Michigan Technological University

Michigan Technological University, Computer Science

MA 5790 Combined Section - Predictive Modeling Assignment 1

Anil Silwal

October 19, 2018

Problem 4.1 Consider the Music Genre data set described in Sect. 1.4. The objective for these data is to use the predictors to classify music samples into the appropriate music genre.

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

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

Solution 4.1(a):

a) Out of 12495 obs to 192 variables, six response variables of music genre are defined. The bar plot and the table representing the frequency distribution of response variable is given below:

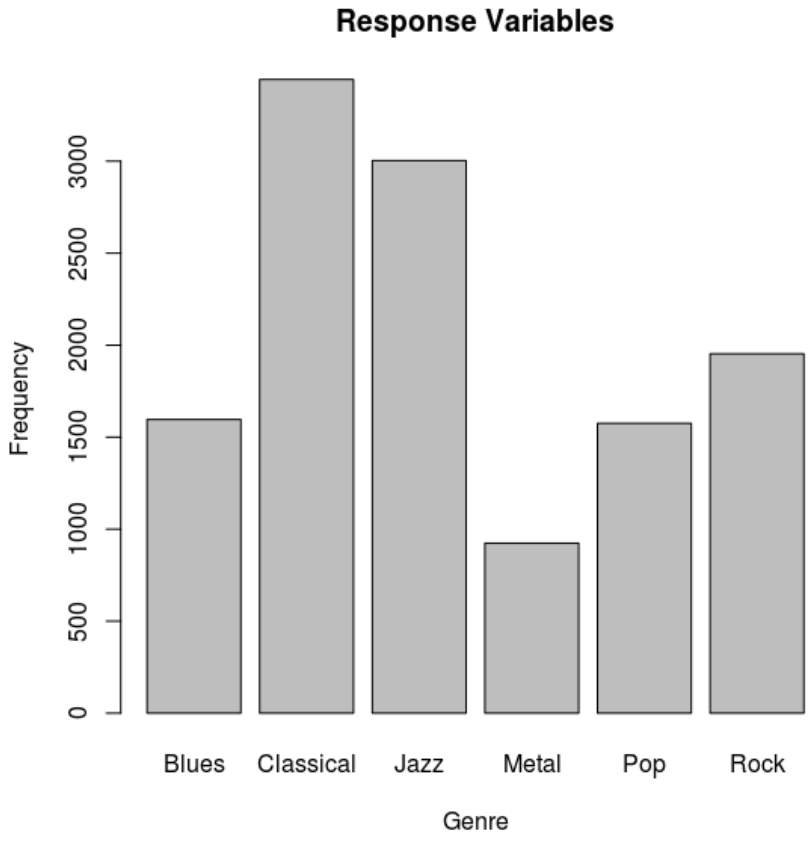


Figure 1: Bar plot showing distribution of response variables

Blue	Classical	Jazz	Metal	Pop	Rock
1596	3444	3003	924	1575	1953

When determining the data splitting method, we should focus on two primary characteristics:

- the number of samples relative to the number of predictors in the data, and
- the distribution of samples across classes.

For these data, there are 12495 samples which is greater than 192, the predictor variables. So, we can split the data into training set and test set. This split would enable us to verify the model performance results without impacting the estimation of optimal tuning parameter(s) selection. However, prior to making this choice, we must examine the class distribution of the response.

The bar plot shows imbalance distribution of response variable with metal genre the lowest percentage of samples(7%) and the classical genre the highest percentage of samples(28%). Depending on this response, stratified random sampling could be the best to split the data set.

Also, because the number of samples in the overall data set is large enough such that resampling or cross-validation techniques have good chance at randomly selection sample across all classes in similar proportion to the entire data set to estimate the model performance. K-Fold cross-validation with k = 5 or 10 would be the less computationally expensive.

Solution 4.1(b):

The package caret i.e library(caret) has various function for data splitting. To use the training/test splits, function createDataPartition could be used.

```
library(caret)
trainingRows =  createDataPartition(classes, p = .80, list= FALSE)
trainClasses = classes[trainingRows]
```

To divide the training data into 5 or 10 folds such that class distribution is sustainably maintained , again package caret is used. The following code could be used to create 10 folds.

```
library(caret)
partitions = createDataPartition(trainClasses, k = 5, returnTrain = TRUE)
```

Seed should be set for reproducibility.

Problem 4.2 Consider the permeability data set described in Section Sect. 1.4. The objective for this data is to use the predictors to model compounds' permeability.

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

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

Solution 4.2(a)

Out of 165 obs to 1107 variables are defined. The histogram plot representing the frequency distribution of response variable is given below:
The figure shows frequency distribution of permeability values.

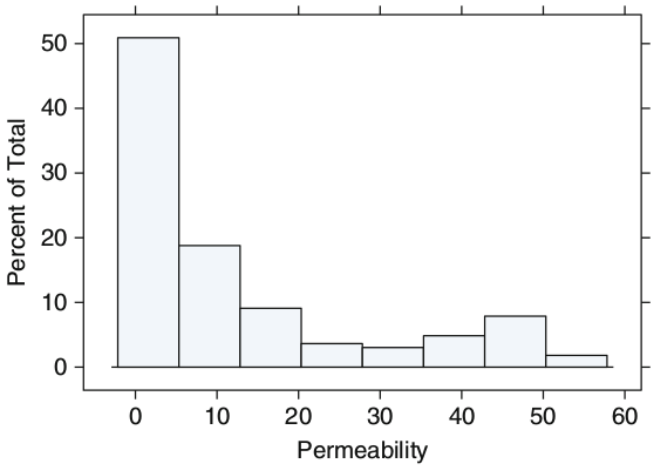


Figure 2: Histogram showing distribution of response variables

When determining the data splitting method, we should focus on two primary characteristics:

- the number of samples relative to the number of predictors in the data, and
- the distribution of samples across classes.

