REPORT FROM FINAL PROJECT – INTRODUCTION TO MACHINE LEARNING

COURSE: DATA SCIENCE FOUNDATIONS II

STUDENT: LEANDRO BARUCH

Identify Fraud from Enron Dataset

*"**Enron Corporation** was an American energy, commodities, and services company based in Houston, Texas. It was founded in 1985 as a merger between Houston Natural Gas and InterNorth, both relatively small regional companies. Before its bankruptcy on December 3, 2001, Enron employed approximately 29,000 staff and was a major electricity, natural gas, communications and pulp and paper company, with claimed revenues of nearly $101 billion during 2000. Fortune named Enron "America's Most Innovative Company" for six consecutive years.*

At the end of 2001, it was revealed that Enron's reported financial condition was sustained by institutionalized, systematic, and creatively planned accounting fraud, known since as the Enron scandal. Enron has since become a well-known example of willful corporate fraud and corruption. The scandal also brought into question the accounting practices and activities of many corporations in the United States and was a factor in the enactment of the Sarbanes–Oxley Act of 2002. The scandal also affected the greater business world by causing the dissolution of the Arthur Andersen accounting firm, which had been Enron's main auditor for years.

*Enron filed for bankruptcy in the Southern District of New York in late 2001 and selected Weil, Gotshal & Manges as its bankruptcy counsel. It ended its bankruptcy during November 2004, pursuant to a court-approved plan of reorganization. A new board of directors changed the name of Enron to **Enron Creditors Recovery Corp**., and emphasized reorganizing and liquidating certain operations and assets of the pre-bankruptcy Enron. On September 7, 2006, Enron sold Prisma Energy International Inc., its last remaining business, to Ashmore Energy International Ltd. (now AEI)." – From Wikipedia, the free encyclopedia*

The objective of this project, is use both financial and email data from 146 executives, and use those data to identify Persons of Interest (POIs). For that, I will analyse the dataset, all the data features, understand their correlations and see which are the best for this purpose. After that I will create a Machine Learning Model to identify the POIs based on those features and check their efficiency.

Summary

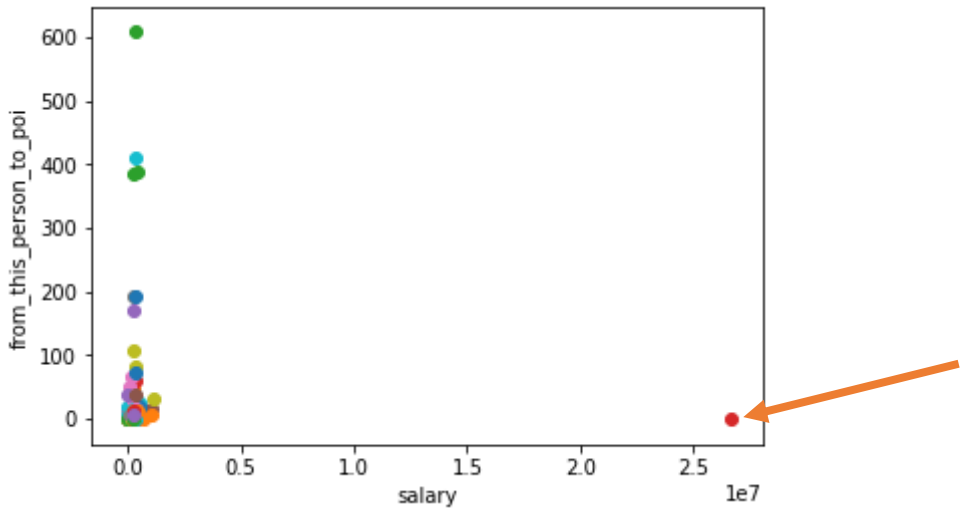
Exploring the Dataset

During dataset exploration, I have found some interesting information:

- We have 146 registers in our dataset, one of them was the 'TOTAL', so it was removed and now we have 145 registers;
- From 145 people, 18 of them are flagged as POI, they are:
    - Hannon Kevin P, Colwell Wesley, Rieker Paula H, Kopper Michael J, Shelby Rex, Delainey David W, Lay Kenneth L, Bowen Jr Raymond, Belden Timothy N, Fastow Andrew S, Calger Christopher F, Rice Kenneth D, Skilling Jeffrey K, Yeager F Scott, Hirko Joseph, Koenig Mark E, Causey Richard A and Glisan JR Ben F;
- The features we have at our disposal are:
    - salary, to_messages, deferral_payments, total_payments, exercised_stock_options, bonus, restricted_stock, shared_receipt_with_poi, restricted_stock_deferred, total_stock_value, expenses, loan_advances, from_messages, other, from_this_person_to_poi, poi, director_fees, deferred_income, long_term_incentive, email_address, from_poi_to_this_person

Checking those features, of course those which more call attention are Salary, Bonus, Total Stock Value. These features can tell us which person are receiving more money from Enron, and outliers may be guilty in this case.

Check #1

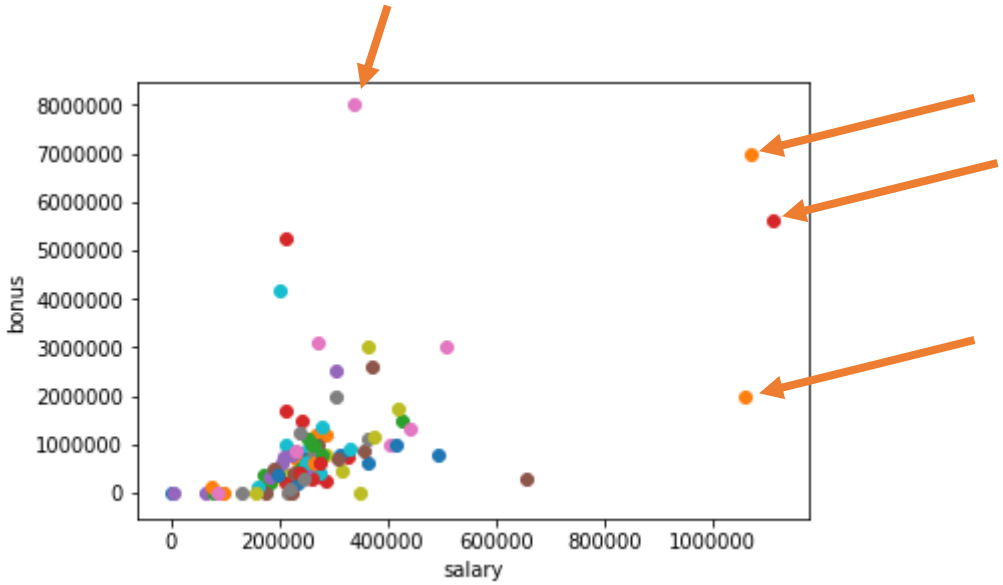


Salary x Messages to POI

We can see a big outlier.

After checking it, its name is 'TOTAL'. Probably when the dataset was created someone forgot to delete the Total Row. I just deleted it.

Check #2



Salary x Bonus

Now we can see 4 more outliers. Three of them with the higher salaries in company, and one with the highest bonus.

The person with the highest bonus, is Lavorato John J, head of operations with a bonus of $8.000.000. And he is not flagged as POI. Suspicious…

The three top salaries of Enron are:

- Skilling Jeffrey K, CEO
  - Salary: $1.111.258
- Lay Kenneth L, founder and CEO
  - Salary: $1.072.321
- Frevert Mark A, CEO
  - Salary: $1.060.932

Those three occupied the highest role in the company, but it means that big difference in salary, since we are talking only about the high level of executives of Enron? Also, Frevert Mark A was not flagged as POI.

Seeing those differences, make me wonder how much is the average salary and bonus for high executive level in Enron. And the result is:

- Average salary: $ 284.088
- Average bonus: $ 1.201.773

It means the examples above are way higher the average.

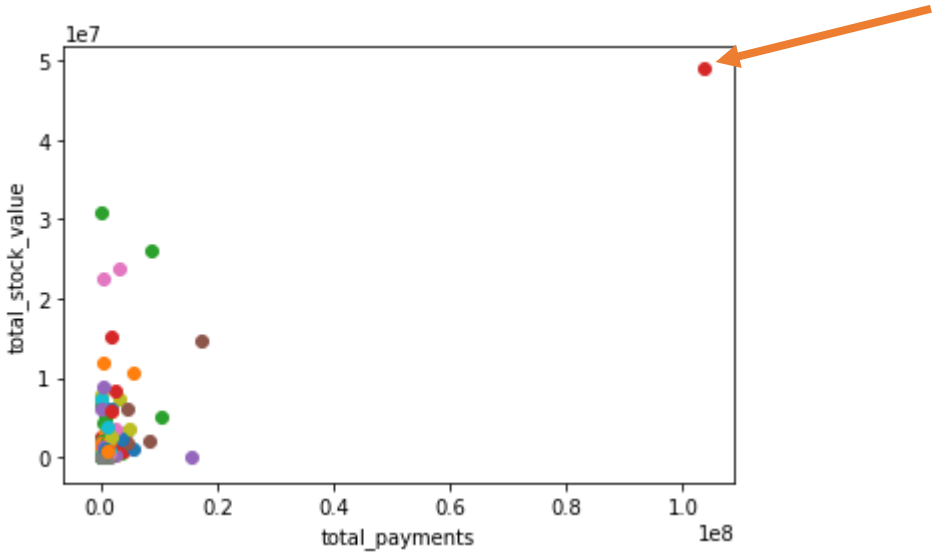
Also, to be honest, this average salary looks weird. The Total was around $26kk and dividing it per number of people, I expected another value.

