Data Summaries and dplyr

Nate Wells

Math 141, 2/4/22

Outline

In this lecture, we will. . .

Outline

In this lecture, we will. . .

- Discuss measurements of center and spread for quantitative data
- Use contingency tables to investigate relationships among categorical variables
- Use the `summarize` function in the `dplyr` package to compute summary statistics

Section 1

Data Summaries

Exam Statistics

Suppose you are an instructor trying to gauge class performance on an exam. You have exam scores for 200 intro stat students.

Exam Statistics

Suppose you are an instructor trying to gauge class performance on an exam. You have exam scores for 200 intro stat students.

What summarizing information would it be helpful to know in order to assess how well the class did?

Exam Statistics

Suppose you are an instructor trying to gauge class performance on an exam. You have exam scores for 200 intro stat students.

What summarizing information would it be helpful to know in order to assess how well the class did?

1. What was the typical value (maybe average or median)?
2. How much variation was there in scores?
3. What was the shape of the data?
4. Were there any outliers?

The Mean

The **mean** or average of a data set is one measure of *center*, obtained by adding all observed values and dividing by their number:

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

where n is the number of observations and x_i is the value of the ith observation.

The Mean

The **mean** or average of a data set is one measure of *center*, obtained by adding all observed values and dividing by their number:

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

where n is the number of observations and x_i is the value of the ith observation.

```
mean(biketown_short$Distance_Miles)
```

```
## [1] 1.677599
```

The Mean

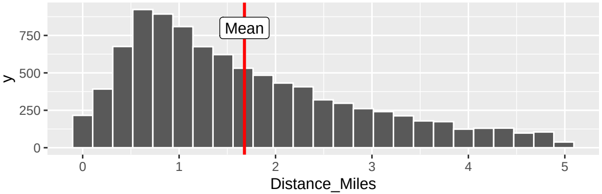
The **mean** or average of a data set is one measure of *center*, obtained by adding all observed values and dividing by their number:

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

where n is the number of observations and x_i is the value of the ith observation.

```
mean(biketown_short$Distance_Miles)
```

```
## [1] 1.677599
```



- If the histogram were made of solid material, the mean would be the point along the horizontal axis where the solid is perfectly balanced.

The Median

The **median** is another measure of *center* and separates data into two equally sized sets.

The Median

The **median** is another measure of *center* and separates data into two equally sized sets.

Suppose the n values are ordered from least to greatest. The median is the value in the middle of the list.

- If n is even, then there are two middle values, and the median is their average.

The Median

The **median** is another measure of *center* and separates data into two equally sized sets.

Suppose the n values are ordered from least to greatest. The median is the value in the middle of the list.

- If n is even, then there are two middle values, and the median is their average.

```
median(biketown_short$Distance_Miles)
```

```
## [1] 1.39
```

The Median

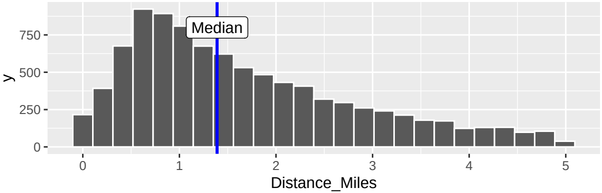
The **median** is another measure of *center* and separates data into two equally sized sets.

Suppose the n values are ordered from least to greatest. The median is the value in the middle of the list.

- If n is even, then there are two middle values, and the median is their average.

```
median(biketown_short$Distance_Miles)
```

```
## [1] 1.39
```



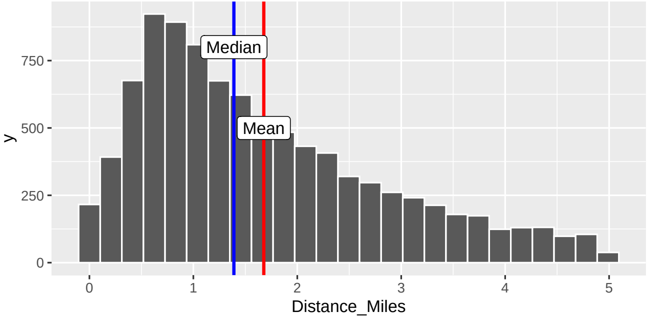
- The median corresponds to the line that divides a histogram into two equal area pieces.

