**Steps used with each classifier**

|  |  |
| --- | --- |
| **LDA (Linear discriminant analysis)** | **k-NN (k-Nearest Neighbour)** |
| * Fit an LDA model using MASS::lda function
* Predict on the test set using predict function
* Extract the predicted classes
* Extract the predicted probabilities
* Evaluate the LDA model using accuracy\_kappa function
* Print the accuracy and kappa values
 | * Choose an optimal value of k using cross-validation on the training set
* Loop through each k value
* Perform 10-fold cross-validation using class::knn
* Calculate the accuracy as the mean of correct predictions
* Plot the accuracy values against k values
* Find the optimal k value as the one with the highest accuracy
* Fit a k-NN model using class::knn function with the optimal k value
* Evaluate the k-NN model using accuracy\_kappa function
* Print the accuracy and kappa values

  |
| **RF (Random Forest)** | **ANN (hidden layer feed-forward neural network)** |
| * Fit a random forest model using randomForest::randomForest function
* Predict on the test set using predict function with type = "prob"
* Extract the predicted classes by choosing the class with the highest probability
* Evaluate the random forest model using accuracy\_kappa function
* Print the accuracy and kappa values
* Extract the predicted classes by choosing the class with the highest probability
 | * Choose an optimal value of size (number of units in the hidden layer) using nnet::cv
* Loop through each size value
* Perform 10-fold cross-validation using nnet::cv
* Extract the accuracy value from the output
* Plot the accuracy values against size values
* Find the optimal size value as the one with the highest accuracy
* Fit an ANN model using nnet::nnet function with the optimal size value
* Predict on the test set using predict function with type = "prob"
* Extract the predicted classes by choosing the class with the highest probability
* Evaluate the ANN model using accuracy\_kappa function
* Print the accuracy and kappa values
 |