Identify Fraud from Enron Email

In this project, the email and financial details collected as part of the investigation of the Enron bankruptcy case is investigated by using machine learning techniques to identify people of interest based on the financial data and their email conversations. The corporate fraud at Enron involved employees being incentivized with bonus and stock options for increasing short term earnings of Enron. The Enron dataset consists of email conversations and financial details of 144 employees. The dataset needed to be cleaned before it could be used for modeling. The Enron dataset consists of a lot of information about individuals which could take a long time to read though the email conversations and the financial documents in conjunction. For this kind of tasks, multiple law firms across the world have been using machines to read through large volumes of texts and financial documents to find trends in the data which can help in identifying suspicious data points. [1]

The dataset used for building a machine learning model was built by combining data from both the financial documents and the details from the emails exchanged between the various employees of Enron. The dataset has missing values for various columns for each employee. The unique id for each employee is their name. The dataset seems to be built by using the financial document as there is an outlier in the dataset which corresponds to the TOTAL row in the financial data PDF. Another outlier or data point with lots of missing data is the entry for "The Travel Agency in the Park" which was a travel agency used for travel bookings of Enron employees. These 2 data points were removed from the dataset as a preprocessing step. Additionally the features are all numeric and some are financial data points which have a high variance in them, scaling were applied to all the features in the dataset. Another important point to note about the dataset is that the dataset labels are unbalanced. There are many more non POI employees than POIs.

The dataset consists of 20 features in all, out of which 19 are numeric and one string feature which is the email address, used to find the emails of the user. Many data points have missing values in the features. As the dataset is unbalanced, it would not make sense to impute the values in any method. Additionally, the number of data points is very low, and so data points with missing features cannot be removed as that mean losing a lot of information. Many of the features in the financial category have missing data as it might not be applicable to all the users.

Feature Engineering

In machine learning, having more features is not always beneficial. Having more meaningful or useful features is more important. For example, the Enron dataset has 20 features, which can be a lot of dimensions for a simple algorithm like k-means. Sometimes it is possible to combine the information from multiple features in a smaller number of features using feature engineering. Feature engineering is a wide topic and some of the tasks which are part of feature engineering are feature selection, dimensionality reduction, etc. The new feature subset can be created automatically by using techniques like principal component analysis or using mutual information to find features which give the maximum information about class labels of the classification task. For this project, mutual information was used to find the top features to retain for the modeling task. For each of the machine learning algorithm, an appropriate value of k best features to choose was found using a grid search with default values for all machine learning algorithms.

The following figure shows the results of the grid search discussed above. The optimal number of features to use for the classification task was 5. This includes 2 features which were created from the original features. In the Enron scandal Wikipedia page, there are details that the employees were making extravagant spending to not think about cost always and promote original thinking. This information was used to create a new variable which captures the ratio of expenses to the salary of employees to find employees who were spending a lot of their total payments from the company. But this feature was not found to be useful by the algorithms. The Enron employees were known to be given a lot of bonuses to appreciate getting more contracts. To find employees who were getting substantial bonuses a new feature was created to compare the ratio of their bonus and their other allowances. This feature was useful in increasing the predictive power of the ML algorithm. Another feature which was useful was the ratio of total stock value and the restricted stock value. Employees who were getting more stock than their entitled RSUs might be involved in fraudulent activities.

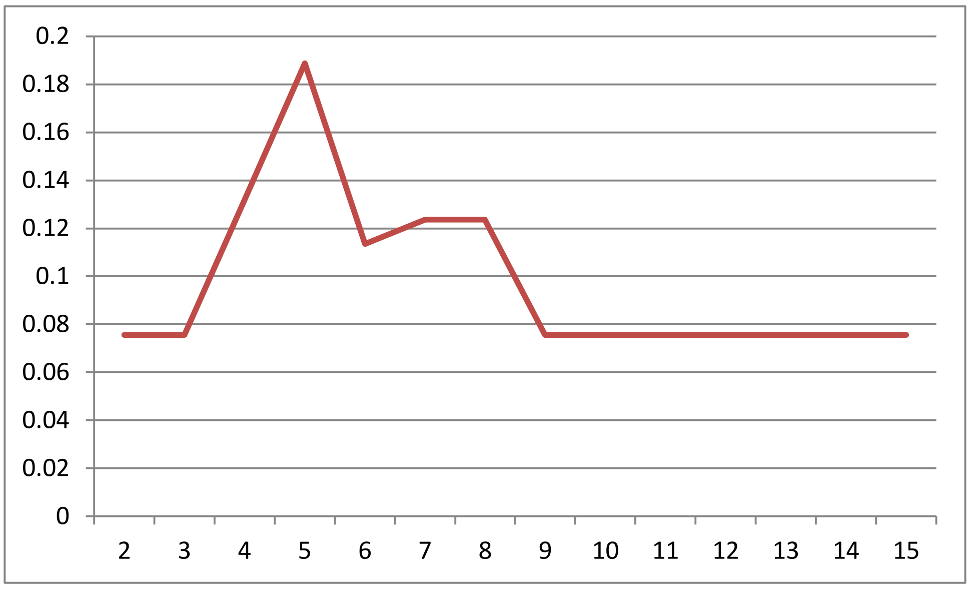

Figure 1: Choosing the best number of features to retain for the model.

