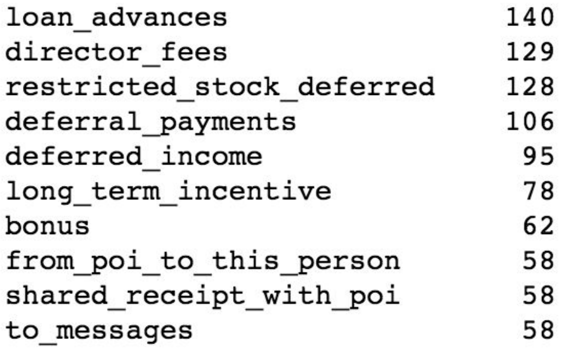
Enron Submission Free-Response Questions

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”]

   Enron scandal led the largest bankruptcy reorganization in American history in the October 2001. This project analyzed data collected from this famous event and built a model to predict whether a person was the ‘person of interest (poi)’ using the financial and email contacts information.
   The original dataset has 146 observations and 21 variables. 14 variables are financial information and 6 variables are email contacts information. The target variable is ‘poi’ indicates whether a person is ‘person of interest’ (‘poi’=1) or not (‘poi’=0).

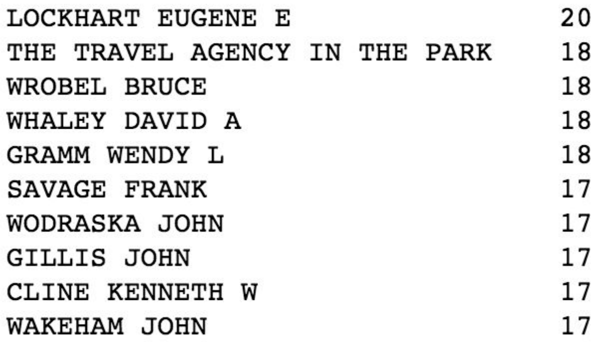
From an initial investigation of the dataset, following problems caught my attention:

- Incomplete Data: many variables contain missing values, for a relative small dataset with only 146 observations, the missing values can significantly affect the analysis.
  The top 10 features have most missing values are as below:



```
loan_advances                140
director_fees                129
restricted_stock_deferred    128
deferral_payments            106
deferred_income               95
long_term_incentive           78
bonus                         62
from_poi_to_this_person       58
shared_receipt_with_poi       58
to_messages                   58
```

  The top 10 observations have most missing values are as below:



```
LOCKHART EUGENE E                20
THE TRAVEL AGENCY IN THE PARK    18
WROBEL BRUCE                     18
WHALEY DAVID A                   18
GRAMM WENDY L                    18
SAVAGE FRANK                     17
WODRASKA JOHN                    17
GILLIS JOHN                      17
CLINE KENNETH W                  17
WAKEHAM JOHN                     17
```

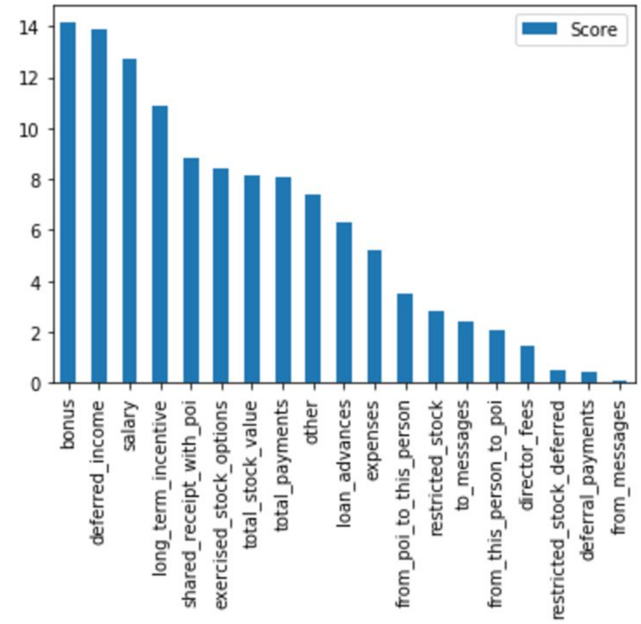
To handle this problem, I filled missing values with ‘0’, which is a reasonable guess since the obligated entries usually means 0.

- Imbalanced Data: from the frequency count of the labels. The count for ‘poi’ is only 18 out of 146. The imbalance may precision problems. To handle this problem, I use various evaluation metrics when selected the best model.
- Outliers: From the box-plots of some variables, I found there is one obvious outlier for almost all variables. By printing out the problematic row, I found the ‘name’ of the row is **‘TOTAL’**, which is a summary row for the dataset. There are two observations (**'BELFER ROBERT'** and **'BHATNAGAR SANJAY'**) have negative stock values which may introduce misleading information. According to the missing values summary, the observation **‘LOCKHART EUGENE E’** has 20 missing values, which means there’s no information about this person except the whether this is a ‘poi’ or not. **‘THE TRAVEL AGENCY IN THE PARK’** is not a person’s name and contains very limited information (18 missing values).
  To get more accurate analysis I removed above observations from the dataset.

