Identifying Enron Fraud.

A write up of my responses to the questions asked as part of the Udacity's Intro to Machine Learning Project

I have tried to answer these questions in the notebook EnronDataset_EDA.ipynb as well, here just formulating the thought process and steps that are documented in the notebook in brief comments.

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?

In the year 2000, Enron was one of the largest companies in the US. but by 2002, Enron had collapsed as a corporation due to widespread corporate fraud.
In the investigation that followed, huge amount of information was released to public, which included information about executive's salary, stock options, employee emails.

We are given a dataset of 146 data points, each having 21 features ranging from salary, bonus etc to email features like number of messages, POI conversations.

This analysis and modelling once implemented can be used to identify a person as POI (person of interest (POI) is someone who was indicted for fraud, settled with the government, or testified in exchange for immunity) or not..

Following are the major steps in the procedure :

- Exploring Enron Data Set
- Feature Selection, engineering and processing
- Choosing and Running an algorithm
- Validation of results from the algorithm

As part of the project I have done some initial EDA of the dataset, which covers analysis like, distribution of poi to non-poi, relation between salary/bonus, distribution of bonus and salary, relation between total stock value and exercised stock options among others. The EDA can be found in EnronDataset_EDA.ipynb

Out of 146 data points, only 18 are POI, due to complexity of the data set using machine learning to find factors governing and patterns that might lead to a person being categorized 'POI' and can be used to identify POIs in new data set.

Outlier Detection.

On exploring some data points and doing basic visualizations we can see that following observed points can be considered as outliers.

‘TOTAL’ - This is a total metric which is not be considered for this analysis as it’s just consolidation in excel.

‘THE TRAVEL AGENCY IN THE PARK’ - This is not a person also, most of the attributes for this data point is NaN.

‘LOCKHART EUGENE E’ is another person who has all of the features as NaN, hence dropping this as well.

Following the data cleaning, 143 data points remained.

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.

- Features Selection

I did not use any algorithm like kBest, while did feature selection manually.
So based on my EDA, while I couldn't clearly see which features were more important than others, but it was clear there were features in the dataset that huge NaNs, which won't contribute to the ultimate classification.

So I started with eliminating the features that had too many NaN(greater than 75)

Then i dropped email address and person, as they were only there for identification and should play no role in the model.

I did an comparison before dropping features and after dropping features

Metric	Before	After
Recall	0.08	0.1
Precision	0.31	0.26
Accuracy	0.84	0.85

Although precision dropped, recall and accuracy increased, I decided to experiment with creating new feature and see if there is any impact

- Features Creation

I ended up creating 3 features ->

eso/tsv → exercied_Stock_options/total_stock_value
from_poi/to_msg → from_poi_to_this_person / to_messages
to_poi/from_msg → from_this_person_to_poi / from_messages

I did compare the performance with selecting these features with a simple decision tree classifier to see what can be expected and decided to keep all these new features for optimum performance.

I also compared the performance of RandomForest against No new features, with a only ‘eso/tsv’, with ‘from_poi/to_msg’ + ‘to_poi/from_msg’ and with all

Features	No New Feature	eso/tsv	from_poi/to_msg + to_poi/from_msg	All 3
Accuracy	0.84	0.85	0.855	0.855
Precision	0.26	0.38	0.38	0.36
Recall	0.08	0.06	0.13	0.13

When using all 3 features, precision decreased and rest all remained same.
I decided to use all three features as results were very similar in the experiments and adding new feature was helping recall values a lot, and i felt that using an algorithm other RandomForest and parameter tuning would result in better results

- Features Scaling

I did not encounter a need for using feature scaling, if I had used an algo like KMeans, I might have needed to do a max min split scaling.

Question 3 : What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms?

I tried multiple algorithms like Naive Bayes, RandomForest, SVC, AdaBoost, Bagging.

Algorithm	**Naive Bayes**	**RandomForest**	**SVC**	**AdaBoost**	**Bagging**
Accuracy	0.20	0.87	0.87755102	0.83	0.83
Precision	0.12	0.35	N/A	0.36	0.37
Recall	0.80	0.12	N/A	0.29	0.17

I found that I was getting best results with RandomForest and AdaBoost and decided to parameter tuning for both of them to see which of them give better metrics.
Details can be seen in the notebook.

