EEE 591 Project 2 Report

I have chosen the below parameters for constructing the MLPClassifier:

```
mlp = MLPClassifier( hidden_layer_sizes=(100), activation='logistic',
                    max_iter=2000, alpha=0.00001, solver='adam', tol=0.0001, random_state=5 )
```

as the accuracy for this architecture is 0.95.

The above parameters are chosen as they gave the best accuracy so far. The model predicts that the chance of surviving the real mine field is 95%. Confusion matrix for number of components = 7 and accuracy of 0.95

N=63	**Class 1 Rock-(predicted)**	**Class 2 Mine-(predicted)**
Class 1 Rock-(Actual)	**28**	**1**
Class 2 Mine-(Actual)	**2**	**32**

This means that total number of data present in test set is 63. The total number of actual data belonging to class 1 i.e. Rock is 29 and total number of actual data belonging to class 2 i.e. Mine is 34. Out of 29 actual Rock data, 28 are predicted right but 1 is predicted to belong to be mine. This situation is called False positive. Out of 34 actual Mine data, 32 are predicted right but 2 are predicted to belong to class rock. This situation is called False Negative. In short, confusion matrix is used to describe the performance of a classification model.

When I have chosen the below parameters for constructing the MLPClassifier:

```
mlp = MLPClassifier( hidden_layer_sizes=(100), activation='logistic',
                    max_iter=2000, alpha=0.00001, solver='adam', tol=0.0001, random_state=None )
```

When the code is run with the random_state of train_test_split left as none, the results will make a huge difference. This happens because random_state determines how to decompose the data set into training and test sets. If it is set to none, each time the code is run, the training set and the test set are different, so the accuracy we obtain is also different. Therefore, I think it would be better to set random_state to a constant.

Setting random_state to none in the MLP classifier will be called different results in different runs. For example, for the above parameters, 5 can be converted to 0.90 precision. In another round, there are 7 components with an accuracy of 0.89, 10 components with an accuracy of 0.94(highest), and so on. This means that setting random_state to none will generate a random number in all runs, resulting in inconsistent results. However, setting random_state to 5 can use 0.95 accuracy for 7 components, this is highest accuracy.

Component = 7 feasible maximum accuracy, which means these 7 features are the most Important functions and drive the entire model. They carry the most information in this data set. The accuracy of including other functions is decreasing because they do not

contain many functions information, so it does not help predict. Therefore, the graph of the graph increases to 7 Components, then, the accuracy begins to decline.