Meets Specifications

Congratulations on meeting all specifications! You did really well to pass this project.

The Enron Corpus is a very useful dataset used for solving a range of natural language processing tasks. The dataset that was provided for this project is a processed version of the original corpus. The dataset is also hosted on Kaggle. Here's the link: https://www.kaggle.com/wcukierski/enron-email-dataset. Take a look at the work other people have done with this dataset by clicking on kernels.

Interesting articles

- http://www.newyorker.com/magazine/2017/07/24/what-the-enron-e-mails-say-about-us
- https://blog.quandl.com/email-receipts-predicted-gopros-q3-earnings
- https://blog.quandl.com/alternative-data-action-email-receipts

Good luck with the rest of the program!

Quality of Code

Code reflects the description in the answers to questions in the writeup. i.e. code performs the functions documented in the writeup and the writeup clearly specifies the final analysis strategy.

poi_id.py can be run to export the dataset, list of features and algorithm, so that the final algorithm can be checked easily using tester.py.

Understanding the Dataset and Question

Student response addresses the most important characteristics of the dataset and uses these characteristics to inform their analysis. Important characteristics include:

- total number of data points
- allocation across classes (POI/non-POI)
- number of features used
- are there features with many missing values? etc.

Student response identifies outlier(s) in the financial data, and explains how they are removed or otherwise handled.

Optimize Feature Selection/Engineering

At least one new feature is implemented. Justification for that feature is provided in the written response. The effect of that feature on final algorithm performance is tested or its strength is compared to other features in feature selection. The student is not required to include their new feature in their final feature set.

Univariate or recursive feature selection is deployed, or features are selected by hand (different combinations of features are attempted, and the performance is documented for each one). Features that are selected are reported and the number of features selected is justified. For an algorithm that supports getting the feature importances (e.g. decision tree) or feature scores (e.g. SelectKBest), those are documented as well.

If algorithm calls for scaled features, feature scaling is deployed.

Pick and Tune an Algorithm

At least two different algorithms are attempted and their performance is compared, with the best performing one used in the final analysis.

Response addresses what it means to perform parameter tuning and why it is important.

At least one important parameter tuned with at least 3 settings investigated systematically, or any of the following are true:

- GridSearchCV used for parameter tuning
- Several parameters tuned
- Parameter tuning incorporated into algorithm selection (i.e. parameters tuned for more than one algorithm, and best algorithm-tune combination selected for final analysis).

Validate and Evaluate

At least two appropriate metrics are used to evaluate algorithm performance (e.g. precision and recall), and the student articulates what those metrics measure in context of the project task.

Suggestion

The response that explains what recall measures is much better. Essentially, recall gauges how many POIs the model is able to flag out of all the POIs available in the dataset.

Response addresses what validation is and why it is important.

Performance of the final algorithm selected is assessed by splitting the data into training and testing sets or through the use of cross validation, noting the specific type of validation performed.

When tester.py is used to evaluate performance, precision and recall are both at least 0.3.