Enron Fraud Detectors using Enron Emails and Financial Data.

Question 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 this project is to develop a machine learning algorism to identify the person of interest (POI) based on the financial and email data from Enron corpus, a public dataset produced by US Federal Energy Regulatory Comission during its investgiation of Enron company. The dataset contains email and financial data of 146 people, most of which are senior management of Enron.

The original Enron dataset final_project_dataset.pkl contains **146** data points, with **21** features. The features in the data fall into three major types, namely financial features, email features and POI labels. Among all the people, there are **18** poi in total, the remaining **128** are non-poi.

Financial features: ['salary', 'deferral_payments', 'total_payments', 'loan_advances', 'bonus',

'restricted_stock_deferred', 'deferred_income', 'total_stock_value', 'expenses',

'exercised_stock_options', 'other', 'long_term_incentive', 'restricted_stock', 'director_fees']

(all units are in US dollars)

Email features: ['to_messages', 'email_address', 'from_poi_to_this_person', 'from_messages',

'from_this_person_to_poi', 'shared_receipt_with_poi'] (units are generally number of emails

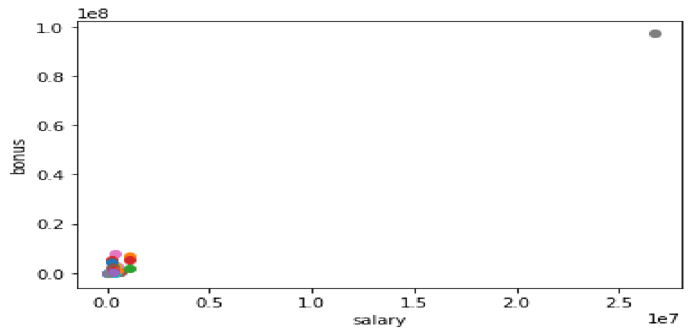
messages; notable exception is ‘email_address’, which is a text string)

POI label: [‘poi’] (boolean, represented as integer)

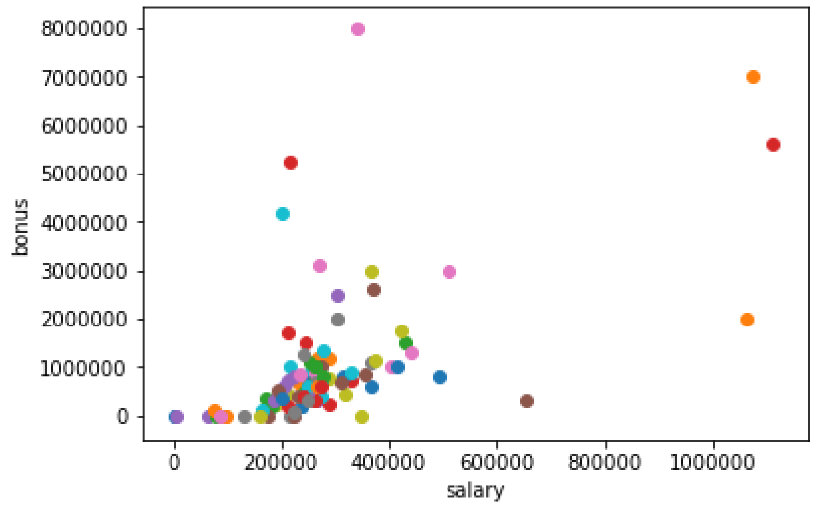
All features contain NaN value except for **‘poi’**. For example, among 146 data points, **‘loan_advances’** contains **142** NaN values, ‘director fees’ contains **129** NaN values, ‘restricted_stock_deferred’ contains **128** NaN values, etc.

Outliers:

I plotted feature 'salary', 'bonus' in a scatter plot. One data point was found as an outlier. After checking the original financial document, this data point represents the total value.



