Dante Ruiz

Enron Submission Free-Response 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"]***

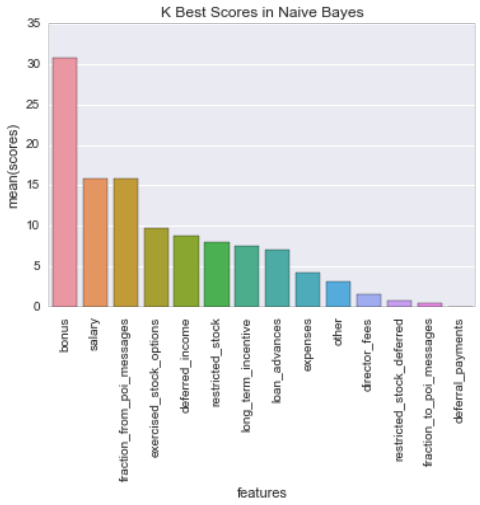
   The objective of the project is to *train and tune a machine learning algorithm that can predict which employees from Enron are "Persons of Interest (POI)" using financial and email data. In this context, machine learning is useful because it has a set of algorithms, supervised learning algorithms, that can achieve this objective.* For instance, the financial and email data provided have 14 features by 143 employees, and we already have a list of employees classified by POIs (POI=18 and non-POI=125). Therefore, it is possible to make an algorithm to learn the patterns of a split of the data, to finally make predictions of POIs on the other split. It is worth to mention that the data provided had to be cleaned first from NaNs and outliers that corresponded to entries different from persons, such as total and a travel agency. NaNs were converted to zeros and outliers were removed before preprocessing the final data.

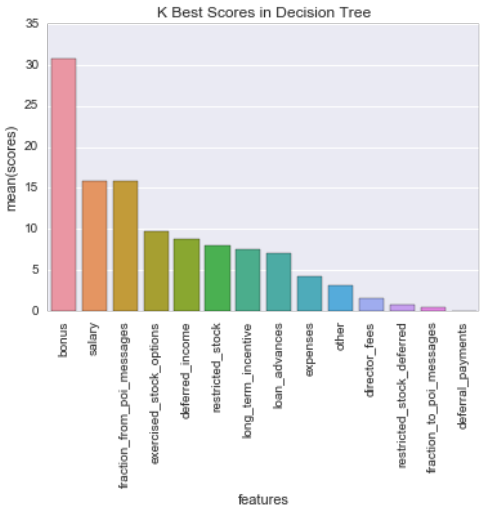
2. ***What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what 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"]***

   To accomplish the objective of the project I experimented with four algorithms: naïve bayes, decision tree, logistic regression and k-nearest neighbors. I conducted an initial estimation using all the features. I found out that each of the algorithms offered different results and that the results were dependent on the training and testing split. Bearing this in mind I fine tuned the algorithms considering the following:

   - ✓ New features: I engineered two additional features "Fraction of emails written to POI from total written emails" and "Fraction of emails received from POI from total received emails". The rational for including this features in the project is that in some sense they reduce the dimensionality of the emails data and also these relative measurements sort of normalize this information across the rest of the employees.
   - ✓ Feature scaling: I scaled the features using minimum and maximum method expecting for the data to be more normalized across the wide array of fields.
   - ✓ Feature selection: I tested each algorithm with different combinations of the 14 features using the Select K Best features as a method of decision making.
   - ✓ Parameter tuning: I systematically tried different parameters on the algorithms.

   Bearing this in mind, I found when using GridSearch and Cross Validation the best estimated classifier for each algorithm, I noticed that each had a different set of features. The following two figures show the mean scores of each features for Naïve Bayes and the Decision Tree classifiers, the two algorithms that yielded the best performance.





- ✓ The Naïve Bayes best classifier considers: salary, bonus, exercised_stock_options and fraction_from_poi_messages.
- ✓ The Decision Tree classifier considers: salary, bonus, long_term_incentive, deferred_income, loan_advances, other, expenses, director_fees, exercised_stock_options, restricted_stock and fraction_from_poi_messages.

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"]***

   As previously mentioned I obtained two algorithms that yielded the best performance. Their, recall, precision and F score are the same. The other Logit and KNN were not very good at predicting True Positives which is the final objective of the project.

   For the tester.py evaluation script I chose the Decision Tree classifier because under shuffle stratified sampling it produced the best F score.

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"]***

   Tuning the parameters of an algorithm can be tricky, as tuning more parameters doesn't necessarily traduce in better performance scores, and on the contrary it can reduce performance. I played with different set of parameters and I chose the set of parameters that helped increase the performance metrics. The following are the set of parameters I modified:

   - ✓ Decision tree:
     - Split node strategy: Best strategy or random strategy.
     - Minimum number of samples required to split an internal node: 2 or 3
   - ✓ Logistic Regression:
     - Inverse of regularization strength (C): { 1,1.1,1.2,1.3,1.4, 1.5,1.6,1.7,1.8,1.9,2 }
     - the algorithm to use in the optimization problem (solver): { 'liblinear','sag' }
   - ✓ K-nearest neighbor
     - number of neighbors used for the classification: ranging from 4 to 7.

- weight function used in prediction: { 'uniform','distance' }

In order to select this parameters, I conducted a marginal analysis at how each parameter affected the F score metric.

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"]***

To validate the performance of my algorithms I conducted the following steps:

- ✓ Step 1: Split data in training and testing sets using a normal train_test_split considering a test size of 30%.
- ✓ Step 2: To produce the fine tuned algorithms I used stratified k fold cross validation in the training sets. I set the number of folds to 10 as it is suggested that on average this

Some common mistakes I was aware of are related with how to split the data in training and testing set. Also, in ensuring that classes are proportionally allocated between training and testing set. Therefore, the initial data was kind of unsorted by class of POIs, and train_test_split was a good choice. In the second step in order to ensure that each of the classes was correctly allocated k fold cross validation was used as this functionality takes charge of doing this task for me.

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 metrics used to compare the models are accuracy, precision, recall and F scores. Also the number of True Positives, False Positives, False Negatives and True Negatives are captured. Both Decision Tree (DT) and Naïve Bayes (NB) reached the best performance metrics after the feature selection and the parameter tuning.

The following table shows that the accuracy of DT and NB is of 90%; however, this is not the most appropriate metric in the light that there are very few POIs in the dataset, and therefore in the testing set. Meaning that the accuracy score is kind of biased in the sense that it is praising the prediction of true negatives, and that is not exactly the purpose of the project.

	Accuracy score	Precision score	Recall score	F score	Predictions	True positives	False positives	False negatives	True negatives
Decision Tree	0.9	0.6	0.5	0.55	43	3	2	3	35
Naive Bayes	0.9	0.6	0.5	0.55	43	3	2	3	35
Logit	0.9	0.2	1.0	0.33	43	1	4	0	38
KNN	0.9	0.0	0.0	0.00	43	0	5	1	37

To overcome the accuracy issue, the precision score and recall score play a better role for interpretation. A recall score of 0.5 says that our models correctly predict 50% of the time POIs that are in fact POIs. Moreover, the precision score of 0.6 means that both classifiers are able to correctly predict 60% of POIs from all those employees that were predicted as POI. Overall, the precision score is better than the recall score meaning that whenever a POI gets flagged in the test set, I know with confidence that it is very likely to be a real POI and not a false alarm. Both classifiers reached a F score of 55% of flagging a POI when it is a POI and not flagging a POI when it is not a POI.