ANLY 500 Laboratory #1 (part 2) – Descriptive Statistics

Evans Chapter 3

“Performance Lawn Equipment Case Study” from Evans, **Business Analytics**

Contents

[Introduction 1](#_Toc492550678)

[Chapter 3 2](#_Toc492550679)

[Part 1 2](#_Toc492550680)

[Step 1 2](#_Toc492550681)

[Step 2 7](#_Toc492550682)

[Part 2 7](#_Toc492550683)

[Step 1 7](#_Toc492550684)

[Part 3 8](#_Toc492550685)

[Step 1 8](#_Toc492550686)

[Step 2 11](#_Toc492550687)

[Part 4 11](#_Toc492550688)

# Introduction

In this part of the lab, we are working on some data visualization techniques using R.

# Chapter 3

## Part 1

You have been tasked with putting together an overview of PLE’s business performance and market position. You have specifically been asked to construct appropriate charts and summarize your conclusions for:

1. Dealer Satisfaction
2. End-User Satisfaction
3. Complaints
4. Mower Unit Sales
5. Tractor Unit Sales
6. On-Time Delivery
7. Defects after Delivery
8. Response Time

### Step 1

To begin this we’ll again need to put together subsets of the data, e.g. Dealer Satisfaction for North America, and so on. Once we are through subsetting the data we can create the plots that describe what is going on performance wise. But first, let’s look at the data:

> str(DealerSatisfaction)

'data.frame': 23 obs. of 9 variables:

$ Region: Factor w/ 4 levels "CH","EU","PA",..: NA NA NA NA NA 4 4 4 4 4 ...

$ Year : int 2010 2011 2012 2013 2014 2010 2011 2012 2013 2014 ...

$ L0 : int 1 0 1 1 2 0 0 0 0 1 ...

$ L1 : int 0 0 1 2 3 0 0 0 1 1 ...

$ L2 : int 2 2 1 6 5 0 0 1 1 2 ...

$ L3 : int 14 14 8 12 15 2 2 4 3 4 ...

$ L4 : int 22 20 34 34 44 6 6 11 12 22 ...

$ L5 : int 11 14 15 45 56 2 2 14 33 60 ...

$ Count : int 50 50 60 100 125 10 10 30 50 90 ...

> str(EndUserSatisfaction)

'data.frame': 23 obs. of 9 variables:

$ Region: Factor w/ 4 levels "CH","EU","PA",..: NA NA NA NA NA 4 4 4 4 4 ...

$ Year : int 2010 2011 2012 2013 2014 2010 2011 2012 2013 2014 ...

$ L0 : int 1 1 1 0 0 1 1 0 0 0 ...

$ L1 : int 3 2 2 2 2 2 3 2 2 2 ...

$ L2 : int 6 4 5 4 3 5 6 6 5 5 ...

$ L3 : int 15 18 17 15 15 18 17 19 20 19 ...

$ L4 : int 37 35 34 33 31 36 36 37 37 37 ...

$ L5 : int 38 40 41 46 49 38 37 36 36 37 ...

$ Count : int 100 100 100 100 100 100 100 100 100 100 ...

There are 23 observations of 9 variables in each data file. There are six levels of satisfaction that have been recorded, which is a bit odd. Usually an odd number of levels of satisfaction are used for a Likert Scale. We could also do some descriptive statistics using the summary() function but since the years and regions would be aggregated I’m not sure that would reveal much.

First, let’s subset the data by region to create data tables. For example, we can start with the first region, North America or NA, and print out the return as:

> dealerSat\_NA <- DealerSatisfaction[1:5, ]

> dealerSat\_NA

Region Year L0 L1 L2 L3 L4 L5 Count

1 <NA> 2010 1 0 2 14 22 11 50

2 <NA> 2011 0 0 2 14 20 14 50

3 <NA> 2012 1 1 1 8 34 15 60

4 <NA> 2013 1 2 6 12 34 45 100

5 <NA> 2014 2 3 5 15 44 56 125

In case you had been wondering the expression “NA” is a standard expression in R and most programming environments or languages. NA typically denotes a missing value. So, R has automatically put brackets around NA in our data files even though NA for us means North America. You can set up objects this same way for South America (SA), Europe (EU), Pacific Rim (PA), and China (CH). Now, for the plotting.

Again, there will be many ways to create the required plots, e.g. the lattice package or ggplot2. We’ll use ggplot2 for this example. You’ll need to install the ggplot2 and labeling packages and attach them using library(ggplot2) and library(labeling) commands. If you have any trouble installing packages or attaching them using the library() function get in touch with me as soon as possible. This should not prevent you from completing your assignments.