Mean, Median, and Skew

Both mean and median represent *typical* values for a data set.

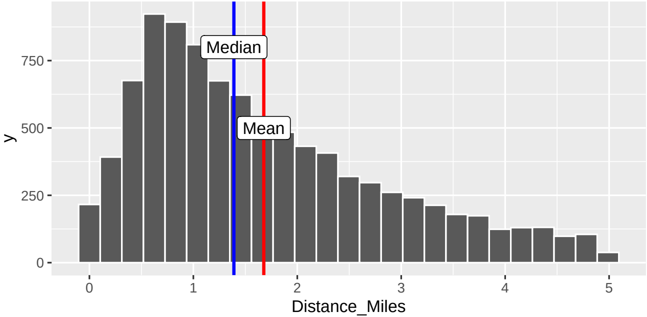
Mean, Median, and Skew

Both mean and median represent *typical* values for a data set.



Mean, Median, and Skew

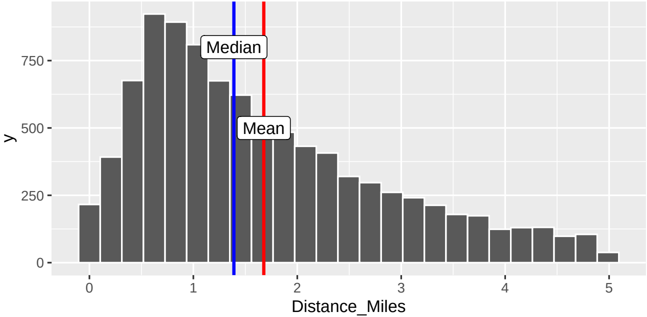
Both mean and median represent *typical* values for a data set.



- In non-symmetric distributions, the mean will further along the direction of skew than the median.

Mean, Median, and Skew

Both mean and median represent *typical* values for a data set.



- In non-symmetric distributions, the mean will further along the direction of skew than the median.
  - Why?

Robustness

Consider two data sets, one with a large outlier and one without:

```
my_data <- c(1, 2, 5, 7, 8, 10)
my_data_with_oulier <- c(1, 2, 5, 7, 8, 100)
```

Robustness

Consider two data sets, one with a large outlier and one without:

```
my_data <- c(1, 2, 5, 7, 8, 10)
my_data_with_oulier <- c(1, 2, 5, 7, 8, 100)
```

The mean value of a dataset is very sensitive to outliers.

Robustness

Consider two data sets, one with a large outlier and one without:

```
my_data <- c(1, 2, 5, 7, 8, 10)
my_data_with_oulier <- c(1, 2, 5, 7, 8, 100)
```

The mean value of a dataset is very sensitive to outliers.

```
mean(my_data)
```

```
## [1] 5.5
```

```
mean(my_data_with_oulier)
```

```
## [1] 20.5
```

Robustness

Consider two data sets, one with a large outlier and one without:

```
my_data <- c(1, 2, 5, 7, 8, 10)
my_data_with_oulier <- c(1, 2, 5, 7, 8, 100)
```

The mean value of a dataset is very sensitive to outliers.

```
mean(my_data)
```

```
## [1] 5.5
```

```
mean(my_data_with_oulier)
```

```
## [1] 20.5
```

The median, however, is not.

```
median(my_data)
```

```
## [1] 6
```

```
median(my_data_with_oulier)
```

```
## [1] 6
```

Measures of Variability

We'd like to assess how variable the data set is.

- Are values usually close to the mean, or are they spread out?

Measures of Variability

We'd like to assess how variable the data set is.

- Are values usually close to the mean, or are they spread out?

How can we find the typical amount an observation differs from the mean observation?

Measures of Variability

We'd like to assess how variable the data set is.

- Are values usually close to the mean, or are they spread out?

How can we find the typical amount an observation differs from the mean observation?

Guess 1: Compute the average difference

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})$$

Distance_Miles	Mean	Deviations
1.57	1.2	0.37
2.09	1.2	0.89
0.38	1.2	-0.82
0.86	1.2	-0.34
1.10	1.2	-0.10

Measures of Variability

We'd like to assess how variable the data set is.