In this case,the number of samples is: 165, given by function nrow(fingerprints) and the number of predictors is: 1107 given by ncol(fingerprints). Since the number of samples are smaller than the number of predictors, it is not good to split the data into train/test test because splitting data sets into training and test set might affect the linkage between the predictor variables and response variable. In such case, resampling technique is suggested to use to select the tuning parameters and estimate the performance.

The figure shows distribution of permeability values are skewed. Hence, given the imbalance in the response variable, it might be suggested to use stratified random sampling method.

Solution 4.2(b)

The package caret i.e library(caret) has various function for data splitting. To create the stratified sampling such that class distribution is sustainable maintained , function createDataPartition would be used.

```
library(caret)
trainingRows =  createDataPartition(classes, p = .80, list= FALSE)
trainClasses = classes[trainingRows]
```

To create the multiple iterations of 5 folds cross-validation, again package caret is used. The following code could be used to create multiple iterations of 5 folds.

```
library(caret)
data(permeability)
multiFolds = createMultiFolds(permeability, k = 5, times = 20)
```

Seed should be set for reproducibility.

Problem 4.3 Partial least squares (Sect. 6.3) was used to model the yield of a chemical manufacturing process (Sect. 1.4).

(a) Using the "one–standard error" method, what number of PLS components provides the most parsimonious model?

(b) Compute the tolerance values for this example. If a 10% loss in R^2 is acceptable, then what is the optimal number of PLS components?

(c) Several other models (discussed in Part II) with varying degrees of complexity were trained and tuned and the results are presented in Figure **??**. If the goal is to select the model that optimizes R^2, then which model(s) would you choose, and why?

(d) Prediction time, as well as model complexity (Sect. 4.8) are other factors to consider when selecting the optimal model(s). Given each model's prediction time, model complexity, and R^2 estimates, which model(s) would you choose, and why?

Solution 4.3(a)

	Resampled R^2	
Components	Mean	Std. Error
1	0.444	0.0272
2	0.500	0.0298
3	0.533	0.0302
4	0.545	0.0308
5	0.542	0.0322
6	0.537	0.0327
7	0.534	0.0333
8	0.534	0.0330
9	0.520	0.0326
10	0.507	0.0324

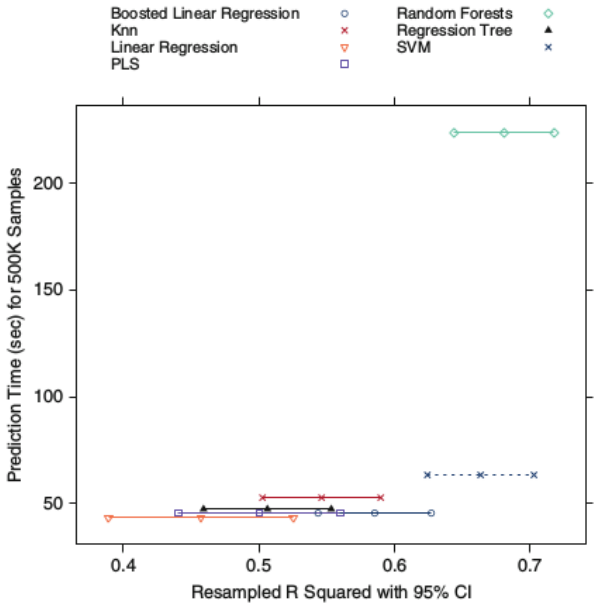
The one-standard error rule would select the simplest model with accuracy no less than 0.5142 (0.545 - 0.0308). This corresponds to 3PLS which is 0.533. The "pick-the-best" solution is component 3 PLS. Although, the 9PLS seems best-choice but the cost for computation in 9PLS is higher than 3PLS, so I suggest a model with 3 PLS components is the most parsimonious model (simpler).

Solution 4.3(b)

Computed the tolerance value:
Mean : 0.4440 0.5000 0.5330 **0.5450** 0.5420 0.5370 0.5340 0.5340 0.5200 0.5070
Error: 0.0272 0.0298 0.0302 0.0308 0.0322 0.0327 0.0333 0.0330 0.0326 0.0324
Tolerance: -0.1853 -0.0826 -0.0220 **0.0000** -0.0055 -0.0147 -0.0202 -0.0202 -0.0459 -0.0697
Tolerance given by (X-O)/O where X = performance value and O = numerically optimal value:

Given, a 10 % loss in accuracy was acceptable as a trade-off for a simpler model, then best optimal number of PLS components is a 2PLS.

Solution 4.3(c)



Looking at figure above, the model with the best R^2 value is random forest. However, the support vector machine (SVM) has nearly equivalent results with some overlap. The Boosted Linear Regression seems like possible next model based on R^2 value but this model has worse value than SVM. So, the best model in terms of R^2 are random forest and SVM.

