Homework 2

Course: MA 5790

Question 4.1

a. What data splitting method(s) would you use for these data? Explain.

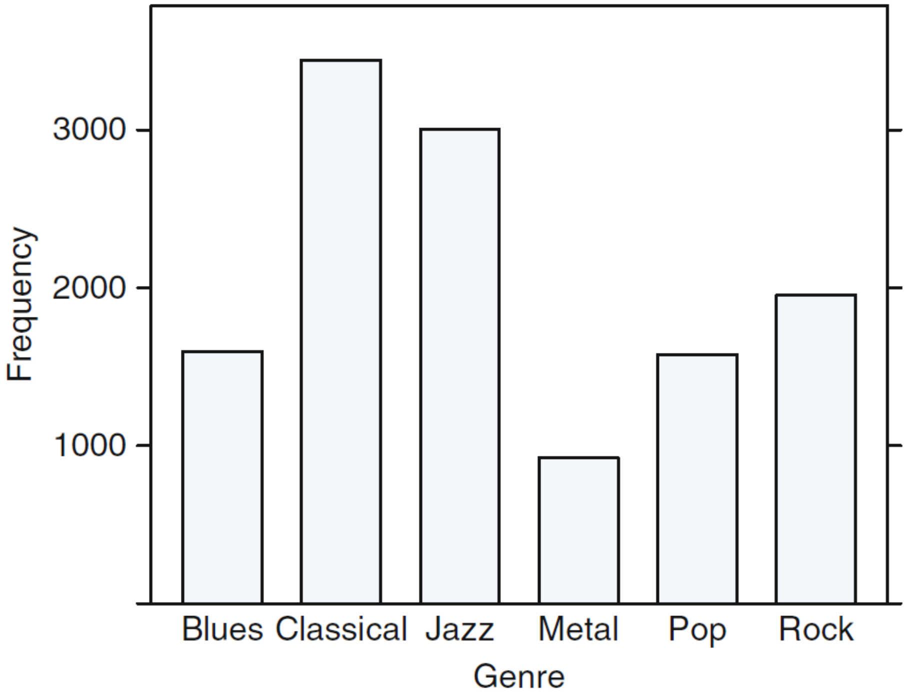


Fig. 1.1: The frequency distribution of genres in the music data

The frequency distribution of the music genres is given as fig 1.1 in the textbook, as shown above. The bar plot indicates an unbalanced response variable distribution, with classical music having the highest frequency (28%) and metal with the least frequencies (7%). All other response variable classes show similar variations in frequency. When determining the data splitting method for data of this form, we will have to consider the distribution of the samples across the different classes and the number of the samples relative to the number of predictors in the data. For these data, there are 12495 samples which are greater than the 192 predictor variables given.

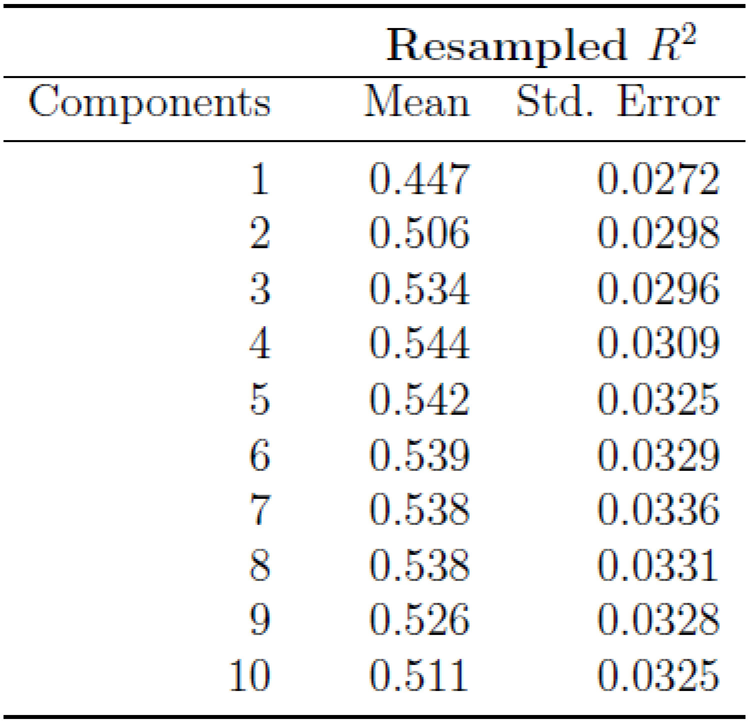
Hence, we can divide our data into training and test sets. This split would verify the model performance results without negatively impacting optimal tuning parameters selection estimation. Also, stratified random sampling could be an appropriate method to split the data set with an unbalanced distribution. In addition, considering the large data set, the computational requirements of the model have to be considered in choosing the resampling technique. Using k-fold cross-validation with k within the range of five to ten will be computationally less expensive than bootstrapping. However, bootstrapping may provide an accurate estimate of the tuning parameters and the model performance.

