STAT 1300: Introduction to Data Science

Andrew Kerr

2/15/23

Table of contents

Introduction

This is a work-in-progress for the book for STAT 1300: Introduction to Data Science offered through the Mathematics Department at Columbus State Community College.

STAT 1300 serves as a quantitative-reasoning based course for data science. While data science is a deep and complicated field, a basic familiarity can go a long way in terms of understanding and using results from data science to inform decision making.

This course is a non-calculus based data analysis course. This course will make heavy use of the R programming language and will utilize R Studio Online through Posit.

In this book, you will cover the basic functions of the R programming language while learning various aspects of data analysis. While there are instances of performing calculations, the vast majority of the work will be done in R.

---

1 preface_1ed

Welcome to the first edition of the textbook for STAT 1300! This is a new class, so there are likely to be several changes and updates over the course of the first few iterations of the class.

Below is a summary of the items we will cover in the book:

- Understand data structures (observations, variables), as well as types of variables (quantitative, qualitative).
- Read in and summarize data in different formats (.csv, .xlsx, from a website)
- Create graphical summaries of variables
- Create and explain the properties of a linear regression model
- Understand and run classification, both supervised and unsupervised, for a simple data set
- Discuss ethical issues surrounding classification methods

2 Introduction

Simple introduction to data and data science.

Part I

collect-summarize-maintain

The fundamental basis of this course is data. Data can come in different forms: raw numbers, results from a survey, and video recordings, for example. By the end of this section, you should be able to:

- Identify the variables and individuals in a data set
- Work with R's internal data sets using simple commands in R
- Read in data from a file
- Read in data from a website
- Identify the types of variables in a data set
- Summarize a data set via measures of center and measures of spread

3 data-structure

A data set is a collection of observations. These observations consist of one or more variables, or quantities that can change from observation to observation.

R has several data structures for storing data. We will focus on the data.frame type.

It's helpful to begin with one of the many data sets that are included with R. In particular, the iris data set is a great data set to begin with.

The iris data set consists of 150 observations with 5 variables. By typing iris in the command line, we will see the entrie data set.

For brevity, we can use the head() function to look at the first few rows of the data set:

```
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

The first four variables represent the length and width measurements of each iris flower's sepal and petal, and the last variable lists the species of the iris flower.

3.1 Extracting Values from a Data Frame

Sometimes we may just care about certain aspects of the data set. There are a few ways to get a specific row or a specfic column of the data set (or even a specific value!)

To access specific rows or columns, we can use square brackets after the name of the data frame.

```
iris["row number","column number"]
```

So, if we wanted to pick out the value in the third row and the fourth column, we would use:

```
iris[3,4]
```

```
[1] 0.2
```

Note that this gives us the third value in the Petal.Width column (the fourth column)

If we want to get an entire row or an entire column, we would leave that corresponding portion blank. For the entire fifth row of the iris data set, we have:

```
iris[5,]
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
5            5         3.6          1.4         0.2  setosa
```

For the entire second column, we can use:

```
iris[,2]
```

```
  [1] 3.5 3.0 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 3.7 3.4 3.0 3.0 4.0 4.4 3.9 3.5
 [19] 3.8 3.8 3.4 3.7 3.6 3.3 3.4 3.0 3.4 3.5 3.4 3.2 3.1 3.4 4.1 4.2 3.1 3.2
 [37] 3.5 3.6 3.0 3.4 3.5 2.3 3.2 3.5 3.8 3.0 3.8 3.2 3.7 3.3 3.2 3.2 3.1 2.3
 [55] 2.8 2.8 3.3 2.4 2.9 2.7 2.0 3.0 2.2 2.9 2.9 3.1 3.0 2.7 2.2 2.5 3.2 2.8
 [73] 2.5 2.8 2.9 3.0 2.8 3.0 2.9 2.6 2.4 2.4 2.7 2.7 3.0 3.4 3.1 2.3 3.0 2.5
 [91] 2.6 3.0 2.6 2.3 2.7 3.0 2.9 2.9 2.5 2.8 3.3 2.7 3.0 2.9 3.0 3.0 2.5 2.9
[109] 2.5 3.6 3.2 2.7 3.0 2.5 2.8 3.2 3.0 3.8 2.6 2.2 3.2 2.8 2.8 2.7 3.3 3.2
[127] 2.8 3.0 2.8 3.0 2.8 3.8 2.8 2.8 2.6 3.0 3.4 3.1 3.0 3.1 3.1 3.1 2.7 3.2
[145] 3.3 3.0 2.5 3.0 3.4 3.0
```

We can also access columns using the dollar sign operator and then using the column name:

```
iris$Sepal.Width
```

```
  [1] 3.5 3.0 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 3.7 3.4 3.0 3.0 4.0 4.4 3.9 3.5
 [19] 3.8 3.8 3.4 3.7 3.6 3.3 3.4 3.0 3.4 3.5 3.4 3.2 3.1 3.4 4.1 4.2 3.1 3.2
 [37] 3.5 3.6 3.0 3.4 3.5 2.3 3.2 3.5 3.8 3.0 3.8 3.2 3.7 3.3 3.2 3.2 3.1 2.3
```

```
 [55] 2.8 2.8 3.3 2.4 2.9 2.7 2.0 3.0 2.2 2.9 2.9 3.1 3.0 2.7 2.2 2.5 3.2 2.8
 [73] 2.5 2.8 2.9 3.0 2.8 3.0 2.9 2.6 2.4 2.4 2.7 2.7 3.0 3.4 3.1 2.3 3.0 2.5
 [91] 2.6 3.0 2.6 2.3 2.7 3.0 2.9 2.9 2.5 2.8 3.3 2.7 3.0 2.9 3.0 3.0 2.5 2.9
[109] 2.5 3.6 3.2 2.7 3.0 2.5 2.8 3.2 3.0 3.8 2.6 2.2 3.2 2.8 2.8 2.7 3.3 3.2
[127] 2.8 3.0 2.8 3.0 2.8 3.8 2.8 2.8 2.6 3.0 3.4 3.1 3.0 3.1 3.1 3.1 2.7 3.2
[145] 3.3 3.0 2.5 3.0 3.4 3.0
```

3.2 Exercises

1. Consider using the mtcars data set in R. What are two different ways that we could get just the values in the wt (weight) variable?
2. Using the mtcars data set in R, how could we find the 18th value in the mpg variable?
3. How could I get just the first 3 rows of this data set?

4 variable-type

In the previous section, we looked at general data structures in R, specifically using a data.frame, as well as how to extract a certain row or column of a data set.

In this section, we will look closer at the variables: specifically, the different *types* of variables we can encounter.

In general, there are two major classes of variables: quantitative variables and qualitative variables.

Quantitative Variables are variables that are numerical, and for which simple arithmetic operations, such as addition and multiplication, make sense. Some examples of quantitative variables include the sepal and petal measurements in the iris data set, as well as all of the variables in the mtcars data set.

Qualitative Variables, also sometimes called categorical variables, describe a quality or category of the observation. Typically these will be text or words. Some examples of qualitative variables include the species variable in the iris data set, as well as the group and ID variables in the sleep data set.

It should be noted that occasionally qualitative variables will use numbers. This is often for convenience. For example, instead of using the species of setosa, veriscolor, and virginica, we could have just labeled them as species 1, 2, and 3.

R gets far more granualr in its classification. Rather than just using quantitative and qualitative as labels, there are further break downs we can use.

Quantitative	Qualitative
numeric	logical
raw	character
integer	factor
complex	

4.1 How to Determine the Variable Type

It may be difficult to know immediately what type each variable fits into. There are a few ways to do this.

The first is to use the class() function. If we specify a column in a data set, we can use the class() function to learn how R is storing the values.

For example, let's look at the class of the Petal.Length and Species variables in the iris data set:

```
class(iris$Petal.Length)
```

```
[1] "numeric"
```

```
class(iris[,5])
```

```
[1] "factor"
```

We see that we get numeric for the Petal.Length variable and we get factor for the Species variable. This means that Petal.Length is a quantiative variable and Species is a qualitative variable.

If we have an entire data frame, we can use the str() function to find the *structure* of the data set. This will list all of the variables (column) along with their types, as well as the first few observations.

If we look at the sleep data set, we have:

```
str(sleep)
```

```
'data.frame':   20 obs. of  3 variables:
 $ extra: num  0.7 -1.6 -0.2 -1.2 -0.1 3.4 3.7 0.8 0 2 ...
 $ group: Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
 $ ID   : Factor w/ 10 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
```

The variable extra is a numeric variable (quantitative), and the variables group and ID are factors (qualitative).

4.2 Examples

1. List the variables of the ToothGrowth data set as well as whether each variable is quantitative or qualitative.
2. List the variables of the USArrests data set as well as whether each variable is quantitative or qualitative.
3. List the variables of the mtcars data set as well as whether each variable is quantitative or qualitative.
4. List the variables of the ChickWeight data set as well as whether each variable is quantitative or qualitative.

5 summary-stats

Now that we have an idea of how to look at different variables and classify as quantiative or qualitative variables, we can look at summaries for both types of variable.

5.1 Summarizing Quantitative Variables

Summary statistics provide snapshots of a variable by providing values that give information about a variable.

5.1.1 The Sample Mean

The *sample mean* of a variable is a measure of center; that is, it gives a value that represents a typical value of the data set. It is calculated by adding up all of the values in a data set divided by the number of values in that sum.

In R, we use the `mean()` function.

Suppose we have a variable defined below:

```
x = c(85, 71, 68, 89, 92)
```

We could find the mean by hand:

$$\frac{85 + 71 + 68 + 89 + 92}{5} = \frac{405}{5} = 81$$

We can also use the `mean()` function:

```
mean(x)
```

```
[1] 81
```

If we want to find the mean `Sepal.Length` measurement in the `iris` data set:

```r
mean(iris$Sepal.Length)
```

```
[1] 5.843333
```

5.1.2 The Sample Median

The *sample median* is another measure of center, but rather than using the raw values, the median considers which value is in the middle position when the data are sorted in ascending order.

The calculation for the median differs depending on the number of observations in the calculation.

1. If there are an even number of observations, then the median is the sample mean of the values in the $\frac{n}{2}$ and $\frac{n}{2} + 1$ positions of the sorted data set.
2. If there is an odd number of observations, then the median is the value in the $\frac{n+1}{2}$ position of the sorted data set.

Consider the two following data sets:

```r
x = c(85, 71, 68, 89, 92)
y = c(103, 110, 93, 81, 109, 100)
```

To calculate the median for x, we first need to sort the data:

68, 71, 85, 89, 92

Now, since there is an odd number of values ($n = 5$), we take the value in position 3, or 85.

So, we have the median of x to be 85.

To calculate the median for y, we first sort the data:

81, 93, 100, 103, 109, 110

Now, since there is an even number of values ($n = 6$), we take the sample mean of the values in positions 3 and 4, or the values 100 and 103. So the median for y is:

$$\frac{100 + 103}{2} = \frac{203}{2} = 101.5$$

Note that the median does not necessarily need to be a value in the data set - it describes a position in the data set.

The median is the value such that *at least* half of the observations are less than or equal to the median, and *at least* half of the observations are greater than or equal to the median. Effecitvely, the median divides the data set into halves.

In R, we can just use the `median()` fnction:

```
median(x)
```

```
[1] 85
```

```
median(y)
```

```
[1] 101.5
```

Once nice aspect here is that the R function takes care of the sorting.

5.1.3 The Sample Quartiles (Q_1 and Q_3)

We just saw how the median can split a data set into halves. On occasion, we will want to chop up the data set into more pieces. Eventually, we will discuss the idea of a *percentile*, but for now we will focus on quartiles.

The *first quartile*, often denoted as Q_1, is the value that divides the bottom 25% of the data set from the upper 75% of the data set. Mechanically, to find Q_1, we find the median of the data that are *less than or equal to the median*.

The *third quartile*, often denoted as Q_3, is the value that divides the bottom 75% of the data set from the upper 25% of the data set. Mechanically, to find Q_3, we find the median of the data that are *greater than or equal to the median*.

To find quartiles, we will use the `quantile` function (notice the difference in spelling!). The syntax is as follows:

`quantile(data,probs)`

Where `data` is your data set and `probs` is a collection of the corresponding percentile values.

6 outliers

7 reading-r-data

Part II

data-visulaization

8 univariate-bar-chart

9 univariate-histogram

10 univariate-boxplot

11 multiple-graphs

12 scatterplot

13 dynamic-graphs

14 misuses-visualozation

15 metric-represent

Part III

statistical-models

16 linear-model-intro

17 linear-assumptions

18 linear-model-part1

19 linear-model-part2

20 simulation-linear

Part IV

statistical-learning

21 learning-intro

22 supervised-learning

23 lda

24 qda

25 cart-class

26 random-forests

27 unsupervised-learning

28 knn

29 h-clustering

Part V

add-topics-learning

30 proper-use

31 unfair-treatment

32 ethics

References