Data Analyst Nanodegree Final Project by Paulo Casaretto

Available online for commenting:
https://docs.google.com/document/d/1nXnNOd9vSk82H6sfEWC7HiyTO7o2f7Tw-zxOqUDGAfM/edit?usp=sharing

Directory structure

Along with project resouces, and expected python scritps there are three more notable files in this directory.

- Machine Learning Final Project.ipynb : iPython notebook used for exploration
- exploration.py : python script used for evaluating and testing different classifiers
- sources.txt : where I note the resources used for this project

Answers for the proposed questions

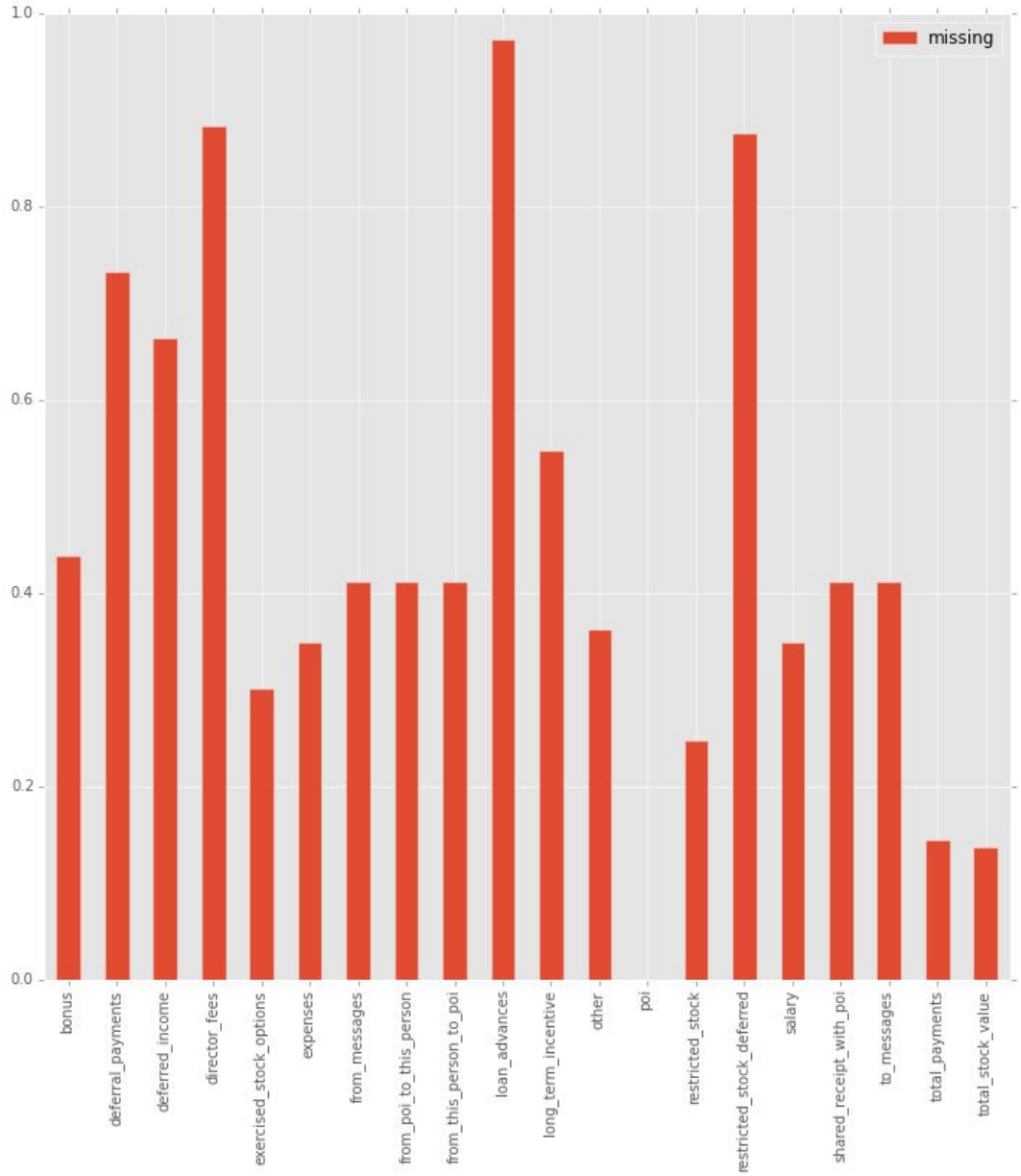
https://docs.google.com/document/d/1NDgi1PrNJP7WTbfSUuRUnz8yzs5nGVTSzpO7oeNTEWA/pub?embedded=true

The major goal of this project is to build an algorithm capable of identifying possible fraudsters from Enron's financial and email dataset.