b. Using tools described in this chapter, provide code for implementing your approach(es).

All code is provided in Appendix A.

<u>Question 4.3</u>

a. Using the “one-standard error” method, what number of PLS components provides the most parsimonious model?



	Resampled R^2	
Components	Mean	Std. Error
1	0.447	0.0272
2	0.506	0.0298
3	0.534	0.0296
4	0.544	0.0309
5	0.542	0.0325
6	0.539	0.0329
7	0.538	0.0336
8	0.538	0.0331
9	0.526	0.0328
10	0.511	0.0325

Using the “one-standard error” method requires selecting the simplest model with accuracy greater than the best setting minus one standard deviation of its error. The best setting uses four PLS components with a lower bound equal to (0.544 – 0.0309). A model using three PLS components satisfies this requirement. Hence using three PLS components provides the most frugal model.

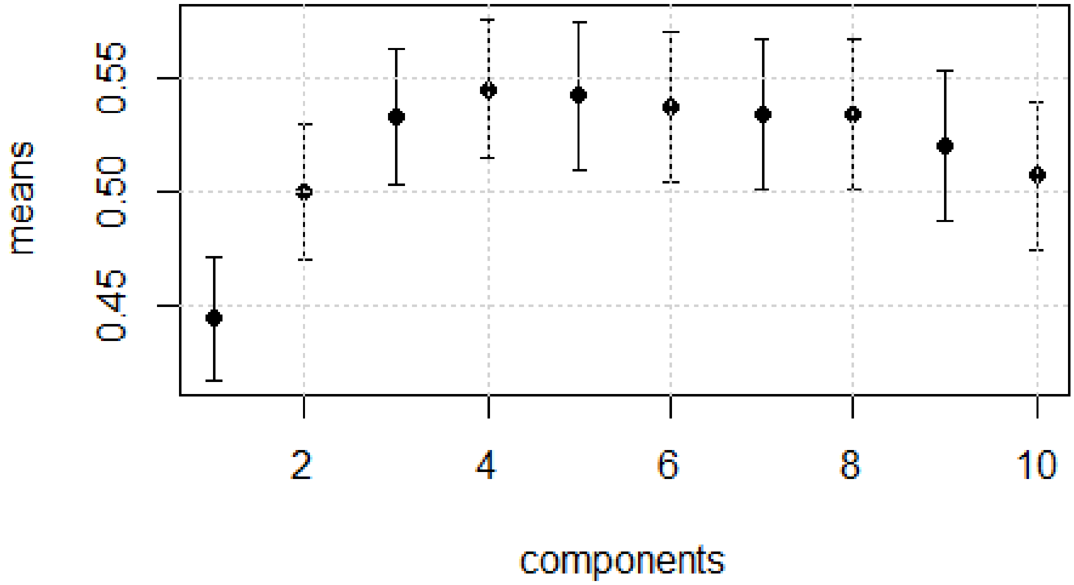


Figure 1. PLS components and means of different models

b.