- Are values usually close to the mean, or are they spread out?

How can we find the typical amount an observation differs from the mean observation?

Guess 1: Compute the average difference

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})$$

Distance_Miles	Mean	Deviations
1.57	1.2	0.37
2.09	1.2	0.89
0.38	1.2	-0.82
0.86	1.2	-0.34
1.10	1.2	-0.10

- What's the problem?

Measures of Variability

We'd like to assess how variable the data set is.

- Are values usually close to the mean, or are they spread out?

How can we find the typical amount an observation differs from the mean observation?

Guess 1: Compute the average difference

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})$$

Distance_Miles	Mean	Deviations
1.57	1.2	0.37
2.09	1.2	0.89
0.38	1.2	-0.82
0.86	1.2	-0.34
1.10	1.2	-0.10

- What's the problem?

Avg_Deviations
0

Measures of Variability

The fix?

Measures of Variability

The fix?

Guess 2: Compute the average *squared* difference

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Distance_Miles	Mean	Sq_Deviation
1.57	1.2	0.1369
2.09	1.2	0.7921
0.38	1.2	0.6724
0.86	1.2	0.1156
1.10	1.2	0.0100

Measures of Variability

The fix?

Guess 2: Compute the average *squared* difference

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

- This is called the **Population Variance**

Distance_Miles	Mean	Sq_Deviation
1.57	1.2	0.1369
2.09	1.2	0.7921
0.38	1.2	0.6724
0.86	1.2	0.1156
1.10	1.2	0.0100

Measures of Variability

The fix?

Guess 2: Compute the average *squared* difference

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Distance_Miles	Mean	Sq_Deviation
1.57	1.2	0.1369
2.09	1.2	0.7921
0.38	1.2	0.6724
0.86	1.2	0.1156
1.10	1.2	0.0100

- This is called the **Population Variance**

Pop_Variance
0.3454

Standard Deviation

The population variance does measure spread of data.

$$\text{Population Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Standard Deviation

The population variance does measure spread of data.

$$\text{Population Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

But it does have two small problems:

Standard Deviation

The population variance does measure spread of data.

$$\text{Population Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

But it does have two small problems:

1. When sampling, it tends to *underestimate* the variability in the population. So we increase it by dividing by something smaller:

Standard Deviation

The population variance does measure spread of data.

$$\text{Population Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

But it does have two small problems:

1. When sampling, it tends to *underestimate* the variability in the population. So we increase it by dividing by something smaller:

$$\text{Sample Variance} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Standard Deviation

The population variance does measure spread of data.

$$\text{Population Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

But it does have two small problems:

1. When sampling, it tends to *underestimate* the variability in the population. So we increase it by dividing by something smaller:

$$\text{Sample Variance} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

2. Because observations are squared, it is no longer measured in same *units* as original data (i.e. if data is in miles, then variance is in sq. miles). So we take square roots:

Standard Deviation

The population variance does measure spread of data.

$$\text{Population Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

But it does have two small problems:

1. When sampling, it tends to *underestimate* the variability in the population. So we increase it by dividing by something smaller:

$$\text{Sample Variance} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

2. Because observations are squared, it is no longer measured in same *units* as original data (i.e. if data is in miles, then variance is in sq. miles). So we take square roots:

$$\text{Standard Deviation} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Visualizing Standard Deviation

The standard deviation measures the typical size of deviations of observations from the mean.

Visualizing Standard Deviation

The standard deviation measures the typical size of deviations of observations from the mean.

For most data sets, almost all observations are within a distance of 2 standard deviations of the mean:

Visualizing Standard Deviation

The standard deviation measures the typical size of deviations of observations from the mean.

For most data sets, almost all observations are within a distance of 2 standard deviations of the mean:

```
sd(biketown_short$Distance_Miles)
```

```
## [1] 1.172257
```

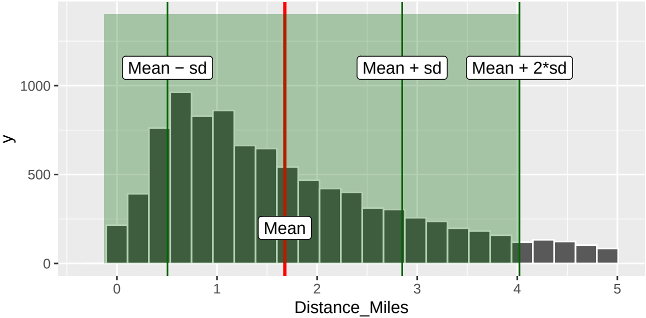
Visualizing Standard Deviation

The standard deviation measures the typical size of deviations of observations from the mean.