For the final model, a Decision Tree classifier was used with gini coefficient criterion and a maximum depth of 25. The following list shows the scores of the variables retained for the final model using the SelectKBest method in sci-kit learn.

```
('bonus_other_ratio', 0.058824165830585029)
('total_stock_restricted_ratio', 0.078502697737206972)
('other', 0.094236797279436901)
('expenses', 0.10753217365233136)
('bonus', 0.12128637478227744)
```

Choosing ML Algorithm

For this project I experimented with Decision Tree and k nearest neighbor (kNN) algorithm. Additionally Gaussian naïve Bayes algorithm was also explored, but the Recall score of GNB was consistently low and so was not considered for the final evaluation. Decision tree algorithm is not particularly well suited for continuous features but it is a simple algorithm with a model that is easy to understand and visualize. kNN was chosen for the relative simplicity of the algorithm. In the feature engineering stage the final number of features that are retained reduced to a quarter and it made kNN more acceptable for the small dataset at hand. kNN did surprisingly better than Decision Trees. This kind of makes sense as the POIs have a lot of common trends among them, but it might be difficult to capture the various trends in the dataset with a single decision tree. Using boosting or ensemble learning might be able to improve the performance.

Hyper-parameter Tuning

Most ML algorithms come with hyper-parameters which can be changed based on the dataset. Hyper-parameters like max tree depth for Decision Tree can help prevent overfitting in the algorithm. Choosing the appropriate parameters for an algorithm can be a daunting task if performed manually. For this very reason a popular method applied to find the best parameters for an algorithm for a given dataset is the Grid Search method, where the algorithm is run on various combinations of the hyper-parameters and the best parameters which have the highest evaluation metric value are used as the final parameters to make the final model. All the algorithms implemented in sklearn come with a default value for all the hyper-parameters. For example, for KNN, the number of neighbors to find to decide the classification is 5. For the small unbalanced Enron dataset, finding 5 neighbors is not an ideal choice. Using grid search it was found that 1 neighbor is the appropriate value to get the best performance. The evaluation metric used for finding the best hyper-parameters was the f1 score. The f1 score is a better measure than accuracy as the dataset is unbalanced.

Validation

Validation is a very important step in machine learning. The model that is built needs to be validated against test data or unseen data to see how well the ML model is able to generalize. A basic way of doing this is to split the dataset into Test and Training sets. But the problem with this simple approach is the difficulty in identifying overfitting. The model might give excellent performance on the training set but not on the testing set. To know if the divergence in train & test dataset performance it is necessary to do a learning curve by varying the test and train split ratio. This can be a time consuming task for large datasets, and while tuning the hyper-parameters.

A common practice in ML model training is to use cross validation to train the model on different subsamples of the dataset and then verify the result on a unseen subset of the training data. Describing Cross Validation is beyond the scope of this paper, but an excellent source to understand it is available at [2]. For this project a 5 fold cross validation was used in the grid search to find the best hyper parameters for both the ML algorithms.

Performance Evaluation

The average performance of the final ML model, built with the kNN algorithm using a single neighbor to determine the classification, is presented below. For the grid search, the metric to optimize was the f1 score, which is a combined measure of the *precision* and *recall.* Precision and recall are more appropriate for this dataset use the unbalanced classes. Precision and recall are ratios from the confusion matrix which help capture more information than the simple measure of accuracy. For example, in the unbalanced dataset of Enron, 88.8% of the employees are not POIs. So a simple classifier which just says all employees are not POIs would have an accuracy of 88.8%. The classification algorithm should generalize for all the classes and not just the majority class. F1 measure is the harmonic mean of precision and recall metrics. In order to get the POIs, it is important to have a high precision and recall. The average performance metrics for the kNN algorithm is as follows.

Average Accuracy: 0.84862
Average Precision: 0.50940
Average Recall: 0.43350
Average F1: 0.46840

Conclusion

The results found with kNN are impressive for the amount of manual work involved. The algorithm is able to generalize well and identify the POIs with a small training sample. When the dataset grows, these techniques are much more useful in automating the discovery of POIs or similar abnormalities in data.

[1] https://www.biggerlawfirm.com/how-artificial-intelligence-and-machine-learning-enhance-the-performance-of-lawyers/

[2] https://www.analyticsvidhya.com/blog/2015/11/improve-model-performance-cross-validation-in-python-r/