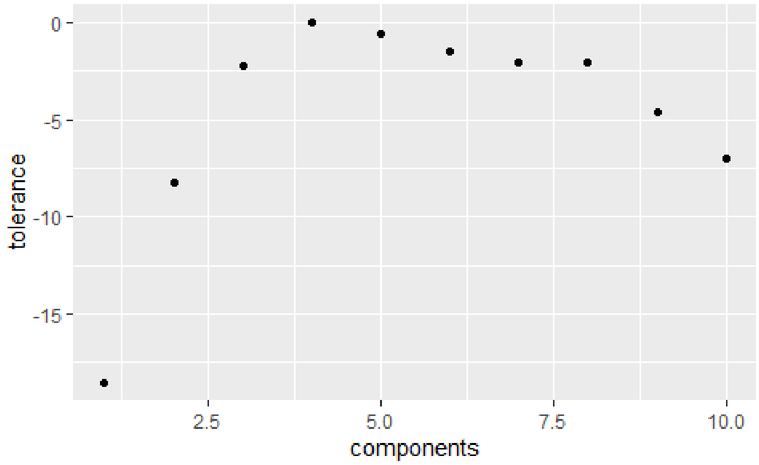
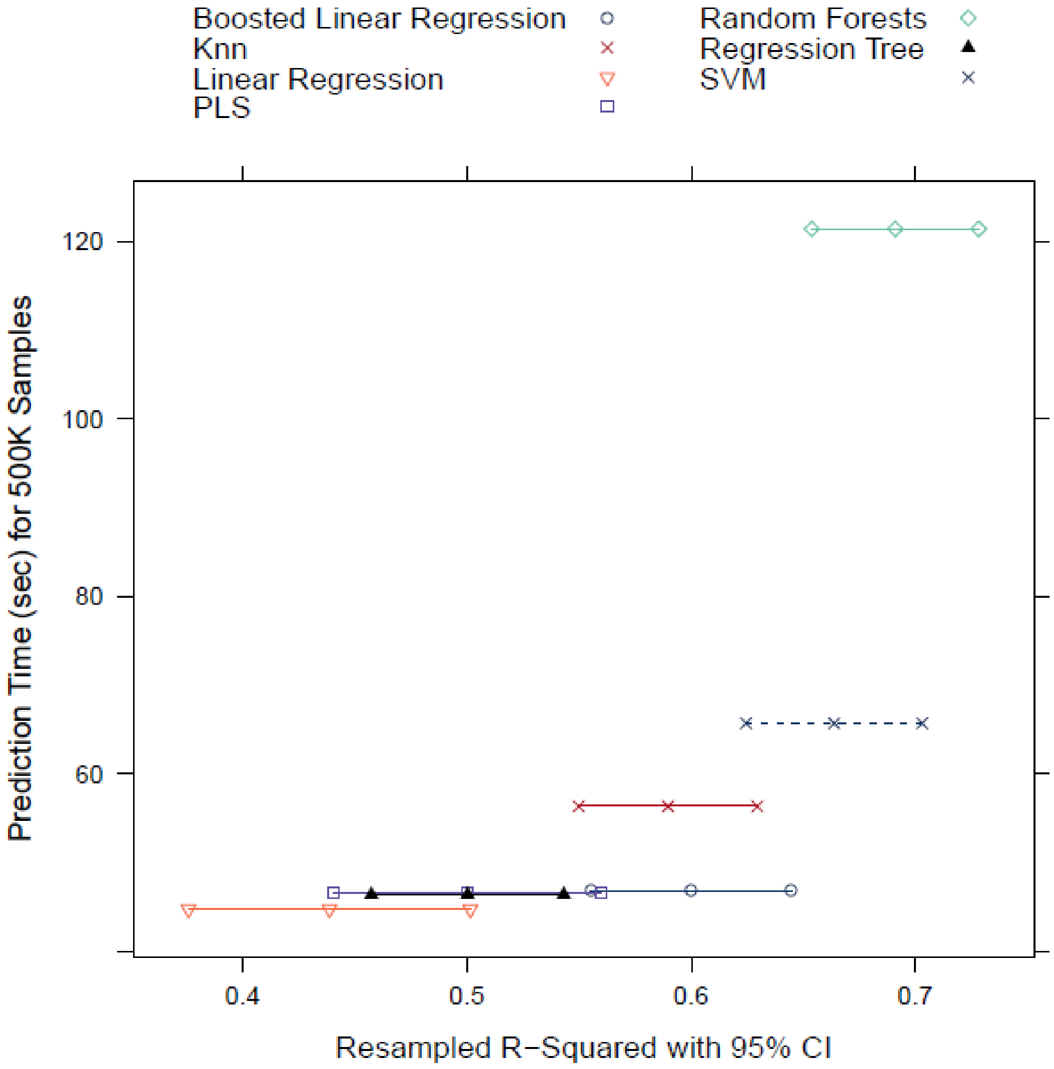