The outlier **'TOTAL' in the dataset was removed.**



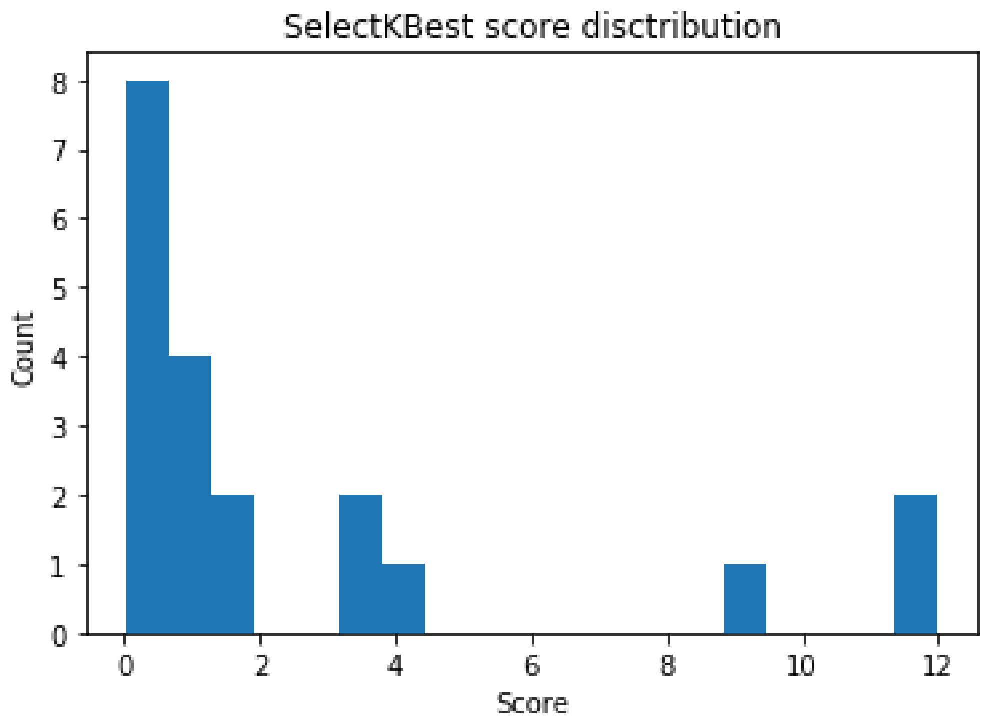
Also, another outlier, **'THE TRAVEL AGENCY IN THE PARK'** was found and removed. It is not a person's name. Finally, the dataset contains 144 people and 21 features.

Question 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.

First I looked the pdf document and found that **'email_address'** is text string that related to person's name, so I excluded it. And it is found that the value of **'loan_advances'** feature is all **NaN**, so this feature was excluded.

Two features were created and added to the feature list, **'from_poi_to_this_person_ratio'f** and **'from_this_person_to_poi_ratio'**. The two features represent the fraction of emails that person sent or received from poi.

I then investigated the importance of each feature through calculating the SelectKBest score method. Among the 20 features, 3 of them obtained scores over 9, and 6 of them have scores over 3, the rest scores are below 2. Considering that scores below 2 were too low to be considered significant, the 6 features that have score over 3 were selected as final features. Among these 6 features, **'from_this_person_to_poi_ratio '** is the new feature created and all other features belong to financial features.



Features	Scores
exercised_stock_options	11.988503931743294
'total_stock_value'	11.358885105643221
from_this_person_to_poi_ratio	9.4008335426221254
expenses	3.9753908186478673
salary	3.3795714918901285
deferred_income	3.27499666260915

Final_features_list = ['poi', 'exercised_stock_options', 'total_stock_value', 'from_this_person_to_poi_ratio', 'expenses','salary', 'deferred_income']

Missing values in the features were investigated. Final results may be influenced by these missing values.

Features	Number of Missing Values	Percentage of Missing Values
poi	0	0
exercised_stock_options	43	29.86%
'total_stock_value'	19	13.19%
from_this_person_to_poi_ratio	58	40.28%
expenses	50	34.72%
salary	50	34.72%
deferred_income	96	66.67%

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

Due to the size of data is small and the data are imbalance (only **144** data points and **18** of them are poi), the StratifiedShuffleSplit method can deal with this situation and was used to repeatedly calculated the performance of each classifier for 1000 times. 6 classifiers were tried and evaluated.

clf_names = ["Naive Bayes","SVM Linear", "SVM RBF","Decision Tree","KMeans","Random Forest"]

Classifiers	Accuracy	Precision	Recall	F1 Score
Naive Bayes	**0.87273**	**0.53031**	**0.39800**	**0.45473**
SVM RBF	NA	NA	NA	NA
Decision Tree	0.83247	0.37270	0.37550	0.37410
KMeans	0.80373	0.28289	0.30750	0.29468
Random Forest	0.86387	0.47597	0.20800	0.28949

After comparison, the Naive Bayes classifier has the highest accuracy, precision, recall and F1 score, however, the Naive Bayes does not have parameters to tune, so that I choose Decision Tree classifier, which has second highest F1 score to be further tuned.

