IS 606: Statistics and Probability for Data Analytics
Hands-On Laboratory Series
Discrete Probability Distributions: Fundamentals

Overview

This exercise is designed to give you practice in working with basic features of a discrete probability distribution in order to provide key summary properties of the distribution.

Prerequisites

You should know the basic concepts of discrete probability distributions, including the essential properties of a probability distribution, the core ideas of expected value and standard deviation, and the basics of a percentile summary approach.

Materials

In order to complete this lab exercise, you will need to obtain the associated CSV file that contains the raw data:

lab-data-file-discrete-distributions.csv

All other necessary information and instruction is contained within the exercise.

Instructions

This lab exercise is to be completed step by step according to the instructions given. If you are struggling with a particular step, then our recommendation is that you look to the solution ***for only that step*** for help. Once you have sorted out the details of the step in question, proceed to the next task.

1. Read the dataset into R and compile the probability distribution by using the table() function.

   The following code reads in the file on my machine and tabulates the distribution. You will need to alter the file's path on your own machine to match the directory where you have stored it. (Note that code that is indented is actually a continuation of the same line.)

```
mydata <- read.csv(file="C:/labfiles/lab-data-file-
     discrete-distributions.csv", header=T)
mycounts <- table(mydata$Outcome)
mydist <- mycounts / sum(mycounts)
print(mydist)
```

The resulting output is:

```
    1     2     3     4     5     6     7     8
0.030 0.095 0.175 0.305 0.220 0.110 0.030 0.035
```

2. Verify that the distribution you have tabulated is a legitimate probability distribution by checking whether it satisfies the three rules of probability distributions, namely:

   - Each probability value is nonnegative.
   - The sum of all probability values is exactly 1. (In general, when working from data like this, you have to watch out for rounding here. In our exercise, the number of observations is 100, so this should not be a problem.
   - Each possible individual outcome in the sample space is assigned one value.

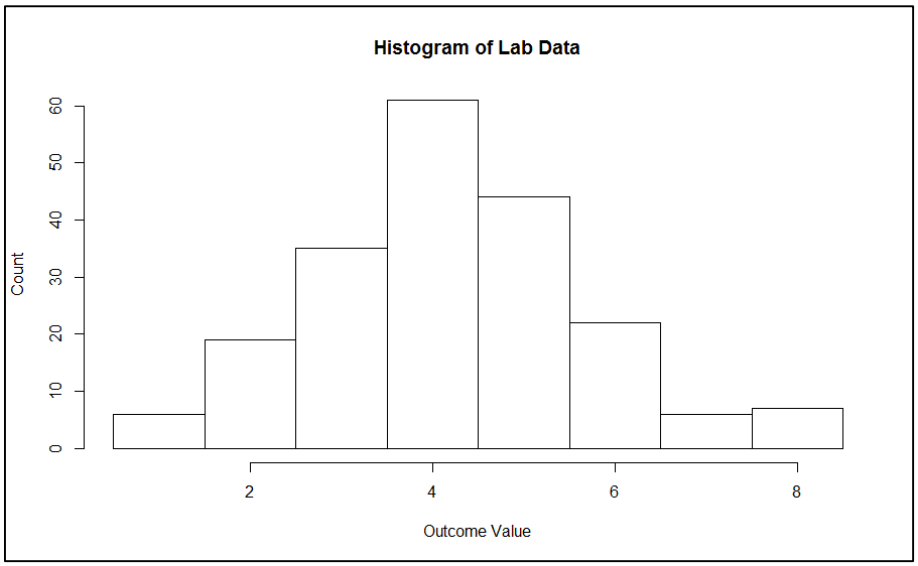
   These are easy enough to verify from the table above. All of the values are, in fact, positive (though 0 would be allowed) and the values add up to one. In addition, we will assume the instructor has provided us with all outcomes and their probabilities. (You can usually verify this from problem context.)

3. Plot a histogram of the raw data. Since the values of the distribution are whole numbers ranging from 1 to 8, I would suggest using the hist() function and specifying break points using the parameter breaks=c(0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5). This gives a visual representation of the probability distribution.

   The following code produces the histogram (note that I am including a couple of cosmetic parameters, such as main, to make it look nicer):

```
hist(mydata$Outcome,
     breaks=c(0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5),
     xlab="Outcome Value", ylab="Count",
     main="Histogram of Lab Data")
```

   Here is the plot that results:



4. Calculate the expected value of the distribution using the theoretical formula:

$$E(X) = \sum_{i=1}^{n} x_i p(x_i)$$

The expected value is calculated as follows:

$$\boldsymbol{E(X) = (0.030)(1) + (0.095)(2) + (0.175)(3) + (0.305)(4) + (0.220)(5) + (0.110)(6) + (0.030)(7) + (0.035)(8) = 4.215}$$

5. Calculate the variance and standard deviation of the distribution using the theoretical formulas:

$$Var(X) = \sum_{i=1}^{n} (x_i - E(X))^2 p(x_i) \qquad and \qquad SD(X) = \sqrt{Var(X)}$$

$$\boldsymbol{Var(X) = (1 - 4.215)^2(0.030) + (2 - 4.215)^2(0.095) + \cdots + (8 - 4.215)^2(0.035)}$$

$$\boldsymbol{Var(X) = 2.27}$$

$$\boldsymbol{SD(X) = \sqrt{2.27} = 1.51}$$

6. Use R to draw a random sample from your distribution (with replacement) of 10,000 observations using the sample() function. Be sure to set a random seed using set.seed() and to save your resulting sample for later use by assigning it to a vector called **mysample**. You can sample directly from the original data or you can specify the details by hand using parameters.