For most data sets, almost all observations are within a distance of 2 standard deviations of the mean:

```
sd(biketown_short$Distance_Miles)
```

```
## [1] 1.172257
```



Quartiles and IQR

Where the median divides data into equal halves, *quartiles* divide data into equal quarters

Quartiles and IQR

Where the median divides data into equal halves, *quartiles* divide data into equal quarters

- 25% of all observations are less than the *first quartile* $Q1$
- 25% of all observations are greater than the *third quartile* $Q3$

Quartiles and IQR

Where the median divides data into equal halves, *quartiles* divide data into equal quarters

- 25% of all observations are less than the *first quartile* $Q1$
- 25% of all observations are greater than the *third quartile* $Q3$

```
quantile(biketown_short$Distance_Miles, c(.25, .75))
```

```
##  25%  75%
## 0.75 2.38
```

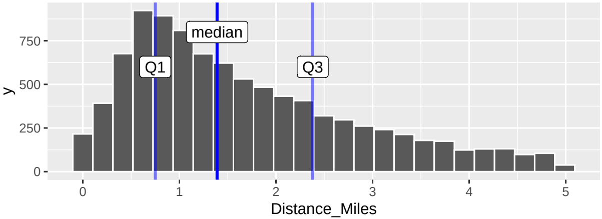
Quartiles and IQR

Where the median divides data into equal halves, *quartiles* divide data into equal quarters

- 25% of all observations are less than the *first quartile* $Q1$
- 25% of all observations are greater than the *third quartile* $Q3$

```
quantile(biketown_short$Distance_Miles, c(.25, .75))
```

```
##   25%  75%
## 0.75 2.38
```



- The *IQR* is the distance between the 1st and 3rd quartile: $\mathrm{IQR} = Q3 - Q1$

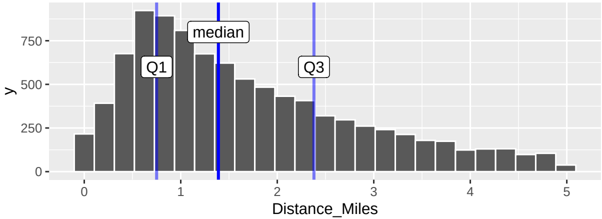
Quartiles and IQR

Where the median divides data into equal halves, *quartiles* divide data into equal quarters

- 25% of all observations are less than the *first quartile* $Q1$
- 25% of all observations are greater than the *third quartile* $Q3$

```
quantile(biketown_short$Distance_Miles, c(.25, .75))
```

```
##  25%  75%
## 0.75 2.38
```



- The *IQR* is the distance between the 1st and 3rd quartile: $\mathrm{IQR} = Q3 - Q1$
- Comparing $\mathrm{Median} - Q1$ and $Q3 - \mathrm{Median}$ can show shape of distribution.

Section 2

Summarizing Categorical Data

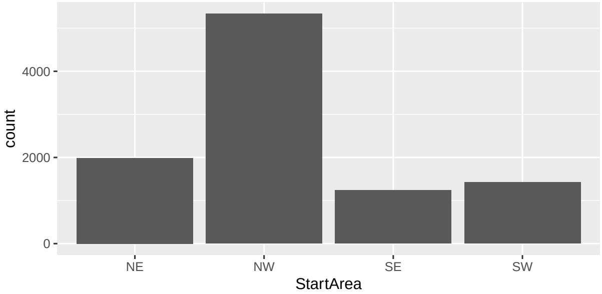
The Distribution of a Categorical Variable

Distributions of categorical variables can be presented in tables and summarized in bar charts:

The Distribution of a Categorical Variable

Distributions of categorical variables can be presented in tables and summarized in bar charts:

StartArea	NE	NW	SE	SW
n	1989	5334	1240	1424



Contingency Tables

- To compare 2 categorical variables, we can use a *contingency table*, which lists the counts for each pair of values of the two variables:

Contingency Tables

- To compare 2 categorical variables, we can use a *contingency table*, which lists the counts for each pair of values of the two variables:

	Casual	Subscriber
NE	1141	848
NW	2586	2748
SE	762	478
SW	865	559

Contingency Tables

- To compare 2 categorical variables, we can use a *contingency table*, which lists the counts for each pair of values of the two variables:

	Casual	Subscriber
NE	1141	848
NW	2586	2748
SE	762	478
SW	865	559

- Contingency tables can be created by applying the `table()` function to 2 colums of a data frame:

Contingency Tables

- To compare 2 categorical variables, we can use a *contingency table*, which lists the counts for each pair of values of the two variables:

	Casual	Subscriber
NE	1141	848
NW	2586	2748
SE	762	478
SW	865	559

- Contingency tables can be created by applying the `table()` function to 2 colums of a data frame:

```
table(biketown$StartArea, biketown$PaymentPlan)
```

Marginal Counts

- Suppose we want to recover the individual distribution of each variable in a table.

```
my_table<-table(biketown$StartArea, biketown$PaymentPlan)
```

	Casual	Subscriber
NE	1141	848
NW	2586	2748
SE	762	478
SW	865	559

- Apply the `margin.table()` function to a `table`. Use 1 for the row variable and 2 for the column variable

Marginal Counts

- Suppose we want to recover the individual distribution of each variable in a table.

```
my_table<-table(biketown$StartArea, biketown$PaymentPlan)
```

	Casual	Subscriber
NE	1141	848
NW	2586	2748
SE	762	478
SW	865	559

- Apply the `margin.table()` function to a `table`. Use 1 for the row variable and 2 for the column variable

```
margin.table(my_table, 1)
```

```
##
##   NE   NW   SE   SW
## 1989 5334 1240 1424
```

```
margin.table(my_table,2)
```

```
##
##      Casual Subscriber
##        5354       4633
```

Frequency Tables

Instead of comparing counts for each pair of values, we can consider the proportion of observations in each pair:

Frequency Tables

Instead of comparing counts for each pair of values, we can consider the proportion of observations in each pair:

```
my_table
```

	Casual	Subscriber
NE	1141	848
NW	2586	2748
SE	762	478
SW	865	559

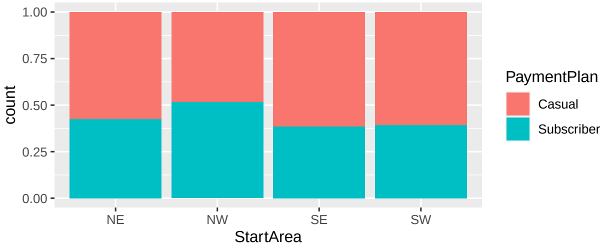
```
prop.table(my_table)
```

	Casual	Subscriber
NE	0.1142485	0.0849104
NW	0.2589366	0.2751577
SE	0.0762992	0.0478622
SW	0.0866126	0.0559728

Row and Column Proportions

How do we create a table version of the segmented bar chart?

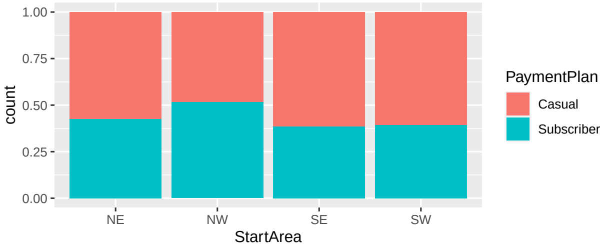
```
ggplot(biketown, aes(x =StartArea, fill =PaymentPlan))+geom_bar(position ="fill")
```



Row and Column Proportions

How do we create a table version of the segmented bar chart?

```
ggplot(biketown, aes(x =StartArea, fill =PaymentPlan))+geom_bar(position ="fill")
```




```
prop.table(my_table, 1)
```