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? What parameters did you tune? (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).

Each and every ML algorithm have some important parameters whose value is fixed before we start the ML process. These parameters govern how the algorithm behaves/works.
For example, in Support Vector Machines within the algorithm there are various kernel that can be specified which can govern algorithm behaviour.
Tuning in terms of machine learning means to fix the values of these parameter in such a combination that the algorithm can maximize the performance in a given situation
Parameter tuning is essentially checking the output against various parameters and using the values that yield best results. In case of SVM, if we are changing kernels, we would try out different kernels and see which give best results for our dataset.
Parameter tuning varies from dataset to dataset and there is no set methods to follow.
Tuning the parameter of the algorithms is essentially changing how the algorithm behaves, this can be done manually with simultaneous plotting, visualizing results and analysing.

Parameter tuning when doing manually is lot of trial and error and kind of an art. It plays an important role in maximizing the performance the algorithm, as even a minute change in the algorithm parameters can lead to a significant increase in the accuracy of the results generated by the algorithm

I opted to use sklearn's GridSearchCV for finding the optimum parameters among a range of parameters.

I did parameter tuning for both RandomForest and AdaBoost classifier as I was getting good enough results for both of them.

For AdaBoost, I chose to tune, n_estimators, learning rate and the type of algorithm between 'SAMME.R','SAMME'

For Random Forest, I chose to tune - max_depth, min_samples_split, n_estimators, min_samples_leaf and criterion (gini or entropy)
For both of these I got following results:

Algorithm	**RandomForest**	**AdaBoost**
Accuracy	0.86	0.825
Precision	0.45	0.34
Recall	0.16	0.36

Although Accuracy and Precision for RandomForest was much better than AdaBoost, Recall was really bad.

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 items: "discuss validation", "validation strategy"]

Validation is the process of testing the our model to see if works as expected or not.
Typically we divide the data into test,train splits so that we train our model on train data and test on test data.

When testing the model, to check if algorithm is working as expected or not we generally rely on various metrics that we calculate on the test data set, which can be accuracy, precision, error rate
If we the model is correct, we should see high accuracy, precision and typically low error rate.

These metrics are calculated on test data by comparing the predicted labels with the known labels and from there we can calculate the metrics.
Most popular ML libraries (sklearn in this case) provide built in metrics to test our model.

If we do not split the data into train and test data sets, the model will be built too specific to that particular data set and won't be able to perform when put to use for predictions. This will result in a classic overfitting issue, in which the models fits the current data perfectly but fails on new data.

Another thing we can do to be more confident of our model is after splitting the data into train and test data. We can further do cross validation or n fold validation within train data so and do actual testing against test data.
Here test data behaves as essentially 'new' data.

We can also leverage sklearns inbuilt function like StratifiedShuffleSplit and StratifiedKFold to validate our model.

I have chosen to use StratifiedShuffleSplit with n_folds = 10 to have confidence on the model.

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"]

There can be multiple metrics to check the performance of the model. We were required to check for Precision and Recall, I also checked on accuracy to see how model was performing.

Precision - 0.33235 (AdaBoost)

It shows that 33.25% people were correctly classified as 'poi'. This is helpful in minimizing the false positive, as we do not want to have a case in which wrong person was classified as 'poi'. Higher the precision, lesser the number of people being incorrectly classified.

Recall - 0.3385 (AdaBoost)

This means 33.8% of data was retrieved correctly by the model. That is 35.7% was correctly labeled as poi.

Accuracy - 0.82113(AdaBoost)

Model was 82.1% accurate in predicting whether a person is POI or not. Higher the accuracy, better the model.

- List of Resources

- scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
- http://www.investopedia.com/updates/enron-scandal-summary/
- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.as_matrix.html
- http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html
- https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-algorithms-machine-learning/
- https://towardsdatascience.com/using-bagging-and-boosting-to-improve-classification-tree-accuracy-6d3bb6c95e5b
-