There is plenty of information online about the ggplot2 package and the ggplot() function. Actually, there is too much information to go into any real detail in this document. The entire series of commands and the respective explanations I’ll use is:

First take the transpose of the desired columns of the original data table to get the data in the proper sequence for the melt command. The melt command is required to create the plot correctly.

> tdealerSat\_NA <- t(dealerSat\_NA[,3:8])

> tdealerSat\_NA

1 2 3 4 5

L0 1 0 1 1 2

L1 0 0 1 2 3

L2 2 2 1 6 5

L3 14 14 8 12 15

L4 22 20 34 34 44

L5 11 14 15 45 56

Next, add the years as column names to get these in the proper sequence in the melded data.

> colnames(tdealerSat\_NA) <- c("2010", "2011", "2012", "2013", "2014")

> tdealerSat\_NA

2010 2011 2012 2013 2014

L0 1 0 1 1 2

L1 0 0 1 2 3

L2 2 2 1 6 5

L3 14 14 8 12 15

L4 22 20 34 34 44

L5 11 14 15 45 56

Use the melt() function to “melt” the data, i.e. put it in the proper sequence for plotting using ggplot(). The melt() function is part of the reshape2 package which you may need to install. Notice that all the levels and counts are sorted by each year. You can check this against the original data to make sure you’ve got the proper sequencing.

> data.m2 <- melt(tdealerSat\_NA, id.vars=var1)

> data.m2

Var1 Var2 value

1 L0 2010 1

2 L1 2010 0

3 L2 2010 2

4 L3 2010 14

5 L4 2010 22

6 L5 2010 11

7 L0 2011 0

8 L1 2011 0

9 L2 2011 2

10 L3 2011 14

11 L4 2011 20

12 L5 2011 14

13 L0 2012 1

14 L1 2012 1

15 L2 2012 1

16 L3 2012 8

17 L4 2012 34

18 L5 2012 15

19 L0 2013 1

20 L1 2013 2

21 L2 2013 6

22 L3 2013 12

23 L4 2013 34

24 L5 2013 45

25 L0 2014 2

26 L1 2014 3

27 L2 2014 5

28 L3 2014 15

29 L4 2014 44

30 L5 2014 56

Add column names to the melded data in order to get the proper axis and legend labels in the plot.

> colnames(data.m2) <- c("Level", "Year", "Counts")

> data.m2

Level Year Counts

1 L0 2010 1

2 L1 2010 0

3 L2 2010 2

4 L3 2010 14

5 L4 2010 22

6 L5 2010 11

7 L0 2011 0

8 L1 2011 0

9 L2 2011 2

10 L3 2011 14

11 L4 2011 20

12 L5 2011 14

13 L0 2012 1

14 L1 2012 1

15 L2 2012 1

16 L3 2012 8

17 L4 2012 34

18 L5 2012 15

19 L0 2013 1

20 L1 2013 2

21 L2 2013 6

22 L3 2013 12

23 L4 2013 34

24 L5 2013 45

25 L0 2014 2

26 L1 2014 3

27 L2 2014 5

28 L3 2014 15

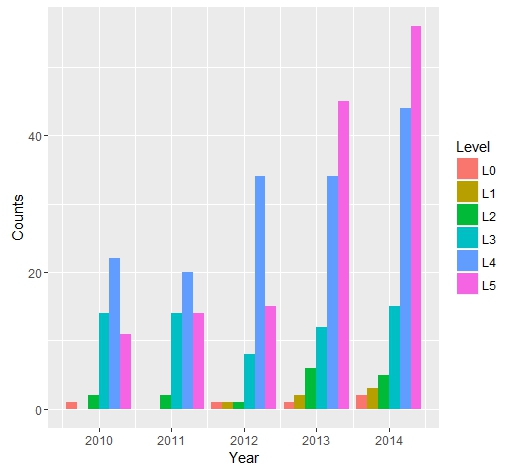
29 L4 2014 44

30 L5 2014 56

Create the plot with the ggplot() function. You can look this up in help() in RStudio or online.

