Identify Fraud from Enron Email

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, a significant amount of typically confidential information entered into the public record, including tens of thousands of emails and detailed financial data for top executives.

Question #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 goal of this project is to use the given ENRON data and come up with a predictive model which will identify an individual as a "Person of Interest (POI)”. Machine learning helps in learning the emailing habits of POIs and non-POIs and find any pattern in their emails and test our predictive model to identify an individual as a POI or not.

The Enron dataset contains 146 records with 1 labeled feature (POI), 14 financial features and 6 email features. The value True/False of the POI feature informs whether the individual is a POI or NON POI.

The ENRON data have 21 features in total. There are 18 POIs and 128 NON POIs. Some of the features have missing values and represented as 'NaN' .POI cannot be 'NaN', so NaN for other features are counted and presented below.

Count of NAN per feature:

Feature	NaN
loan_advances	142
director_fees	129
restricted_stock_deferred	128

deferral_payments	107
deferred_income	97
long_term_incentive	80
bonus	64
to_messages	60
shared_receipt_with_poi	60
from_poi_to_this_person	60
from_messages	60
from_this_person_to_poi	60
other	53
salary	51
expenses	51
exercised_stock_options	44
restricted_stock	36
email_address	35
total_payments	21
total_stock_value	20

The plot for salary vs bonus helped in visualizing the data and to locate the outlier.

Top 3 salary and bonus data points are gathered, from that an outlier is found.

- TOTAL is one among the high salary and bonus data points.Other high salary and bonus data points are POI. Total has a salary of 26704229 and bonus of 97343619 and is removed.

A list of datapoints that has 'NaN' value for both salary and bonus is collected . On going through the values of the above list ,two records are found.

- THE TRAVEL AGENCY IN THE PARK did not represent an individual.so it is removed.

- LOCKHART EUGENE E has 'NaN' for all the features except 'POI' as POI can only be True/False.

These two data points are removed as they don't have much information to come up with a predictive model.

Once the data points are removed,once again the data is visualized through scatter plot. There are few data points have high bonus and (or) high salary.

Question #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"]

Two new features are created.

- 'Poi_ratio' is the ratio of total number of messages sent to/from from POI to the total number of messages sent and received.The interaction between two POIs will be much frequent/high than the interaction between a POI and a NON POI.
- 'Total earnings' is the sum of salary ,total_stock_value and exercised_stock_options .

When the new features are included in the features list,the algorithm performed the same way in the small dataset file (poi_id.py).

Algorithm	Accuracy	Precision score	Recall score
GaussianNB	0.600	0.600	0.600

When the **new features are not included** in the features list,the performance is

Accuracy: 0.84840 Precision: 0.41988 Recall: 0.35900 F1: 0.38706 F2: 0.36972
Total predictions: 15000 True positives: 718 False positives: 992
False negatives: 1282True negatives: 12008

When the **new features are included** in the features list,the performance is

Accuracy: 0.85020 Precision: 0.42058 Recall: 0.32700 F1: 0.36793 F2: 0.34223
Total predictions: 15000 True positives: 654 False positives: 901 False
negatives: 1346 True negatives: 12099

When the new features are included ,the accuracy and precision score improved slightly whereas recall score is decreased .For Final submission the new features will not be included.

Using feature selection/dimensionality reduction on sample sets, either improves estimators' accuracy scores or to boost their performance on very high-dimensional datasets.

Selectkbest from sklearn is used to select the most influential features. 10 features are selected and its scores are published. These features will be used to identify the POI.

Feature	score
Total_earnings	25.373
exercised_stock_options	24.815
total_stock_value	24.183
bonus	20.792
salary	18.290
deferred_income	11.458
long_term_incentive	9.922

restricted_stock	9.213
total_payments	8.773
shared_receipt_with_poi	8.589

feature scaling is the method for rescaling the features. It still contains the same information but expressed in different form. MinMaxScaler() is used for this Project.The financial features have the values in USD where as email features have just numbers. The value of email features are very small compared to financial features. Feature scaling gives a reliable value.

Question #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”]

GaussianNB is the algorithm which is chosen for the project submission after trying 8 different algorithms. GaussianNB implements the Gaussian Naive Bayes algorithm for classification. Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of independence between every pair of features. They require a small amount of training data to estimate the necessary parameters.

Selectkbest is used with GaussianNB to come up with a good predictive model. SelectKbest finds the k best features from the features list for this algorithm to achieve the goal. The value of K is set to different values and then found that k=5 gives the better result with the function 'f-classif'. MinMaxScaler is used for feature scaling.

Algorithm	Accuracy	Precision score	Recall score
GaussianNB	0.600	0.600	0.600

Accuracy: 0.84840 Precision: 0.41988 Recall: 0.35900 F1: 0.38706 F2: 0.36972
Total predictions: 15000 True positives: 718 False positives: 992

False negatives: 1282True negatives: 12008

Decisiontreeclassifier is one among the algorithms ,I tried.
MinMaxscaler is used for feature scaling. SelectKbest finds the k best features from the features list for this algorithm to achieve the goal.The output is better when the value of k set to 7.Decision Tree Classifier is a simple and widely used classification technique. It applies a straightforward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record.

