**BT2103 Assignment 1**

**Group Members: Toh Zhan Ting (A0234206R), Chen Haoli (A0234102B), Lo Zhi Hao (A0236437B), Luah Jun Yang (A0233519E)**

**What does each line of the R code above do? Explain any output that is produced by the R command.**

| **For the first chunk of R code, we are setting up the packages and classifying the existing data into training and test sets** | | |
| --- | --- | --- |
| **R code** | **Purpose of R Code** | **Output and explanation (if any)** |
| library(e1071) | Loading the e1071 package, to create SVM model | No Output |
| index <- 1:nrow(rice) | Find the total number of rows and create a vector with the number of rows in rice dataset  Variable *index* contains 3810 indexes since there are 3810 rows in rice dataset | No Output |
| set.seed(12345) | seed(n): n is the starting point used in the generation of a sequence of random numbers  Set seed to make sure the pseudorandom number generator in following lines always generates the same output and helps in debugging | No Output |
| testindex <- sample(index, trunc(length(index)/3)) | Takes a sample from the rice rows vectors that is 1/3 of the original size by splitting the dataset with sample().  Variable *testindex* will have 1270 indexes. | No Output |
| testset <- rice[testindex,] | Creating a test dataset to prepare for testing of the SVM model with all rows entries in rice dataset which have indexes that are in *testindex*. | No Output |
| trainset <- rice[-testindex,] | Creating a training dataset to prepare for the running of SVM model with all rows entries in rice dataset which have indexes that are not in *testindex*. | No Output |

| **Second chunk of code is about running and printing out the results from the SVM model** | | |
| --- | --- | --- |
| **R code** | **Purpose of R Code** | **Output and explanation (if any)** |
| svm.model <- svm(as.factor(CLASS) ~ . , data = trainset,type="C-classification", kernel="linear") | svm() trains the support vector machine (SVM) algorithm on the training dataset.  “as.factor(CLASS)~.” refers to the class column in the rice dataset which is the dependent variable and all other attributes in the rice dataset are used as independent variables.  Since it’s a CLASS is a factor, the type is “C-classification”.  Since we are looking for a linear hyperplane, the kernel is set to “linear”, so the output will be a linear boundary. | No Output |
| svm.model | Calling a summary of the svm model which has 483 support vectors. | Parameters of the SVM model as explained above |
| svm.model$SV | Returns the 483 resulting support vectors that lie on the hyperplane, which include the coordinates of the respective features (e.g. Area, perimeter major axis, convex area etc) | svm.model$SV  AREA PERIMETER MAJORAXIS MINORAXIS ECCENTRICITY CONVEX\_AREA EXTENT  31 0.2789724658 0.3956633177 0.431313789 -0.013734801 0.544255346 2.995925e-01 -1.122897971  … 482 more rows |
| svm.model$index | Returns the index of the 483 resulting support vectors in the data matrix, which are picked out by the model to form the “linear boundary” or the separation of the 2 classes. | [1] 24 65 66 71 72 73 80 81 83 93 95 96 99 112 113 117 118 120 131 134 137 173 176 177 187 188 189 190 192 199 201 202  [33] 205 207 209 210 214 220 230  … up to a total of 482 indexes of the supporting vectors |

| **For the third chunk of code, we are utilising the svm model to predict and measure the accuracy of the model on test dataset by using predict function to predict values of class based on the SVM model and the remaining independent variables** | | |
| --- | --- | --- |
| **R code** | **Purpose of R Code** | **Output and explanation (if any)** |
| results\_train <- predict(svm.model,trainset[,-8]) | predict() functions will predict values of CLASS, generating prediction results from training data by using the svm model.  -8 refers to deleting the CLASS column | No Output |
| results\_test <- predict(svm.model, testset[,-8]) | predict() functions will predict values of CLASS, generating prediction results from test data by using the svm model.  -8 refers to deleting the CLASS column | No Output |
| table(pred=results\_train,actual=trainset$CLASS) | Creating a contingency table to represent the confusion matrix of the model in the training dataset | Pred means whether a rice in *results\_train* has been predicted to be of class Cammeo or Osmancik using the svm model  Actual means whether a rice is actually of class Cammeo or Osmancik.  A contingency table is printed out to show the confusion matrix computed by the model.  Taking Cammeo as the model’s positive value, we can see that :  TP = 982  TN = 1376  FP = 95  FN = 87 |
| table(pred=results\_test,actual=testset$CLASS) | Creating a contingency table to represent the confusion matrix of the model in the test dataset | Pred means whether a rice in *results\_train* has been predicted to be of class Cammeo or Osmancik using the svm model  Actual means whether a rice is actually of class Cammeo or Osmancik.  A contingency table is printed out to show the confusion matrix computed by the model.  Taking Cammeo as the model’s positive value, we can see that :  TP = 516  TN = 674  FP = 35  FN = 45 |
| mean(results\_train == trainset$CLASS) | Getting the accuracy of estimations of predicted results in training dataset (true positive, true negative) | Accuracy of a model is calculated as such:  The accuracy of the predicted results in the training dataset is derived as such:  The accuracy of the predicted result is quite high. |
| mean(results\_test == testset$CLASS) | Getting the accuracy of estimations of predicted results in test dataset (true positive, true negative) | The accuracy of the predicted results in the test dataset is derived as such:  The accuracy of the predicted results is higher in the test dataset than that in the training dataset. |

