**Machine learning of person of interest in the Enron scandal**

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?

Goal: To identify person of interest (poi) involved in the Enron Scandal using financial and email data. Machine learning provides a powerful prediction tool bases on features by building the classifier in the training dataset. This classifier could tell us whether a person is poi or not.

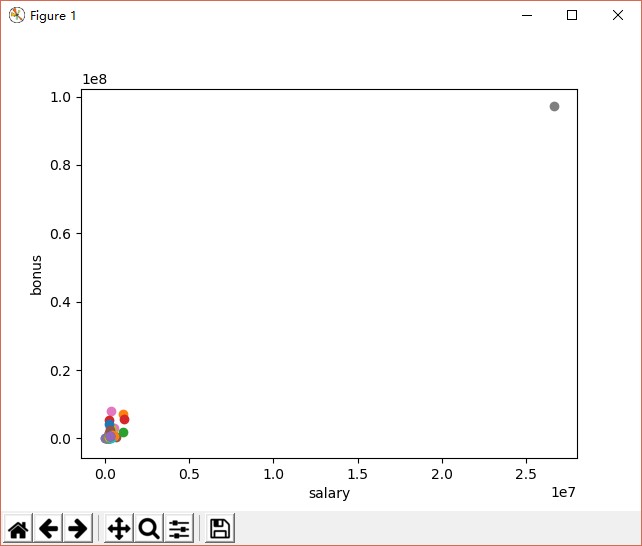
Dataset:

The original dataset contains data for 146 data points with a total of 21 features. The features can be categorized into 3 types: financial, email and poi label.

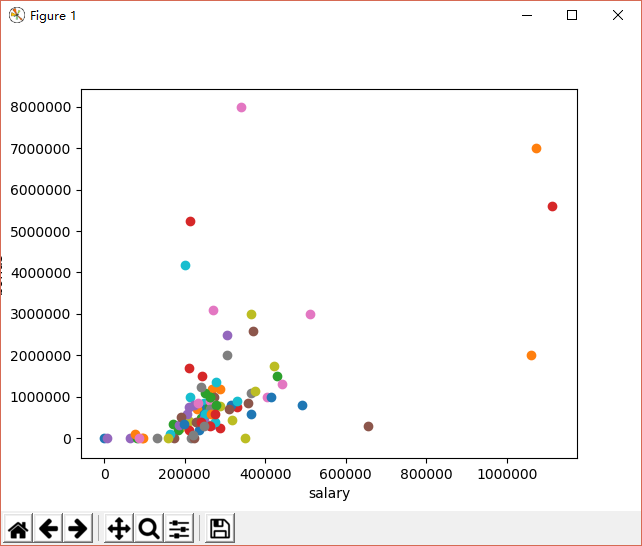
* financial features: ['salary', 'deferral\_payments', 'total\_payments', 'loan\_advances', 'bonus', 'restricted\_stock\_deferred', 'deferred\_income', 'total\_stock\_value', 'expenses', 'exercised\_stock\_options', 'other', 'long\_term\_incentive', 'restricted\_stock', 'director\_fees'] (all units are 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)

Outliers:

The salary and bonus are plotted in a scatter plot as these are the two most evident financial features. One clear outlier is found as shown below, which is the total value.



After removing this point as shown below, another 4 points are prominent. However, they are not too far and these points belong to top person in the company like CEO, etc after checking the names. Hence, these 4 points are remained in the analysis. In addition, after looking at all the names of the data points, there is one point called ‘The travel agency in the park’, which obviously is not a person’s name. Hence this point is removed, too.



Therefore, there are 144 data points with a total of 21 features. However, only 18 persons (12.5%) are labeled as poi; it is challenging to build a classifier on such skewed dataset. The stratified shuffle split is used to evaluate the performance.

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”]

The ‘total\_payments’ and ‘total\_stock\_value’ are the sum of other variables. Hence they are removed for the analysis. The ‘email address’ is a string text and is unique to person (no correlation with poi); hence this is removed, too.