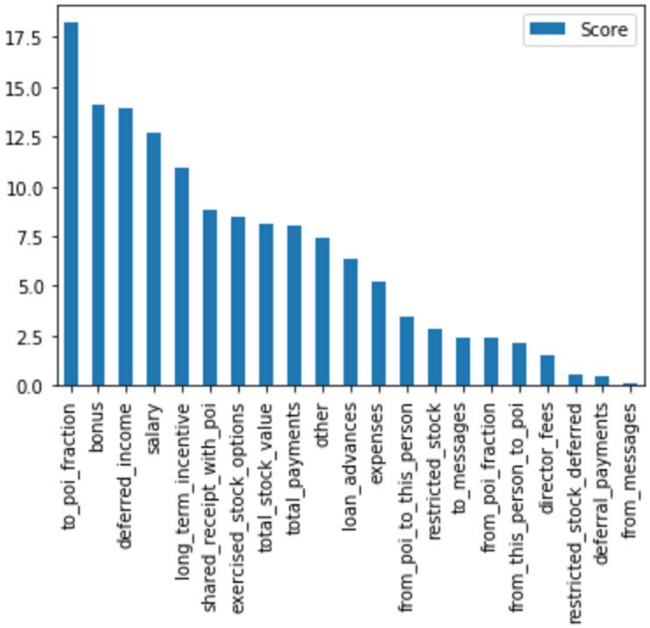
2. What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what feature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

- New Features:
  [“from_poi_fraction”, “to_poi_fraction”],
  “from_poi_fraction”= 'from_poi_to_this_person'/ “from_message”
  “to_poi_fraction”=”to_poi_from_this_person”/”to message”
  The reason to add the new features is to standardized email contact information since some roles tend to have more email communication in general. The proportion of their communication with the person of interest is more accurate to describe their interaction with the person of interest.

  Following bar plots illustrate the effect of adding new features. The new feature ‘to_poi_fraction’ has the highest score from the SelectKBest method. Therefore, in this project I kept the new features in my analysis.



Scores of features for original dataset



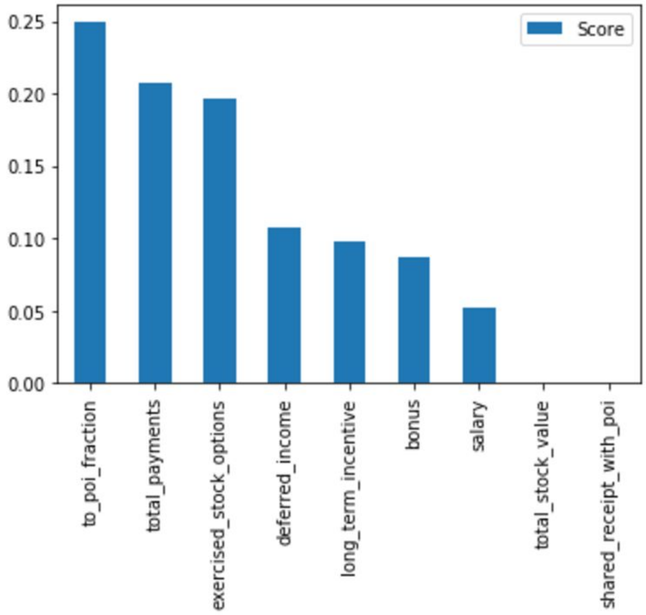
Scores of features for dataset with new features

- Feature scaling:
  SVC classifier is a distance based algorithm, therefore, for SVC classifier I applied a MinMaxScaler to scale all features to 0-1. No scaler was applied to Naive Bayes and Decision tree classifiers since they are independent of feature scale.

- Final feature_list:

  ['poi', 'salary', 'total_payments', 'bonus', 'deferred_income', 'total_stock_value', 'exercised_stock_options', 'long_term_incentive', 'shared_receipt_with_poi', 'to_poi_fraction']

  Features that are included in the final prediction model were selected by a GridSearchCV process on a Decision Tree Classifier Pipeline.



