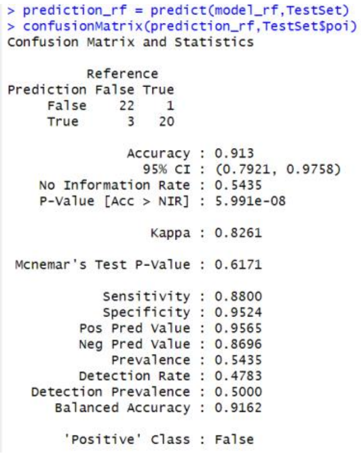
Report Of Supervised Learning Algorithms Based On Enron Dataset

A. Description of Supervised Learning Algorithms

Before we talk about the performance of the algorithms, let's take a look at the data processing. The dataset contains 22 columns in which two of them are names and email addresses. We shall remove these two characters columns since they provide few information in supervised learning. Then, I cleaned all the datapoint whose salary is NaN. Since the number of the data whose salary is missing is small, this won't badly affect our training. Then, I filled all the other NaN by 0. Then, I turned dataset to data frame. Finally, I use SMOTE method to balance the data and take poi as the target.

- **Description of Random Forest**

  After processing the data, we have 19 attributes and 1 target. Since 4<sqrt(19)<5, I tried mtry=4 and 5. I found when mtry equals to 4, random forest performs better than mtry equals to 5.



```
> prediction_rf = predict(model_rf,TestSet)
> confusionMatrix(prediction_rf,TestSet$poi)
Confusion Matrix and Statistics

          Reference
Prediction False True
     False    22    1
     True      3   20

               Accuracy : 0.913
                 95% CI : (0.7921, 0.9758)
    No Information Rate : 0.5435
    P-Value [Acc > NIR] : 5.991e-08

                  Kappa : 0.8261

 Mcnemar's Test P-Value : 0.6171

            Sensitivity : 0.8800
            Specificity : 0.9524
         Pos Pred Value : 0.9565
         Neg Pred Value : 0.8696
             Prevalence : 0.5435
         Detection Rate : 0.4783
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.9162

       'Positive' Class : False
```

Figure 1. Result of Random Forest (mtry=4)

  From figure 1, you could see the accuracy of Random Forest is 0.913.

- **Description of Neural Network**

  In Neural Network experience, I set the iterators from 1 to 600 with a step length of 100. This allows us to see the result when training epochs are 1, 101, 201, …, 601. Then, we get the best performance of the model and see the test result. From figure 2 you could see when training epoch is growing, the error rate is decreasing, but there is still a large randomness in Neural Network. You can easily see the best performance here is when epoch = 500.

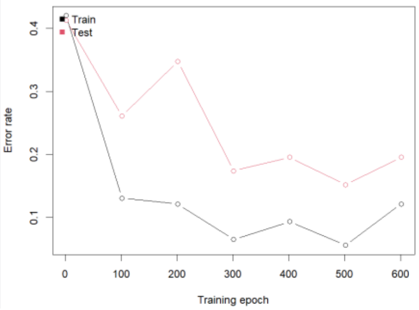
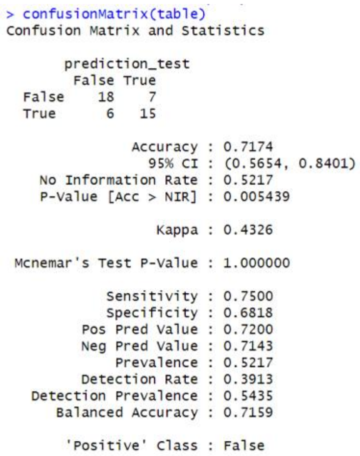


Figure 2. Trend when training epoch grows.

As we set the seed as 2023, we could train 500 epochs again to get the same model. Figure 3 shows the performance of the best NN module we got, whose accuracy is 0.7174.



```
> confusionMatrix(table)
Confusion Matrix and Statistics

       prediction_test
        False True
  False    18    7
  True      6   15

               Accuracy : 0.7174
                 95% CI : (0.5654, 0.8401)
    No Information Rate : 0.5217
    P-Value [Acc > NIR] : 0.005439

                  Kappa : 0.4326

 Mcnemar's Test P-Value : 1.000000

            Sensitivity : 0.7500
            Specificity : 0.6818
         Pos Pred Value : 0.7200
         Neg Pred Value : 0.7143
             Prevalence : 0.5217
         Detection Rate : 0.3913
   Detection Prevalence : 0.5435
      Balanced Accuracy : 0.7159

       'Positive' Class : False
```

Figure 3. Result of NN

B. Which one is more suitable for this case

From the experiment above, we could have 2 accuracies from Random Forest(0.913) and Neural Network(0.7174). Since 0.7174 < 0.913, we could get a conclusion that ***in enron case, random forest is more suitalbe than neural networks.***

Due to the randomness of these two algorithms and dataset segmentation, I made more than one experiment. I found that Random Forest performs significantly better than NN in Enron case. The following is some potential reasons.

1. **Dataset is small**

   From Enron dataset, we could see there are only 146 objects, which is not large enough for a well performed Neural Network.

2. **Features are low-level**

   From Enron dataset, we could see there are 19 useful attributes. These features are low-level, which means, we are hardly to find high-level attributes hidden behind these attributes.

3. **Complex interacted data**

   The features contain salary, messages, payments, etc. These features highly interacted with each other, which could be handled well by Random Forest.

4. **Outliers and noise is common**

   In Enron dataset, there are tons of NaNs. From the last assignment, we could find if we take the intersection of these data points, it will return an empty set. These outliers and noise could be handled by Random Forest easily.