1. The goal of this project is to identify persons of interest for potentially having committed fraud. Machine learning, particularly supervised learning techniques, is useful in this effort because it can effectively explore patterns within the data and attempt to correlate these patterns with expected outcomes.

   Outliers were managed by implementing an outlier removal function, which removed 10% of instances with the highest error level.

2. I used all features from the two feature groups, except for salary. I did not use feature scaling, as my algorithm performed well without.

   For feature creation, I used principal component analysis, combining with K Nearest Neighbor and optimizing with GridSearchCV to maximize performance. In the end, however, PCA did not end up in the final model. As can be seen below, the performance of the K Nearest Neighbor with PCA was worse than the unoptimised K Nearest Neighbor in terms of accuracy and precision, and also worse than the final model in terms of accuracy, precision & recall.

	Accuracy	**Precision**	**Recall**
KNearestNeighbor (unoptimised)	0.8736	0.57602	0.197
KNearestNeighbor (with PCA)	0.79353	0.22256	0.22
KNearestNeighbor (final)	0.8892	0.62176	0.4315

I used the SelectKBest function in two ways to narrow the list of features from an original list of all features. First, I used the SelectKBest as the first function in a Pipeline before I ran the selected algorithm, and then I used the GridSearchCV function to optimize the K parameter. The GridSearchCV resulted in an optimal parameter of 19 (meaning that all features is the optimal number). After optimal parameters were found for all functions in the pipeline, I used the SelectKBest function again, but this time running a fit_transform to change the features. K values of 18 and 19 both had the same results that optimized the model; as this meant that the extra feature ("salary") was not required, I eliminated from the final model.

Here are the feature scores for all features:

Feature	Feature Score
salary	0.001184122
deferral_payments	0.365213143
total_payments	0.319758983
loan_advances	2.52741924
bonus	0.018005019
restricted_stock_deferred	0.319261787
deferred_income	0.105706207
total_stock_value	0.178222708
expenses	0.0163341
exercised_stock_options	0.245479475
other	0.082767571
long_term_incentive	0.015099532
restricted_stock	0.034427158
director_fees	0.459755924
to_messages	0.375062928
from_poi_to_this_person	1.93875606
from_messages	0.117352032
from_this_person_to_poi	2.3781999
shared_receipt_with_poi	3.77224875

3. After attempting a number of algorithms including Decision tree and Naïve Bayes, I ended up using the K Nearest Neighbor algorithm. Results as follows:

	Accuracy	Precision	Recall
Decision Tree	0.7888	0.20445	0.202
Naïve Bayes	0.3362	0.1472	0.83

KNearestNeighbor (unoptimised)	0.8736	0.57602	0.197
KNearestNeighbor (with PCA)	0.79353	0.22256	0.22
KNearestNeighbor (final)	0.8892	0.62176	0.4315

4. Algorithms may have many potential settings that may yield a wide variety of results. Tuning the algorithm means to adjust the parameters of the algorithm to achieve the best desired result—typically highest level of accuracy. I used GridSearchCV to tune my algorithm, setting the parameters and possible values in a dictionary, and feeding those parameters into the algorithms to achieve the highest possible target value (which itself can be set as a parameter; in this case, I aimed for a high recall value because this value felt particularly difficult to get above the desire .3 value.
5. Validation is testing the model against a set of data that was not used in the development of the model, to ensure that the model is generalizable and can be used in circumstances beyond model configuration. One classic mistake is overfitting the data, meaning that the model is created so specifically to the model training data, that the performance of the model against another similar set of data would be significantly lower.

   I validated the data using the pre-built classifier test script (which uses stratified shuffle split cross validation).

6. For the final model, the precision was .62176; this value means that out of all of the instances that were identified as positive, 62% were correct ("true positives").
   The recall for the final model was .4315; this value means that out of all true positive items, 43% were identified correctly.