Here is some code that creates the sample (there are several ways to achieve the same result):

```
set.seed(135246)
mysample <- sample(x=c(1,2,3,4,5,6,7,8), size=10000,
      replace=TRUE, prob=mydist)
```

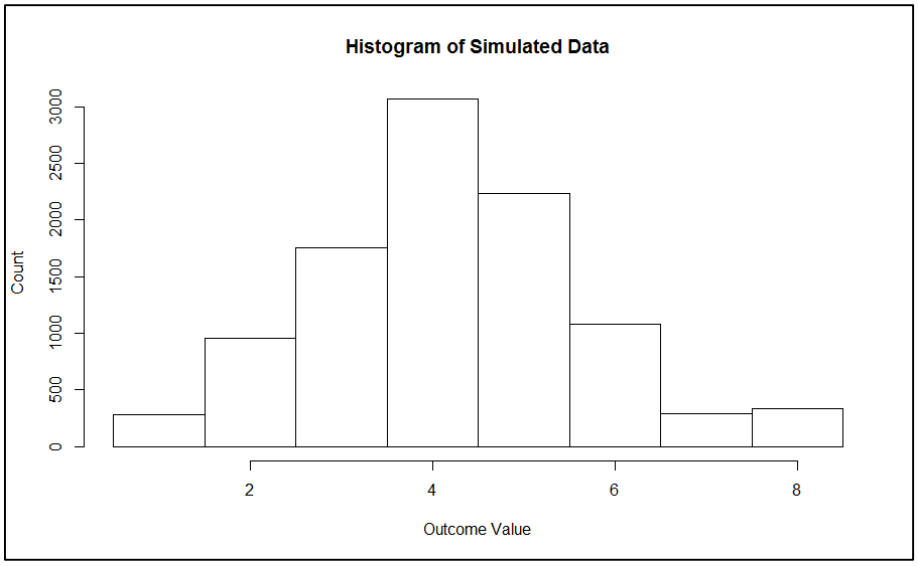
It is worth noting that the sample() function expects prob to be a vector of weights. The output in my code above is a table of weights, but R is able to coerce it into a vector. Not a big deal, but worth mentioning. This might not work for more complicated table results.

7. Plot a histogram of your simulated data using the same technique used in step 3 above. Does it look similar to the histogram from the original data?

Here is the code:

```
hist(mysample,
      breaks=c(0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5),
      xlab="Outcome Value", ylab="Count",
      main="Histogram of Simulated Data")
```

The histogram output is given at the top of the next page.



8. Using the sample you drew in step 6 (mysample), calculate the simulated mean using the mean() function. How close is it to the theoretical value?

   The following code calculates the mean and prints it to the console:

   ```
   mean(mysample, na.rm=TRUE)
   ```

   The outputted mean value is 4.2055, very close to the theoretical value of 4.215.

   (Note that the mean() function won't work on data with missing values, so I have included the parameter na.rm=TRUE in order to remove any missing values before computing the mean. In our lab data there are no missing values. This is just for illustrative purposes!)

9. Using the sample you drew in step 6 (mysample), calculate the simulated standard deviation using the sd() function. How close is it to the theoretical value?

   The following code calculates the sample standard deviation and prints it to the console:

   ```
   sd(mysample, na.rm=TRUE)
   ```

   The outputted mean value is 1.4896, very close to the theoretical value of 1.51.

   (Note that the mean() function won't work on data with missing values, so I have included the parameter na.rm=TRUE in order to remove any missing values before computing the mean. In our lab data there are no missing values. This is just for illustrative purposes!)

10. Construct the cumulative distribution function from the probability distribution.

    The cumulative distribution is also a table of probabilities, but gives the probability that X takes on a value less than or equal to the given outcome. Thus, we add all probabilities for these outcomes. The result is as follows (with the last row giving the cumulative distribution):

Value	1	2	3	4	5	6	7	8
P(X = k)	0.030	0.095	0.175	0.305	0.220	0.110	0.030	0.035
P(X ≤ k)	0.030	0.125	0.300	0.605	0.825	0.935	0.965	1.000

11. Obtain the five-number summary (Min-Q1-Med-Q3-Max) using your cumulative distribution function constructed from the original data.

    The minimum is clearly 1. The 25th percentile occurs, according to the cumulative distribution, at the value where the cumulative probability (which is a percentile) is greater than or equal to 0.25 but where the previous value has cumulative probability less than 0.25. Thus, the 25th percentile, or Q1, is 3. The median (50th percentile) is 4 by the same logic. Q3 is 5. Finally, the maximum is 8. Presented visually, we have a five-number summary of:

    1 – 3 – 4 – 5 – 8

12. Obtain the same five-number summary from the original raw data using R and the quantile() function. (You can specify the percentiles you want using the probs parameter. To get the five-number summary, you can give probs=c(0,0.25,0.5,0.75,1). Note that you can also get this information with the summary() function, but the quantile() function is more general and you can get other percentiles as well.)

    Here is the code that generates the five-number summary using the quantile() function:

```
quantile(mydata$Outcome, probs=c(0,0.25,0.5,0.75,1))
```

    Here is the output on the command line:

```
  0%  25%  50%  75% 100%
   1    3    4    5    8
```

    Note that we can also get this information using the summary() function, but quantile is more general since you can specify any percentiles you want. Here is the code for using summary():

    summary(mydata$Outcome)

    Here is the command line output:

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   3.000   4.000   4.215   5.000   8.000
```

    Note that the mean is also given.

Summary

The exercise above is a very basic summarization approach when working with discrete probability distributions. These sorts of steps are crucial to understanding the structure and content of data that might be used in modeling applications. Generally, each variable of a data set should be examined carefully for these sorts of properties. In an applications lab, we'll look at how we can use a discrete distribution combined with other information to make important decisions.