Two features are created as ‘from\_poi\_to\_this\_person\_ratio’ and ‘from\_this\_person\_to\_poi\_ratio’ to capture the percentage of the emails related to this person. The ratio is more meaningfull than the absolute value because a person with higher level might have a larger number of emails. Higher ratio may suggest a higher interest, hence these 2 ratios are added as new feature.

The features are scaled with MinMaxScalar, as it is crucial for the support vector machine(SVM) algorithm for the training process. The selectKBest are used to select the features, as we know the total number of features and we can pick up those important ones for the analysis. The best selector is shown as below:

Pipeline(memory=None,

steps=[('features', FeatureUnion(n\_jobs=1,

transformer\_list=[('univ\_select', SelectKBest(k=2, score\_func=<function f\_classif at 0x12C28F70>))],

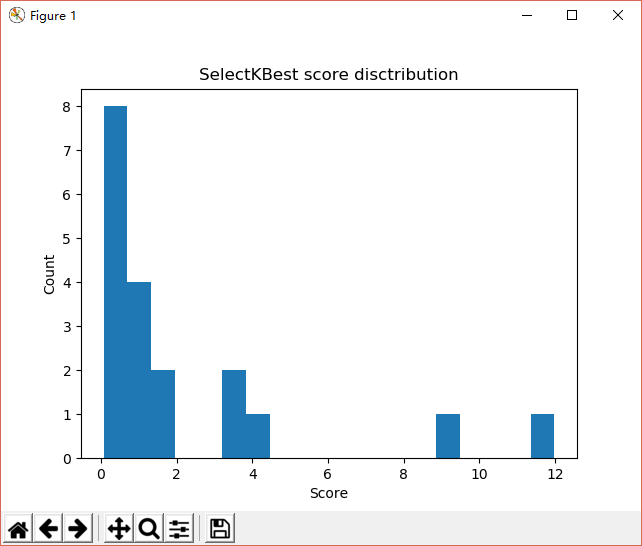
transformer\_weights=None)), ('svm', SVC(C=10, cache\_size=200, class\_weight=None, coef0=0.0,

decision\_function\_shape='ovr', degree=3, gamma='auto', kernel='linear',

max\_iter=-1, probability=False, random\_state=None, shrinking=True,

tol=0.001, verbose=False))])

It suggests that there are 2 most important features. However, if the importance for each figure is plotted with histogram as shown below, It could be found that there are 5 features with score>3.



These 5 features are: ‘exercised\_stock\_options’ (11.988), ‘from\_this\_person\_to\_poi\_ratio’ (9,400), ‘expenses’ (3.975), ‘salary’ (3.379), and ‘deferred\_income’ (3.275). However, the deferred income may be related to other type of incomes. Hence I decide not to use this feature.

