P5- 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 Corporation was an American energy company which was once ranked sixth-largest energy company in the world. At Enron's peak, its shares were worth $90.75, but after the company declared bankruptcy on December 2, 2001, they plummeted to $0.67 by January 2002 due to widespread corporate fraud.

   To this day, many wonder how such a powerful business disintegrated almost overnight and how it managed to fool the regulators with fake, off-the-books corporations for so long. As part of this project, I would try investigating the Enron fraud using machine learning skills I learned to try building a person of interest identifier based on financial and email data made public as a result of the Enron scandal.

   As preprocessing to this project, the Enron email and financial data into a dictionary, where each key-value pair in the dictionary corresponds to one person. The dictionary key is the person's name, and the value is another dictionary, which contains the names of all the features and their values for that person. The features in the data fall into three major types, namely financial features, email features and POI labels.

   financial features: ['salary', 'deferral_payments', 'total_payments', 'loan_advances', 'bonus', 'restricted_stock_deferred', 'deferred_income', 'total_stock_value', 'expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'restricted_stock', 'director_fees'] (all units are in US dollars)

   email features: ['to_messages', 'email_address', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi'] (units are generally number of emails messages; notable exception is ‘email_address’, which is a text string)

   POI label: [‘poi’] (boolean, represented as integer)

   The dataset contained:

   Number of data points: 146
   Number of person of interest: 18
   Number of non-person of Interest: 128
   Number of features per person: 21

   Missing values :

Salary	51
to_messages	60

deferral_payments	107
total_payments	21
long_term_incentive	80
loan_advances	142
Bonus	64
restricted_stock	36
restricted_stock_deferred	128
total_stock_value	20
Expenses	51
from_poi_to_this_person	60
exercised_stock_options	44
from_messages	60
Other	53
from_this_person_to_poi	60
Poi	0
deferred_income	97
shared_receipt_with_poi	60
email_address	35
director_fees	129

I identified the below 3 outliers in the data set through exploratory data analysis:

- **TOTAL:** This outlier was the sum total of all the payments and stock values carried out by people in the Enron financial data set. So it didn't hold any further significance in my analysis, so I removed it.
- **THE TRAVEL AGENCY IN THE PARK**: This outlier doesn't seem to correspond to a real person with all the payments and stock value data point having 0 values barring other expenses and total payments so it this data point has also been removed from the data set.
- **LOCKHART EUGENE E**: This person has all the payments and stock value data point values as 0 so I removed it from the data set.

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"]

I believe for people to be involved in the fraud, they must have had some financial communication with POI. So to identify features like this which could help in my further analysis, I created the following 3 new features on the basis of e-mails exchanged between non POI and POI:

- fraction_from_poi: proportion of messages received from POI

- fraction_to_poi: proportion of messages sent to POI
- messages_with_poi: sum total of messages exchanged with POI

Features like from_poi_to_this_person, from_this_person_to_poi, shared_receipt_with_poi used to engineer new features can potentially create a data leakage. This can hamper the model's ability to generalize on unseen data and can give the false effect that the model performs really well. I tested the impact of my selected features on my final Decision Tree classifier:

- **With all the features**(newly created+already provided)
  Accuracy: 0.83940 Precision: 0.38348 Recall: 0.33650 F1: 0.35846 F2: 0.34495
- Without fraction_from_poi
  Accuracy: 0.81327 Precision: 0.28293 Recall: 0.26100 F1: 0.27152 F2: 0.26511
- Without fraction_to_poi
  Accuracy: 0.84253 Precision: 0.34095 Recall: 0.19400 F1: 0.24729 F2: 0.21230
- Without messages_with_poi
  Accuracy: 0.85773 Precision: 0.46539 Recall: 0.45050 F1: 0.45783 F2: 0.45340
- **Without new features**
  Accuracy: 0.87133 Precision: 0.52489 Recall: 0.36900 F1: 0.43335 F2: 0.39230

On the basis of these scores, I can determine that without my newly chosen features, the accuracy, precision and recall had been better than with it. So I have chosen to not use my newly created features in my analysis any further.