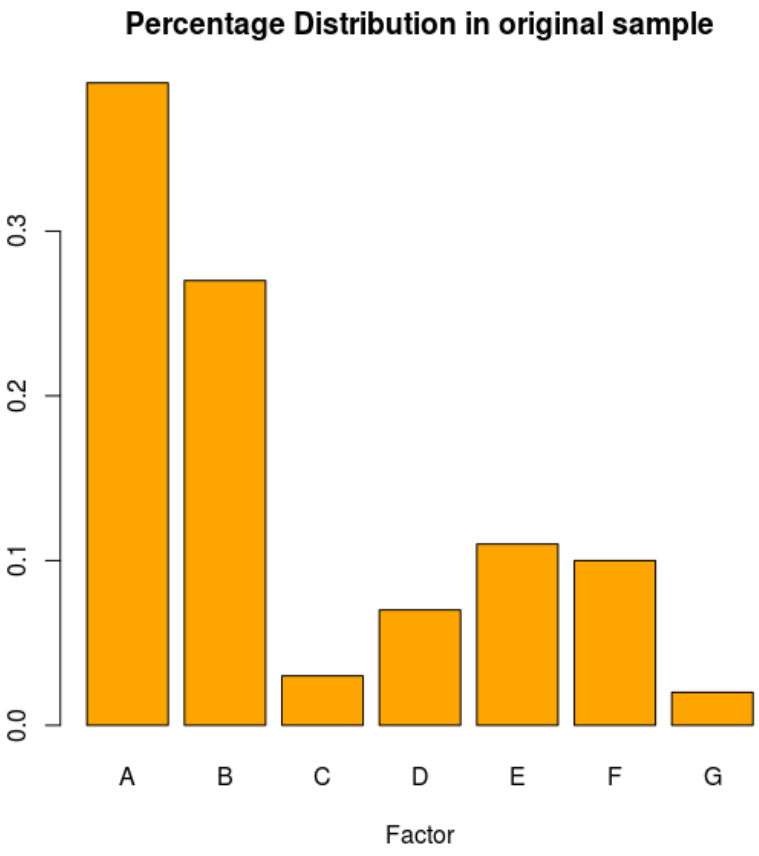
Solution 4.3(d)

Th best model in terms of R^2 alone, random forest and SVM models would be the ones. Considering execution time, SVM model clearly wins since it is far faster. But if, prediction function is needed to be recorded, PLS and regression tree models would be considered although they give low R^2. Hence the selection of best model depends highly on the implementation.

Problem 4.4

Solution 4.4(a)

Given Sample Oil Type has following data with total value = 96. The original frequency distribution of given sample is:



A	B	C	D	E	F	G	Total
37	26	3	7	11	10	2	**96**

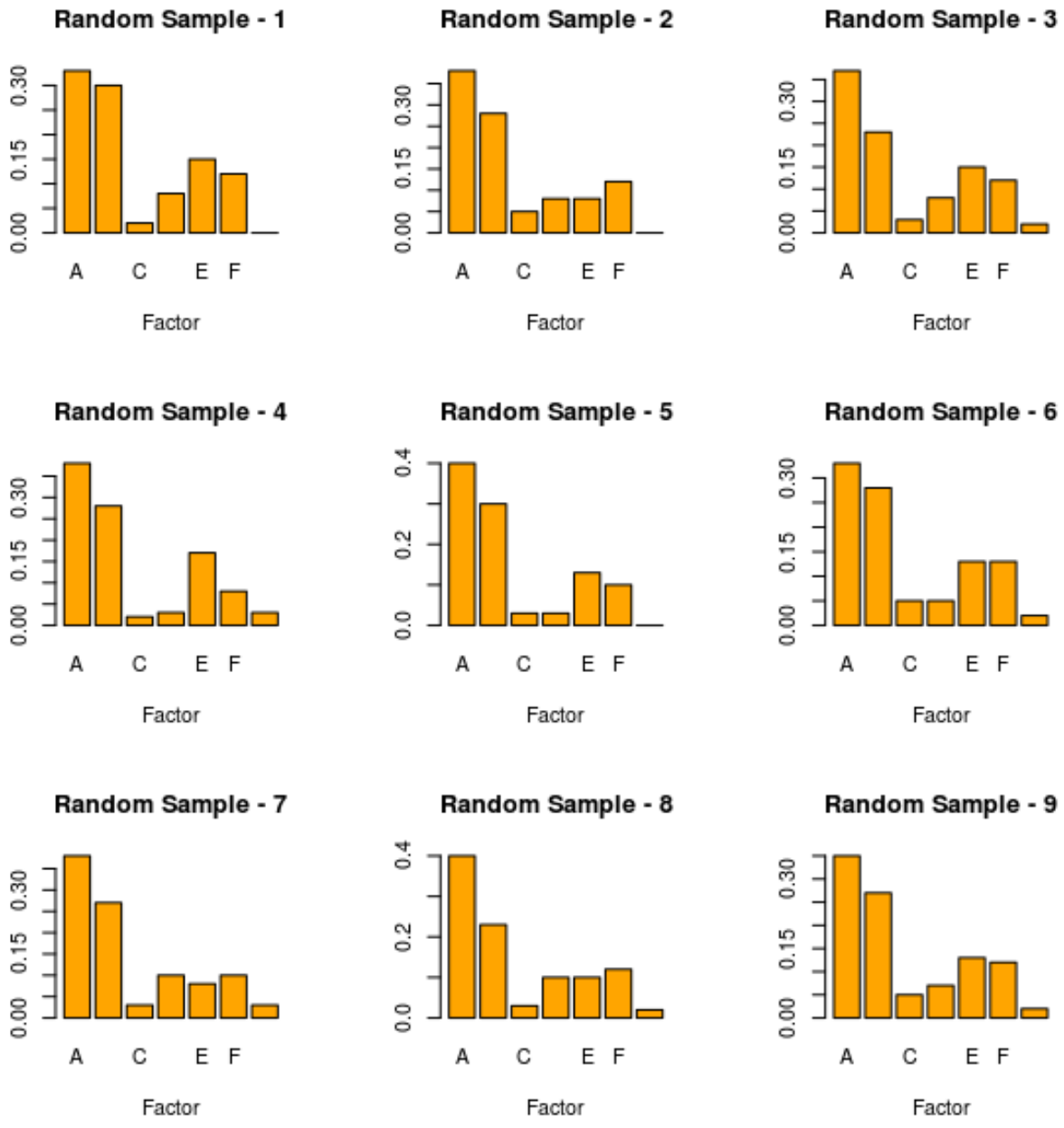


Figure 3: Bar plot showing percentage distribution of random variables

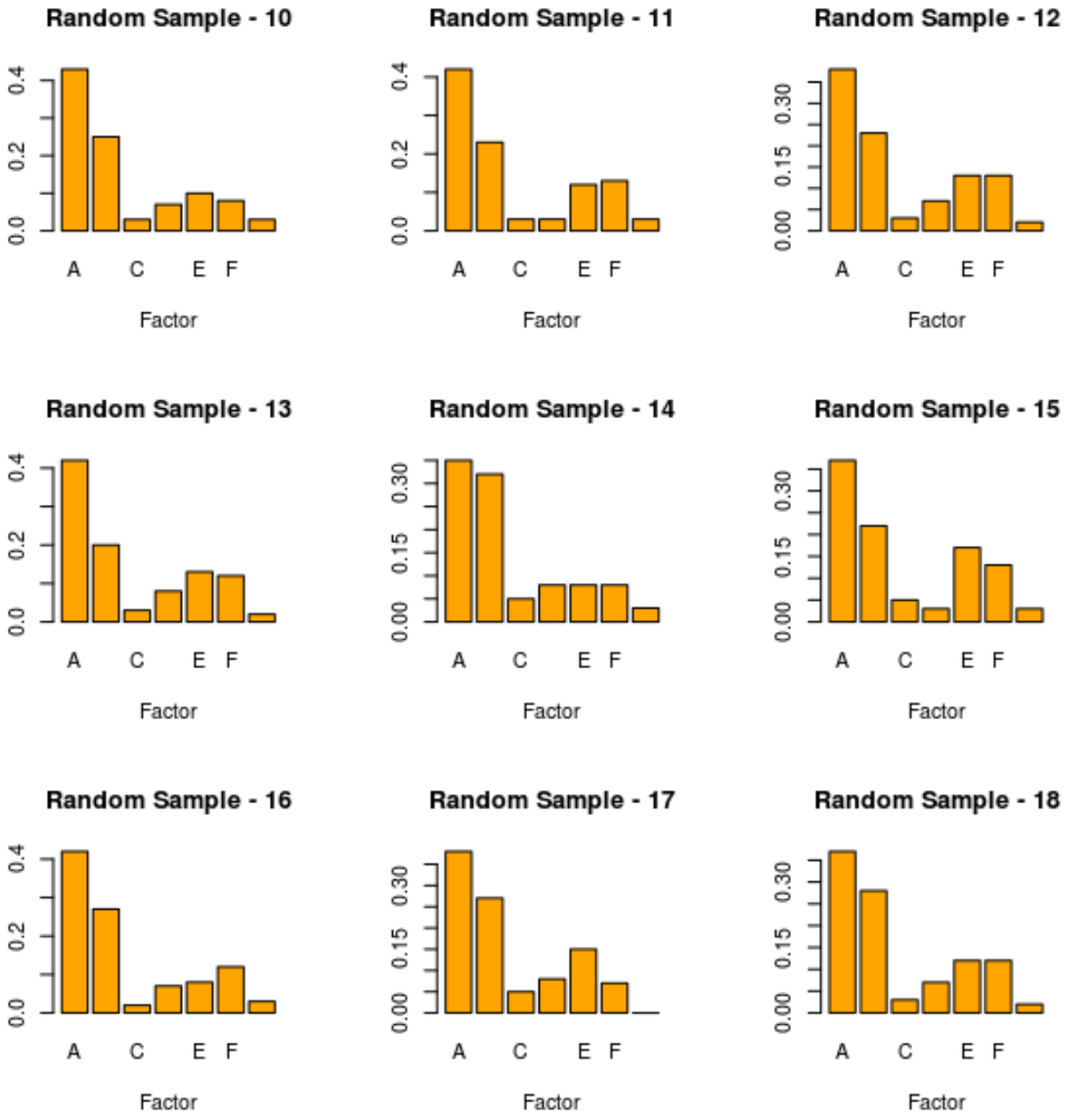


Figure 4: Bar plot showing percentage distribution of random variables

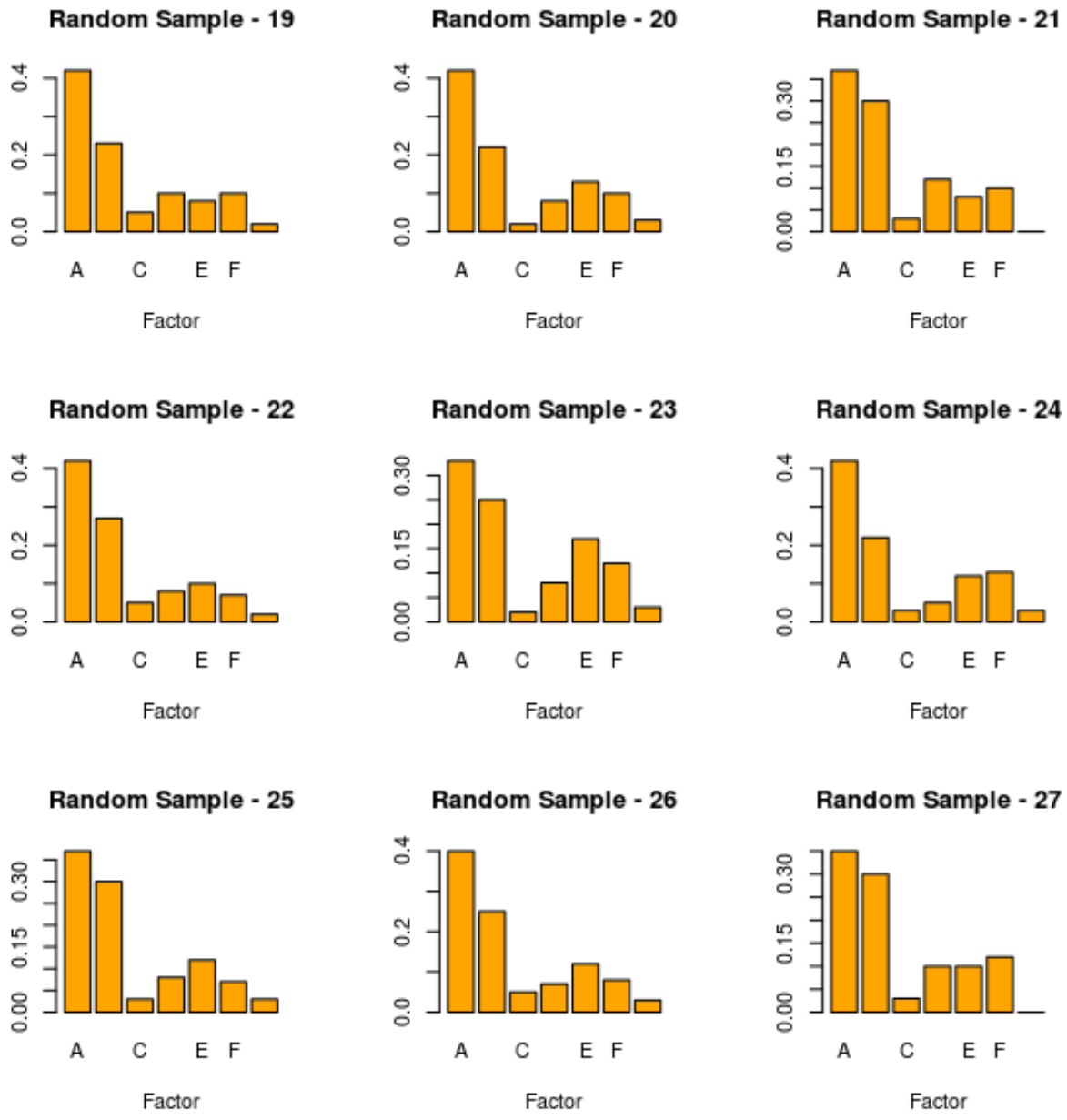


Figure 5: Bar plot showing percentage distribution of random variables

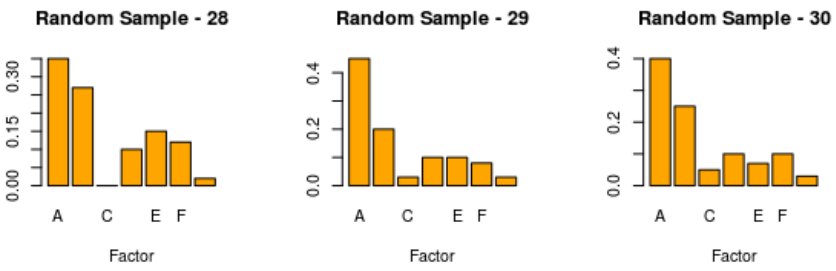


Figure 6: Bar plot showing percentage distribution of random variables

The 30 random observation in above bar plot clearly explains that each of the random sample are different from the original and these 30 different random sampling of 60 samples each were looked when further deep then, also the frequencies were different. In some instance, **frequency of G, C goes to 'zero'**. Hence training set will not use the information of all classes if random samples were considered for data splitting. Because of this reason, the data splitting using random sampling is not effective for modeling.