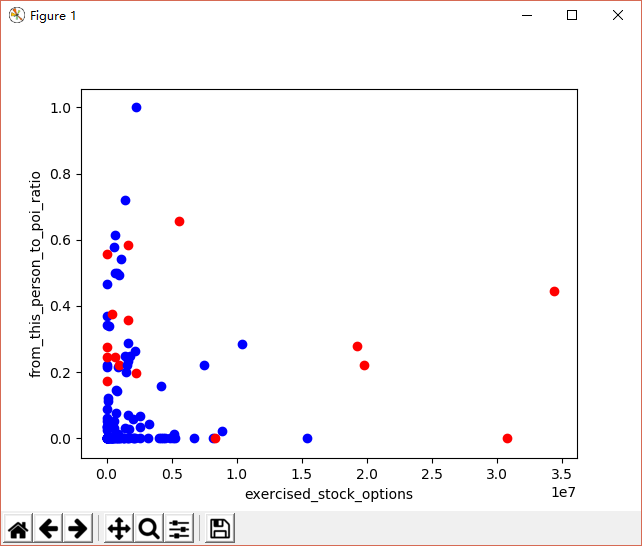
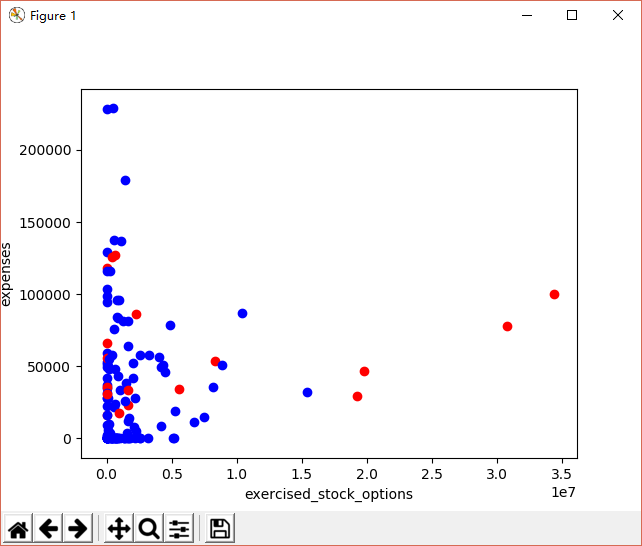
As a result, the final selected features are:

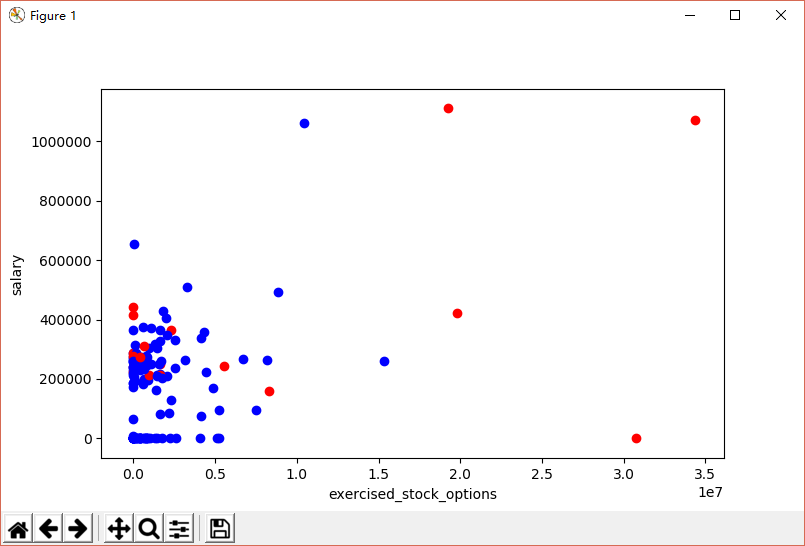
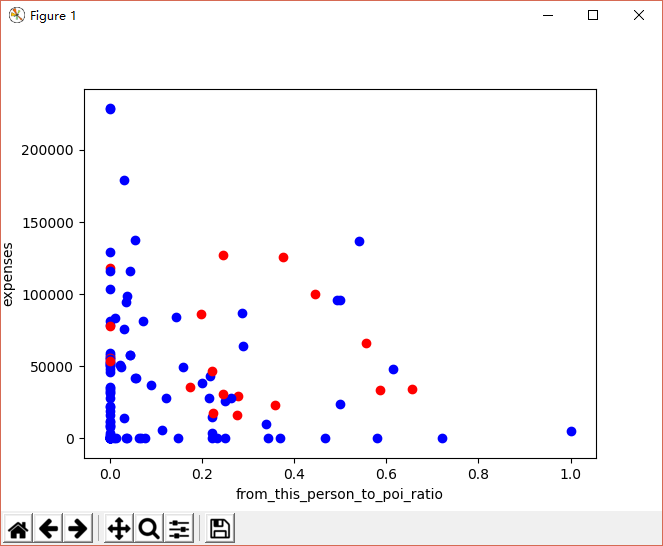
features\_list = ['poi', 'exercised\_stock\_options', 'from\_this\_person\_to\_poi\_ratio', 'expenses','salary']

