_incentive	10.072454529369441
restricted_stock	9.3467007910514877
total_payments	8.8667215371077717
shared_receipt_with_poi	8.7464855321290802
loan_advances	7.2427303965360181
from_poi_to_this_person	5.3449415231473374
from_this_person_to_poi	2.4265081272428781
director_fees	2.1076559432760908
to_messages	1.6988243485808501
deferral_payments	0.2170589303395084
from_messages	0.16416449823428736
restricted_stock_deferred	0.06498431172371151

As can be seen the above tables shows the features score for the features list with and without the new features.

In addition: The features that we selected manually are which select based on its score that shown above:

['poi','exercised_stock_options','total_stock_value','bonus','salary','to_poi_ratio','deferred_income','long_term_incentive']

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”]

I ended by using Random forest which gave good results in terms of precision and recall. Further , I have used GaussianNB but the performance was very bad.

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”]

Tuning the parameters means the process of impacting on the model by manipulating some parameters that we created to perform best and give better results from the performance and recall perspective.

First algorithm that I used was GaussianNB, it does not have any parameters to tune. Second, for Random Forest algorithm I have tuned the parameters min_samples_split and n_estimators to get better performance as can be seen as below.

min_samples_split=8	n_estimators=25	Accuracy: 0.84407 Precision: 0.44804 Recall: 0.39450	
min_samples_split=10	n_estimators=50	Accuracy: 0.84836 Precision: 0.46768 Recall: 0.44500	

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 machine-learning technique, which is used to evaluate and compare machine-learning algorithm by splitting the data into two datasets, one for training and the other one for testing purposes. The classic mistake that usually happens in this case is overfitting. To avoid overiftting , split the data into training and testing dataset and use testing dataset at the end after training your model well.

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"]

   First, I have used three evaluation metrics including performance , precision and Recall in my project and the results for the Random forest algorithm by using the train_test_split as below:

Accuracy: 0.904761904762
Precision = 0.666666666667
Recall = 0.666666666667
Second, I have used the tester.py which uses StratifiedShuffleSplit function which you recommended my in your review to use in my code and the result were as below:

```
Accuracy: 0.83671	Precision: 0.42712	Recall: 0.41900
```

As can be seen above the results differ for these two evaluations. T he three parameters values were low when we use the test.py and it makes around Total predictions: 14000.

- Precision is “True Positive(838) / (True Positive(838) + False Positive(1124)). Out of all the items labeled as positive, how many truly belong to the positive class (person of interest)”, so my algorithm predicts the person as a person of interest(poi) about %42 of the time this person is in reality the poi[2]

- Recall is “True Positive (838) / (True Positive(838) + False Negative(1162). Out of all the items that are truly positive, how many were correctly classified as positive. Or simply, how many positive items were 'recalled' from the dataset”, so my algorithm was able to recognize a person of interest (poi) about %41 of the time out of all POIs [2].

Refernces:

1. https://www.quora.com/What-is-the-definition-of-precision-in-machine-learning
2. udacity forums and lessons.