So, I have checked for 'NaN' values and I have found this:

- Amount of NaN salary: 51
- Amount of NaN bonus: 64

Looks like the data is far from complete and it may interfere the capability of our model detect new POIs. We will have to use the data from e-mails to be more precise.

Check #3



Total Payments x Total Stock Value

Now, considering all payments and stock value, a big outlier appears. And it is his second time, Lay Kenneth L with the big value of $103.559.793 in total payments and $49.110.078 of total stock value.

Features with missing data

As we see in Check#2, salary and bonus have a lot of missing values. Let's check if we have other cases with other features.

After a check, there is a lot of missing data in our dataset, below the results:

- Amount of NaN to_messages: 59
- Amount of NaN deferral_payments: 107
- Amount of NaN total_payments: 21
- Amount of NaN exercised_stock_options: 44
- Amount of NaN restricted_stock: 36
- Amount of NaN shared_receipt_with_poi: 59
- Amount of NaN restricted_stock_deferred: 128
- Amount of NaN total_stock_value: 20
- Amount of NaN expenses: 51
- Amount of NaN loan_advances: 142

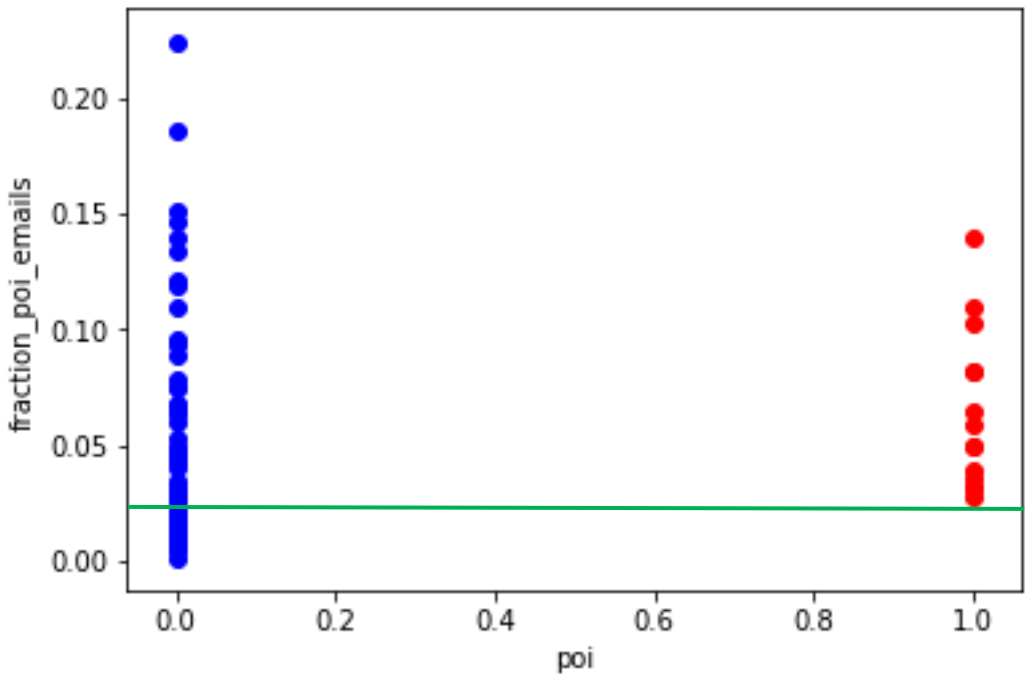
- Amount of NaN from_messages: 59
- Amount of NaN other: 53
- Amount of NaN from_this_person_to_poi: 59
- Amount of NaN director_fees: 129
- Amount of NaN deferred_income: 97
- Amount of NaN long_term_incentive: 80
- Amount of NaN from_poi_to_this_person: 59

Some features are expected to have missing values like 'loan_advances', 'long_term_incentives', 'from_this_person_to_poi' and 'from_poi_to_this_person', but I see it's a big amount event to other features. It can impact in the performance of our models, that's why we will need to do various tests to find the best scenario.

