Random Sample

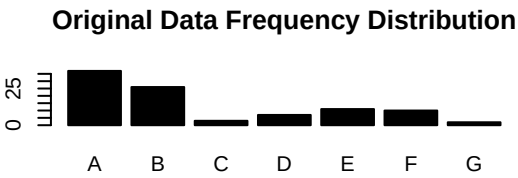
Amin Fesharaki

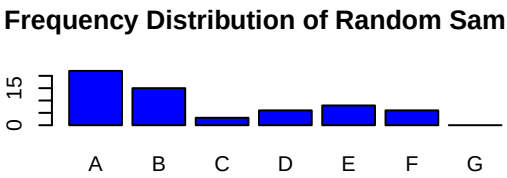
5/21/2022

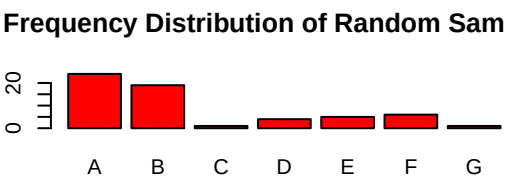
Brodnjak-Vonina et al. (2005) develop a methodology for food laboratories to determine the type of oil from a sample. In their procedure, they used a gas chromatograph (an instrument that separates chemicals in a sample) to measure seven different fatty acids in an oil. These measurements would then be used to predict the type of oil in a food sample. To create their model, they used 96 samples of seven types of oils. These data can be found in the caret package using data(oil). The oil types are contained in a factor variable called oilType. The types are pumpkin (coded as A), sunflower (B), peanut (C), olive (D), soybean (E), rapeseed (F) and corn (G).

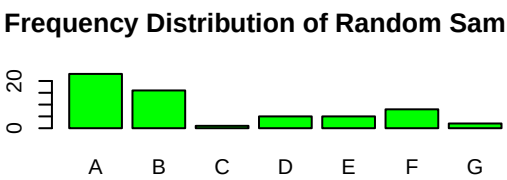
A) Use the sample function in base R to create a completely random sample of 60 oils. How closely do the frequencies of the random sample match the original samples? Repeat this procedure several times to understand the variation in the sampling process.

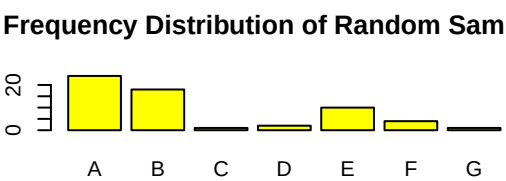
```
set.seed(20) #Set seed for reproducible results
R1 <- sample(oilType, size = 60) #Random sample with size 60 #1
R2 <- sample(oilType, size = 60) # Random Sample #2
R3 <- sample(oilType, size = 60) # Random Sample #3
R4 <- sample(oilType, size = 60) # Random Sample #4
R5 <- sample(oilType, size = 60) # Random Sample #5
par(mfrow=c(3,2)) # Display all bar charts right next to each other for analysis
org <- plot(oilType, main = 'Original Data Frequency Distribution', type = 'bar', col= 'black') #Bar ch
#Bar chart to show frequencies
plot(R1, main = 'Frequency Distribution of Random Sample 1', type = 'bar', col = 'blue')
plot(R2, main = 'Frequency Distribution of Random Sample 2', type = 'bar', col = 'red')
plot(R3, main = 'Frequency Distribution of Random Sample 3',  type = 'bar', col = 'green')
plot(R4, main = 'Frequency Distribution of Random Sample 4', type = 'bar', col = 'yellow')
plot(R5, main = 'Frequency Distribution of Random Sample 4', type = 'bar', col = 'purple')
```

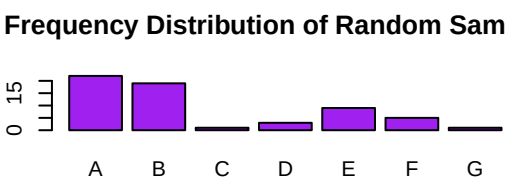








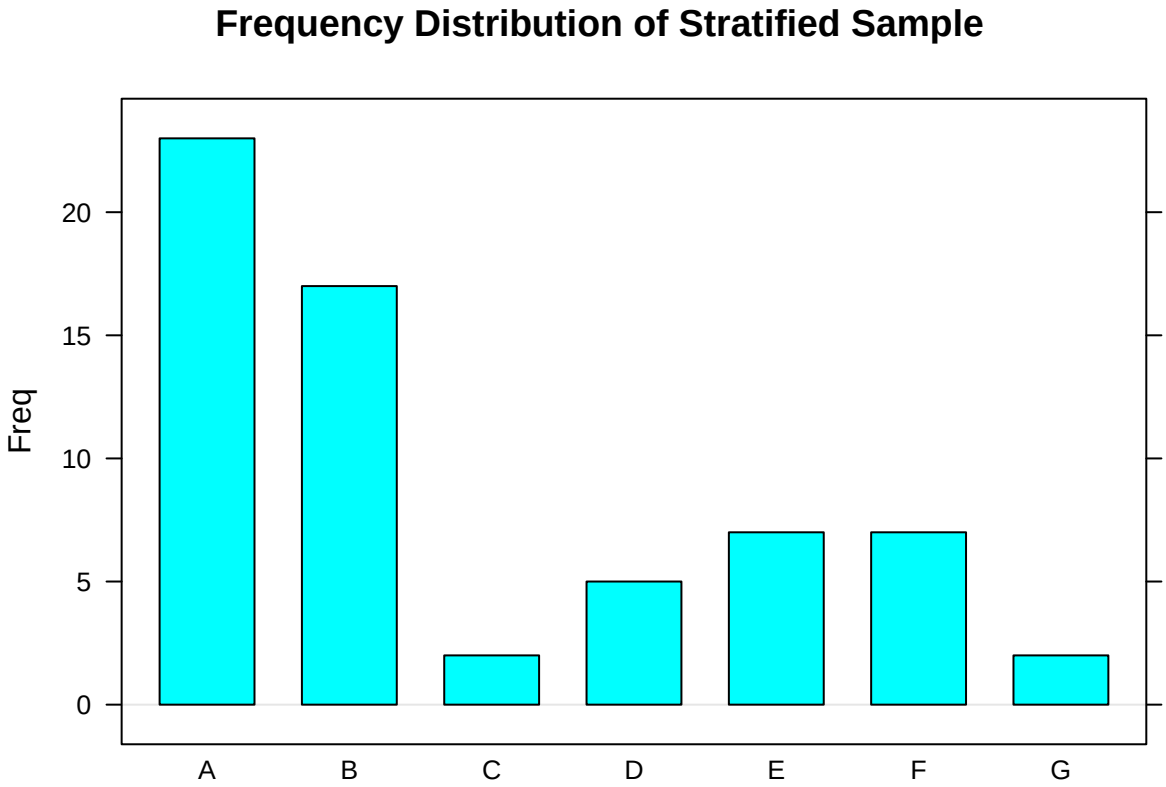




* According to the figures above, the random samples do not look significantly different for being random samples of the data. All figures capture a high frequency with oil types A and B, with A always being slightly bigger than B. However, oil types C, D, E, F, and G vary between the figures. Random samples 2 and 3 closely resemble the original distribution. It is important to note that random samples do not always come out with a desirable frequency distribution. As shown in random sample 1, the G class has little to no samples within that class, which throws of an accurate model representation for class G if it was used. Random sample 4 shows a significant less class distribution for oil types C, D, F, and G when compared to the original distribution. In conclusion, when using the sample function, there is a chance where the sample closely represents the original distribution. However, there is also a chance where it gives a distribution that is vastly different from the original.

B) Use the caret package function createDataPartition to create a stratified random sample. How does this compare to the completely random samples?

```
Strat = createDataPartition(oilType, p = 0.62) # Stratified Random Sample (.62 roughly equates to sampl
Strat_t = lapply(Strat, function(x, y) round(table(y[x])), y = oilType) #Incorporate Oil Types into bar
barchart(Strat_t[[1]],  main = 'Frequency Distribution of Stratified Sample', horizontal = FALSE)
```



* The stratified random sample resembles the original distribution very closely. Each class has roughly the same amount with the same order of classes in terms of frequency (A>B>E>F>D>C>G) just like the original distribution. Stratified sampling is more effective in returning a similar frequency distribution than using simple random sampling.

C) With such a small sample size, what are the options for determining performance of the model? Should a test set be used?

- For samples that are not large, there is a strong case that a test set should be avoided because every sample may be needed for a sufficient model. In addition, the test size may not have sufficient precision to make a reasonable inference. To combat this issue, resampling methods such as cross-validation can be used to produce appropriate estimates of model performance, Resampling techniques can produce performance estimates superior to a single test set since they evaluate many alternatives to the data. In regards to a small sized data set, leave on out cross validation (LOOCV) can be used with a relatively good computational time.