Solution 4.4(b)

In package caret, the createDataPartition() function generates random samples considering each class. This means the random sample are created from each of the class so that frequency distribution of stratified random variables are significantly closer to the original sample in terms of frequency distribution.

Comparing to the random sampling, this gives better results in frequency and data selection and is considered thus, a reasonably good strategy to model the data set. Following is a near-by observation of 30 stratified random variable sampling method which will explain in better.

If following graph you will observe that frequency distribution in all of the 30 random samples gives near to original data frequency distribution. And you will also observe that, **using this method in graph generate below, no any variables A,B,C,D,E,F,G have became 'zero'** during stratified random sampling for all of 30 random variables.

Because of this, stratified random variable sampling is better than random variable in this type of given data.

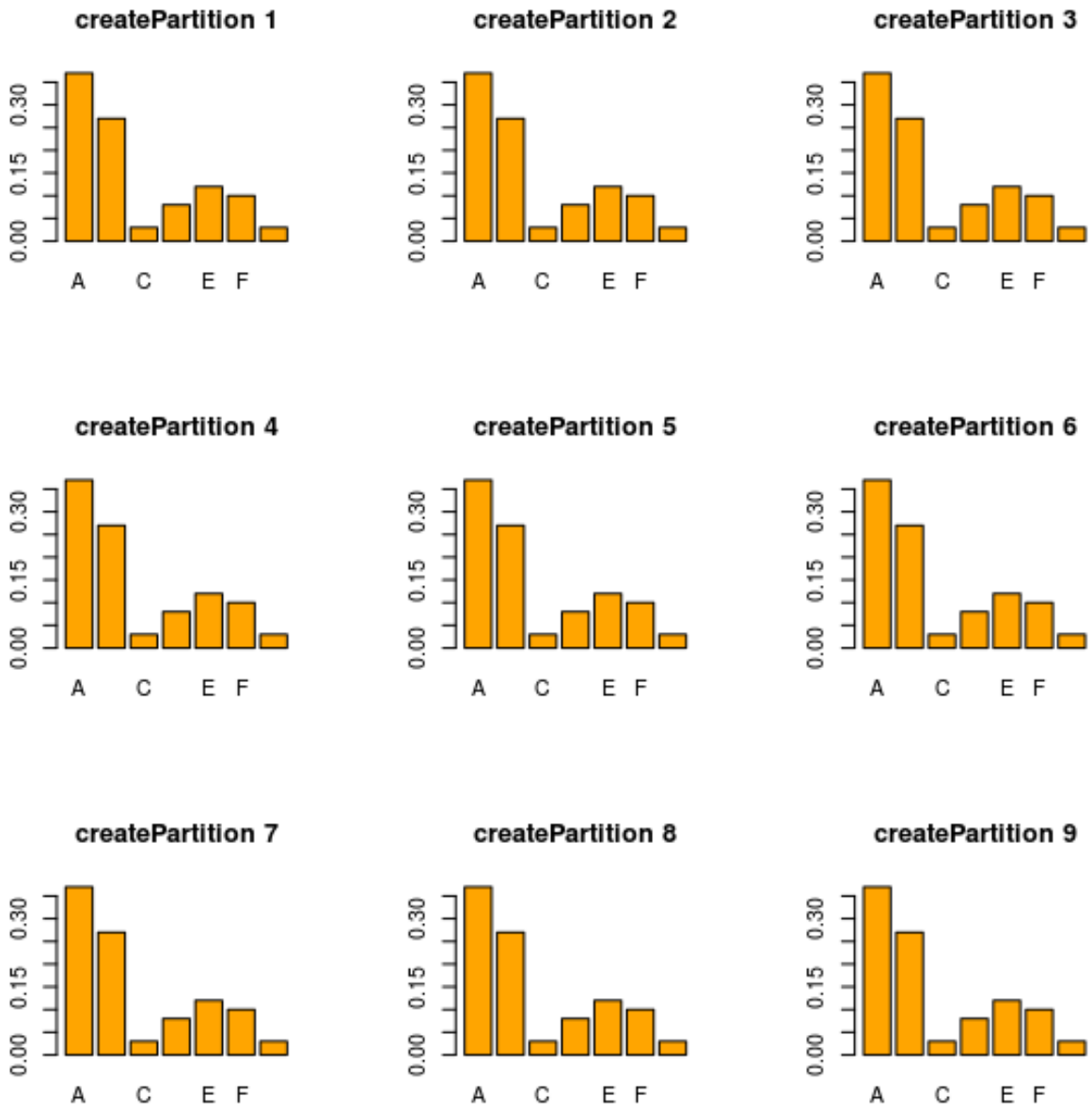


Figure 7: Bar plot showing percentage distribution of stratified random variables