Checking the Features

Now we have removed an outlier and detected some behaviors from dataset, we need to check the features, and see what feature is more important in detecting a POI.

As I said before, use the amount of e-mails sent and received from POIs may be a good indicative to find a POI. Also, we have a few examples that received a big amount of cash but it's not a POI.



As we can see with the green line, POIs tend to exchange e-mail among them at least 3% of their total e-mails while not POIs sometimes have 0%.

Also, we have a considerable difference between their averages:

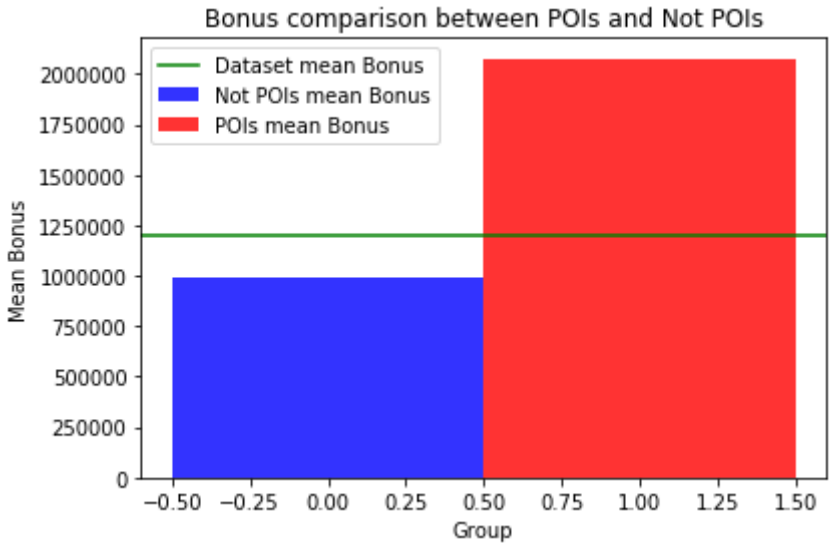
POIs use to have an average of 5% of e-mails exchanged among them, while Not POIs uses to have an average of 2,5% of e-mails exchanged with POIs.

It's worth to use this feature in our classifier to see if improves it.

POIs and Not POIs comparison

According the information we got earlier regards the average salary and bonus, lets check in a plot, how is the difference of average salary and bonus considering the two groups.

As we can see in the image above, the mean salary from people flagged as POIs is way higher than Not POIs and the dataset mean salary.



Now using bonus information. The difference is even higher in favour of POIs. Looks like fat bonus is a good way to convince people to change side.

The conclusion after seeing those informations is that the people flagged as POI was receiving much more money than the not POIs.

In numbers, we have:

Salary difference from POI and Not POI: 1.46

Bonus difference from POI and Not POI: 2.10

Metrics

First, I believe we have too much features for the training points we have.

Let's check:

```
Number of training points:  100
```

```
Number of features:  20
```

We will be looking for the training and prediction time that takes to created the model and some metrics;

Accuracy: is the fraction of predictions our model got right. Accuracy has the following formula:

$$\frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Precision: proportion of positive identifications was actually correct. Precision has the following formula:

$$\frac{True\ positives}{True\ positives\ + False\ positives}$$

Recall: proportion of actual positives was identified correctly. It's formula is:

$$\frac{True\ positives}{True\ positives + False\ negatives}$$

F1 Score: it is the balance between Precision and Recall. The F1 Score formula is:

$$F1 = 2\ X\ \frac{Precision * Recall}{Precision + Recall}$$

<u>The performance requested for this project, is Precision and Recall above 0.30</u>

Checking Classifiers with all features

SVM LinearSVC (failed)

```
        Accuracy: 0.74020       Precision: 0.21422      Recall: 0.35550 F1: 0.26734
F2: 0.31407
        Total predictions: 15000        True positives:  711    False positives: 2608
False negatives: 1289   True negatives: 10392
```

Naive Bayes (failed)

```
        Accuracy: 0.74713       Precision: 0.23578      Recall: 0.40000 F1: 0.29668
F2: 0.35109
        Total predictions: 15000        True positives:  800    False positives: 2593
False negatives: 1200   True negatives: 10407
```

Decision Tree (failed)

```
        Accuracy: 0.80020       Precision: 0.23098      Recall: 0.21400 F1: 0.22216
F2: 0.21719
        Total predictions: 15000        True positives:  428    False positives: 1425
False negatives: 1572   True negatives: 11575

Feature Ranking:
1 feature salary (0.226)
2 feature to_messages (0.224643000805)
3 feature deferral_payments (0.193350505689)
4 feature total_payments (0.148191898441)
5 feature exercised_stock_options (0.123346130488)
6 feature bonus (0.0675824026312)
7 feature restricted_stock (0.0168860619469)
8 feature shared_receipt_with_poi (0.0)
9 feature restricted_stock_deferred (0.0)
10 feature total_stock_value (0.0)
```

Random Forest (failed)

```
        Accuracy: 0.85587       Precision: 0.37615      Recall: 0.12300 F1: 0.18538
F2: 0.14213
        Total predictions: 15000        True positives:  246    False positives:  408
False negatives: 1754   True negatives: 12592
```

So, our F1 Score is too low as expected. Let's use the more important features from Decision Tree and check again.

Checking Classifiers with 5 top features

The five features that we will use in this test are: salary, to messages, deferral payments, total payments and exercised stock options.

SVM LinearSVC (failed)

```
        Accuracy: 0.62033       Precision: 0.12365      Recall: 0.30350 F1: 0.17571
F2: 0.23511
        Total predictions: 15000        True positives:  607    False positives: 4302
False negatives: 1393   True negatives: 8698
```

Naive Bayes (failed)

```
        Accuracy: 0.84753       Precision: 0.35607      Recall: 0.17750 F1: 0.23690
F2: 0.19729
        Total predictions: 15000        True positives:  355    False positives:  642
False negatives: 1645   True negatives: 12358
```

Decision Tree – (approved)

```
        Accuracy: 0.82187       Precision: 0.33954      Recall: 0.35550 F1: 0.34734
F2: 0.35219
        Total predictions: 15000        True positives:  711    False positives: 1383
False negatives: 1289   True negatives: 11617
```

Random Forest (failed)

```
        Accuracy: 0.86233       Precision: 0.45649      Recall: 0.17050 F1: 0.24827
F2: 0.19492
        Total predictions: 15000        True positives:  341    False positives:  406
False negatives: 1659   True negatives: 12594
```

Checking Classifiers with 3 top features

The three features that we will use in this test are: salary, to messages and deferral payments.

SVM LinearSVC (failed)

```
        Accuracy: 0.64317       Precision: 0.15277      Recall: 0.25100 F1: 0.18994
F2: 0.22240
        Total predictions: 12000        True positives:  502    False positives: 2784
False negatives: 1498   True negatives: 7216
```

Naive Bayes (failed)

```
        Accuracy: 0.77258       Precision: 0.22239      Recall: 0.14600 F1: 0.17628
F2: 0.15677
        Total predictions: 12000        True positives:  292    False positives: 1021
False negatives: 1708   True negatives: 8979
```

Decision Tree (failed)

```
        Accuracy: 0.74508       Precision: 0.19481      Recall: 0.16900 F1: 0.18099
F2: 0.17360
        Total predictions: 12000        True positives:  338    False positives: 1397
False negatives: 1662   True negatives: 8603
```

Random Forest (failed)

```
        Accuracy: 0.79408       Precision: 0.13146      Recall: 0.04200 F1: 0.06366
F2: 0.04862
        Total predictions: 12000        True positives:   84    False positives:  555
False negatives: 1916   True negatives: 9445
```

No good results came from 3 features. Next scenario will be using the top feature, salary.

Checking Classifiers with the top features

The top feature that we will use in this test is salary.

SVM LinearSVC (failed)

```
        Accuracy: 0.32060       Precision: 0.20000      Recall: 0.79900 F1: 0.31992
F2: 0.49969
        Total predictions: 10000        True positives: 1598    False positives: 6392
False negatives:  402   True negatives: 1608
```

Naive Bayes (failed)

```
        Accuracy: 0.79660       Precision: 0.46398      Recall: 0.10950 F1: 0.17718
F2: 0.12925
        Total predictions: 10000        True positives:  219    False positives:  253
False negatives: 1781   True negatives: 7747
```

Decision Tree (failed)

```
        Accuracy: 0.70170       Precision: 0.25114      Recall: 0.24800 F1: 0.24956
F2: 0.24862
        Total predictions: 10000        True positives:  496    False positives: 1479
False negatives: 1504   True negatives: 6521
```