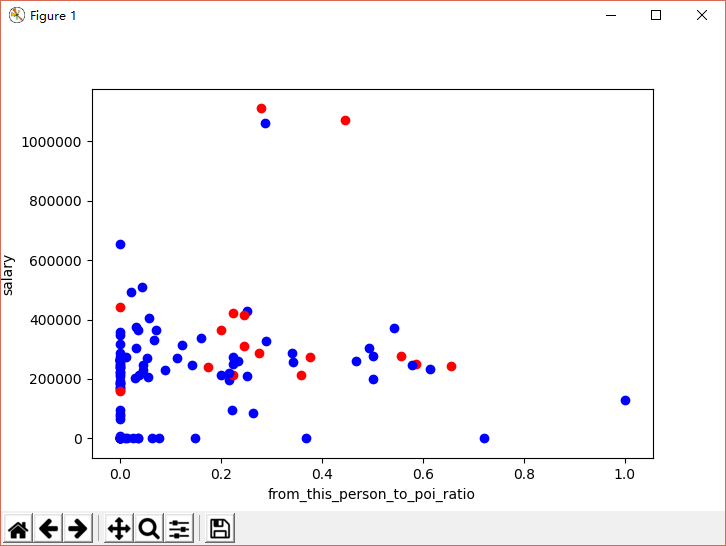
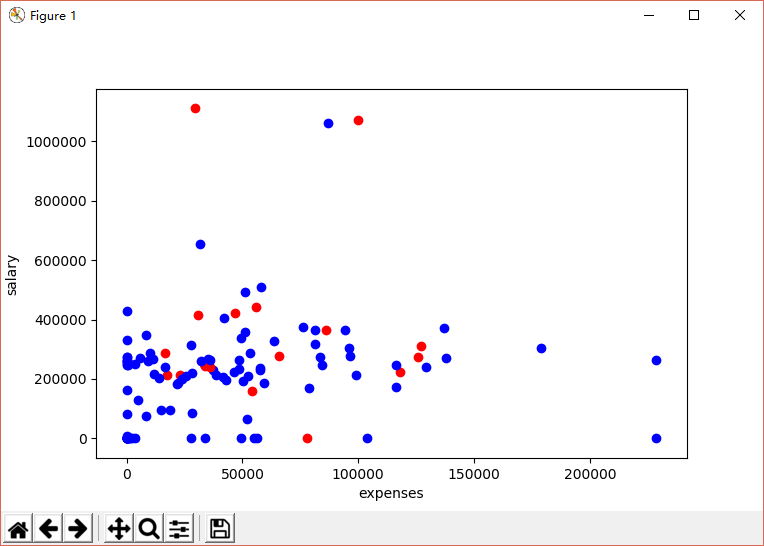
There are two financial features and one NEW email feature.

However, it is further found that with 144 data points and the selected features, some values are ‘NaN’, which may greatly affect the performance because any data point with non-valid value is not included in the training and testing process.

If the pairs of features are plotted with poi labeled in red, it could be seen that exercised\_stock\_options may separate the poi clearly, however, no further feature is able to draw a simple decision line. Therefore, the training purpose is necessary to contain all the important features.

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”]

Because of the limited and skewed dataset, the Stratified Shuttle Split is used to split the training and testing data with replaced repetition of 1000 times. The metrics for the naïve Bayes, linear SVM, RBF SVM, decision tree based learning process are shown below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm | Accuracy | Precision | Recall | F1 score |
| Naive Bayes | 0.64186 | 0.25575 | 0.78900 | 0.38629 |
| Linear SVM | 0.81164 | 0.33890 | 0.33500 | 0.33694 |
| RBF SVM | 0.82064 | 0.36630 | 0.35000 | 0.35796 |
| Decision Tree | 0.75686 | 0.30281 | 0.539 | 0.38777 |

Naive Bayes shows the highest F1 score but the lowest accuracy; it is doing well on identifying a potential target (poi in this case), although it may be a false positive.