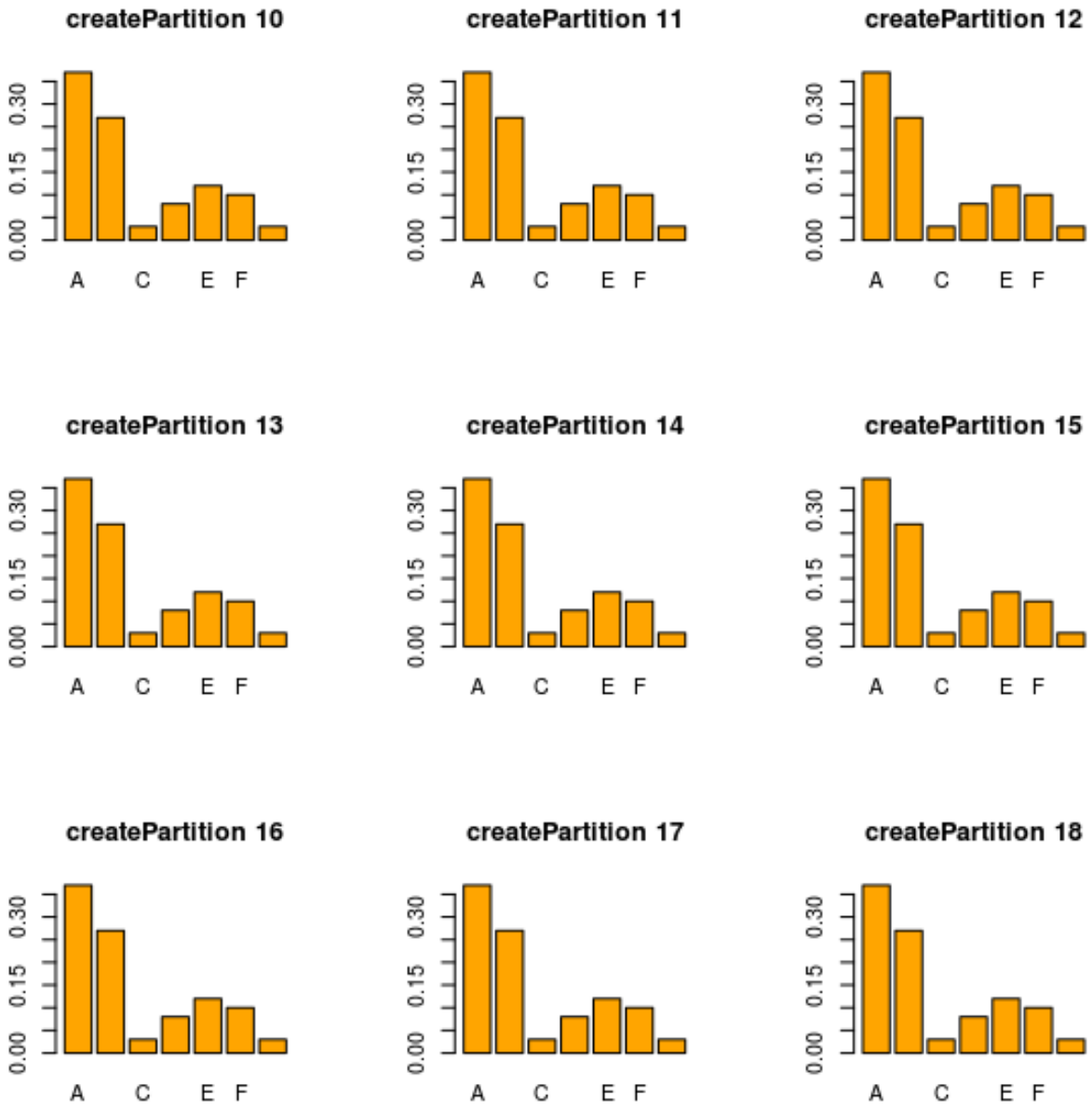


Figure 8: Bar plot showing percentage distribution of stratified random variables