\Random Forest (failed)

```
        Accuracy: 0.71150       Precision: 0.23645      Recall: 0.19850 F1: 0.21582
F2: 0.20508
        Total predictions: 10000        True positives:  397    False positives: 1282
False negatives: 1603   True negatives: 6718
```

We could see in our last two scenarios, that decreasing the number of features didn't work well. Let's try using the 7 best features from 'DecisionTree feature_importances_'

Checking Classifiers with 7 top features

The seven features that we will use in this test are: salary, to messages, deferral payments, total payments, exercised stock options, bonus and restricted stock.

SVM LinearSVC (failed)

```
        Accuracy: 0.68353       Precision: 0.18862      Recall: 0.41600 F1: 0.25955
F2: 0.33519
        Total predictions: 15000        True positives:  832    False positives: 3579
False negatives: 1168   True negatives: 9421
```

Naive Bayes (failed)

```
        Accuracy: 0.84373       Precision: 0.35017      Recall: 0.20100 F1: 0.25540
F2: 0.21972
        Total predictions: 15000        True positives:  402    False positives:  746
False negatives: 1598   True negatives: 12254
```

Decision Tree (failed)

```
        Accuracy: 0.79840       Precision: 0.25526      Recall: 0.26700 F1: 0.26100
F2: 0.26457
        Total predictions: 15000        True positives:  534    False positives: 1558
False negatives: 1466   True negatives: 11442
```

Random Forest (failed)

```
        Accuracy: 0.85480       Precision: 0.37139      Recall: 0.12850 F1: 0.19094
F2: 0.14784
        Total predictions: 15000        True positives:  257    False positives:  435
False negatives: 1743   True negatives: 12565
```

Checking Classifiers with 10 top features

The ten features that we will use in this test are: salary, to messages, deferral payments, total payments, exercised stock options, bonus and restricted stock.

SVM LinearSVC (failed)

```
        Accuracy: 0.69007       Precision: 0.19587      Recall: 0.42650 F1: 0.26845
F2: 0.34520
        Total predictions: 15000        True positives:  853    False positives: 3502
False negatives: 1147   True negatives: 9498
```

Naive Bayes (failed)

```
        Accuracy: 0.66713       Precision: 0.22677      Recall: 0.62100 F1: 0.33222
F2: 0.46079
        Total predictions: 15000        True positives: 1242    False positives: 4235
False negatives:  758   True negatives: 8765
```

Decision Tree (failed)

```
        Accuracy: 0.80540       Precision: 0.26851      Recall: 0.26650 F1: 0.26750
F2: 0.26690
        Total predictions: 15000        True positives:  533    False positives: 1452
False negatives: 1467   True negatives: 11548
```

Random Forest (failed)

```
        Accuracy: 0.85853       Precision: 0.42078      Recall: 0.16200 F1: 0.23394
F2: 0.18472
        Total predictions: 15000        True positives:  324    False positives:  446
False negatives: 1676   True negatives: 12554
```

Checking Classifiers with new feature

In this scenario, we will use poi plus our top 1 feature and our new feature. So, we will have: fraction poi emails and salary.

SVM LinearSVC (failed)

```
        Accuracy: 0.73400       Precision: 0.22604      Recall: 0.19100 F1: 0.20705
F2: 0.19711
        Total predictions: 11000        True positives:  382    False positives: 1308
False negatives: 1618   True negatives: 7692
```

Naive Bayes (failed)

```
        Accuracy: 0.81945       Precision: 0.51675      Recall: 0.10800 F1: 0.17866
F2: 0.12830
        Total predictions: 11000        True positives:  216    False positives:  202
False negatives: 1784   True negatives: 8798
```

Decision Tree (failed)

```
        Accuracy: 0.70227       Precision: 0.14326      Recall: 0.12800 F1: 0.13520
F2: 0.13079
        Total predictions: 11000        True positives:  256    False positives: 1531
False negatives: 1744   True negatives: 7469
```

Random Forest (failed)

```
        Accuracy: 0.77300       Precision: 0.16098      Recall: 0.05900 F1: 0.08635
F2: 0.06756
        Total predictions: 11000        True positives:  118    False positives:  615
False negatives: 1882   True negatives: 8385
```

Checking Classifiers with top 7 features from SelectKBest

As we saw, the new feature didn't improve our results, what did was using the top 7 features. We had a good result with SVM and Naive Bayes, but I believe we can get even better, maybe using SelectKBest.

