Machine Learning Project – Enron Fraud

Question 1: Understanding the dataset and question¶

The goal of this project is to identify people who may have been involved in fraud at Enron by applying machine learning to the Enron dataset. Machine learning is useful for this project because it uses algorithms to learn from and make predictions based on data, ultimately detecting new patterns and trends in the dataset.

To begin the machine learning process, I familiarized myself with the Enron dataset by loading the data into a pandas dataframe. Starting with data exploration allowed me to get an intuitive sense of the data and consider which features might be best for further analysis, before applying my machine learning model. For example, I plotted and gathered descriptive statistics for the various features.

This is a small dataset: 21 features (columns) and 146 rows. The features can be broadly classified into two groups: financial data and email data. As noted in the introduction for this project, the dataset also contains the "poi" feature: "a hand-generated list of persons of interest in the fraud case, which means individuals who were indicted, reached a settlement or plea deal with the government, or testified in exchange for prosecution immunity." The poi or non-poi label is what I will assign through the machine learning model. It is important to note that the poi feature is unevenly split in the dataset: only 18 pois and 128 non-pois. This will be important when considering a cross-validation method for the machine learning model.

In the process of data exploration, I found and removed a misleading outlier: the TOTAL row for the financial data.

I removed the email address feature because it is not numerical data but otherwise, I retained the features and selected features for my machine learning model programmatically.

Question 2: Optimize Feature Selection/Engineering

I modified code from the lesson on feature selection to add two new email features. Since some people in the dataset had sent or received more emails overall, the new features are a proportion of emails sent to or from a poi out of the total number of emails to or from a person.

Therefore, the various features of the dataset have vastly different scales. The new email proportion features ranged from 0 to 1, while salary and total payments are in the 10^4 to 10^8 range. Therefore, I used MinMax Scaling as part of my model pipeline. The MinMax Scaler translates the values of a feature to be within a given range (here 0-1), so that all features can be compared to each other.

I used the Select K Best method for feature selection of my model. Select K Best scores each feature and selects the top features for the model. After testing different parameters for K, my model used the top 5 features with these scores:

exercised_stock_options: 25.10

total_stock_value: 24.47

fraction_to_poi: 22.34

bonus: 21.06

salary: 18.58

Questions 3&4: Pick and Tune an Algorithm

I tested both Gaussian Naïve Bayes and Decision Tree algorithms for this project. The Gaussian Naïve Bayes algorithm is one of the simplest algorithms available – other than assumed prior probabilities, it does not have any parameters to tune. For the Decision Tree algorithm, I used GridSearchCV to systematically test many possible combinations of parameters, such as the criterion and min samples split, and find the best choice. Tuning parameters is an important step because finding the optimal parameters finds the best possible performance for an algorithm.

While both algorithms exceeded the 0.3 threshold for precision and recall that is required for the project, Gaussian Naïve Bayes had higher values for both precision and recall:

Gaussian Naïve Bayes precision – 0.46284

Gaussian Naïve Bayes recall – 0.37050

Decision Tree precision – 0.35664

Decision Tree recall – 0.32980

Therefore, I chose Gaussain Naïve Bayes for my algorithm.

Questions 5&6: Validate and Evaluate

After the model is fit to training data, validation is used to predict the label of the testing data. Validation is important because it verifies that the model is accurate in its classification of the data. When performed correctly, validation can prevent overfitting. Overfitting is a problem where the model matches the training data too closely, thereby incorrectly assigning future data points. I validated my model using GridSearchCV and Stratified Shuffle Split cross validation. GridSearchCV combines cross validation with parameter tuning, thereby avoiding overfitting. The Enron dataset is small and the poi feature is unevenly split in the data: only 18 pois. Therefore, stratified shuffle split was a useful cross validation method because it can use the entire dataset for training and testing by splitting and testing multiple times.

Finally, precision and recall were the evaluation metrics used in this project. Precision is defined as true positives / (true positives + false positives). Recall is defined as true positives / (true positives + false negatives). In this context, precision tells us how many people were correctly identified as pois, out of all the people identified as pois and recall tells us how many people were identified as pois out of all the people who should have been identified as pois.

Resources

Udacity Feature selection lesson

Udacity forums

Scikit Learn documentation