Data Exploration

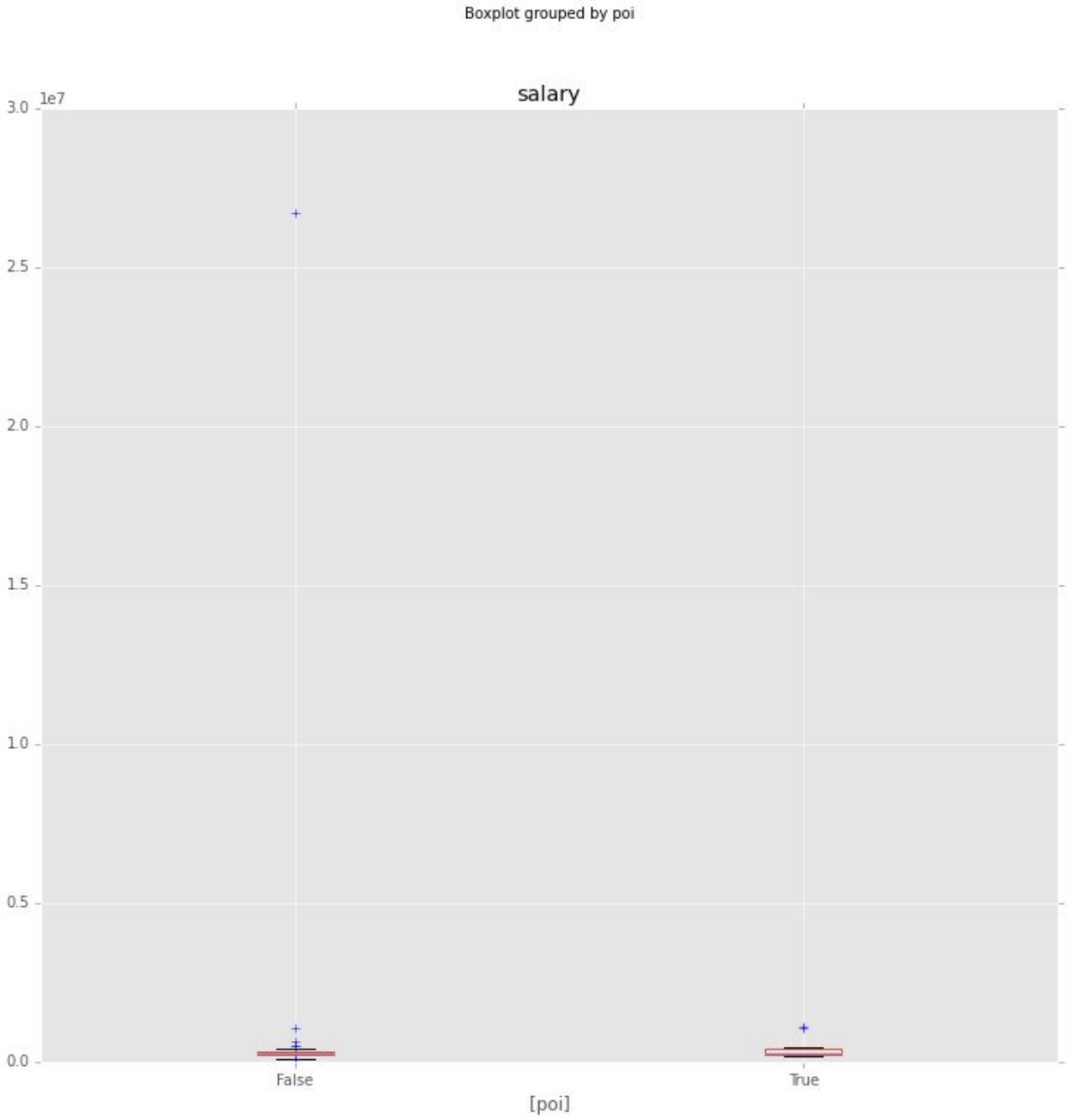
This dataset has 146 data points and is very unbalanced with only 18 data points marked as POIs and 128 as not.

Most of it's 21 features are numeric and there is a lot of missing data. The average percentage of missing data between the numeric features is 45%. I have included a chart to further illustrate the problem.



Outlier investigation

Analyzing salary data, I found a clear outlier that was the sum of all other data points. A clear data entry mistake. This data point was excluded from all posterior analysis.