After user SelectKBest, I get to another top 7 of features: deferra_payments, total_payments, exercised_stock_options, bonus, restricted_stock_deferred, director_fees, deferred_income

Lets try again and see if we get some improvement.

SVM LinearSVC (fail)

```
        Accuracy: 0.72440       Precision: 0.21348      Recall: 0.39750 F1: 0.27778
F2: 0.33905
        Total predictions: 15000        True positives:  795    False positives: 2929
False negatives: 1205   True negatives: 10071
```

Naive Bayes (fail)

```
        Accuracy: 0.34000       Precision: 0.15658      Recall: 0.90050 F1: 0.26678
F2: 0.46175
        Total predictions: 15000        True positives: 1801    False positives: 9701
False negatives:  199   True negatives: 3299
```

Decision Tree (approved)

```
        Accuracy: 0.81393       Precision: 0.31630      Recall: 0.34050 F1: 0.32796
F2: 0.33537
        Total predictions: 15000        True positives:  681    False positives: 1472
False negatives: 1319   True negatives: 11528
```

Random Forest (fail)

```
        Accuracy: 0.87133       Precision: 0.53715      Recall: 0.25300 F1: 0.34398
F2: 0.28293
        Total predictions: 15000        True positives:  506    False positives:  436
False negatives: 1494   True negatives: 12564
```

Considering all tests made, Decision Tree was approved in 2 scenarios, using top 5 features and the top 7 features selected by SelectKBest.

Also, RandomForest had promising results, and maybe tuning it with GridSearchCV we can reach our objective

Checking DecisionTree and RandomForest with adjusted parameters using GridSearchCV

DecisionTree final test

The parameters selected are:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=200,
            max_features=2, max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=2,
            min_samples_split=0.1, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='best')
```

The result was next to using default parameters but worse Recall, as you can see below:

```
        Accuracy: 0.85447       Precision: 0.41953      Recall: 0.23850 F1: 0.30411
F2: 0.26103
        Total predictions: 15000        True positives:  477    False positives:  660
False negatives: 1523   True negatives: 12340
```

DecisionTree Conclusion

Our best result was using Top 5 features from DecisionTree, and default parameters as you can see in the Overall Table

RandomForest final test

Parameters suggested by GridSearchCV:

```
rfc = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=30, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=3,
            min_weight_fraction_leaf=0.0, n_estimators=20, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

Our result was improved and approved by the rounded Recall value. The other results are better than DecisionTree, but since RandomForest was approved because the rounded value, I will choose DecisionTree as our best classifier.

```
        Accuracy: 0.87687       Precision: 0.57537      Recall: 0.29200 F1: 0.38740
F2: 0.32390
        Total predictions: 15000        True positives:  584    False positives:  431
False negatives: 1416   True negatives: 12569
```

Check the table of Overall Results

	All Features					Top 5 Features				
	Accuracy	Precision	Recall	F1	F2	Accuracy	Precision	Recall	F1	F2
SVM	0,74	0,21	0,34	0,26	0,3	0,62	0,12	0,29	0,17	0,23
Naive Bayes	0,74	0,24	0,4	0,3	0,35	0,85	0,36	0,18	0,24	0,2
DecisionTree	0,8	0,23	0,22	0,22	0,22	0,82	0,34	0,36	0,35	0,35
RandomForest	0,85	0,34	0,11	0,17	0,13	0,86	0,44	0,17	0,24	0,19

	Top 7 Features					Top 10 Features				
	Accuracy	Precision	Recall	F1	F2	Accuracy	Precision	Recall	F1	F2
SVM	0,67	0,19	0,43	0,26	0,34	0,69	0,2	0,42	0,27	0,34
Naive Bayes	0,84	0,35	0,2	0,26	0,22	0,67	0,23	0,62	0,33	0,46
DecisionTree	0,8	0,25	0,26	0,25	0,25	0,8	0,27	0,27	0,27	0,27
RandomForest	0,86	0,4	0,14	0,21	0,16	0,85	0,37	0,15	0,21	0,17

