Heki Wong

Udacity Account: yanheki530@gmail.com

Enron POI Identifier Report

Introduction

Enron Corporation was an American energy, commodities, and services company based in Houston, Texas. Enron Corporation bankrupted on December 2, 2001 because of corporate fraud and corruption. as one of the largest corporate fraud cases in American history. The purpose of this project is to identify persons of interest in the fraud case from 146 executives at Enron. A person of interest (POI) is defined as a person who indicted for fraud, settled with the government, or testified in exchange for immunity. This report tried to use machine learning techniques to construct a POI identifier by given information.

The Enron Data

The Enron data includes 146 executives in the dataset. Each data point contains 14 financial features, 5 email features and 1 target label. The target label flag if this specific person is POI or not. However, the target labels in the Enron data is unbalance where there are only 18 POIs in the dataset. The first step is to screen the available data points in each feature (14 financial features, 5 email features and 1 target label). Table 1 shows all available data points in each feature. Those features with insufficient data are not taken into consideration, since the lack of data issue might affect the generalization of the algorithms in the future. The threshold for the first feature selection is at least the features have more than 85(~60%) available data points.

Table 1. The available data points in each feature.

Feature	Available data	Feature	Available data
Email features		restricted_stock_deferred	36
to_messages	86	deferred_income	49
from_poi_to_this_person	86	total_stock_value	126
from_messages	86	expenses	95
from_this_person_to_poi	86	exercised_stock_options	102
shared_receipt_with_poi	86	other	93
Financial features		long_term_incentive	72
salary	95	restricted_stock	110
deferral_payments	39	director_fees	17
total_payments	125		
loan_advances	4	Target label	
bonus	82	poi	146

The next step is to remove any outliers or obvious mistakes. Two outliers were removed which are “TOTAL” and “THE TRAVEL AGENCY IN THE PARK”, since these two data do not represent any characters in the Enron company. Other than that, all the data points are reserved for further analysis.

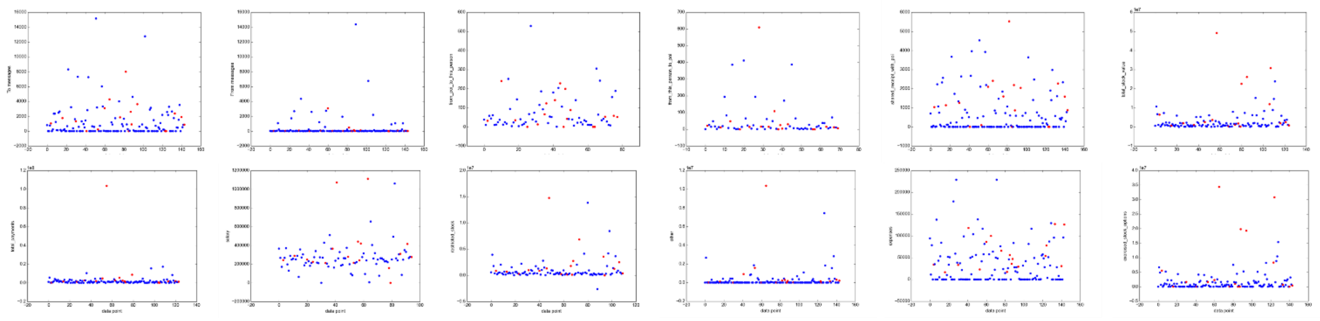
Figure 1. After the first feature selection, there are 12 features for further analysis.