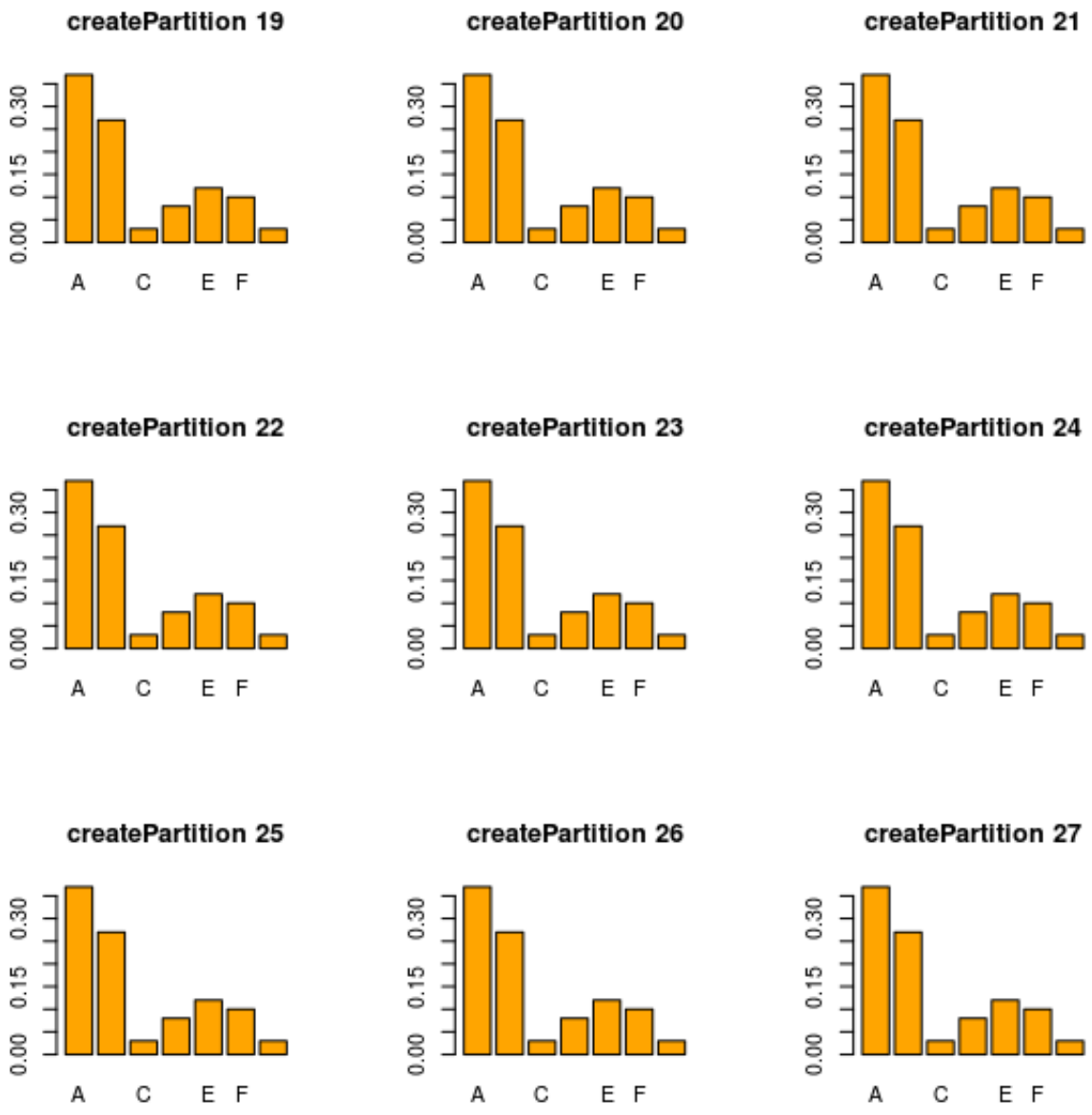


Figure 9: Bar plot showing percentage distribution of stratified random variables

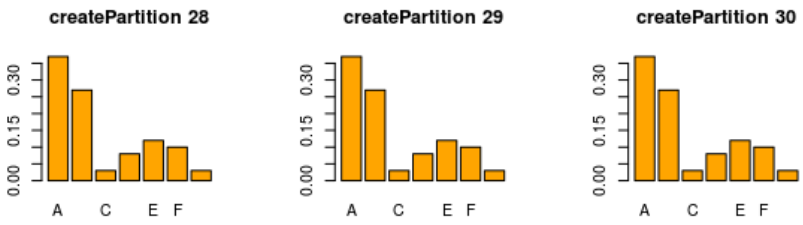


Figure 10: Bar plot showing percentage distribution of stratified random variables

Solution 4.4(c)

If the size of sample is small, then partitioning the data set into train and test data sets can be inefficient and the reason is: train data set can not capture all aspects of predictor in such case.
For the test set, Leave-one out cross validation (LOOCV) would be the good option to determine the performance of a model.

```
library(mlbench)
library(caret)
library(e1071)
library(AppliedPredictiveModeling)
library(dplyr)
library(data.table)

##################################
#Question 4.1(a)
##################################


musicGenre = read.csv(" /Downloads/MA 5790/genresTrain.csv", header=TRUE)
counts = table(musicGenre$GENRE)
barplot(counts, main="Response Variables", xlab="Genre", ylab="Frequency")

##################################
#Question 4.3(b)
##################################


print(mean)
mean = c(0.444,  0.500, 0.533, 0.545,  0.542, 0.537, 0.534, 0.534, 0.520, 0.507)
print(error)
error = c(0.0272,  0.0298,  0.0302,  0.0308, 0.0322,  0.0327,  0.0333,  0.0330, 0.0326, 0.0324)
print(toler)
toler = round((mean - 0.545) / 0.545, 4)


##################################
# Question 4.4(a)
# normal random variable
##################################


data(oil)
str(oilType)
print(table(oilType))

# total variable count is 96, so 96 is taken
tb = round(table(oilType) / 96, 2)

par(mfrow = c(1,1))
barplot(tb, main="Percentage Distribution in original sample",
        xlab="Factor", col="orange")


# this is done to divide the graph in 4 columns
par(mfrow = c(3,3))

# total number of random variable size is considered 60
sampleSize= 60
set.seed(12345)
for (i in 1:30)-
  random oilType = round(table(sample(oilType, size =sampleSize)) / sampleSize, 2)
  barplot(random oilType, main=paste("Random Sample -",i),
          xlab="Factor", col="orange")


##################################
# Question 4.4(b)
# stratified random variable
##################################
```

```
set.seed(234569)
list caret = createDataPartition(oilType, p = 0.59, times = 30)
perc caret = lapply(list caret, function(x, y) round(table(y[x])/60, 2), y = oilType)

# this is done to reset the page
par(mfrow = c())
# this is done to divide the graph in 4 columns
par(mfrow = c(3,3))

for (i in 1:30)-
  barplot(perc caret[[i]], horizontal = F, main = paste('createPartition',i), col="orange")
```

References:

1. Applied Predictive Modeling : @authors Max Kuhn. Kjell Johnson
2. https://archive.ics.uci.edu/ml/index.php