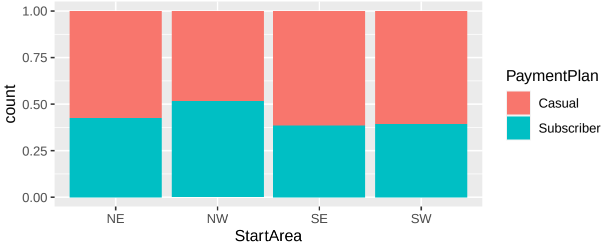
```
##
##         Casual Subscriber
##    NE 0.5736551  0.4263449
##    NW 0.4848144  0.5151856
##    SE 0.6145161  0.3854839
##    SW 0.6074438  0.3925562
```

- Each row gives breakdown of `PaymentPlan` by levels of `StartArea`

Row and Column Proportions

How do we create a table version of the segmented bar chart?

```
ggplot(biketown, aes(x =StartArea, fill =PaymentPlan))+geom_bar(position ="fill")
```




```
prop.table(my_table, 1)
```

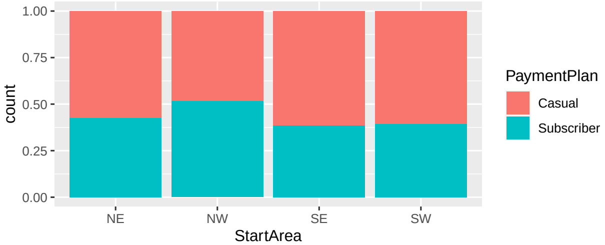
```
##
##         Casual Subscriber
##   NE 0.5736551  0.4263449
##   NW 0.4848144  0.5151856
##   SE 0.6145161  0.3854839
##   SW 0.6074438  0.3925562
```

- Each row gives breakdown of `PaymentPlan` by levels of `StartArea`
- Note row proportions add to 1.

Row and Column Proportions

How do we create a table version of the segmented bar chart?

```
ggplot(biketown, aes(x =StartArea, fill =PaymentPlan))+geom_bar(position ="fill")
```




```
prop.table(my_table, 1)
```

```
##
##         Casual Subscriber
##    NE 0.5736551  0.4263449
##    NW 0.4848144  0.5151856
##    SE 0.6145161  0.3854839
##    SW 0.6074438  0.3925562
```

- Each row gives breakdown of PaymentPlan by levels of StartArea
- Note row proportions add to 1.
- Do column proportions?

Row and Column Proportions

Compare the results in the following tables:

Row and Column Proportions

Compare the results in the following tables:

```
prop.table(my_table, 1)
```

```
##
##         Casual Subscriber
##    NE 0.5736551  0.4263449
##    NW 0.4848144  0.5151856
##    SE 0.6145161  0.3854839
##    SW 0.6074438  0.3925562
```

```
prop.table(my_table, 2)
```

```
##
##         Casual Subscriber
##    NE 0.2131117  0.1830348
##    NW 0.4830034  0.5931362
##    SE 0.1423235  0.1031729
##    SW 0.1615614  0.1206562
```

Row and Column Proportions

Compare the results in the following tables:

```
prop.table(my_table, 1)
```

```
##
##          Casual  Subscriber
##   NE 0.5736551   0.4263449
##   NW 0.4848144   0.5151856
##   SE 0.6145161   0.3854839
##   SW 0.6074438   0.3925562
```

```
prop.table(my_table, 2)
```

```
##
##          Casual  Subscriber
##   NE 0.2131117   0.1830348
##   NW 0.4830034   0.5931362
##   SE 0.1423235   0.1031729
##   SW 0.1615614   0.1206562
```

And compare to the total proportion table:

```
prop.table(my_table)
```

```
##
##          Casual Subscriber
##   NE 0.11424852 0.08491038
##   NW 0.25893662 0.27515771
##   SE 0.07629919 0.04786222
##   SW 0.08661260 0.05597276
```

Section 3

Summarizing with dplyr

The `dplyr` package

- The `dplyr` (dee-plier) package provides a set of specialized tools for manipulating dataframes.

