Enron Submission Free-Response Questions

- 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? [relevant rubric items: “data exploration”, “outlier investigation”]

Circa 2000, Enron was a top ten largest company in America. By 2002, Enron was bankrupt with billions of dollars gone, tens of thousands of jobs lost and dozens of people in jail. Enron went from being a multi-billion dollar energy trading company to the biggest case of corporate fraud in American history. Enron, the company, collapsed but Enron survives as one of the most famous datasets of all time, the Enron email corpus. The Enron corpus is the largest set of emails publicly available with correspondences numbering into the hundreds of thousands. It offers fascinating insight into the inner voice of widespread corporate malfeasance and with the aid of Machine Learning, can be used to forensically identify patterns of fraud and persons of interest.

The dataset is comprised of 146 rows and 21 columns for 3066 data points, it is allocated across two classes, 18 POI and 126 non-POI, there are 21 features which are: 'poi', ‘from_messages', 'from_poi_to_this_person', 'from_this_person_to_poi', 'shared_receipt_with_poi', 'to_messages', 'salary', 'bonus', 'long_term_incentive', 'loan_advances', 'other', 'expenses', ‘director_fees', 'total_payments', 'exercised_stock_options', 'restricted_stock', 'restricted_stock_deferred', and ‘total_stock_value’**.**

Many features with NaN data points were replaced with zeros.

Two outliers were found by visually examining the enron61702insiderpay.pdf, ‘TOTAL’ and ‘THE TRAVEL AGENCY IN THE PARK’ and removed with df_dict.pop. Two other salary outliers were found by graphing the data with matplotlib.pyplot.scatter, 'LAY KENNETH L' and ‘SKILLING JEFFREY K’, but kept in as valid data points.

- 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”]

GridSearchCV was utilized as an automated feature selection function with 7 features selected out of a manually chosen range of 6-10 features as pre-determined by SelectKBest.

The following features with the following scores were selected:

'from_messages', '19.69', '0.000',
'expenses', '5.02', '0.027',
'restricted_stock_deferred', '4.35', '0.040',
'restricted_stock', '2.37', '0.127',
'long_term_incentive', '2.01', '0.159',
'other', '1.75', '0.189',
'to_messages', '1.29', '0.259’.

According to: https://stats.stackexchange.com/questions/244507/what-algorithms-need-feature-scaling-beside-from-svm

“Graphical-model based classifiers, such as Fisher LDA or Naive Bayes, as well as Decision trees and Tree-based ensemble methods (RF, XGB) are invariant to feature scaling, but still it might be a good idea to rescale/standartize your data.”

MinMaxScaler was utilized as it is an important requirement in PCA in which we are interested in the components which maximize variance.

Three custom features were engineered: restricted_stock_to_total_stock_value_ratio, bonus_to_total_payments_ratio and expenses_to_total_payments_ratio. Restricted stock, bonuses and expenses were chosen as they are seen to more uniquely characterize executives and management, those with a greater ability to perpetrate and profit from fraud. When added to the data frame these features boosted performance in Precision from 0.33214 to 0.33125 but at a cost to Recall, decreasing from 0.37250 to 0.37100.

- 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”]

In the final determination, the algorithm used was GaussianNB as a part of a Pipeline.

Other algorithms attempted were Gaussian Naive Bayes, DecisionTreeClassifier, C-Support Vector, RandomForestClassifier, NearestNeighbors. and AdaBoost.

GaussianNB() resulted in Precision: 0.23989 and Recall: 0.44500.

DecisionTreeClassifier() resulted in Precision: 0.24440 and Recall: 0.24000.

C-Support Vector resulted in resulted in "Precision or recall may be undefined due to a lack of true positive predicitons."

RandomForestClassifier resulted in Precision: 0.35585 and Recall: 0.10800 despite much parameter tuning.

NearestNeighbors() returned Precision: 0.35585 and Recall: 0.10800 despite parameter tuning.

Finally, AdaBoost returned a max of Precision: 0.39049 and Recall: 0.27100 with n_estimators=4, which fell short of the project requirements, even after much parameter tuning.

Having exhausted these more obvious approaches, after extensive research, a Pipeline of MinMaxScaler(), PCA(), SelectKBest(), and GaussianNB() was utilized to successfully meet the project requirement of at least 3.0 in both precision and recall with Precision: 0.33214 and Recall: 0.37250.

- 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"]

Parameter tuning is the art of optimizing the parameters of an algorithm to fit a data set. This can result in overfitting if an algorithm tales a particular data set too literally.

According to https://stackoverflow.com/questions/22903267/what-is-tuning-in-machine-learning, "Translating this into common sense, tuning is essentially selecting the best parameters for an algorithm to optimize its performance… Performance can be measured by minimizing the number of adjustments needed for a self-driving car to parallel park, or the number of false predictions in autocomplete; or it could be maximizing the time an average visitor spends on a website based on advertisement dimensions, or the number of in-app purchases in Candy Crush."

GridSearchCV was used for parameter tuning as a part of Pipeline. GridSearchCV which according to scikit-learn.org, “exhaustively generates candidates from a grid of parameter values specified with the param_grid parameter.”

As a part of initial project exploration, AdaBoost was tuned utilizing it’s n_estimators parameter which determines the maximum number of estimators at which boosting is terminated. Various numbers of estimators were tried, from 500 to 1.

500 estimators resulted in Precision: 0.32707 and Recall: 0.24350.

Performance increased in precision as the number of estimators decreased, peaking at 0.46354 for 3 estimators.

Performance also increased for recall peaking at 0.27100 for 4 estimators but never reached the .3 project requirement threshold.

Ultimately, 4 estimators yielded the best results in both precision and recall with Precision: 0.39049 and Recall: 0.27100.

After 4 estimators, performance in precision and recall decreased to Precision: 0.26692 and Recall: 0.07100 for 1 estimator.

- 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 the practice of holding out a subset of the data in addition to the training and test subsets. If the training subset is used to train candidate algorithms then the validation subset is used to compare the algorithm’s performances and decide which to accept before tuning that particular algorithm on the test set. If an algorithm performs much better on the train set than on the test set it is a sign of overfitting. Cross-validation, the process of repeating the Holdout method of setting aside a portion of the training dataset, usually a 70%/30% split, was utilized as the validation strategy of this project.

- 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”]

Recall and precision were used as project evaluation metrics which resulted in Precision: 0.33214 and Recall: 0.37250. In this project, recall is defined as the probability of an algorithm to identify a person of interest or POI given that that person is a POI and precision is defined as the probability, if after having identified a person as a POI, that that person actually is a POI. This project’s algorithm identified 33% of the POI’s and when identifying a POI, was correct 37% of the time.