Feature Processing

1. Select K Best with one duplicated feature

Once the data was cleaned of outliers, the next step was going through the second time of feature selection. This Enron dataset is difficult to visualize which feature could be a potential identifier. For this reason, the manner here is using univariate feature selection by selecting the best features based on univariate statistical tests. Since the data are interval data point, the select k best method uses ANOVA to compute the features. The number of k features is 6 according to the best result I got. The features that the select k best method picked up are "total_stock_value", "restricted_stock", "exercised_stock_options", "total_payments", "salary", and "shared_receipt_with_poi". In addition, if duplicating "one" feature which the F value greater than 8.0 ("shared_receipt_with_poi", "salary", "total_payments", "total_stock_value", "exercised_stock_options", "restricted_stock") added into the algorithms, the average accuracy, precision and recall would greatly improve. In this project, I selected "exercised_stock_options" for this repeated feature, so the select k best method picked "total_stock_value", "restricted_stock", "exercised_stock_options" (twice), "total_payments" and "salary".

Table 2. The F score and P value calculated by ANOVA from the select k best method. The features inputted in the select k best method are picked by the first step determined by the available data points.

Feature	F score	P value
to_messages	1.6463	.2016
from_poi_to_this_person	5.2434	.0235
from_messages	0.1697	.6810
from_this_person_to_poi	2.3826	.1249
shared_receipt_with_poi	8.5894	.0039
salary	18.2897	< .0001
total_payments	8.7728	.0036
total_stock_value	24.1829	< .0001
expenses	6.0942	.0148
exercised_stock_options	24.8151	< .0001
other	4.1875	.0426

restricted_stock	9.2128	.0029

Before putting these features into the algorithms to learn, all the selected features were performed feature scaling which rescale the features into the range from 0 to 1.

2. Select K Best with three new features

In the result of select K best method, the ANOVA table provides the information about the relative important features. The interaction between features might make the target stands out from among the data points. The three features with the best result according to lots of experiments are:

A. Adding "total_stock_value" and "exercise_stock_options" together

B. Adding "total_stock_value" and "salary" together

C. Multiplying "shared_receipt_with_poi" with "total_payments"

Since the "total_stock_value" has been considered twice in the new features, I removed it from the feature list. After adding these three new features, the select K best method (k = 4, which gets the best result) has been re-run again to allow the method to pick up the important features. The selected features are "salary", "exercised_stock_options", "new_feature_A", and "new_feature_B". These two new selected features might displays that the stock value that a person owned and exercised might determine if he/she was doing fraud.

3. Principle Component Analysis

In this project, I also tested the principle component analysis (PCA) to transform the features to several components and tried to use logistic regression to test the performance of the algorithm.

Algorithm Selection and Tuning

Select K Best

The AdaBoost classifier was finally selected as the algorithm for the POI identifier One advantage of ensemble learning algorithms is that the ensemble methods use a combination of multiple base models to improve predictions, which sometimes can have better outputs. In addition, the AdaBoost function allows to tune the parameters of its algorithm such as different base estimators. The reason to try different parameters is to determine the performance or complexity of the algorithm. Take the AdaBoost function for example, the number of estimators determines the complexity of the algorithm. It might produce a good performance, but it might be overfit the data or the high complexity might need the higher computation, meaning that it could take relative longer time for the prediction. In this project, no base estimator (None), SVM, Decision Tree Classifier and Random Forest Classifier are selected to compare the results. In these four tables, two features list selected by the selected the k best method were tested. (Fa: "total_stock_value", "restricted_stock", "exercised_stock_options", "total_payments", "salary", and "shared_receipt_with_poi"; Fb: "total_stock_value", "restricted_stock", "exercised_stock_options" (twice), "total_payments" and "salary")

Table 3. The result from AdaBoost classifier without base estimator.

Base estimator = None		
	Number of estimators = 20	Number of estimators = 50

	SAMME		SAMME.R		SAMME		SAMME.R	
	Fa	Fb	Fa	Fb	Fa	Fb	Fa	Fb
Accuracy	0.8602	0.8951	0,8319	0.8601	0.8530	0.8672	0.8460	**0.8740**
Precision	0.4167	0.7778	0.3222	0.4333	0.4111	0.3333	0.3333	**0.6095**
Recall	0.2778	0.2778	0.2778	0.3889	0.2778	0.2778	0.1667	**0.3889**

Table 4. The result from AdaBoost classifier with SVC as base estimator. The SVC does not support probability method, it can not be calculated by SAMME.R.

