Enron Submission Free-Response Questions

1. 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”]

A. This project focuses on building a machine-learning algorithm that can identify the people who involved in the downfall of the Enron Company. The dataset used in this project is a combination of financial and email data of Enron employees. This dataset contains 146 data points and 21 features in which 14 related to financial, 6 are email and 1 POI. The goal of this project is to optimize POI identifier by using different machine-learning tools.

These are some wrangling steps taken:

1. There is an outlier named Total that stores aggregate of all values of the dataset. This point is misleading so removed it.
2. There is an irrelevant data point named The Travel Agency In The Park. This point is removed.
3. There is shift in the financial data of two employees Belfer Robert and Bhatnagar Sanjay. I am not sure whether their email data is right and they are not POI’s. So, removed these two data points.
4. There is one data point with no data available for all features so removed it.

In the final dataset there are total of 141 employees. Out of which there are 18 POI’s and 123 non POI’s. Majority of employees are not persons of interest.

2. 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”]

A. In this project, I used SelectKBest for selecting best features. I sent select k-best in a pipeline with grid search to select the best parameter combination. There are total of 19 features and all are selected by GridSearchCV. k=19 is giving better precision and recall scores when compared to other K values. I also tried manually changing the K values and the overall performance of the algorithm is better when K=19. These are the scores of the final features selected.

Features	**Scores**
fraction_to_poi	18.24

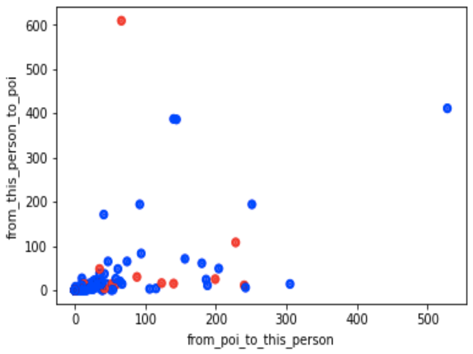
bonus	14.15
deferred_income	13.91
salary	12.72
long_term_incentive	10.90
shared_receipt_with_poi	8.81
exercised_stock_options	8.44
total_stock_value	8.16
total_payments	8.06
other	7.39
loan_advances	6.32
expenses	5.23
restricted_stock	2.79
to_messages	2.43
fraction_from_poi	2.41
director_fees	1.47
restricted_stock_deferred	0.52
deferral_payments	0.45
from_messages	0.08

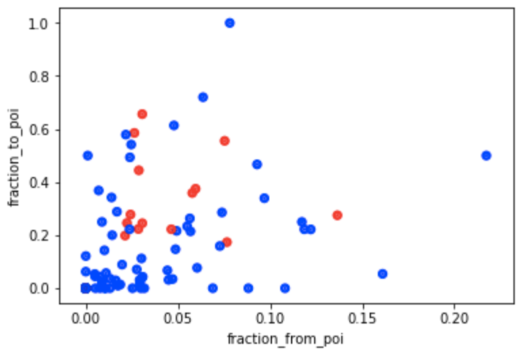
New Feature Implementation

These are the two new features created

1) fraction_to_poi
2) fraction_from_poi

When a scatter plot is drawn between features from_poi_to_this_person and from_this_person_to_poi, POI's(red color dots) are distributed along with non-POI's(blue color dots). From this visualization, these features are not that helpful in predicting POI's. So I added two new features fraction_to_poi and fraction_from_poi. These new features are created by scaling the above features by total number of messages to or from this person. These new features might help us identify those who have low amounts of email activity overall but high percentage of emails from POI's. Now when we see the visualization of fraction_to_poi and fraction_from_poi, POIs are concentrated in one part of the graph.





After adding these new features in the final algorithm there is improvement in precision and recall scores. These features fraction_to_poi(18.24) and fraction_from_poi(2.41) got pretty good scores.

3. 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”]

A. Decision tree is the final algorithm used in this project because its precision and recall scores are better when compared to others. These are the scores of different algorithms when classification report is used to check the results.

Algorithm	**Precision**	**Recall**
Decision tree	0.33	0.50
GaussianNB	0.33	0.50
Random_forest	0.67	0.50
AdaBoost	0.50	0.50

When tested using tester.py evaluation metric the results are as follows.

Algorithm	**Precision**	**Recall**
Decision tree	0.39	0.33
GaussianNB	0.39	0.32
Random_forest	0.58	0.16
AdaBoost	0.36	0.29

Decision tree is chosen as final model because it is performing better in both cases.

4. 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”]

A. Tuning of parameters of an algorithm essentially mean to select best parameter values for an algorithm to optimize its performance. There are two ways to tune parameter 1) manually check different combinations 2) To use GridSearchCV, it will automatically checks different combination and gives the best parameter combination. If these parameters are not tuned well it might lead to overfitting or underfitting and therefore fail to fit additional data or predict future observations reliably. For this project, I have used GridSearchCV.

While implementing SelectKBest and decision tree algorithm, these are the parameter values sent to GridSearchCV

3) feature_selection__k=[10,13,15,19]
4) decision_tree__min_samples_split=[10,20,30,50]

    5) decision_tree__criterion=['entropy','gini']

5. 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"]

A. Validation is used in machine learning to test the performance of a model. The classic mistake is testing the algorithm on the same data that is used for training. Then the result will be perfect because the model already knew the labels of the sample and it will fail to predict future observations.

   In this project, train_test_split function is used to split training and testing data where 70% of the data is used for training and 30% is used for testing. I also tested my model with *tester.py* where it uses Stratified Shuffle Split. This cross validation technique is mainly used when classes are unbalanced. It gives accurate results because it creates multiple datasets out of one. And it will make sure that the proportion of each class in all folds is approximately equal. It works very well in our case because Enron dataset is small and unbalanced where non-POI's are six times more than POI's. In this project, this splitter creates 1000folds, where in each fold training is done on 90% of the data and testing on remaining 10%.

6. 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"]

A. The evaluation metrics used for this POI identifier are precision and recall. This metric is very useful when classes are imbalanced. In our case, non-POI's are six times more than POI's. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned. The precision of my decision tree is 0.33 and its recall score is 0.50. My recall score is high, that means nearly every time a POI is shown up in my test set, I am able to identify him or her. The cost of this is sometimes non-POI's gets flagged. I chose my recall score to be high because it is important to not to miss any POI's. Even if non-POI's are flagged, it is all right because they will not be put in jail right away. Instead, they will all be further prosecuted and after necessary investigation real POI's are determined. So, I tried my best to not to miss any person of interest.