Question 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).

Different classifiers have various of parameters that can be tuned to improve the performance according to different datasets and tasks. The goal for this task is to identify the poi in the dataset. Ideally, we want both precision and recall. Only high precision may lead to low recall, which will miss many targets, accordingly, only high recall may lead to low precision, which will include lots of false positives. So I want a higher F1 score, which balance the recall and precision.

I chose the decision tress to further tune the classifier, the parameters include “**max_depth**” and “**max_features**”. “**max_features**” determines how many features the tree is randomly assigned. The smaller, the less likely to overfit, but too small will start to introduce under fitting. Values include: [‘sqrt’, ‘log2’, None] were tested. ‘sqrt’ means to use the sqrt(original number of features), ‘log2’ means to use $\log_2$(original number of features) and None means to use original number of features. ‘sqrt’ and ‘log2’ got same results and both better than None. Since in this case the original number of features = 6, and sqrt(6) is nearly equal to $\log_2(6)$, so the results are same for this two parameters.

“**max_depth**” is the number of depth for the decision tree, in general, the larger number leads to better result, however, it is more likely to overfit. Values include: [5,10,20,40, None] were tested. None means as much depths as needed.

max_features = 'sqrt'

max_depth	Accuracy	Precision	Recall	F1 Score
5	**0.84293**	0.39039	0.31700	0.34989
10	0.83007	0.36350	0.36550	0.36450
20	0.83067	0.36459	0.36350	0.36405
40	0.83693	0.38611	**0.37800**	0.38201
None	0.83887	**0.39180**	0.37750	**0.38452**

max_features = 'log2'

max_depth	Accuracy	Precision	Recall	F1 Score
5	**0.84000**	0.37294	0.29350	0.32848
10	0.83100	0.36359	0.35650	0.36001
20	0.83633	**0.37969**	0.35900	**0.36906**
40	0.83053	0.36188	0.35500	0.35840
None	0.83133	0.36750	**0.36750**	0.36750

max_features = None

max_depth	Accuracy	Precision	Recall	F1 Score
5	0.82867	0.34561	0.31900	0.33177
10	0.83080	0.36891	0.37850	0.37364
20	**0.83387**	**0.37941**	**0.38700**	**0.38317**
40	0.82967	0.36483	0.37450	0.36960
None	0.83340	0.37679	0.38150	0.37913

After comparison, when **max_features = 'sqrt'** and **max_depth = None**, best F1 score can be obtained.

Question 5: What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis?

The aim of validation is to assess how the results of a classifier will generalize to an independent dataset. In machine learning and prediction, we need to evaluate the performance of the model using different performance metrics. It is important to do cross validation. Usually the dataset is divided into training set and testing set to avoid overfitting.

Cross validation were performed by splitting the data into training and test dataset in feature selection, algorithm selection, and algorithm tuning. Since the size of data is small and the data are imbalance, I used stratified shuffle split cross validation to evaluate the performance of classifiers.

Question 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.
The **accuracy, precision, recall,** and **F1 score** were calculated to evaluate the classifier. After the tuning the final results are **Accuracy:** 0.84293, **Precision:** 0.39180, **Recall:** 0.37750 and **F1 score**: 0.38452.

The accuracy is the percentage of right prediction in all predictions, the accuracy of the final classifier is good, over 83% predictions are correct. Precision is the percentage of true positive predictions in all positive predictions, which means among all people predicted to be poi, about 37.29% of them are true poi. Recall is the percentage of the true positive predictions in all true cases, which means among all poi in the dataset, 36.45% of them were identified by the classifier. So the precision and recall are not very ideal and could be further improved. However, the extremely limited amount of data is a major reason for the relatively low precision and recall.

If new features were not included, the final results are **Accuracy:** 0.82313, **Precision:** 0.33418, **Recall:** 0.32900 and **F1 score**: 0.33157. These results show that the created new features are useful to increase the overall prediction results.

Decision tree	Accuracy	Precision	Recall	F1 Score
With new features	0.83353	0.37289	0.36450	0.36865
Without new features	0.82313	0.33418	0.32900	0.33157