Base estimator = SVC								
	Number of estimators = 20				Number of estimators = 50			
	SAMME		SAMME.R		SAMME		SAMME.R	
	Fa	Fb	Fa	Fb	Fa	Fb	Fa	Fb
Accuracy	0.8741	0.8741			0.8741	0.8741		
Precision	0	0			0	0		
Recall	0	0			0	0		

Table 5. The result from AdaBoost classifier with decision tree classifier as base estimator.

Base estimator = Decision Tree Classifier								
	Number of estimators = 20				Number of estimators = 50			
	SAMME		SAMME.R		SAMME		SAMME.R	
	Fa	Fb	Fa	Fb	Fa	Fb	Fa	Fb
Accuracy	0.8112	0.8039	0.8112	0.8039	0.8112	0.8039	0.8112	0.8039
Precision	0.3036	0.2883	0.3036	0.2883	0.3036	0.2883	0.3036	0.2883
Recall	0.2778	0.2222	0.2778	0.2222	0.2778	0.2222	0.2778	0.2222

Table 6. The result from AdaBoost classifier with random forest classifier as base estimator.

Base estimator = Random Forest Classifier								
	Number of estimators = 20				Number of estimators = 50			
	SAMME		SAMME.R		SAMME		SAMME.R	
	Fa	Fb	Fa	Fb	Fa	Fb	Fa	Fb

Accuracy	0.8830	0.8811	0.8533	0.8741	0.8883	0.8811	0.8394	0.8741
Precision	0.5556	0.5333	0.2778	0.3667	0.5556	0.5333	0.3556	0.4778
Recall	0.2222	0.3333	0.1111	0.2222	0.2222	0.3333	0.2222	0.2778

The result shows that the AdaBoost algorithm with no base estimator, 50 estimators and SAMME.R algorithm predicts the best precision and recall.

Select K Best with three new created features

The AdaBoost is also used in this analysis. The base estimators includes None, Decision Tree Classifier (min_samples_split=5, max_features = None), Random Forest Classifier (min_samples_split=3, max_features = None). The reason to not allow the algorithm such as Decision Three classifier and Random Forest Classifier to choose the important features is that I have already gone through the select K best method to pick up the important features. The best result from this algorithm is as follow:

Table 7. The result from AdaBoost with the select K best method with three new created features

Accuracy	0.8924
Precision	0.8333
Recall	0.3333

Principle Component Analysis

Table 8. Choosing different number of components in the PCA to calculate the performance of logistic regression.

Component			Component		Component	
1	Accuracy	0.8811	4	0.8672	7	0.8672
	Precision	0.3333		0.0		0.0
	Recall	0.0556		0.0		0.0
2	Accuracy	0.8812	5	0.8672	8	0.8672
	Precision	0.2222		0.0		0.0
	Recall	0.1111		0.0		0.0
3	Accuracy	0.8812	6	0.8672	9	0.8672
	Precision	0.2222		0.0		0.0
	Recall	0.1111		0.0		0.0

The average performance of using logistic regression with PCA is worse than using AdaBoost method. After four components, the logistic regression with PCA features starts to predict all the people as non POI. However, the interesting point that can be seen obviously is that the average accuracy is higher than using AdaBoost method. This will be discussed in the discussion section.

Important of parameter tuning

Algorithms may perform differently using different parameters depending on the structure of data. It can over-tune an algorithm to predict training data extremely well, but fail miserably on unseen data. Each algorithm was tuned using an exhaustive grid search over any major tune-able parameters, over 1000 randomized stratified cross-validation stratified splits. The results were scored in each split on the hold-out testing portion, and the score was averaged over all 1000 splits. The parameters which gave the highest average score were selected for the final model.

