

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”]

My goal for the project was to end up with an automated classifier that took would build automated as much as possible the Classifier choosing and parameter tuning. The data use used is fairly broad and didn't look like a ton of varied data, it also wasn't strictly numeric. The majority of it was clustered in similar ways with ridiculous outliers. The first steps were to convert the relevant data to numerals (when applicable) to allow for easy analysis, Next the outliers had to be stripped out so the classifiers didn't spend all of their time handling those exceptions.

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”]

For my POI identifiers I initially started with a large feature set. I then plotted them in Pyplot to get a feel for the shaped of the data. I chose the features by using judgment for areas where people could easily manipulate payments without raising too many eyebrows. I settled on my final list ('poi', 'salary', 'total_payments', 'loan_advances', 'bonus', 'restricted_stock_deferred', 'deferred_income', 'from_poi_to_this_person', 'exercised_stock_options', 'long_term_incentive', 'from_this_person_to_poi') based on that premise. I did impment some scaling for the features, but only a generic scaling while using the SVM classifier as a runtime optimization.

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”]

I ended up using an Adaboosted Decision tree. I tried a Naïve Bayes, Decision Tree and SVM and the SVM tuned actually approached the Adaboosted Decision tree in terms of performance.

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"]

I used a few mechanisms to choose and tune the algorithm. Firstly I ran the basic classifiers, Naive Bayes, Decision Tree and SVM with no parameters on my training data then used the testing data to score them. I then took the top two performers and used Grid Search to tune the parameters. I then reran the comparison between these two classifiers and then reran the tuning for the Adaboost with the optimized Decision tree.

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"]

For validation I initially tried to use the f1 scoring, however it doesn't work with SVM. I then changed it to average precision. Using the test classifier function gave more details like recall and the number of false positives and false negatives among others.

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 classifier test initially reported the following metrics

Accuracy: 0.77320	Precision: 0.15468	Recall: 0.15700	F1: 0.15583	F2: 0.15653
Total predictions: 15000	True positives: 314	False positives: 1716	False negatives: 1686	True negatives: 11284

After optimizing and filtering through Adaboost the following metrics were observed.

Accuracy: 0.82033	Precision: 0.28563	Recall: 0.23150	F1: 0.25573	F2: 0.24062
Total predictions: 15000	True positives: 463	False positives: 1158	False negatives: 1537	True negatives: 11842

We can see the accuracy increases by 5% and precision by 13%. In real world the number of True positives increased while false positive and false negatives dropped. This makes the classifier more reliable and reflects the increase in accuracy.