RBF SVM performs better than linear SVM in all metrics. In addition, RBF has the highest precision within these 4 algorithms, which indicates that when it flags a potential target, it is most likely to be true.

Decision tree shows modest accuracy and precision, and second highest recall and F1 score.

I will further optimize the parameter for RBF SVM, in order to have a higher F1 score.

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”]

Machine learning algorithm has many parameters and the result could be tuned for a given problem with different parameters.

Since our goal is to identify poi, it is better to improve both the recall and precision score; therefore we could identify poi as many as possible with high precision. Due to the skewed dataset here, the accuracy is not the best choice because we may pick more non-poi correctly but overlook poi. If precision is monitored alone, we may be too cautious in the prediction and let poi escape from the prediction; on the other hand, if recall is monitored alone, we may identify false positive poi who are indeed innocent.

Since F1 score depends on both recall and precision, and higher recall and higher precision lead to higher F1 score. Both C and gamma parameters are tuned in order to improve the F1 score. The best RBF parameters are shown below:

SVC(C=10000, cache\_size=200, class\_weight=None, coef0=0.0,

decision\_function\_shape='ovr', degree=3, gamma=0.1, kernel='rbf',

max\_iter=-1, probability=False, random\_state=None, shrinking=True,

tol=0.001, verbose=False)

Accuracy: 0.80821 Precision: 0.35076 Recall: 0.40250 F1: 0.37485 F2: 0.39097

It can be seen that F1 score is improved to 0.37485.

In addition, I also test that if I remove the added feature, what is the performance with the optimized RBF classfier.

Accuracy: 0.80921 Precision: 0.32679 Recall: 0.31650 F1: 0.32156 F2: 0.31851

Although the accuracy is slightly higher, the precision, recall and F1 score are significantly lower. This clearly suggests that the classifier works better with the new features added.

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 a method to assess how the results of analysis will generalize to an independent data set. In the machine learning and prediction, it is important to validate the performance of our prediction. In general, we can save a small part of the data as the test data and train on the remaining part only. In the end, we could evaluate the model on the test data and have an idea how well it is. If all the data is used for the training process, we may overfit the model which is not desired.

In the study here, the cross validation is used in the feature selection, algorithm selection and algorithm tuning. In addition, because of the skewed dataset, the stratified shuffle split is used to help on the accuracy. Multiple times of training is also conducted to make sure the splitting is fair and the contribution of each variable is averaged for the accuracy.

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”]

The accuracy, presicion, recall and F1 score are calculated for each model. With the optimized parameters, the results for RBF SVM are:

Accuracy: 0.80821 Precision: 0.35076 Recall: 0.40250 F1: 0.37485 F2: 0.39097

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Prediction | |
| poi | Non-poi |
| Reality | poi | True positive | False negative |
| Non-poi | False positive | True negative |

Accuracy is defined as the ratio of true predictions to the total predictions (a poi is predicted as poi, a non-poi is predicted as a non-poi). Precision is defined as the ratio of true positives to the sum of true positives and false positives. Recall is defined as the ratio of true positives to the sum of true positives and false negatives.

My algorithms use exercised\_stock\_options, the percentage of emails to poi in all emails sent, expenses and salary, to predict whether a person is a poi. 35% of time is correct in identifying a true poi, which is better than guessing with a probability of 12.5%. 40% of time is correct in identifying a true poi when it is poi; which means that 60% of time it will be missed. In general 80% of time the prediction is correct.

Future works:

More features are necessary in order to further improve the precision and recall. It may be possible to look at the email contest itself and some works for the text learning are already done besides this work. It may be used here to identify poi.