I used sklearn's SelectKBest for feature selection in Pipeline with Grid Search CV to determine optimal k value and that turned out to be k= 13

Selected features:
salary: 18.289684
total_payments: 8.772778
bonus: 20.792252
deferred_income: 11.458477
total_stock_value: 24.182899
expenses: 6.094173
exercised_stock_options: 24.815080
other: 4.187478
long_term_incentive: 9.922186
restricted_stock: 9.212811
from_poi_to_this_person: 5.243450
from_this_person_to_poi: 2.382612
shared_receipt_with_poi: 8.589421

To scale the features, I used MinMaxScaler from sklearn to normalize the features as financial/email data numbers vary in high magnitude.

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 tried algorithms like **Nearest Neighbors, SVM(Linear/Rbf), Decision Tree, Random Fores, AdaBoost,Naive Bayes,Extra Trees** for my classifiers with their default values. Following are the result of the algorithms:

Algorithm	Accuracy	Precision	Recall
K Nearest Neighbors	0. 860465	0.0	0.0
Linear SVM	0. 883721	0.0	0.0
RBF SVM	0. 883721	0.0	0.0
Decision Tree	0. 837209	0. 500000	0. 400000
Random Forest	0. 883721	0. 333333	0. 200000
AdaBoost	0. 813953	0. 200000	0. 200000
Naive Bayes	0. 883721	0.500000	0. 600000
Extra Trees	0. 837209	0. 250000	0. 200000

From these Algorithms since Nearest K neighbor and SVM show precision/recall of 0, these are ignored for my further analysis. Random forest/extra trees seem similar to decision trees so I ignore these as well. From Naives Bayes and Decision Tree which have similar accuracies, I chose to go with Decision Tree algorithm

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 ultimate goal of machine learning is to make a machine system that can automatically build models from data without requiring tedious and time consuming human involvement. One of the difficulties is that learning algorithms (eg. decision trees, random forests, clustering techniques, etc.) requires you to set parameters before you use the models. Our goal, is usually to set those parameters to such optimal values that enable you to complete a learning task in the best way possible. Thus, tuning an algorithm or machine learning technique, can be simply thought of as process which one goes through in which they optimize the parameters that impact the model in order to enable the algorithm to perform the best.

The more tuned the parameters of an algorithm would be, more biased will it be towards the training and testing data. One tradeoff for this could be that it might lead to more fragile models that overfit the test data and don't perform well as expected.

I tuned the parameters of Decision Tree algorithm which I had chosen using scikit-learn's GridSearchCV which exhaustively considers all parameter combinations,to which give the best performance:

For my DecisionTree classifier, I chose the following parameters:

```
parameters = {'max_depth': [1,2,3,4,5,6,8,9,10],
          'min_samples_split':[2,3,4,5,6,7,8],
          'min_samples_leaf':[1,2,3,4,5,6,7,8,9,10],
          'criterion':('gini', 'entropy')}
```

GridSearch gave the following best estimator:

```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=2,
        max_features=None, max_leaf_nodes=None,
        min_impurity_split=1e-07, min_samples_leaf=5,
        min_samples_split=2, min_weight_fraction_leaf=0.0,
        presort=False, random_state=42, splitter='best')
```

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 consists of techniques for assessing that our predictive model generalizes to the entire dataset normally. The advantage of this is that it gives performance of an independent dataset and it serves as a check on over-fitting. A classic mistake one make with validation is over-fitting, case where our model performs very well on training data and poorly on the cross-validation and testing dataset. Meaning our algorithm does not perform well on generalized datasets.

In order to validate my analysis, I used tester.py's test_classifier which used StratifiedShuffleSplit with folds=1000 and random state=42 for providing metrics for my classifier.

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"]

For my DecisionTree classifier, I used the following evaluation metrics:

- Precision: 0.36524
- Recall: 0.29950

Precision is defined as (true positive)/(true positive+false positive) while recall is defined by (true positive)/(true positive+false negative)

With precision score 0.37, it tells us if this model predicts 100 POIs, there would be 37 percent chance of people actually being POIs. With recall score of 0.30, this model finds 30% of all real POIs in prediction. Since the dataset is small is size with limited number of observations, accuracy may not be a good measure.