Prepared by: Rajvir Reehal

Udacity Enron Project Question Responses

1. The goal of this project is to use machine learning to build a predictive model which identifies persons of interest (POI) that were involved in one of the biggest accounting frauds in history. The model will determine POIs based on features included in the Enron dataset.

   The dataset consists of 146 data points with 21 features. Of the 146 data points, 18 are identified as POI. They could be split into 3 different categories: financial features (salary, bonus, total stock value, etc.), email features (to messages, from messages, etc.), and the POI label.

   To identify any outliers, I plotted bonus vs salary and discovered an outlier "TOTAL" which summed up the values of all data points. It was removed along with "THE TRAVEL AGENCY IN THE PARK" as it is not a person.

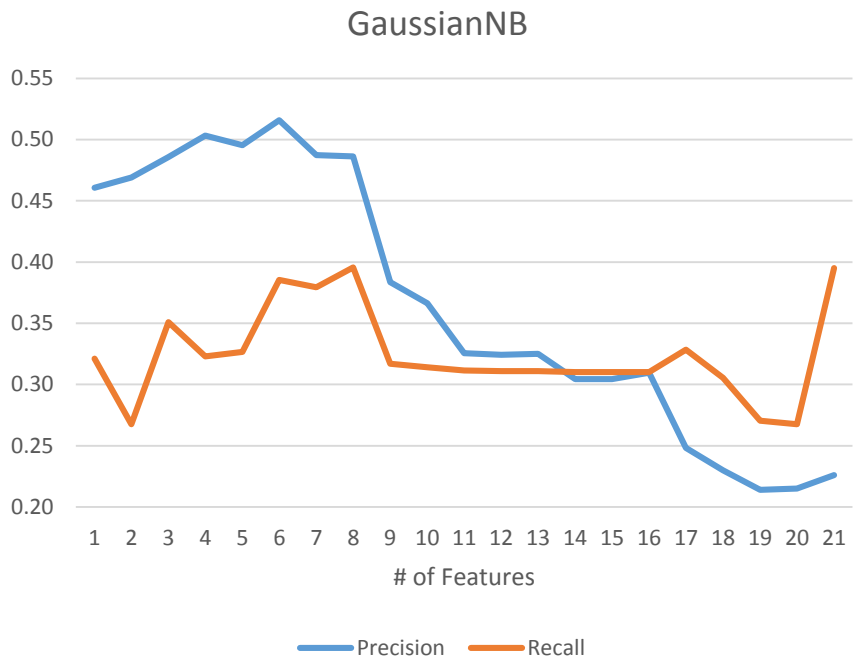
2. The features I ended up using in my POI identifier were selected using scikit's SelectKBest feature selection. Before I started the feature selection process, I created two new features: "ratio_of_messages_from_poi" and "ratio_of_messages_to_poi". The former measures the proportion of a person's inbox that were from a POI and the latter measures the proportion of a person's outbox that were to a POI. After performing SelectKBest feature selection, "ratio_of_messages_to_pio" was selected as one of the features to move forward with in the analysis. Feature scaling was performed using MinMaxScaler to make sure the features were equally weighed before applying the classifer.

   Below are the feature scores from the SelectKBest function:

Feature	Score
exercised_stock_options	24.82
total_stock_value	24.18
bonus	20.79
salary	18.29
ratio_of_messages_to_poi	16.41
deferred_income	11.46
long_term_incentive	9.92
restricted_stock	9.21
total_payments	8.77
shared_receipt_with_poi	8.59
loan_advances	7.18
expenses	6.09
from_poi_to_this_person	5.24
other	4.19
ratio_of_messages_from_poi	3.13

from_this_person_to_poi	2.38
director_fees	2.13
to_messages	1.65
deferral_payments	0.22
from_messages	0.17
restricted_stock_deferred	0.07

For each classifier I used, I chose different values for the number of features using the SelectKBest function and searched for if both the precision and recall scores were above 0.3. Below is a graph which plots the values of precision and recall for different K values for GaussianNB.



3. The algorithms I used were GaussianNB, KMeans, and the KNeighborsClassifier. Below are the best output scores I was able to achieve for each algorithm:

GaussianNB
Accuracy: 0.86050 Precision: 0.51572 Recall: 0.38550 F1: 0.44120 F2: 0.40600
Total predictions: 14000 True positives: 771 False positives: 724 False negatives: 1229
True negatives: 11276

KMeans
Accuracy: 0.79636 Precision: 0.29355 Recall: 0.30250 F1: 0.29796 F2: 0.30067
Total predictions: 14000 True positives: 605 False positives: 1456 False negatives: 1395
True negatives: 10544

KNeighborsClassifier
Accuracy: 0.87079 Precision: 0.61329 Recall: 0.25850 F1: 0.36370 F2: 0.29232
Total predictions: 14000 True positives: 517 False positives: 326 False negatives: 1483
True negatives: 11674

I was able to achieve the best precision and recall scores with GaussianNB with K=6. I was able to achieve the highest precision score with KNeighbors at 0.61329, but it came along with a low recall score. KMeans provided low scores for both precision and recall, therefore GaussianNB was the best performer.

4. To tune the parameters of an algorithm means to optimize the parameters that effect the out of model in order to find the best combination of parameters that complete the learning task in the most accurate way possible. Classifiers such as KMeans require you to set parameters before we can use the classifier (n_init, max_iter, init, etc). However, when adjusting the parameters, you must be cautious of over-fitting the classifier to the learning set.

   My best performing algorithm, GaussianNB does not have any parameters to tune, therefore I will use KNeighbors as an example. For the KNeighbors classifier, I tuned the "n_neigbors" and "weights" paramaters. I used GridSearch to find the optimal set of parameters.

5. Validation is the process in which a sample of data is held back from the training dataset that is used to give an estimate of the model performance while tuning the parameters. Our purpose is to train the model on one dataset (training dataset) and to test it on an independent dataset (test dataset) to measure the unbiased performance of a model.

   A classic mistake you can make is over-fitting data to your training set which then results in poor performance on an independent test set. To prevent over-fitting, it is essential to split the data you have into a test and training set. However, we reach a dilemma as we want to maximize both the training and test set to get the best learning results. There are multiple ways in which you can split and train/test your dataset. One example is K-Fold Cross Validation. K-Fold partitions the dataset into K bins and run K separate learning experiments and average the K different performances. This method allows you to use all the data for the testing and training dataset.

   Stratified Shuffle Split (SSS) was used in my analysis through the test_classifier of the tester.py file. SSS is the preferred validation method for the project task since the dataset is small. This prevents the training or testing dataset from being over-represented in the dataset.

6. GaussianNB provided a precision and recall score of 0.51572 and 0.38550, respectively.

   Recall is the ability of a model to final all of the relevant cases within a dataset. More precisely, it is the proportion of true positives of all true positives and false negatives. In the context of the Enron case, the model can identify a POI as a POI 38.55% of the time.

Precision is the ability of a model to only find the relevant cases within a dataset. It is the proportion of the data points the model identifies as relevant, that actually are relevant. In the context of the Enron case, if the model identifies a POI, 51.57% of the time the person is a POI.