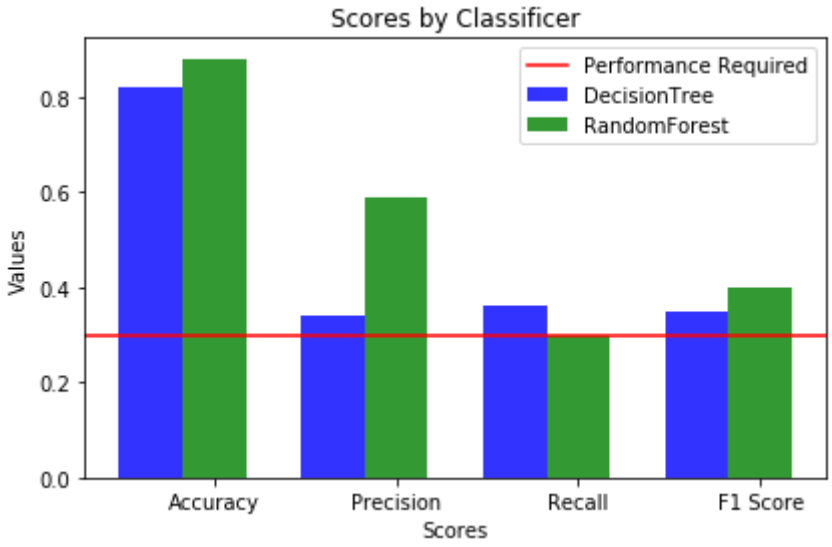
	Top 3 Features					Top 1 Feature				
	Accuracy	Precision	Recall	F1	F2	Accuracy	Precision	Recall	F1	F2
SVM	0,64	0,16	0,26	0,19	0,23	0,31	0,2	0,81	0,32	0,5
Naive Bayes	0,77	0,22	0,15	0,18	0,16	0,8	0,46	0,11	0,18	0,13
DecisionTree	0,74	0,19	0,17	0,18	0,17	0,7	0,25	0,25	0,25	0,25
RandomForest	0,79	0,13	0,04	0,06	0,05	0,71	0,24	0,2	0,22	0,21

	New Feature					SelectKBest				
	Accuracy	Precision	Recall	F1	F2	Accuracy	Precision	Recall	F1	F2
SVM	0,73	0,23	0,2	0,21	0,2	0,73	0,22	0,39	0,28	0,33
Naive Bayes	0,82	0,51	0,11	0,18	0,13	0,34	0,16	0,9	0,27	0,46
DecisionTree	0,7	0,14	0,13	0,14	0,13	0,81	0,32	0,34	0,33	0,34
RandomForest	0,77	0,14	0,05	0,08	0,06	0,87	0,57	0,27	0,36	0,3

	Tuning with GridSearchCV				
	Accuracy	Precision	Recall	F1	F2
SVM					
Naive Bayes					
DecisionTree	0,86	0,43	0,23	0,3	0,25
RandomForest	0,88	0,59	0,3*	0,4	0,33

*Rounded up

Below a plot comparing best result from DecisionTree and RandomForest



Why all the tests?

We did a lot of tests, considering all features, features selected by DecisionTree.features_importances_, and SelectKBest. The reason to that, is because we need consider many variables that can occur with our dataset. For example, the number of features can be overfitting, which means we have more features than necessary, some of them may decrease of result because of missing data or even conflict with another feature. Also, we have the classifier parameters, which GridSearchCV help us to find the best variation in an automatic manner.

All of that steps are needed to insure the result with more proximity from reality with the given data.

Validation

Our validation model, creates 1000 combinations of train and test values.

It' is very important, because when we split our train and test data, we can have many different variations. Creating a scenario that is possible to test it with 1000 different combinations and use the average results can guarantee a fine result of our model considering the most of situations possible.

Conclusion

Best result was using DecisionTree with those data below:

```
features_list = ['poi', 'salary', 'to_messages', 'deferral_payments', 'total_payments', 'exercised_stock_options']

data = featureFormat(my_dataset, features_list, sort_keys = True)
labels, features = targetFeatureSplit(data)

features_train, features_test, labels_train, labels_test = train_test_split(features, labels, test_size = 0.3, random_state =42)

#DecidionTree
clf = DecisionTreeClassifier()
t0 = time()
clf.fit(features_train, labels_train)
print "training time:", round(time()-t0, 3), "s"
t0 = time()
pred = clf.predict(features_test)
print "prediction time:", round(time()-t0, 3), "s"
test_classifier(clf, my_dataset, features_list)
```

References

I used the site https://scikit-learn.org to check parameters from classifiers and my own mini projects from this course.

I used the Git Hub from user zelite to find a function to calculate my new feature: https://github.com/zelite

I used ProgramCreek website to get examples of how to use SelectKBest: https://www.programcreek.com/python/example/93974/sklearn.feature_selection.SelectKBest