| **Rerun the model using additional cross validation and penalising parameters** | | |
| --- | --- | --- |
| **R code** | **Purpose of R Code** | **Output and explanation (if any)** |
| svm.crossmodel <- svm(as.factor(CLASS) ~ . , data=rice, cross=10,type="C-classification",kernel="linear",cost=1) | Rerun the svm model with more arguments (cross for k-times cross validation and cost for penalising constraint violations) | No Output |
| results\_rice <- predict(svm.crossmodel,rice[,-8]) | Generating prediction results from the entire rice dataset by using the svm model | No Output |
| table(pred=results\_rice,actual=rice$CLASS) | Creating a contingency table to represent the confusion matrix of the model in the test dataset | Pred means whether a rice in *results\_train* has been predicted to be of class Cammeo or Osmancik using the svm model  Actual means whether a rice is actually of class Cammeo or Osmancik.  A contingency table is printed out to show the confusion matrix computed by the model.  Taking Cammeo as the model’s positive value, we can see that :  TP = 1496  TN = 2953  FP = 127  FN = 134 |
| mean(results\_rice == rice$CLASS) | Getting the accuracy of estimations of predicted results in rice dataset (true positive, true negative) | The accuracy of svm.crossmodel is derived as such: |

| **Rerun the model using additional weight parameter** | | |
| --- | --- | --- |
| **R code** | **Purpose of R Code** | **Output and explanation (if any)** |
| svm.weightedmodel <- svm(as.factor(CLASS) ~ . , data=rice, cross=10, type="C-classification",kernel="linear",cost=1, class.weights=c(Cammeo=0.9,Osmancik=0.1)) | Rerun the svm model with more arguments (class.weight to take into account the asymmetrical nature of cammeo and osmancik class size) | No Output |
| results\_rice <- predict(svm.weightedmodel,rice[,-8]) | Generating prediction results from the entire rice dataset by using the svm model | No Output |
| table(pred=results\_rice,actual=rice$CLASS) | Creating a contingency table to represent the confusion matrix of the model in the test dataset | Pred means whether a rice in *results\_train* has been predicted to be of class Cammeo or Osmancik using the svm model  Actual means whether a rice is actually of class Cammeo or Osmancik.  A contingency table is printed out to show the confusion matrix computed by the model.  Taking Cammeo as the model’s positive value, we can see that :  TP = 1593  TN = 1715  FP = 465  FN = 37 |
| mean(results\_rice == rice$CLASS) | Getting the accuracy of estimations of predicted results in rice dataset (true positive, true negative) | The accuracy of svm.weightedmodel is derived as such:    = 0.8682415  This accuracy of svm.weightedmodel is slightly lower than that of 0.9314961 using the svm.crossmodel. |

**Suggestions**

1. Increase the amount of penalty on incorrectly classified data by increasing the cost to better take into account incorrect data classification

2. Using different kernels to take into account different kinds of possible mathematical relationships other than linear relationships

3. Use different type of classification such as the `nu-classification` method

4. Apply different weightage for the two rice classes (weightage of actual data is 1630 : 2180)

5. Apply feature selection to the independent variables in the model in order to eliminate irrelevant independent variables and produce better results. With methods such as filter (e.g. Chi-Square Test, Fisher Score), wrapper (e.g. Recursive feature elimination, sequential feature selection algorithms (Forward / Backwards)) and embedded(e.g. L1 (LASSO) regularisation, decision tree).

6. Higher accuracy could be due to overfitting, test model on another dataset to check if model that it generalises well, as a model with a lower accuracy might generalise better

**Findings after testing**

1. weights improve after changing to a weight similar to real distribution of two rice classes, for example cammeo 0.4, osmancik 0.6

2. changing kernels does not help

3. changing times of cross validation does not help

4. Changing to type `nu-classification` classification improves the model

5. Increasing cost (penalty on wrongly classified values) improves the model slightly