Enron Submission 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"]*

The goal for this project is to create an algorithm that predicts POIs (Persons of Interest who have been convicted of fraudulent behaviour) from a given list of Enron employees using the features supplied. Machine Learning is useful for this task as it can quickly, and somewhat accurately, classify data points as POIs using an optimal set of features and parameters from those provided. In this project, there are 3066 data points, of which, 378 or 12.3% relate to POIs and the remaining 2688 data points or 87.7% relate to non-POIs. There are 146 'employees' in the data set, 18 of these are identified as POIs. The 21 features of the dataset are listed below with their amount and percentage of missing values:

Feature	POI: NaN	POI: NaN%	non-POI: NaN	non-POI: NaN%
salary	**1**	**5.56%**	**50**	**39.06%**
deferral_payments	13	72.22%	94	73.44%
total_payments	0	0.00%	21	16.41%
loan_advances	17	94.44%	125	97.66%
bonus	**2**	**11.11%**	**62**	**48.44%**
restricted_stock_deferred	18	100.00%	110	85.94%
deferred_income	7	38.89%	90	70.31%
total_stock_value	**0**	**0.00%**	**20**	**15.63%**
expenses	0	0.00%	51	39.84%
exercised_stock_options	**6**	**33.33%**	**38**	**29.69%**
other	0	0.00%	53	41.41%
long_term_incentive	6	33.33%	74	57.81%
restricted_stock	1	5.56%	35	27.34%
director_fees	18	100.00%	111	86.72%
to_messages	4	22.22%	56	43.75%
email_address	0	0.00%	35	27.34%
from_poi_to_this_person	4	22.22%	56	43.75%

from_messages	4	22.22%	56	43.75%
from_this_person_to_poi	4	22.22%	56	43.75%
shared_receipt_with_poi	4	22.22%	56	43.75%
poi	0	0.00%	0	0.00%

**Bolded features were used in final POI identifier.

I wrote ‘employees’ as such, because there were two names who are not employees, ‘TOTAL’ and ‘THE TRAVEL AGENCY IN THE PARK.’ These two names were removed from the dataset to provide a more accurate analysis. ‘LOCKHART EUGENE E’ was also removed as all of his features had null values. The values of total_payments and total_stock_value were checked by summing their respective parts. For both total_payments and total_stock_value, the values of ‘BELFER ROBERT’ and ‘BHATNAGAR SANJAY’ were corrected. Using the remaining employees and the corrected features, we can use machine learning to detect who is most likely a POI based on the features’ values.

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”, “properly scale features”, “intelligently select feature”]*

I used the following features for my POI identifier: 'salary', ‘bonus', ‘total_stock_value’, 'exercised_stock_options', ‘bonus_salary_ratio', ‘total_payments_and_stock'. These features were selected using GridSearchCV and selectKBest. Numerous tests of my POI identifier were conducted with a variety of different values for my parameters. These six features led to the best POI identifier based on precision, recall, and F1 scores. Their scores based on f_classif are:

exercised_stock_options: 22.42, total_stock_value: 21.55, bonus: 18.60, salary: 17.69, total_payments_and_stock: 15.07, bonus_salary_ratio: 9.16.

I used the MinMaxScaler on my selected features. This scaling improved the scoring for my POI identifier as it provides more disparity in the data points, thus making it easier to find patterns among the POI data points.

	With MinMaxScaler	Without MinMaxScaler
Precision	**0.30301**	0.17746
Recall	**0.82450**	0.26450
F1	**0.44316**	0.21241

I created the following features (followed by the reasoning for creating them):

- bonus_salary_ratio: POIs have a larger bonus relative to their salary compared to non-POIs.
- total_payments_and_stock: The sum of POIs' total_payments and total_stock_value should be greater than those of non-POIs.
- to_and_from_poi_emails: The sum of emails to and from POIs is great among POIs than non-POIs.
- total_poi_emails: The same as to_and_from_poi_emails, but includes shared_receipt_with_poi. I was hoping that the difference between these two measures of communication could lead to some interesting findings.
- percent_of_poi_to_emails: A greater percentage of POIs' 'to emails' are sent to other POIs compared to non_POIs.
- percent_of_poi_from_emails: A greater percentage of POIs' 'from emails' are received from other POIs compared to non-POIs.
- percent_poi_emails: A greater percentage of POIs' emails are either sent to or received from POIs compared to non-POIs.