> ggplot(data.m2, aes(x=Year, y=Counts)) + geom\_bar(aes(fill=Level), position="dodge", stat="identity")

The plot this produces is:

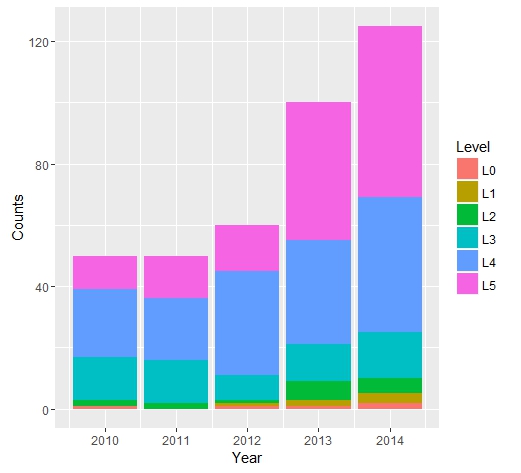


where the number of counts for a particular level of satisfaction for the given year is shown.

Once you have your data in the proper sequence for plotting everything else is easy. To create the stacked bar charts just use:

> ggplot(data.m2, aes(x=Year, y=Counts, fill=Level)) + geom\_bar(stat="identity")

The resulting plot is:



You can follow this same procedure for each Region for Dealer Satisfaction and End-User Satisfaction.

### Step 2

There are also many ways in R to do the line plots, such as for Complaints. I’ll show you an example of a “brute force” method using the simple plot() function as follows:

> plot(Complaints$World, ylim=range(c(0,400)), type="l", xlab="Month", ylab="Number of Complaints")

> par(new=TRUE)

> plot(Complaints$NA., ylim=range(c(0,400)), type="l", col="red", axes = FALSE, xlab = "", ylab = "")

> par(new=TRUE)

> plot(Complaints$SA, ylim=range(c(0,400)), type="l", col="green", axes = FALSE, xlab = "", ylab = "")

> par(new=TRUE)

> plot(Complaints$Eur, ylim=range(c(0,400)), type="l", col="blue", axes = FALSE, xlab = "", ylab = "")

> par(new=TRUE)

> plot(Complaints$Pac, ylim=range(c(0,400)), type="l", col="magenta", axes = FALSE, xlab = "", ylab = "")

> par(new=TRUE)

> plot(Complaints$China, ylim=range(c(0,400)), type="l", col="deeppink4", axes = FALSE, xlab = "", ylab = "")

In this case the axes labels are entered in the first command. The command “par(new=TRUE)” follows each line added to the plot so that you can continue to add lines to the same plot. After the first line the parameters “axes=FALSE, xlab=””, ylab=””” are added so that the axes labels are not continuously overwritten. You can use this or any other plotting methods, e.g. ggplot(), to create the line plots to finish Part 1 of Chapter 3.

## Part 2

For this part of the exercise you are tasked with comparing the costs of shipping between existing locations and proposed locations using quartiles. If you have questions about quartiles the textbook can help or there is a lot of information online.

### Step 1

We can lump all the costs for existing plants into one group and all the costs for proposed plants into a second group. Then compute the quartiles for shipping costs based on those groups.

This is actually quite simple in R. Just use the summary() function that we’ve been using to look at our data as follows:

> summary(ShippingCost\_Existing$Mowers)

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.000 1.312 1.480 1.420 1.528 1.720

> summary(ShippingCost\_Existing$Tractors)

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.260 1.768 1.840 1.879 2.105 2.340

> summary(ShippingCost\_Proposed$Mowers)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.910 1.400 1.520 1.514 1.660 1.980

> summary(ShippingCost\_Proposed$Tractors)

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.170 1.775 2.010 1.958 2.170 2.680

## Part 3

In the third part of this exercise you’ve been tasked with developing a summary about customer attributes. This summary is to be built on the average responses from customers in the 2014 Customer Survey. It should be done by region and include frequency distributions, histograms and quartiles as appropriate. The attributes in the survey include: Quality, Ease of Use, Price and Service.

### Step 1

The only new function we’ll use to complete this part of the exercise is the hist() function to create the required histograms. But first we’ll need to subset the data in order to get it in the proper sequence to calculate the averages and frequency distributions.

First, look at the data as usual:

> str(CustomerSurvey2014)

'data.frame': 200 obs. of 5 variables:

$ Region : Factor w/ 4 levels "China","Eur",..: NA NA NA NA NA NA NA NA NA NA ...

$ Quality : int 4 4 4 5 5 5 5 5 4 4 ...

$ Ease.of.Use: int 1 4 5 4 4 5 4 5 4 5 ...

$ Price : int 3 4 4 4 5 3 4 4 4 4 ...

$ Service : int 4 5 3 4 4 5 2 5 5 5 ...