D) One method for understanding the uncertainty of a test set is to use a confidence interval. To obtain a confidence interval for the overall accuracy, the based R function binom.test can be used. Try different sample sizes and accuracy rates to understand the trade-off between the uncertainty in the results, the model performance, and the test set size.

```
Example <- binom.test(16,20) #Example Test
Accuracy_Percent <- c(.95,.85,.75, .95,.85,.75, .95,.85,.75, .95,.85,.75, .95,.85,.75) #Accuracy Percen
```

```r
Sample_Size <- c(20,20,20,17,17,17,15,15,15,10,10,10,5,5,5) #Sample Size
Accuracy <- round(Sample_Size*Accuracy_Percent) #Return integer for Binom.Test

#Create initial Data Frame to be combined later
BT1 <- binom.test(Accuracy[1], Sample_Size[1]) #First Binomial Test
p_val <- t(as.data.frame(BT1$p.value)) # Create p-value data frame
conf_int <- t(as.data.frame(round(BT1$conf.int,3))) # Create confidence int. data frame
estimate <- t(as.data.frame(round(BT1$estimate,2))) # Create estimate data frame

#Create Column Names
colnames(p_val) = c('P-value')
colnames(conf_int) = c('Lower Bound', 'Upper Bound')
colnames(estimate) = c('Accuracy')

#15 different Binomial test with various Sample Sizes and Accuracy (Manual Inspection)
BT2 <- binom.test(Accuracy[2], Sample_Size[2])
BT3 <- binom.test(Accuracy[3], Sample_Size[3])
BT4 <- binom.test(Accuracy[4], Sample_Size[4])
BT5 <- binom.test(Accuracy[5], Sample_Size[5])
BT6 <- binom.test(Accuracy[6], Sample_Size[6])
BT7 <- binom.test(Accuracy[7], Sample_Size[7])
BT8 <- binom.test(Accuracy[8], Sample_Size[8])
BT9 <- binom.test(Accuracy[9], Sample_Size[9])
BT10 <- binom.test(Accuracy[10], Sample_Size[10])
BT11 <- binom.test(Accuracy[11], Sample_Size[11])
BT12 <- binom.test(Accuracy[12], Sample_Size[12])
BT13 <- binom.test(Accuracy[13], Sample_Size[13])
BT14 <- binom.test(Accuracy[14], Sample_Size[14])
BT15 <- binom.test(Accuracy[15], Sample_Size[15])


# For Loop for an easier comparison (Want to create a table of values)
for (i in 2:15)
{
  Bt <- binom.test(Accuracy[i], Sample_Size[i]) #Binom Test
  p_val2 <- t(as.data.frame(Bt$p.value)) # Pull p-value from BinomTest
  conf_int2 <- t(as.data.frame(round(Bt$conf.int,3))) # Pull conf. int. from BinomTest
  estimate2 <- t(as.data.frame(round(Bt$estimate,2))) # Pull accuracy from BinomTest

  # Add on values into data frame
  p_val <- rbind(p_val, p_val2 )
  conf_int <- rbind(conf_int, conf_int2)
  estimate <- rbind(estimate, estimate2)
}

Size <- as.data.frame(Sample_Size) #Turn Sample Size into a data frame
colnames(Size) <- c('Sample Size')
Table <- cbind(Size, estimate, conf_int, p_val) #Bind Sample Size, Esimate, Confidence interval, and P
rownames(Table) <- NULL #Remove the names of the rows

head(Table,9)
```

```
##   Sample Size Accuracy Lower Bound Upper Bound      P-value
## 1          20     0.95       0.751       0.999 4.005432e-05
```

```
## 2          20     0.85       0.621       0.968 2.576828e-03
## 3          20     0.75       0.509       0.913 4.138947e-02
## 4          17     0.94       0.713       0.999 2.746582e-04
## 5          17     0.82       0.566       0.962 1.272583e-02
## 6          17     0.76       0.501       0.932 4.904175e-02
## 7          15     0.93       0.681       0.998 9.765625e-04
## 8          15     0.87       0.595       0.983 7.385254e-03
## 9          15     0.73       0.449       0.922 1.184692e-01
```

- According to the table, as the accuracy decreases (with the same sample size), the lower bound and upper bound confidence interval decreases. As the sample size decreases (with the same accuracy), the upper bound decreases very slightly while the lower bound decreases at a much higher rate. In other words, the range of the confidence interval decreases as the sample size and accuracy increase. To increase model performance with the same sample size, one should increase the accuracy. Likewise, if accuracy is held constant, then a larger sample size would result in a better model.