Boxplot grouped by poi
salary
1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
False
True
[poi]

	poi	person	salary
104	False	TOTAL	26704229
95	True	SKILLING JEFFREY K	1111258
65	True	LAY KENNETH L	1072321
129	False	FREVERT MARK A	1060932
100	False	PICKERING MARK R	655037
57	False	WHALLEY LAWRENCE G	510364
128	False	DERRICK JR. JAMES V	492375
85	True	FASTOW ANDREW S	440698
52	False	SHERRIFF JOHN R	428780
88	True	RICE KENNETH D	420636

Create new features

Looking at the mail exchanged features I had an insight. The idea was to extract what percentage of total messages each person exchanged with a POI. The hypothesis is that a higher ratio of messages exchanged with POIs might indicate that you are a POI yourself.

The results were interesting. The ratio between messages sent to POIs and total messages sent had a higher correlation coefficient than the original features.

	poi
poi	1.000000
to_poi_ratio	0.408599
from_this_person_to_poi	0.261205
to_messages	0.197932
from_poi_ratio	0.224880
from_poi_to_this_person	0.305633
from_messages	0.045687

Intelligently select features

To intelligently select features, I used a feature union between PCA and SelectKBest. I then used grid search to select both the optimal number of components for the PCA and also the K original features to use.

I ran SelectKBest before optimizing the classifiers and obtained the following table:

exercised_stock_options	25.09754153
total_stock_value	24.46765405
bonus	21.06000171
salary	18.57570327
deferred_income	11.59554766
long_term_incentive	10.07245453
restricted_stock	9.346700791
total_payments	8.866721537
shared_receipt_with_poi	8.746485532
loan_advances	7.242730397
expenses	6.234201141
from_poi_to_this_person	5.344941523
other	4.204970858
from_this_person_to_poi	2.426508127
director_fees	2.107655943
to_messages	1.698824349
deferral_payments	0.2170589303
from_messages	0.1641644982
to_poi_ratio	0.142
restricted_stock_deferred	0.06498431172
from_poi_ratio	nan

It is unfortunate that the engineered features performed so poorly. Given we used Select K Best in our parameter search, it is highly unlikely these will be selected for the final model.

Properly scale features

Features were normalized using MinMaxScaler.

Pick and tune an algorithm & Validate and Evaluate

A multitude of different classifiers and parameters were tested for this dataset.

With such a small an unbalanced dataset, it is imperative that we cross-validate to avoid overfitting. Cross validating our models allows us to check if it properly generalizes the model or if it memorized the training set and is not capable of extrapolating the predictions to unseen data.

Because we cannot afford to reserve a fraction of the dataset for testing, I opted for using a shuffle split validation which essentially shuffles the dataset and partitions it in a training and a test dataset while preserving the ration between classes. Preversing the ratio is important because of the poor distribuition between the classes, if we had chosen another technique we might have ended up with a split with only one class.

Parameter tuning

Grid search was used to select an optimal classifier. The parameters of each classifier are tuned in this process to ensure we have the maximum performance. This is done 1000 times for every classifier and parameter combination and the performance of each run is averaged to produce the final score.

Accuracy would be a very poor choice for validation since the dataset is very unbalanced. If I had chosen it I could have ended up with a classifier that classified all data points as non-POIs. I chose to use the f1 score to evaluate and choose the best algorithm.

Evaluated algorithms and scores at default parameters and optimum parameter settings (post tuning)

	Pre-tuning		Post-tuning	
Classifier	Precision	Recall	Precision	Recall

Adaboost	0.33723	0.21650	0.39943	0.39943
SVC	0.34914	0.27250	0.41594	0.12000
Random Forest	0.43529	0.16650	0.39365	0.28500
K nearest neighbors	0.22593	0.03050	**0.54062**	0.18300
Logistic Regression Classifier	0.34901	0.07050	0.39677	**0.60250**
LDA	0.21056	0.18350	0.48405	0.21250

Logistic Regression showed the best results having precision and recall of 0.39, 0.61 respectively.

The interpretation of these is highly dependent on the application of the project.

As I imagine this project being used in assisting finding possible fraudsters that might have slipped under the radar. In order to bring these to trial, there would be a thorough investigation.

Under this assumption we can interpret a higher precision score as the likelihood of our classifier having fewer false positives. In other terms, a higher precision score would mean that we can be sure that the people classified as POIs are in fact POIs. In the same manner, we can interpret recall as having a low false negative ratio, that is, a higher recall score would mean that the classifier is not likely to miss any POIs.

If this was not the case, for example, if the algorithm was being used to generate a public list of possible suspects, we would need to be more careful and would favor algorithms with higher precision.

I also tested the effect of the engineered features on the final algorithm. Removing them from the dataset and retraining the classifier yielded the exact same results. This is most likely caused by SelectKBest not selecting them for the final algorithm as we saw in the relevant section.