The dplyr package

- The `dplyr` (dee-plier) package provides a set of specialized tools for manipulating dataframes.
- While `dplyr` contains many functions (we'll see at least 6 over the next few days), for now we focus on just one: `summarize` (or `summarise`)

The dplyr package

- The `dplyr` (dee-plier) package provides a set of specialized tools for manipulating dataframes.
- While `dplyr` contains many functions (we'll see at least 6 over the next few days), for now we focus on just one: `summarize` (or `summarise`)
- Previously, we applied functions like `mean()`, `sd()` and `quantile()` to columns of a data frame to get summary statistics:

The dplyr package

- The dplyr (dee-plier) package provides a set of specialized tools for manipulating dataframes.
- While dplyr contains many functions (we'll see at least 6 over the next few days), for now we focus on just one: summarize (or summarise)
- Previously, we applied functions like mean(), sd() and quantile() to columns of a data frame to get summary statistics:

```
mean(biketown$Distance_Miles)
```

```
## [1] 2.047225
```

The dplyr package

- The `dplyr` (dee-plier) package provides a set of specialized tools for manipulating dataframes.
- While `dplyr` contains many functions (we'll see at least 6 over the next few days), for now we focus on just one: `summarize` (or `summarise`)
- Previously, we applied functions like `mean()`, `sd()` and `quantile()` to columns of a data frame to get summary statistics:

```
mean(biketown$Distance_Miles)
```

```
## [1] 2.047225
```

- But it would be nice to have an easy way to store multiple summary statistics in a data frame

The `summarize` function

The `summarize` function takes a data frame, applies specified summary functions to 1 or more columns, and returns a data frame of the results.

The summarize function

The summarize function takes a data frame, applies specified summary functions to 1 or more columns, and returns a data frame of the results.

```
library(dplyr)
summarize(
  biketown,
    Mean_Distance = mean(Distance_Miles),
    SD_Distance = sd(Distance_Miles),
    Median_StartHour = median(StartHour),
    IQR_StartHour = IQR(StartHour)
)
```

```
## # A tibble: 1 x 4
##   Mean_Distance SD_Distance Median_StartHour IQR_StartHour
##           <dbl>       <dbl>            <int>         <dbl>
## 1          2.05        1.95               15             7
```

- Note that code is separated by line breaks for improved readability
- New column names can be arbitrary (but it's nice if they are informative)

The summarize function

The summarize function takes a data frame, applies specified summary functions to 1 or more columns, and returns a data frame of the results.

```
library(dplyr)
summarize(
  biketown,
    These = mean(Distance_Miles),
    Can = sd(Distance_Miles),
    Be = median(StartHour),
    Whatever = IQR(StartHour)
)
```

```
## # A tibble: 1 x 4
##   These   Can    Be Whatever
##   <dbl> <dbl> <int>    <dbl>
## 1  2.05  1.95    15        7
```

- Note that code is separated by line breaks for improved readability
- New column names can be arbitrary (but it's nice if they are informative)

Extending `summarize`

- The `summarize` function can be combined with many common R functions that take a list of values and return a single value:

Extending summarize

- The summarize function can be combined with many common R functions that take a list of values and return a single value:
  - mean()
  - sd()
  - median()
  - IQR()
  - quantile()
  - sum()
  - min()
  - max()
  - n()

Extending `summarize`

- The `summarize` function can be combined with many common R functions that take a list of values and return a single value:
    - `mean()`
    - `sd()`
    - `median()`
    - `IQR()`
    - `quantile()`
    - `sum()`
    - `min()`
    - `max()`
    - `n()`
- It's helpful to save the summarize dataframe for later access:

Extending summarize

- The summarize function can be combined with many common R functions that take a list of values and return a single value:
  - mean()
  - sd()
  - median()
  - IQR()
  - quantile()
  - sum()
  - min()
  - max()
  - n()
- It's helpful to save the summarize dataframe for later access:

```
distance_summary <- summarise(biketown,
                              mean_dist = mean(Distance_Miles),
                              sd_dist = sd(Distance_Miles))
```

Extending summarize

- The summarize function can be combined with many common R functions that take a list of values and return a single value:
  - mean()
  - sd()
  - median()
  - IQR()
  - quantile()
  - sum()
  - min()
  - max()
  - n()
- It's helpful to save the summarize dataframe for later access:

```
distance_summary <- summarise(biketown,
                              mean_dist = mean(Distance_Miles),
                              sd_dist = sd(Distance_Miles))
```

```
distance_summary$mean_dist
```

```
## [1] 2.047225
```

```
distance_summary$sd_dist
```

```
## [1] 1.950687
```