Figure 2. Tolerance against number of PLS components

We can use the equation (X – O)/O to calculate the tolerance, where X = performance value and O is the numerically optimal value. Looking at the figure above Figure 2. The lowest setting that does not exceed the 10% tolerance is a 2-component model.

c.



The model with the best R-squared value is random forest from the figure above. The support vector machine (SVM) has nearly equivalent results with some overlap for the R-squared values. The Boosted Linear regression is the next model, although it is worse than the support vector machine. However, the random forest or SVM models would be best based on R-squared alone.

d.

Using only R-squared values, random forest, and SVM models would be the best models. Considering execution time as given above in the figure. SVM model wins over the random forest. However, this may not hold in all implementations of the models. For implementations in which the prediction function will be recorded, neither of these models would be preferred. In those implementations, the PLS model or the regression tree model would be a better choice, nevertheless, with a significant drop in R-squared.

All code in Appendix B

Appendix A

```
#Load libraries and data
library(dplyr)
library(caret)
library(corrplot)
library(ggplot2)
library(e1071)
library(Hmisc)
library(mlbench)
library(reshape2)
library(subselect)
library(vcd)
library(AppliedPredictiveModeling)

#Question 4.1

#b

#set seed for reproducibility

set.seed(25)

# Create unbalanced classes

n <- 12495

classes <- sample(c(12,24,30), n, replace = TRUE, prob = c(0.7,0.2,0.1))

# Print table of proportions of classes

print(table(classes))

# Split classes using createDataPartition function
```

```
training <- createDataPartition(classes, p = 0.7)


# Verify creation of stratified datasets


table_training <-  table(classes[training$Resample1])


print(table_training/sum(table_training))


# Sample code for Using k = 10 fold cross-validation on training set classes
(train_classes)


ten_foldCV <- createDataPartition(train_classes, k = 10, returnTrain = TRUE)
```

Appendix B

```
#Question 4.3

#a
data(ChemicalManufacturingProcess)

save_plots = FALSE
set.seed(25)

# Get the given data into a form we can plot:
#
components <- 1:10
means <- c( 0.444, 0.500, 0.533, 0.545, 0.542, 0.537, 0.534, 0.534, 0.520, 0.507 )
std_errors <- c( 0.0272, 0.0298, 0.0302, 0.0308, 0.0322, 0.0327, 0.0333, 0.0330, 0.0326, 0.0324 )
data = data.frame( components, means, std_errors )

if(save_plots){postscript()}
errbar( components, means, means+std_errors, means-std_errors )
grid()
max_index = which.max( means )


#b
optimal_value <- subset(data, components == which.max(data$means) )
data$tolerance <- (data$means - optimal_value$mean)/optimal_value$means * 100

#Plot tolerance against number of PLS components

qplot(components, tolerance, data = data)
```