Importances of features in the final model

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 implemented three classifiers including Naive Bayes Gaussian Classifier, Decision Tree Classifier, and SVC Classifier in the project. Best parameters and features were selected via a GridSearchCV procedure for each algorithm, and the final models were tested using the provided test_classifer function. A comparison of the performances of three algorithms is as below:

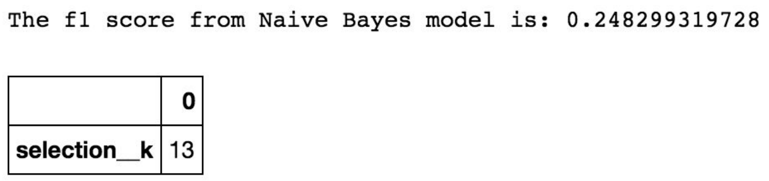
Algorithms	Accuracy	Recall	Precision	f1	f2
Naive Bayes	0.81620	0.30850	0.30989	0.30920	0.30878
Decision Tree	0.85400	0.31850	0.43511	0.36778	0.33654
SVC	0.79993	0.31900	0.30717	0.31298	0.31656

From the comparison, I decided to use the Decision Tree algorithm in the final model since it gave the best scores for all evaluation metrics.

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"]

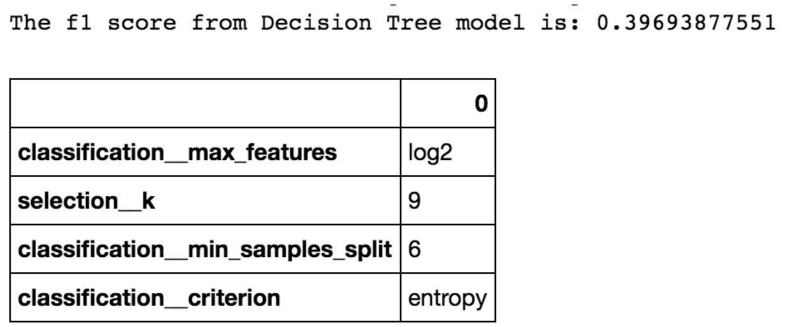
   The process of tuning the parameters of an algorithm is the process of helping the machine to find the most efficient way to 'learn' from the data. Since machine learning models are parameterized, their behaviors are highly dependent on the given parameters. Failed to tuned the model appropriately may end up with a longer learning time, waste of resources, and a poorly fitted model.

   For this project I used GridSearchCV to systematically select the optimal parameters. 'Select_models' file documents pipelines and parameters spaces for three algorithms. The optimal parameters and f1 scores for each algorithms are as below:



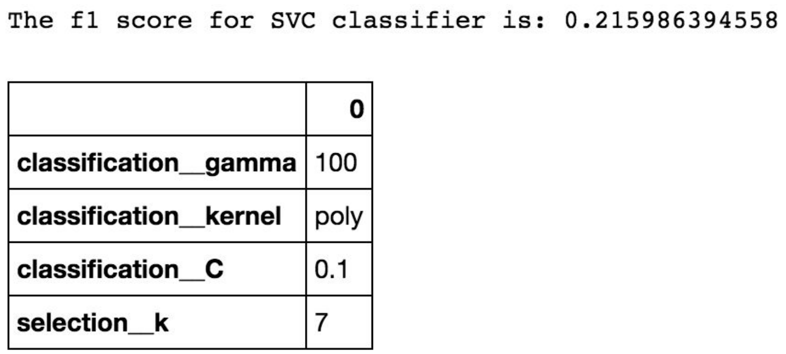
```
The f1 score from Naive Bayes model is: 0.248299319728
```

	0
selection__k	13



```
The f1 score from Decision Tree model is: 0.39693877551
```

	0
classification__max_features	log2
selection__k	9
classification__min_samples_split	6
classification__criterion	entropy



```
The f1 score for SVC classifier is: 0.215986394558
```

	0
classification__gamma	100
classification__kernel	poly
classification__C	0.1
selection__k	7

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"]

Validation is the process of assessing how the model will generalize to an independent data set. A classic mistake would be overfitting, which means you fit the model on the training dataset very well but the model performs poorly when fit to new data.

During the model selection process I use 10-fold cross-validation to validate my models by setting the parameter 'cv=10' in GridSearchCV. A similar validation procedure with 1000-fold cross validation was used in tester.py to evaluate the resulting final models that were selected. The final model was validated using 1000 randomized stratified cross-validation splits and selecting the parameters which performed best on average over the 1000 splits.

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 enron dataset has only 18 out of 146 person of interest. The imbalance makes accuracy a poor evaluation metric. So for evaluating models in this project, I used precision, recall, and F measures:

$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$, is the number of correct positive results(identify a person is 'poi' when the person is indeed a 'poi') divided by the number of all positive results (number of predicted 'poi')

$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$, is the number of correct positive results (identify a person is 'poi' when the person is indeed a 'poi') divided by the number of positive results that should have been returned (number of actual 'poi')

$f1 = 2 * \frac{Precision * Recall}{Precision + Recall}$, is a weighted average of the precision and recall. It is an efficient metric to evaluate imbalanced dataset.

The result for the final Decision Tree Model is:

Algorithms	Accuracy	Recall	Precision	f1	f2
Decision Tree	0.85400	0.31850	0.43511	0.36778	0.33654