Enron Submission Free-Response Questions

A critical part of machine learning is making sense of your analysis process and communicating it to others. The questions below will help us understand your decision-making process and allow us to give feedback on your project. Please answer each question; your answers should be about 1-2 paragraphs per question. If you find yourself writing much more than that, take a step back and see if you can simplify your response!

When your evaluator looks at your responses, he or she will use a specific list of rubric items to assess your answers. Here is the link to that rubric: Link to the rubric Each question has one or more specific rubric items associated with it, so before you submit an answer, take a look at that part of the rubric. If your response does not meet expectations for all rubric points, you will be asked to revise and resubmit your project. Make sure that your responses are detailed enough that the evaluator will be able to understand the steps you took and your thought processes as you went through the data analysis.

Once you've submitted your responses, your coach will take a look and may ask a few more focused follow-up questions on one or more of your answers.

We can't wait to see what you've put together for this project!

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 of this project is to choose and train ML classifier to identify person-of-interest using financial and email data of Enron employees. It's unlikely that a trained classifier will identify POIs absolutely precise but it could help to identify suspects which can be checked further using other techniques.

The data provided has got lot's of NaNs for different features. Distribution of NaNs in the whole data set:

{'salary': 50, 'to_messages': 58, ***'deferral_payments': 106***, 'total_payments': 21, 'long_term_incentive': 79, ***'loan_advances': 141***, 'bonus': 63, 'restricted_stock': 35, ***'restricted_stock_deferred': 127***, 'total_stock_value': 19, 'shared_receipt_with_poi': 58, 'from_poi_to_this_person': 58, 'exercised_stock_options': 43, 'from_messages': 58, 'other': 53, 'from_this_person_to_poi': 58, ***'deferred_income': 96***, 'expenses': 50, 'email_address': 33, ***'director_fees': 128***}

The data set consists of 146 data point. Among them there are 2 data points which are considered as outliers as they don't represent any person: "TOTAL" and "THE TRAVEL AGENCY IN THE PARK". There are also 3 data points with all features equal to 0 or NaN, 2 data points with inconsistent data (sum of all payments is not equal to total of payments or sum of all stocks is not equal to total stocks). After removal of outliers the data set is 139 data points. Among the data points left there are 18 POIs and 121 non-POIs.

There quite a few features which has got NaN for most data points. They may be not useful for classification.

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"]

Final features list: ['poi', 'salary', 'bonus', 'total_payments', 'exercised_stock_options', 'ratio_bonus_totalp', 'ratio_exso_totals']

The following features were constructed from source data:
ratio_bonus_totalp: Ratio of bonus to total_payments (to check if bonuses in POI's total payments are significantly bigger than for non-POI)
ratio_exso_totals: Ratio of exercised_stock_options to total_stocks (to check if POI's exercised stock options are significantly more/less than others)
~~to2from_poi_to_this_person: to_messages / from_poi_to_this_person (If a person receives lots of emails from POIs he/she can be POI as well)~~
~~from2from_this_person_to_poi: from_messages / from_this_person_to_poi (If a person sends lots of emails to POIs he/she can be POI too)~~ **(There appears to problems with using this features due to data leakage from test to train dataset).**

Some classifiers require scaled features, for feature scaling I used MinMaxScaler with feature range (0,1).

SelectKBest was used for feature selection. Parameters for the feature selector was selected by GridSearchCV (quite often, number of features selected was 3 or 4, depending on classifier).

For the finally chosen algorithm scores (chi2) are:
Salary: 1.43381345e+06
Bonus: 1.81466679e+07
Total_payments: 1.04734265e+07
Exercised_stock_options: 1.26964007e+08
Ratio_bonus_totalp: 5.62366171e+00
Ratio_exso_totals: 2.22233723e-02

Highlighted above features were chosen by SelectKBest in GridSearchCV.

DecisionTree classifier feature importances:
Salary: 0.
Bonus: 0.18686078
Total_payments: 0.17283675
Exercised_stock_options:0.43957445
Ratio_bonus_totalp: 0.04408694
Ratio_exso_totals: 0.15664108]

3.What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms?

I have tried several different classifiers:
- GaussianNB; Doesn't have much parameters to tune. Couldn't make it get recall > 0.3.
- DecisionTree. The third best classifier which satisfies the >3 condition.

- RandomForrests (took much longer to train/test than DecisionTrees); the second best results on training and test sets;
- SVC (fast but couldn't find parameters to satisfy condition on precision and recall > 0.3);
- AdaBoost. Takes quite long time to fit/test it. Couldn't get results satisfying the task conditions.;
- KnearestNeighbours. This classifier showed the best results for accuracy, precision and recall. I ended up using it as a classifier.

Classifier	Accuracy	Precision	Recall
GaussianNB	0.849	0.454	0.274
DecisionTree	*0.802*	*0.316*	*0.334*
RandomForest	*0.872*	*0.593*	*0.326*
SVM (SVC)	0.816	0.328	0.272
AdaBoost	0.826	0.351	0.26
KNearestNeighbors	*0.87*	*0.59*	*0.312*

4.***What does it mean to tune the parameters of an algorithm, and what can happen if you don't do this well?***

Tune the parameters of a classifier is a process of finding optimal parameters for the best result (score/accuracy/precision/recall).

To tune parameters I ended up using GridSearchCV. This cross validation method searched for optimal parameters for chosen score function (e.g. precision, recall, f1 etc).

5.What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis?

Validation is a check of model for over-fitting when a classifier performs well on training set and poorly on other data (e.g. test or real data).
A classic mistake is not splitting data available into train and test sets. As follows from datasets names one part was used for training of classifier another part for testing.

It is possible to use many cross validation techniques provided by sklearn like cross-validation, k-fold cross validation, GridSearchCV etc.

I used StratifiedShuffleSplit for splitting data into train and test sets. The method is a merge of StratifiedKFold and ShuffleSplit and provides splits of data into train and test sets. Then GridSearchCV was used to find optimal parameters to get the best value for recall. Fitted parameters were used for the final classifier.

6.***Give at least 2 evaluation metrics and your average performance for each of them.***

The chosen classifier (KnearestNeigbors) showed the following performance on test set:
Accuracy: 0.87071 Precision: 0.58979 Recall: 0.31200 F1: 0.40811 F2: 0.34445

Accuracy tells us that the classifier marked POI correctly in 87% cases.

Precision value close to 0.6 tells us that number of true positives and false positives are close. That

means that there is 60% chance that POI will be marked correctly..

Recall is a measure of completeness and the value 0.3 tells us that the trained classifier identified correctly 30% of all POIs.