Important of Validation

Validation is the processed of checking to see how the model performs on unseen data. A classic mistake would be tuning the model be able to predict your training data very well , but then having it perform poorly on unseen out-of-sample testing data. This is called overfitting. One of the major goals in validation is to avoid overfitting, which can be accomplished through a process called cross-validation.

Tuning by Grid Search

Parameters were tuned with a grid-search (GridSearchCV31) over 1000 stratified shuffled cross-validation 90%-training/ 10%-testing splits to emulate the same testing procedure used in tester.py. This is because the KBest selection and PCA reduction were done within each 1000 cross-validation split, instead of outside the selection process. K-best selection and PCA reduction being in the inner loop was done to give a less biased estimate of performance on any new unseen data that this model might be used for.

Analysis Validation and Performance

This process was validated using 3-fold cross-validation. The reason to do the 3-fold cross validation rather than performing the validation on all dataset is to prevent overfitting issue. The idea behind cross-validation is picking part of data as the training set (67% of the data in this 3-fold cross-validation), and the rest of it is test set (33% of the data). Using this cross-validation method can allow to have the idea about the generalization of an algorithm. The accuracy, precision and recall were averaged among these 3 folds to quantify the performance of different algorithms.

Discussion and Conclusions

classifier	**AdaBoost**					**logistic regression**
	no base estimator (None)				K best method (after create two features)	PCA
	Number of estimators = 50					
	SAMME		SAMME.R			
	Fa	Fb	Fa	Fb		

Accuracy	0.853	0.8672	0.846	0.874	0.8924	0.8812
Precision	0.4111	0.3333	0.3333	0.6095	0.8333	0.2222
recall	0.2778	0.2778	0.1667	0.3889	0.3333	0.1111

FeaturesList: ['poi', 'from_poi_to_this_person','from_this_person_to_poi', 'shared_receipt_with_poi', 'salary','total_payments', 'expenses','exercised_stock_options','other','restricted_stock', 'new_features_1', 'new_features_2', 'new_features_3']

The precision can be interpreted as the likelihood that a person who is identified as a POI is actually a true POI. The best performance in this project is 0.6095 which means that if the algorithm flag a POI, there would be 39% that this person might not be POI (false alarms). On the other hand, gives 0.3889 of recall measurement, leading to the fact that as a potential POI, there is 39% chance that this algorithm would flag this person as a POI. The best precision is 0.8333 with two new created features, meaning that it decreases the chance that a person is falsely recognized as a POI from 39% to 16.7%. However, this algorithm also decreases 6% (recall = 33%) of the ability to flag a person as a POI. As a trade-off, this might be a good algorithm, since if a person is recognized as a POI, the chance of false alarm is only 16.7%. This means that among 18 POIs, only 6 POIs can be flagged and one of them is the false alarm.

Even though this is the best result I got in this project, the ability of this identifier is still far from the practical application. The identifier is still difficult to detect a potential POI. Although the identifier has 39% of chance to have the false alarm, the relatively low recall value shows that this identifier can not successfully detect a POI. A good identifier should have higher precision and recall value (close to 1.0), which means the identifier is able to detect a potential POI and the possibility of being false detection is low.

The biggest challenge in this project is that only 18 POIs in the dataset. This unbalance dataset would bias the result if accuracy of prediction is the only evaluation metric, meaning that if the algorithm predicts all the people as non-POI, the overall accuracy would be 0.8767 (126/146). The lack of POI in the dataset issue also results in the difficulty to train a good algorithm. This can be observed in the logistic regression with PCA where the average accuracy is high, but the precision and recall value is lower comparing to AdaBoost. However, this unbalance dataset issue is very hard to overcome, since it is impossible that half of people in the company are doing fraud and the company can still alive.

Another possible way to improve the identifier could be finding the interaction among different features. This extract some deep information that display some sufficient characteristics of POIs in the dataset.