The following additional feature selections were used, plus their parameters:

- PCA(n_components = 5, whiten = False):

PCA__whiten	True	**False**
Precision	0.19209	**0.30301**
Recall	0.35950	**0.82450**
F1	0.25039	**0.44316**

PCA__n_components	Precision	Recall	F1
1	0.29251	0.84550	0.43465
2	0.29439	0.83150	0.43483
3	0.30660	0.81800	0.44602
4	0.30277	0.82400	0.44283
5	**0.30301**	**0.82450**	**0.44316**
6	0.30244	0.82400	0.44248

- Logistic_Regression(C = 1, class_weight = {True: 2, False: 1}, fit_intercept = False): Although C = 1 is the default, I tested other values. I wanted to bias the weight towards True, as there are fewer POI data points than non_POI. Having fit_intercept = False, this increased the scoring for the POI identifier, as did the other parameters.

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"]*

My POI identifier uses the LogisticRegression algorithm. It provided the highest F1 and recall scores, with the precision score being above the required 0.3. I also tried (ranked by F1 score) KNeighborsClassifier, AdaBoostClassifier, DecisionTreeClassifier, and SVC. After using GridSearchCV and tuning various parameters (see poi_id.py file) these are the best scores I achieved for each algorithm.

Algorithm	Precision	Recall	F1
LogisticRegression	**0.30301**	**0.82450**	**0.44316**
KNeighborsClassifier	0.66721	0.20650	0.31539
AdaBoostClassifier	0.32256	0.26950	0.29365
DecisionTreeClassifier	0.33420	0.25650	0.29024
SVC	0.42571	0.07450	0.12681

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? (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 item: "tune the algorithm"]*

To tune the parameters of an algorithm, we adjust the parameters' values to have the algorithm best learn in the desired way. This 'desired way of learning' could be the algorithm having a higher accuracy, precision, recall, or F1 score, etc. If we do not tune our algorithm well, it will learn suboptimally and make more mistakes than it could have during the prediction stage of the analysis.

I experimented with many parameters for my LogisticRegression algorithm: penalty, dual, C, fit_intercept, intercept_scaling, and class_weight, however, I only changed class_weight and fit_intercept from their default for my final POI identifier. To tune these parameters, I went through an iterative process to find the optimal value using GridSearchCV. For example, to determine the best 'C' value, I began with the values of 0.1, 1, 10, 100; after '1' was seen to be the best (by leading to the highest values of precision, recall, and F1), my next iteration included values closer to 1: 0.5, 1, 5; again '1' was the best, and I continued this process until I could safely conclude that I should use '1' as my final 'C' value.

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 item: "validation strategy"]*

Validation is the process of measuring the effectiveness of an algorithm. We do this by training our algorithm on a separate set of data than that which we test the algorithm on. A classic mistake is to train and test an algorithm on the same set of data, which will create overfitting and an ineffective algorithm.

I validated my analysis by using StratifiedShuffleSplit(n_splits = 200, test_size = 0.1), as well as observing the accuracy, precision, recall, and F1 scores. StratifiedShuffleSplit is very useful for this dataset as the amount of POI to non-POI data points is very unbalanced, but when training and testing is conducted StratifiedShuffleSplit will balance the ratio of data points. Since the dataset is also rather small, StratifiedShuffleSplit helps use to find the optimal parameter values as there is less randomness when many iterations of training and testing are conducted.

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"]*

The following evaluation metrics and their average performances are from my final POI identifier.

Precision: 0.30. My POI identifier predicts many more employees to be POIs than is true. 30% of predictions are correct, which means the predicted POI is a POI, but 70% of predicted POIs are non-POIs.
Recall: 0.82. My POI identifier rarely identifies someone who is a POI as a non_POI. 82% of POIs are correctly predicted as POIs, but 18% of POIs are incorrectly predicted as non-POIs.
F1: 0.44. Typically my POI identifier inaccurately predicts if an employee is or is not a POI. Although my recall score is rather high, the low precision score lowers the value of my F1 score.

I wanted my POI identifier to have the highest recall score with a precision score greater than the required 0.3. This is because I believe it is more important to not miss a POI (recall), than it is to incorrectly identify a non-POI as a POI (precision).

Note: Accuracy is left out here because it would be a rather poor evaluation metric given the imbalance between POI and non-POI data points.