> summary(CustomerSurvey2014)

Region Quality Ease.of.Use

China: 10 Min. :1.000 Min. :1.000

Eur : 30 1st Qu.:4.000 1st Qu.:4.000

Pac : 10 Median :5.000 Median :4.000

SA : 50 Mean :4.395 Mean :4.165

NA's :100 3rd Qu.:5.000 3rd Qu.:5.000

Max. :5.000 Max. :5.000

Price Service

Min. :1.00 Min. :1.00

1st Qu.:3.00 1st Qu.:4.00

Median :4.00 Median :4.00

Mean :3.67 Mean :4.14

3rd Qu.:4.00 3rd Qu.:5.00

Max. :5.00 Max. :5.00

This has already given us the frequency distribution by region. That is, 100 responses or 50% come from North America and so on. But, we have a problem because the data file codes “North America” as “NA” which is a standard phrase in R for a missing value. This makes the straightforward application of functions a mess. There is a compounding problem in that the variable concerned, “Region”, is a factor variable. So…

If you have already imported the data file CustomerSurvey2014.csv into RStudio you will want to remove it using:

> rm(CustomerSurvey2014)

In order to get the “NA” in this file to just be NA you need to have the parameter “stringsAsFactors” equal FALSE. In order to do that, if you are using the pull down menu to import data you need to uncheck the box “Strings as Factors” before you import the data. If you are using the command line use:

> CustomerSurvey2014 <- read.csv("~/MyRWork/data/Evans/CustomerSurvey2014.csv", stringsAsFactors=FALSE)

Unfortunately, this is not all you need to do. R/RStudio will still recognize the text NA as representing missing values but now it is a text string and not the <NA> symbol R/RStudio uses. This means that you can replace “NA” with another abbreviation for North America that will not cause problems with the functions we are trying to execute. For example, if you want to replace “NA” with “NorthA” you can use this and look at what it returns as:

> CustomerSurvey2014[is.na(CustomerSurvey2014)] <- "NorthA"

> str(CustomerSurvey2014)

'data.frame': 200 obs. of 5 variables:

$ Region : chr "NorthA" "NorthA" "NorthA" "NorthA" ...

$ Quality : int 4 4 4 5 5 5 5 5 4 4 ...

$ Ease.of.Use: int 1 4 5 4 4 5 4 5 4 5 ...

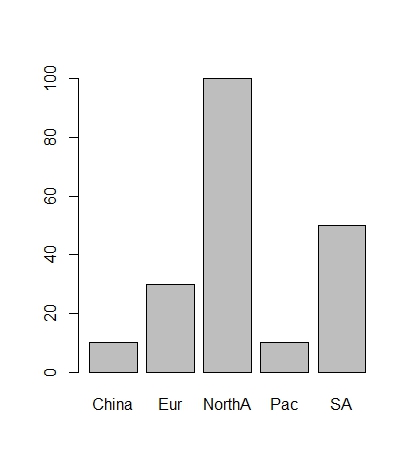
$ Price : int 3 4 4 4 5 3 4 4 4 4 ...

$ Service : int 4 5 3 4 4 5 2 5 5 5 ...

Now, we can easily create a histogram, or because the variable is a factor variable the bar chart, as follows:

> barplot(table(CustomerSurvey2014$Region))

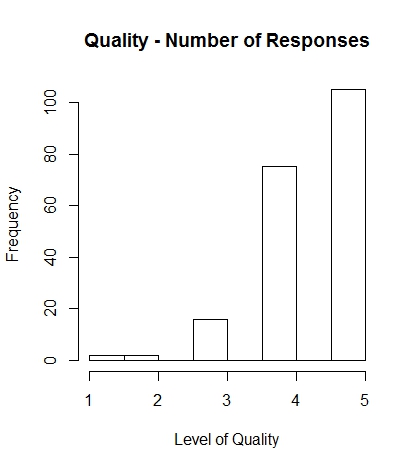
Which produces the plot below.



Because the remaining variables in the CustomerSurvey2014 data file are numeric variables we can use the hist() function as follows:

> hist(CustomerSurvey2014$Quality, main="Quality - Number of Responses", xlab="Level of Quality")

Which produces the following plot.



There are many additional things you can do to make plots in R/RStudio look very professional. I encourage you to explore all the options using the R/RStudio documentation and available information online, e.g. at <https://www.datacamp.com/community/tutorials/15-questions-about-r-plots#gs.kBDIbjY>

### Step 2

The remaining portion of this part of the exercise asks you to compute the Quartiles. You’ve done that before so it isn’t necessary to repeat that here.

## Part 4

You are tasks with proposing a dashboard of the most important business information needed on a routine basis. You are free to complete this part of the exercise as you think best.