Identifying Fraud from Enron Email

Sai Venkat Kotha

The goal of this project is to build an identifier which uses the provided financial and email information of various Enron employees to figure out whether the employee is a Person of Interest (POI) or not. This project uses multiple machine learning techniques to build the identifier. One of the supervised classification algorithms available, Naive Bayes Classifier is used to flag the POIs. The Enron dataset has information about 145 people and this information comes in the form of various features. These features play a crucial role in identifying the POIs. The Naive Bayes Classifier is trained on a part of this dataset to figure out what features help in distinguishing POIs from Non-POIs.

The dataset had 146 data points in which there are 18 POIs and 128 Non-POIs.The histograms for ‘salary’, ‘bonus’, ‘total_payments’ and ‘total_stock_value’ features revealed an outlier. Upon further investigation this outlier turned out to be irrelevant as it was a data point that contained the sum of all the numerical features of the employees in the dataset. The key of this data point was labeled ‘*TOTAL*’ and this data point was removed from the data dictionary.

Identifying the proper features to be fed to the classifier is very important.
The dataset came with the following features

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)

A new feature has been engineered to help identify the POIs better. This feature is named ‘percentage_extra_money’ which represents the percentage of excess money received by the employees with respect to their salary. Intuitively speaking the employees involved in the Enron fraud might have made a lot of money other than their salary income and also these employees would have been at better positions in the firm than the other employees. This new feature sums the bonus, long term incentives, expenses and other money received by the employees and calculates how much percentage it is of the employee’s salary. An employee who had received excess amounts of money other than the salary might be a POI.

Feature scaling was performed on all the features except for the “email_address” feature. Feature scaling was necessary because most of the features had different ranges and variances and some features could have been easily neglected because of the features with large ranges such as salary, bonus, total_payments etc.

The feature selection for this project is done using the ‘SelectKBest’ feature selection technique. After experimenting with various values of ‘k’, it was found that the classifier performed well with 5 features i.e k=5. The performance scores for Gaussian Naive Bayes algorithm for different values of k are listed below

k	Accuracy	Precision Score	Recall Score
5	0.91	0.6	0.6
6	0.89	0.5	0.6
7	0.89	0.5	0.6
8	0.86	0.4	0.4
9	0.89	0.5	0.6
10	0.89	0.5	0.6

The feature scores assigned to the features by ‘SelectKBest’ technique are listed below

Feature	Feature Score
bonus	15.81
deferral_payments	0.001
deferred_income	8.96
director_fees	30.65
exercised_stock_options	0.68
expenses	8.49
from_messages	10.81
from_poi_to_this_person	4.31
from_this_person_to_poi	9.96
long_term_incentive	3.19
other	7.53
percentage_extra_money	8.05
restricted_stock	1.64
restricted_stock_deferred	2.61
salary	4.94
shared_receipt_with_poi	0.43
to_messages	0.10
total_payments	10.67
total_stock_value	0.11

The top 5 features fed into the classifier are *director_fees*, *bonus*, *from_messages*, *total_payments* and *from_this_person_to_poi*

After experimenting with three different classification algorithms namely Gaussian Naive Bayes Classifier, Decision Tree Classifier and Random Forest Classifier, the algorithm which performed better was Gaussian Naive Bayes Classifier. The classifier had an accuracy of 0.9 with a precision score of 0.6 and a recall score of 0.6. The Decision Tree Classifier had an accuracy of 0.84 with a precision score of 0.33 and a recall score of 0.4. The Random Forest Classifier had an accuracy of 0.89 with a precision score of 0.5 and a recall score of 0.2 .The Naive Bayes Classifier was also much faster than the other two algorithms.

Parameter tuning is very important to improve the performance of an algorithm. The process of tuning the parameters means that running the algorithm with different combinations of values for its parameters to find out the combination which produces the best result. If very little parameter tuning is done, the algorithm could become under-fit and become more generic. If too much parameter tuning is done, the algorithm could become over-fitted to the training data and would perform bad on test data.

Parameter tuning can be done manually by running the algorithm several times with different values of its parameters and can also be done programmatically by making use of available techniques such as GridSearchCV.

The Gaussian Naive Bayes classifier does not have any parameters that need tuning but the Decision Tree algorithm has parameters which can be tuned. Some of such parameters are *min_samples_split*, *criterion* etc. In this project GridSearchCV is used to tune the parameters of the Decision Tree Classifier. The algorithm was tuned for different values of ‘criterion’ and ‘min_samples_split’.

Validation is the process of evaluating our trained algorithm using a testing set. If the algorithms are not validated using a testing set and are trained on the entire dataset, it could cause overfitting of the algorithm. In this project, *StratifiedShuffleSplit* is used for validation. The algorithms are trained on 70% of the data and the remaining 30% of the data is used to test the algorithm. This training and testing is done in 10 iterations.

The Gaussian Naive Bayes classifier had an average accuracy of 0.85 with an average precision score of 0.45 and an average recall score of 0.36. The accuracy score indicates that out of all the predicted POIs, 85% were correctly predicted. The precision score indicates that out of all the employees identified as POI, 45% are truly POIs. The recall score indicates that out of all the employees that are actually POI, 36% are correctly identified as POIs.

References:
Udacity - https://www.udacity.com/
Sklearn documentation - http://scikit-learn.org/stable/index.html