Udacity Data Analyst Nanodegree
Project 5: Identify Fraud from Enron Email

1) The goal of this project is to identify persons involved in fraud in the Enron scandal (Persons of Interest). The dataset consists of a list of persons involved with Enron and their personal and financial attributes. It has a total of 20 features, such as email, salary, bonus, stock value, etc. These can be used with machine learning techniques to identify patterns of potential persons of interest. There are a total of 146 records in the dataset, where 18 are a person of interest while the rest are not.

   The dataset contains a few outliers; there were a few data points, which had some values for total payments much higher than other nodes. A linear regression was used in an attempt to estimate data points with total payments, which differed significantly from the dataset. With this method two outliers were removed: "Total and "The Travel Agency in the Park.

   Also, for a few persons, all variables had no values. These included: Cline Kenneth W, Piro Jim, Lockhart Eugene E, Hayslett Roderick J. Without these and by removing 7% of outliers 133 records were kept for further analysis.

2) The dataset contains several features. Not all of them were used for the machine-learning algorithm. The features were selected based on a Random Forest Classifier. The features with an importance higher than 8% were kept for further analysis, these included and their importance were: total payments (11.9%), exercised stock options (10.5%), bonus (11.0%) and deferred income (8.2%). The threshold of 8% was selected because it would result in approximately 20% of the features being selected. Also, the these features have significantly higher importance were compared to the non-selected features.

   A Principal Component Analysis was used to create three new features. Three features were used as they produced the best results for the algorithm. The PCA was tested with both scaled and non-scaled features, on which the non-scaled version performed better. The results for scaled and non-scaled algorithms were as follows:

	Scaled	Non - Scaled
Accuracy	0.910	0.914
Prediction	0.702	0.769
Recall	0.647	0.428

3) A few algorithms were tested, which included: Gaussian Naïve Baise; K-Means; Random Forests; Support Vector Machine. These were tuned to improve the results but Random Forests was selected as the one, which had the best performance.

4) Tuning the parameters of an algorithm can be used to improve its performance. Changing the parameters can be helpful to improve its predictive power or to train it better. If this is not done well the algorithm might not perform well with a new dataset. For this project, a RandomForestClassifier was used and the parameters tuned were the number of estimators and warm start. A GridSearchCV was used the best parameters chosen were warm start as True and X estimators.

5) For validation, a Stratified Shuffle Split was used. With this technique the algorithm was better trained and provided better results. A classic mistake during validation is to over fit the algorithm or to not train the algorithm well. It is important to validate so that the algorithm is robust enough to have a good performance for future unseen data.

   With a Stratified Shuffle Split used to separate the data into several different train and test subsets, the algorithm could be re-trained several times and in conjunction with the "warm start" parameter. This allowed the classifier to reuse the solution from the previous training iteration; which improved the results of the algorithm. Also this parameter was used in conjunction with the number of estimators parameter, in each iteration the number of estimators was increased, also improving performance.

   The ideal values for these parameters were defined manually since they are directly related to how a Stratified Shuffle Split works and the GridSearchCV does not capture that effect.

   Other parameters were also tested: min_samples_split and min_samples_leaf. However, for both, the default values were more effective.

6) The evaluation metrics used were Accuracy, Precision and Recall.

   Accuracy indicates how many POIs or Non-POIs were identified correctly, this metric alone is not enough to evaluate the algorithm due to POI and Non-POI being imbalanced classes – it could be possible to have an algorithm with high accuracy which identifies Non-POIs very well but does a poor job for POIs.

   Precision indicates the percentage of POI who were identified correctly. Recall is a metric that indicates the fraction of POIs that were correctly identified relative to the number POIs that should have been identified.

   For each of these metrics, the algorithm scored 93.5%, 77.9% and 65.7%, respectively.

Sources:

Tuning the parameters of your Random Forest model
https://www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/

Feature Selection for Machine Learning in Python
http://machinelearningmastery.com/feature-selection-machine-learning-python/