Poi_id.py results

Algorithm	Accuracy	Precision score	Recall score
Decision Tree Classifier	0.400	0.400	0.400

Tester.py results
Accuracy: 0.82113 Precision: 0.32621 Recall: 0.32050 F1: 0.32333 F2: 0.32163
Total predictions: 15000 True positives: 641 False positives: 1324 False negatives: 1359 True negatives: 11676

Support Vector Classifier(SVC) is another algorithm ,I would like to present.Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.MinMaxScaler and SelectKbest are used here as well.

Algorithm	Accuracy	Precision score	Recall score
svc	0.316	0.214	0.600

Accuracy: 0.70087 Precision: 0.20624 Recall: 0.43650 F1: 0.28012 F2: 0.35682
Total predictions: 15000 True positives: 873 False positives: 3360 False negatives: 1127 True negatives: 9640

The Accuracy ,precision and recall scores are better when the parameter,'C' set to a high value and 'gamma' set to a low value.

The other algorithms were tuned experimentally, with unremarkable improvement.

Question #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 is giving possible different values to the parameters of an algorithm and measures the performance for the desired output .
GridsearchCV is an approach to parameter tuning that will methodically build and evaluate a model for each combination of algorithm parameters specified in a grid.This is an important step for improving algorithm performance right before presenting results.

The parameters grid values of Decision Tree Classifier are discussed below.

```
params1 = dict(

                selectkbest__k=[7],
                selectkbest__score_func= [f_classif],
                decisiontreeclassifier__criterion = ['gini','entropy'],
                decisiontreeclassifier__max_features =['auto']
                  )
```

Tuning #1
All the parameters are kept same except the value of K.
When the algorithm set to different values of K,the performance changes to

Poi_id.py results

Score	K=1	k=5	k=7	k=10	k=11
Accuracy	0	0.364	0.400	0.333	0
Precision score	0	0.333	0.400	0.286	0
Recall score	0	0.400	0.400	0.400	0

Tester.py results

Value	Performance
K=1	Accuracy: 0.81940 Precision: 0.22752 Recall: 0.14800 F1: 0.17934 F2: 0.15912 Total predictions: 15000 True positives: 296 False positives: 1005 False negatives: 1704 True negatives: 11995
K=5	Accuracy: 0.80700 Precision: 0.26897 Recall: 0.26050 F1: 0.26467 F2: 0.26215 Total predictions: 15000 True positives: 521 False positives: 1416 False negatives: 1479 True negatives: 11584
k=7	Accuracy: 0.82113 Precision: 0.32621 Recall: 0.32050 F1: 0.32333 F2: 0.32163 Total predictions: 15000 True positives: 641 False positives: 1324 False negatives: 1359 True negatives: 11676
k=10	Accuracy: 0.81307 Precision: 0.30020 Recall: 0.30200 F1: 0.30110 F2: 0.30164 Total predictions: 15000 True positives: 604 False positives: 1408 False negatives: 1396 True negatives: 11592
k=11	Accuracy: 0.79980 Precision: 0.23139 Recall: 0.21600 F1: 0.22343

	F2: 0.21891 Total predictions: 15000 True positives: 432 False positives: 1435 False negatives: 1568 True negatives: 11565

Tuning #2
All the parameters are kept same except the criterion parameter. When the algorithm set to different criterion values, the performance is presented below

poi_id.py

	Criterion = 'Gini'	Criterion='Entropy'
Accuracy	0.400	0
Precision score	0.400	0
Recall score	0.400	0

Tester.py

Criterion ='gini'
Accuracy: 0.82107 Precision: 0.33153 Recall: 0.33650 F1: 0.33400 F2: 0.33549 Total predictions: 15000 True positives: 673 False positives: 1357 False negatives: 1327 True negatives: 11643

Criterion ='entropy'
Accuracy: 0.81787 Precision: 0.31496 Recall: 0.31150 F1: 0.31322 F2: 0.31219 Total predictions: 15000 True positives: 623 False positives: 1355 False negatives: 1377 True negatives: 11645

The different values of parameters impacts the outcome. Tuning is a must step to come up with good predictive model.

Question #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 performed to ensure that a machine learning algorithm generalizes well.Overfitting is the classic mistake that happens when we try to come up with the predictive model. Overfitting makes test performance relatively low.

A single split into a training & test set would not give a better estimate of error accuracy. Hence StratifiedSuffleSplit is used to randomly split the data into multiple trials while keeping the fraction of POIs in each trials relatively constant. The validation carried out with the values n_iter=1000 and test_size =0.25 . It took quite a time to finish ,but gave the same results as before.

Question #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 two evaluation metrics used are Precision score and Recall score .
Often, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other.

Precision: Out of all the items that are truly positive, how many were correctly classified as positive. i.e. since we got 42%, then out of 100 people the model predicted as POI, we can be sure that 42 of them are indeed POIs.

Recall: Out of all the items that are truly positive, how many were correctly classified as positive. Or simply, how many positive items were 'recalled' from the dataset.
i.e. with score of 36%, given total 100 POIs in a dataset, this model can find 36 of them.

Accuracy 85% means "out of all my predictions, I can be sure 85% are correct"

RESOURCES: