EDA Techniques Summary

Brandon Booth Connor Crane Amy Ly Abraham Mendoza Miles Moran

5/15/2022

Responsibilities / Sections Covered:

- 7.3.2: Typical Values (Abraham)
- 7.3.3: Variation: Unusual Values (Connor)
- 7.4: Missing Values (Amy)
- 7.5.3: Two continuous variables (Miles)
- 7.6: Patterns and models (Brandon)

7.3.2: Typical Values (Abraham)

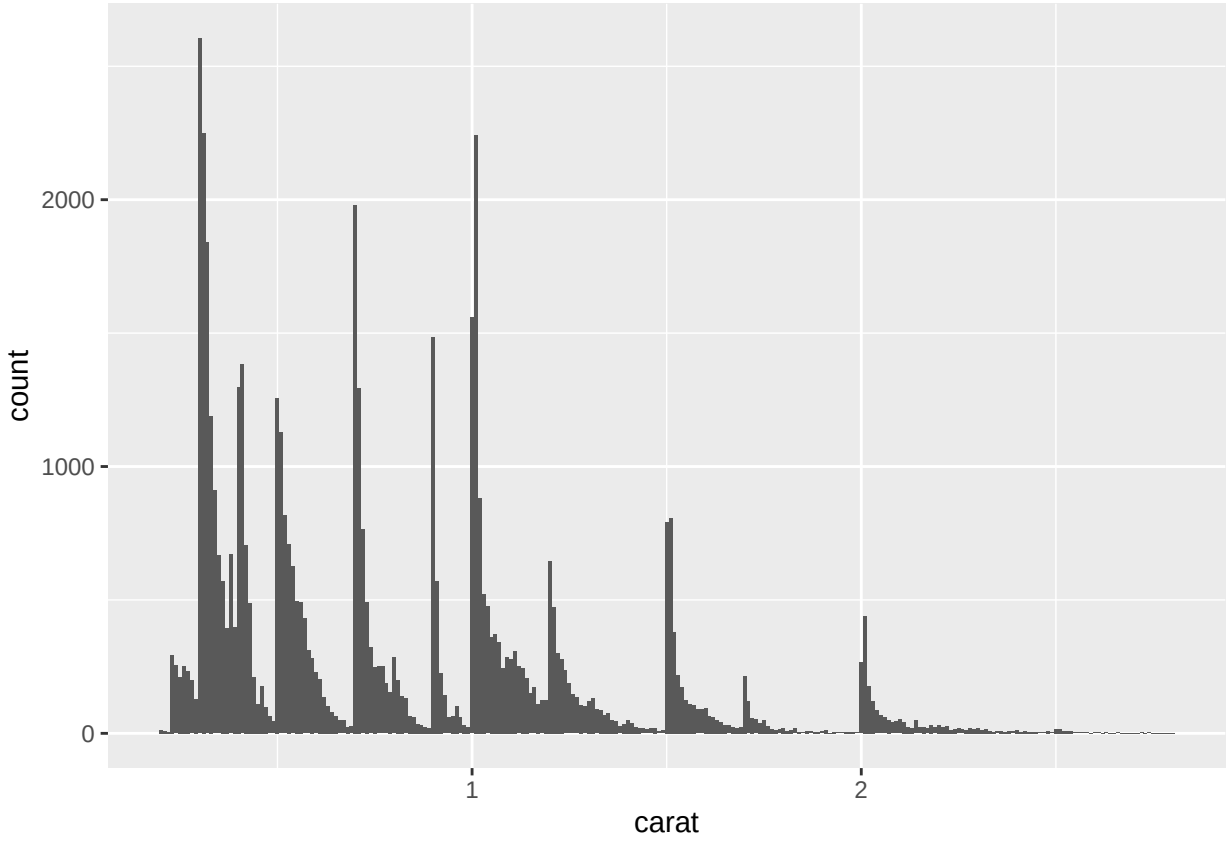
Summary

Typical values can show the common variables. One can then create useful questions out of such data. Questions like, "which values are most common and why?" One can also find clusters of similar values that can help us find subgroups within our data which can then help understand other questions pertaining to clusters.

Example

Below is an example of the count of diamond with a corresponding carat weight. One can see that there are spikes which correspond with a higher cont in that carat weight. We see higher counts at 0.25, 0.5, 0.75, 1, 1.5, and 2 carats.

```
smaller <- diamonds %>% filter(carat < 3)

ggplot(data = smaller, mapping = aes(x = carat)) +
    geom_histogram(binwidth = 0.01)
```



7.3.2: Typical Values (continued)

Why it is useful

Typical values can be useful because they prompt us to further explore the relationships that we see between the variables and further create more questions to investigate such relationships and clusters.

When it can be useful

This can be useful in the example above, when trying to further understand why there are specific typical values such as carat weights, as well as other typical values that we see in the data from the world around us. We could explore typical values in tax brackets to gain a better understanding on how large different socio-economic segments are and then compare them with other countries. Typical values help us better understand typical observations and help us create further questions to investigate.

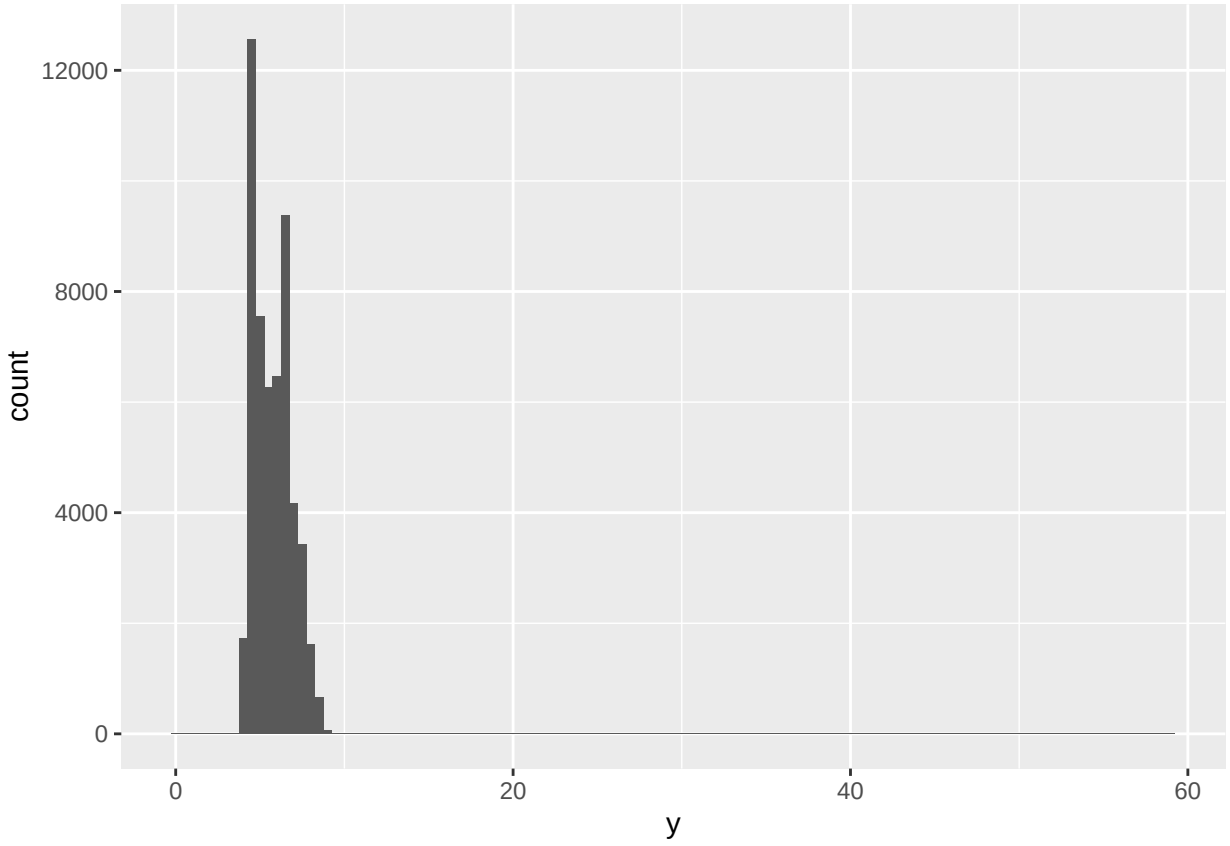
7.3.3: Variation: Unusual Values (Connor)

Very often during exploratory data analysis we will run into values such as outliers that do not fit with the majority of our data and should most likely be investigated more, or at the very least, kept in mind during further analysis. Sometimes these values are mistakes or biased errors but sometimes they are also relevant observations in our data that can have a substantial effect on our conclusions.

Plotting Unusual Values

Below is one not so obvious example of an unusual value.

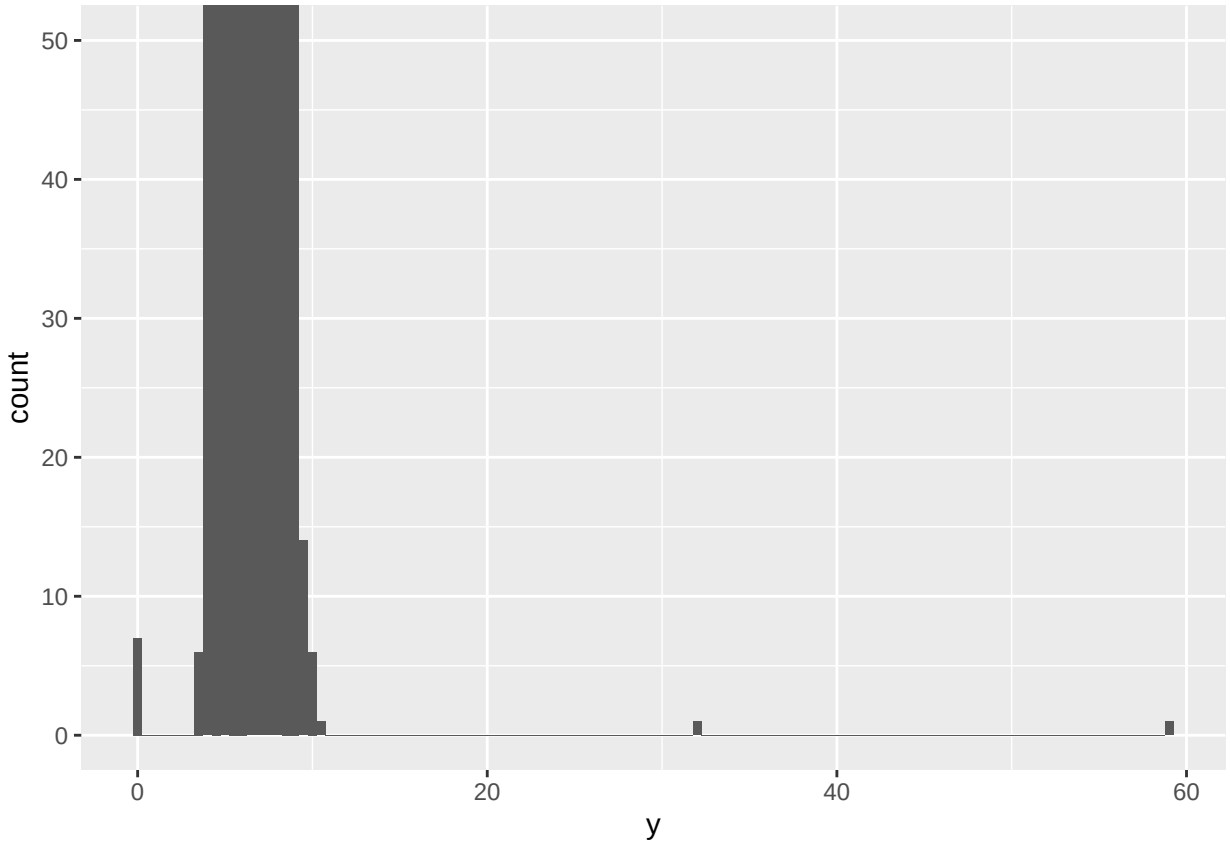
```
ggplot(diamonds) +
    geom_histogram(mapping = aes(x = y), binwidth = 0.5)
```



The only real evidence of unusual values is the disproportionate scaling of the x-axis. The lack of visible outliers is due to the other bins of more common values having so many more observations. One way to adjust for this is to change the scaling of the axes and then examine the plot again.

7.3.3: Variation: Unusual Values (continued)

```
ggplot(diamonds) +
    geom_histogram(mapping = aes(x = y), binwidth = 0.5) +
    coord_cartesian(ylim = c(0, 50))
```



Now that the scale of the plot is adjusted we can better see where our unusual values are located on the plot. This adjustment was made using the `coord_cartesian()` function. We can now take a closer look at these values using `dplyr`.

7.3.3: Variation: Unusual Values (continued)

Investigating with `dplyr`

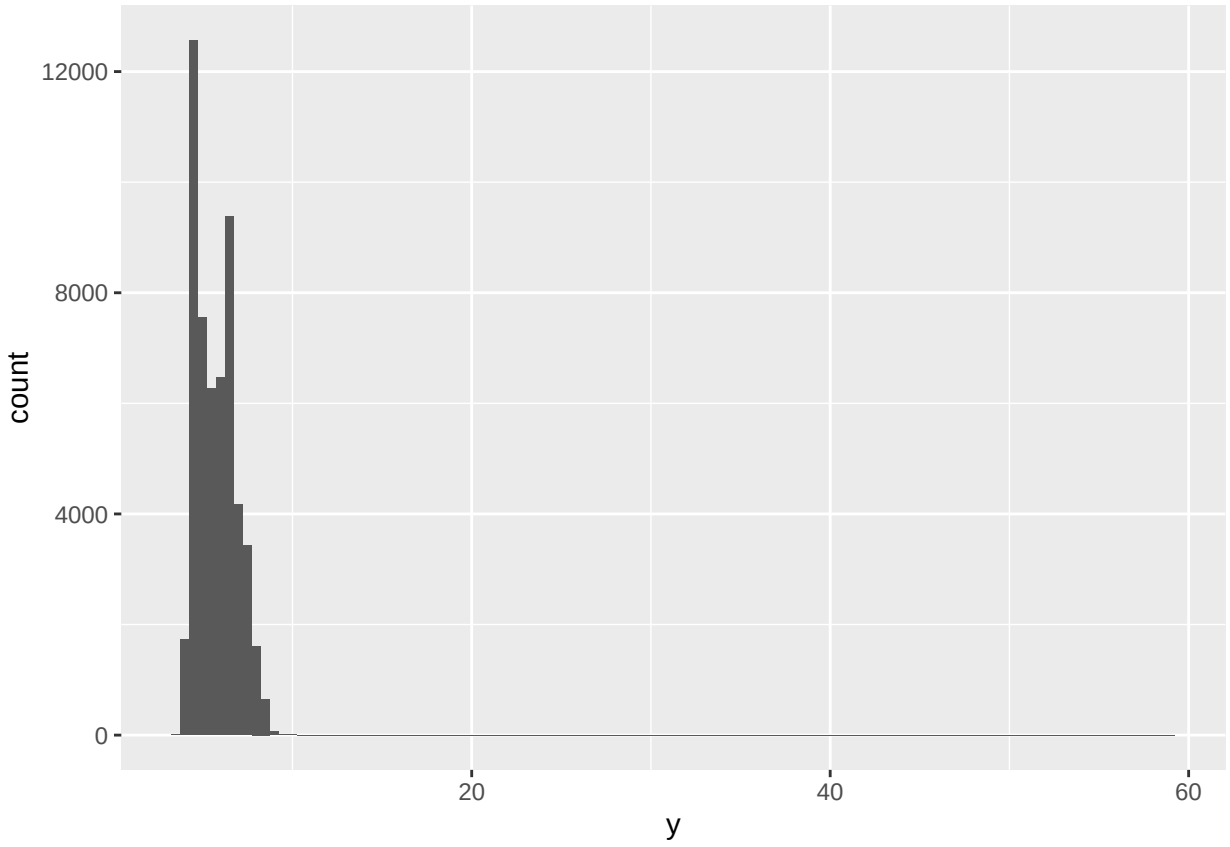
```
unusual <-
    diamonds %>%
    filter(y < 3 | y > 20) %>%
    select(price, x, y, z) %>%
    arrange(y)
unusual
```

```
## # A tibble: 9 x 4
##   price     x     y     z
##   <int> <dbl> <dbl> <dbl>
## 1  5139  0      0    0
## 2  6381  0      0    0
## 3 12800  0      0    0
## 4 15686  0      0    0
## 5 18034  0      0    0
## 6  2130  0      0    0
## 7  2130  0      0    0
## 8  2075  5.15  31.8  5.12
## 9 12210  8.09  58.9  8.06
```

Looking at this output we can see that our unusual values are due to values of 0 for the x, y, and z variables or fairly large values for these variables. Since diamonds can not have a width (y) of 0mm then we know these observations are mistakes. A next step would be removing these likely errors and seeing how it effects our initial plot.

7.3.3: Variation: Unusual Values (continued)

```
diamonds2 <- diamonds %>% filter(y > 0)

ggplot(diamonds2) +
    geom_histogram(mapping = aes(x = y), binwidth = 0.5)
```



Removing those potentially incorrect observations does not effect the plot all that much and it is probably fine for us to adjust those observations to being missing values/NAs.

When is this Useful?

Like I mentioned earlier, having abnormal or unusual values in one's dataset is not uncommon. When presented with this scenario it is important to know what steps to take to discover what is going on with said values. Sometimes, all it takes is a quick peak at the data with `dplyr` to discover the values are errors and can be safely accounted for. Other times a deeper dive and more coding is necessary to understand what is going on. For example, maybe a clustered dataset is being examined and upon further inspection one grouping was incorrectly treated and that is why the data is skewed. Unusual data points are in most datasets and make them interesting, knowing how to diagnose them is important to make sure your plots and conclusions are correct in the long run.

7.4: Missing Values (Amy)

Missing values in a dataset are important and may indicate that more investigation is required. If there are outliers or unusual data points, it's not a good idea to just remove them from the data set. Instead, those values can be recoded to "NA" and R will ignore the values when plotting or performing functions. This is for your awareness.

Additionally, groups of observations that have missing values may have something in common that makes them different from cases with recorded values. This would be important when making comparisons.

I will use the iris dataset to demonstrate some of the techniques.

You can identify if there are any missing values originally from a data set by using the summary function. Then the number of NA's will be tallied.

```
data("iris")

iris_NAs <- iris %>%
    mutate(Sepal.Length = ifelse(Sepal.Length < 4.5, NA, Sepal.Length),
           Sepal.Width = ifelse(Sepal.Width < 2.5, NA, Sepal.Width))

summary(iris_NAs)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
##  Min.   :4.500   Min.   :2.500   Min.   :1.000   Min.   :0.100   setosa    :50
##  1st Qu.:5.125   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
##  Mean   :5.884   Mean   :3.119   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.400   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##  NA's   :4       NA's   :11
```

Alternatively, you can recode unusual values into NA's. Here, I am recoding values that are beyond the 3rd quantile:

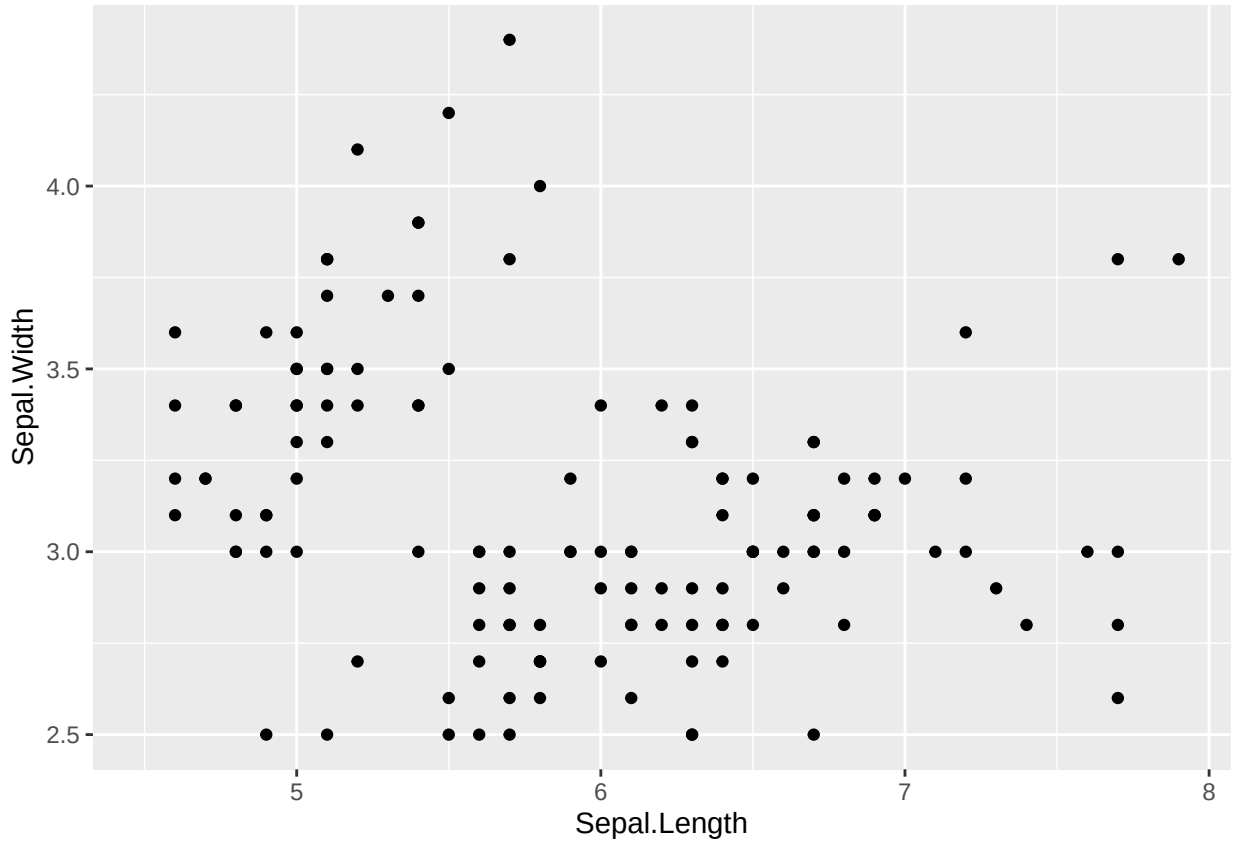
```
iris_recode <- iris %>%
    mutate(Sepal.Length = ifelse(Sepal.Length > 6.4, NA, Sepal.Length),
           Sepal.Width = ifelse(Sepal.Width > 3.3, NA, Sepal.Width),
           Petal.Length = ifelse(Petal.Length > 5.1, NA, Petal.Length),
           Petal.Width = ifelse(Petal.Width > 1.8, NA, Petal.Width))

summary(iris_recode)
```

```
##   Sepal.Length  Sepal.Width     Petal.Length    Petal.Width          Species
##  Min.   :4.3   Min.   :2.000   Min.   :1.000   Min.   :0.1000   setosa    :50
##  1st Qu.:5.0   1st Qu.:2.700   1st Qu.:1.500   1st Qu.:0.2000   versicolor:50
##  Median :5.5   Median :2.900   Median :3.850   Median :1.1000   virginica :50
##  Mean   :5.5   Mean   :2.867   Mean   :3.152   Mean   :0.9147
##  3rd Qu.:6.0   3rd Qu.:3.100   3rd Qu.:4.525   3rd Qu.:1.5000
##  Max.   :6.4   Max.   :3.300   Max.   :5.100   Max.   :1.8000
##  NA's   :35    NA's   :37      NA's   :34      NA's   :34
```

7.4: Missing Values (continued)

When plotting, R will warn us that observations with NA's are removed. If we don't care about the warning, then set na.rm = TRUE to surpress the warning.

```
ggplot(data = iris_NAs, mapping = aes(x=Sepal.Length, y = Sepal.Width)) +
    geom_point()
```



Lastly, we should explore the relationship between the missing values and other factors of the dataset. For the iris dataset, perhaps certain species have more variation and result in more unusual values?

```
iris_recode %>%
    group_by(Species) %>%
    summarize(Count = sum(!complete.cases(iris_recode)))
```

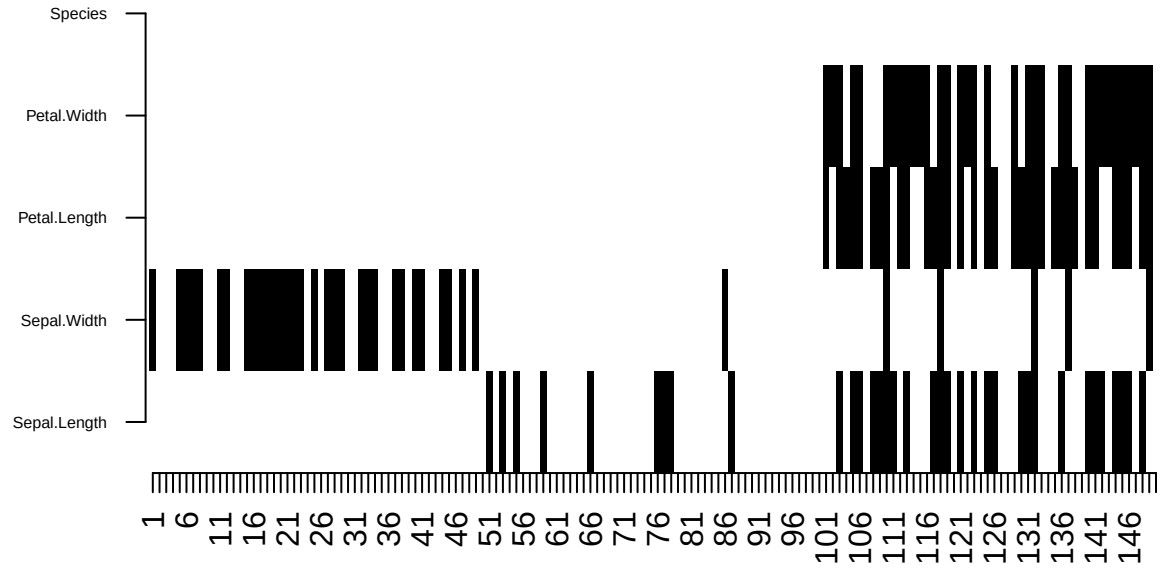
```
## # A tibble: 3 x 2
##   Species    Count
##   <fct>      <int>
## 1 setosa        83
## 2 versicolor    83
## 3 virginica     83
```

In this case, it's not very helpful. Each of the species have the same number of rows that have missing values (which are actually outliers).

7.4: Missing Values (continued)

Another technique that I learned from a previous class is to create a "heatmap" of where missing values tended to come from. In the plot below, it's easy to see for which observations, where the NAs are clustered. For example, a lot of the later observations (103 to 150) tend to have missing data for Petal.Width, Petal.Length, and Sepal.Length. Meanwhile, a lot of the earlier observations (1-49) tend to have missing values in Sepal.Width.

```
image(is.na(iris_recode), axes=FALSE, col=gray(1:0))
par(mar = c(bottom = 5.1, left = 4, top = 4.1, right = 0.5))
axis(2, at=0:4/4, labels = colnames(iris_recode), las=2, cex.axis=0.5)
axis(1, at=0:149/149, labels=row.names(iris_recode), las=2, cex.axis=1)
```

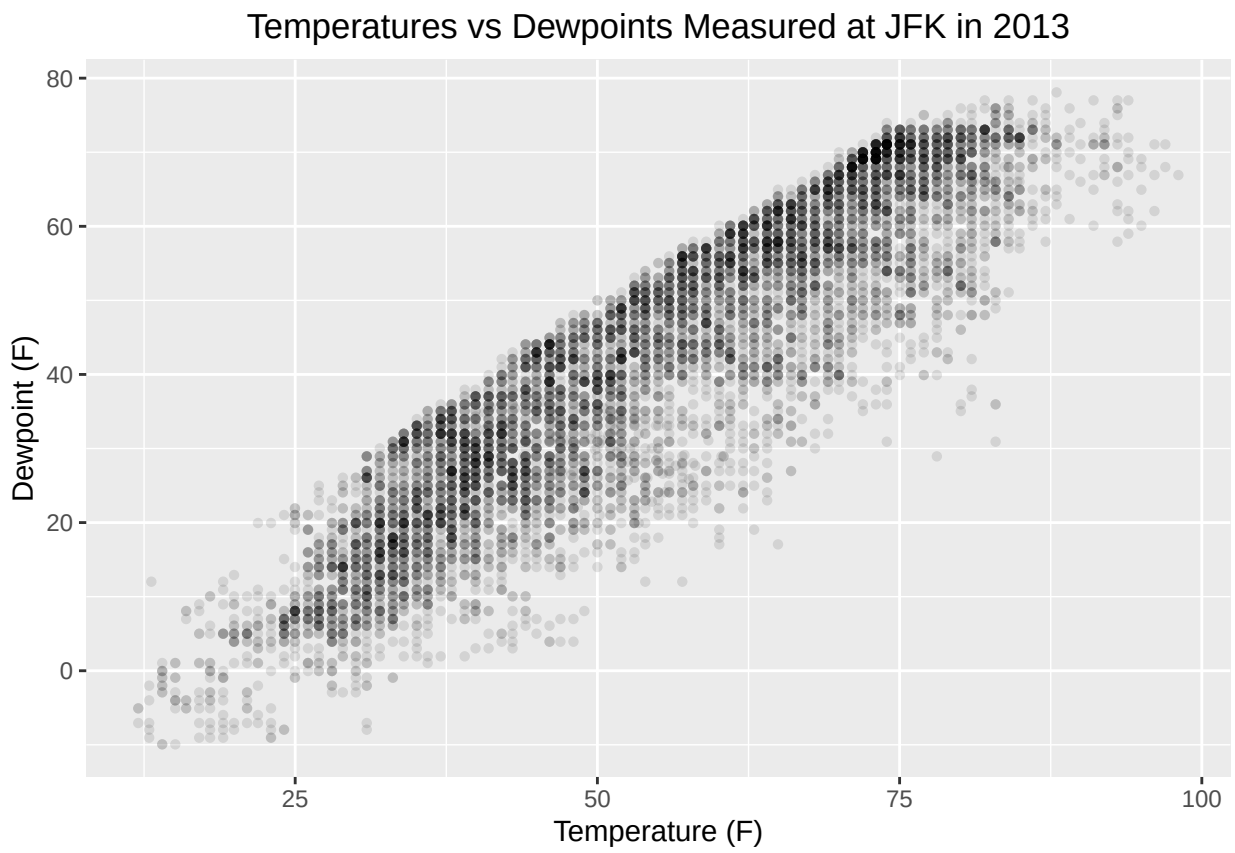


7.5.3: Two continuous variables (Miles)

Plotting the covariation between two continuous variables is difficult because, unless you are willing to bin one or both of the variables, the options are limited. Without binning, the primary way of displaying a continuous×continuous relationship is a scatterplot (or some variation thereof). Consider the `weather` dataframe from the `nycflights13` package used last week: suppose we'd like to know about the relationship between temperature and dewpoint for the measurements taken at JFK international airport.

In the first plot, we scatterplot the tempereature and dewpoint for all measurements taken in 2013

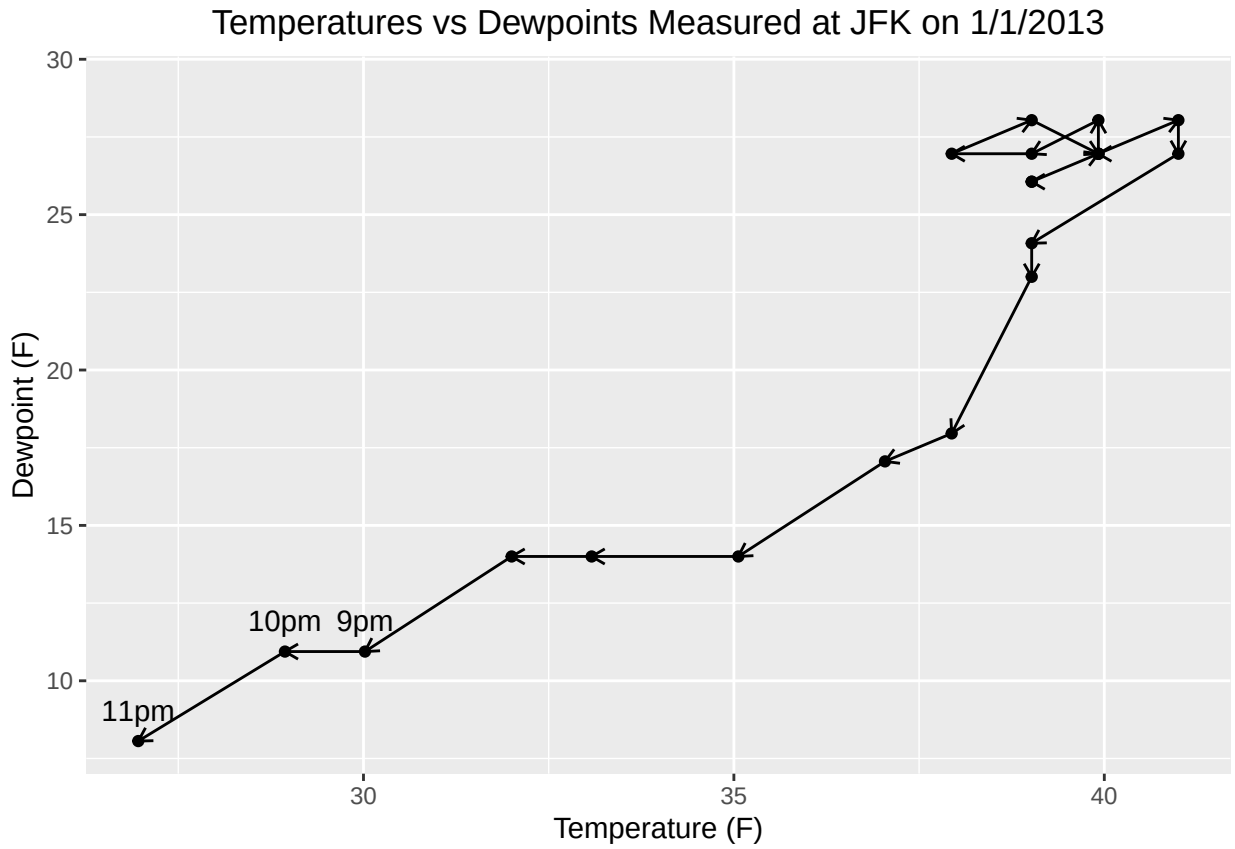
```
varsOfInterest <- c("temp", "dewp", "humid", "wind_speed", "precip", "pressure")
weather <-
    nycflights13::weather %>%
    filter(origin == "JFK") %>%
    drop_na(varsOfInterest)

ggplot(weather) +
    geom_point(aes(x=temp, y=dewp), pch=16, alpha=0.1) +
    labs(title="Temperatures vs Dewpoints Measured at JFK in 2013",
                x="Temperature (F)", y="Dewpoint (F)")
```



7.5.3: Two continuous variables (continued)

In the second plot, we scatterplot the *progression* of tempereature and dewpoint over time for *just* the measurements taken on 1/1/2013

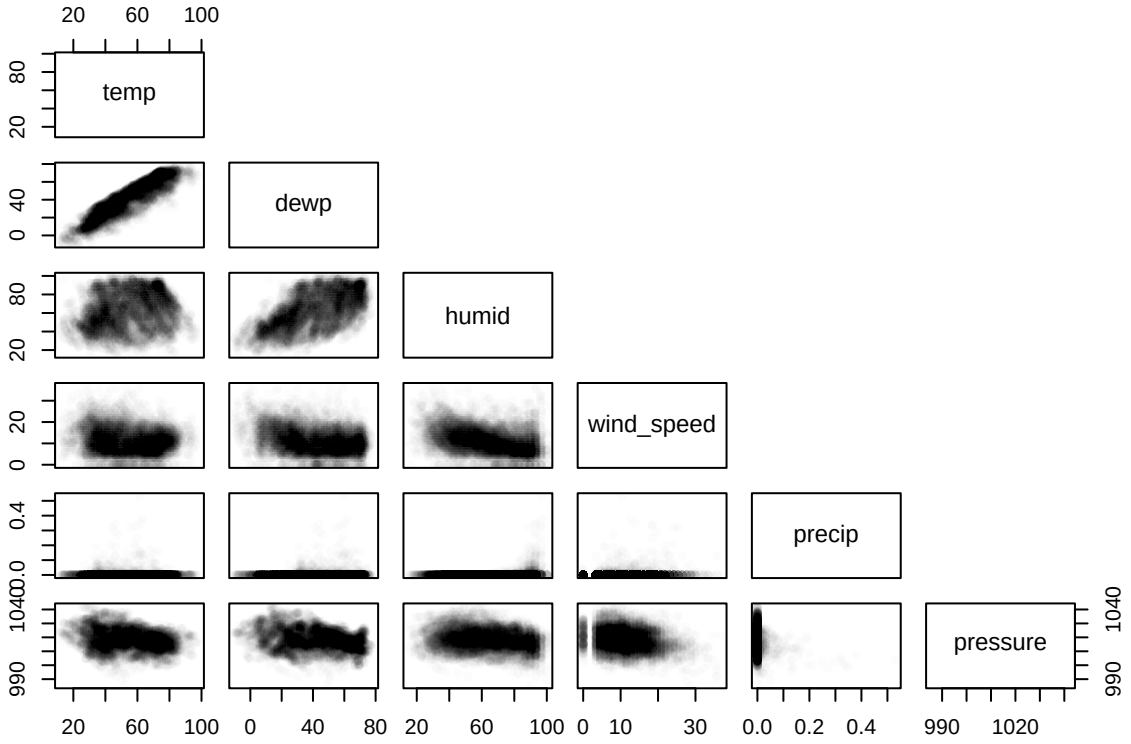
```
weather %>%
    filter(year == 2013, month == 1, day == 1) %>%
    mutate(hourLbl = c(rep("", 18), "9pm", "10pm", "11pm")) %>%
    ggplot(., aes(x=temp, y=dewp, label=hourLbl)) +
        geom_point() +
        geom_text(nudge_y=1) +
        geom_segment(aes(xend = c(tail(temp, n=-1), NA),
                         yend = c(tail(dewp, n=-1), NA)),
                     arrow=arrow(length=unit(0.2, "cm"))
                    ) +
        labs(x="Temperature (F)", y="Dewpoint (F)",
             title="Temperatures vs Dewpoints Measured at JFK on 1/1/2013")
```



7.5.3: Two continuous variables (continued)

In the third plot, we scatterplot *all* combinations of several variables in the dataset (not just the tempereature and dewpoint). This is called a scatterplot matrix or a correlogram.

```
### Note: GGalley is not updated; so, we don't have access to ggpairs() ###

weather %>%
    select(varsOfInterest) %>%
    pairs(., pch=16, col=scales::alpha(1, 0.01), upper.panel = NULL)
```

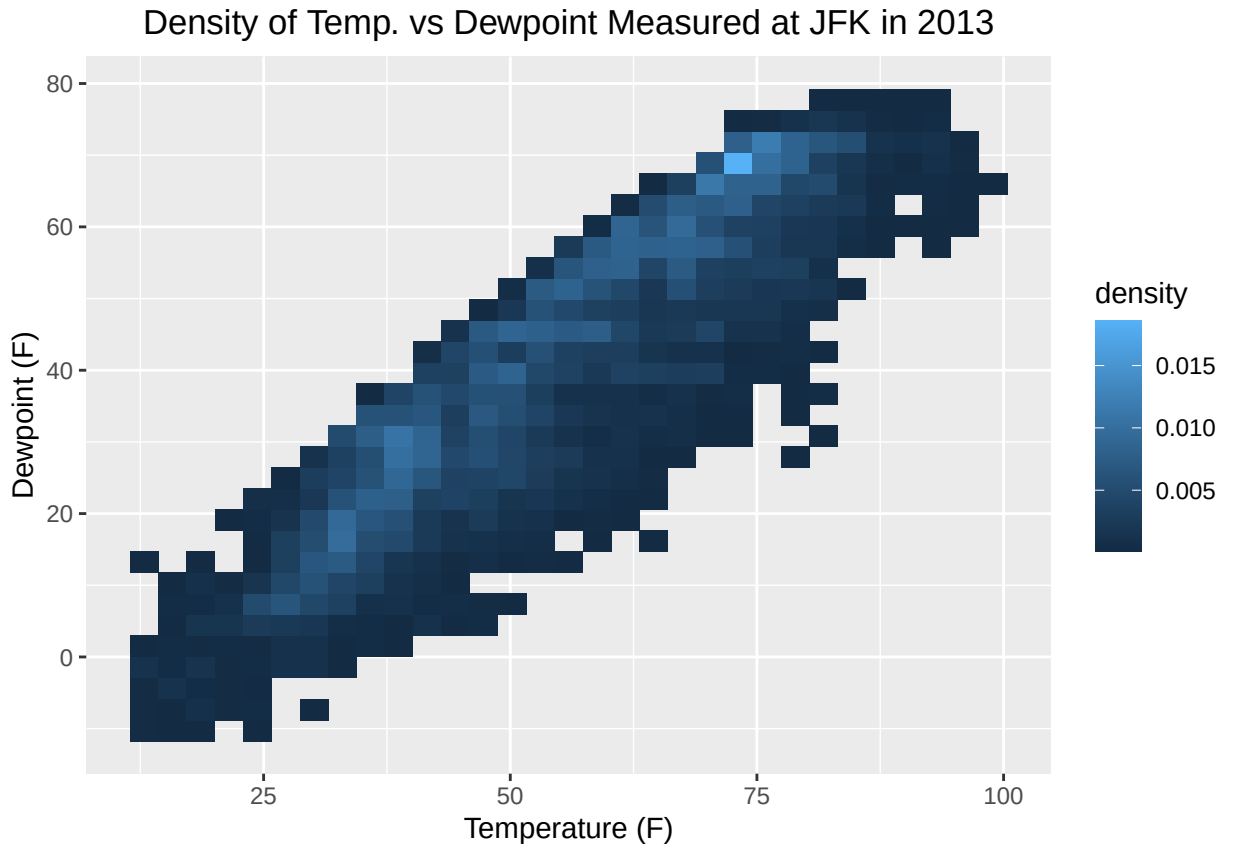


7.5.3: Two continuous variables (continued)

There are more ways to describe the shape of the relationship between two continuous variables; but, all of the following require us to employ binning of some kind. Although this means we lose some information, binning continuous variables *can* be the optimal way to visualize them: scatterplots often suffer from over-plotting, even when we set the alpha level to something really small.

In the first plot, we bin both variables into equal-area rectangular regions and plot their joint density (i.e. the color-scale is continuous)

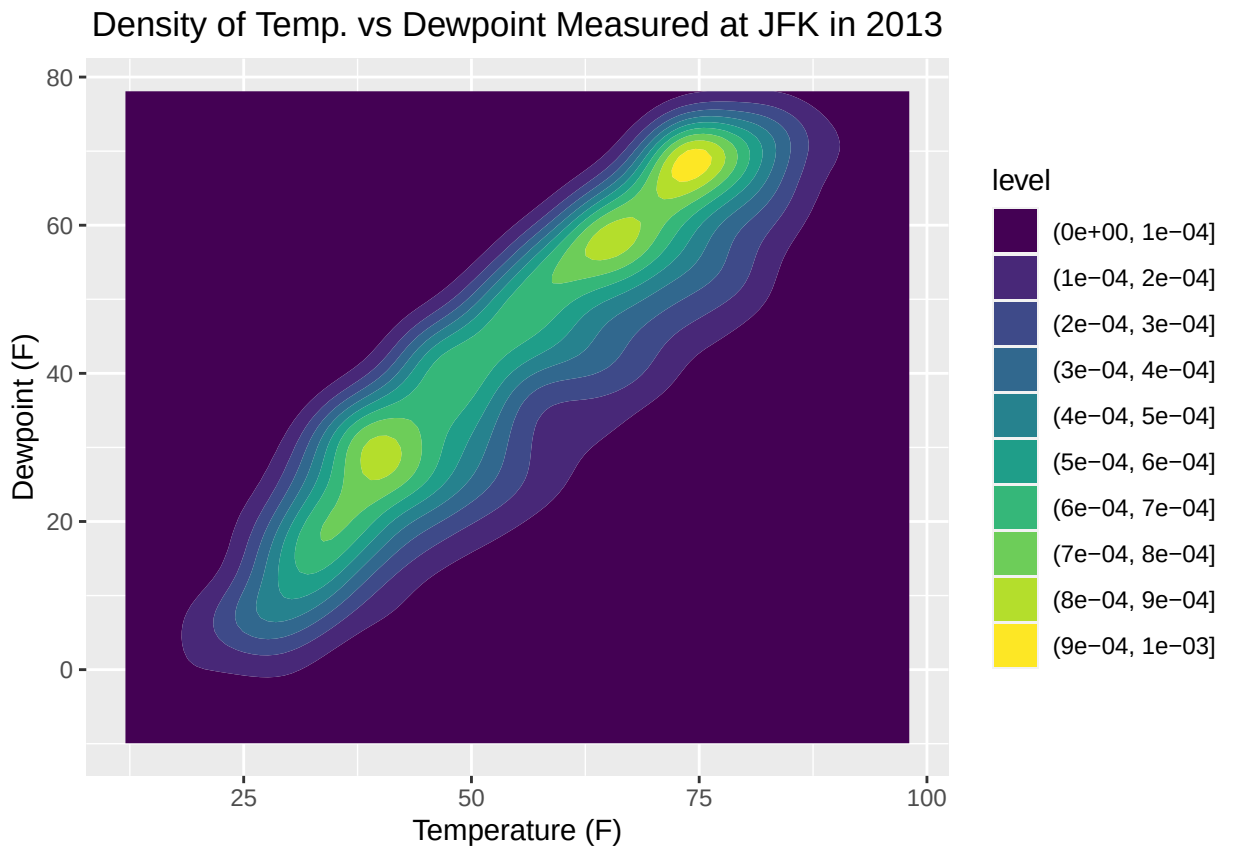
```
weatherLabs <- labs(title="Density of Temp. vs Dewpoint Measured at JFK in 2013",
                    x="Temperature (F)", y="Dewpoint (F)")
ggplot(weather) + weatherLabs +
    geom_bin2d(aes(x=temp, y=dewp, fill=..density..))
```



7.5.3: Two continuous variables (continued)

In the second plot, we *don't* bin either variable; but, we *do* bin the joint density level (i.e. the color-scale is discrete)

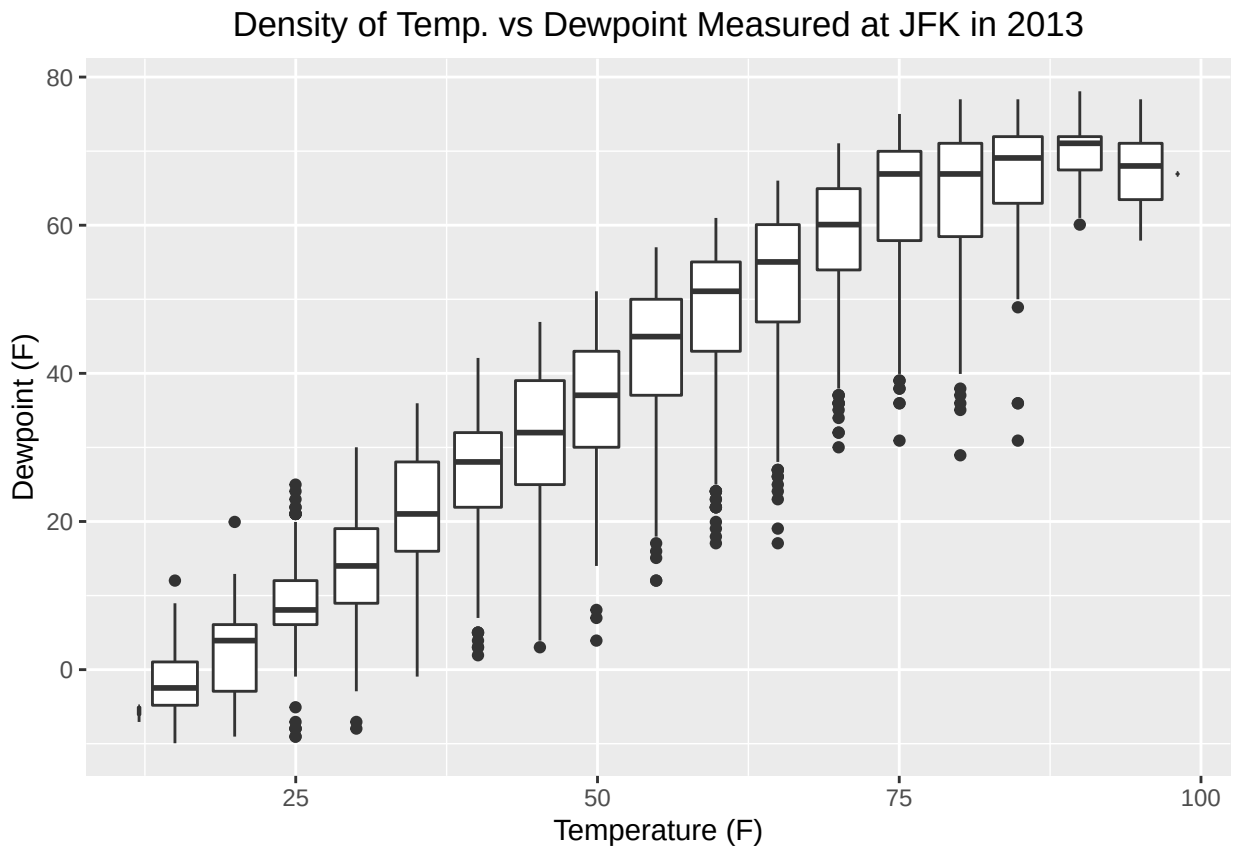
```
ggplot(weather) + weatherLabs +
    geom_density2d_filled(aes(x=temp, y=dewp))
```



7.5.3: Two continuous variables (continued)

In the third plot, we only bin *one* of the variables (here, it's `temp`) and plot the *marginal density* of the other variable for several different values of `temp`.

```
ggplot(weather) + weatherLabs +
    geom_boxplot(aes(x=temp, y=dewp, group = cut_width(temp, 5)))
```



7.6: Patterns and models (Brandon)

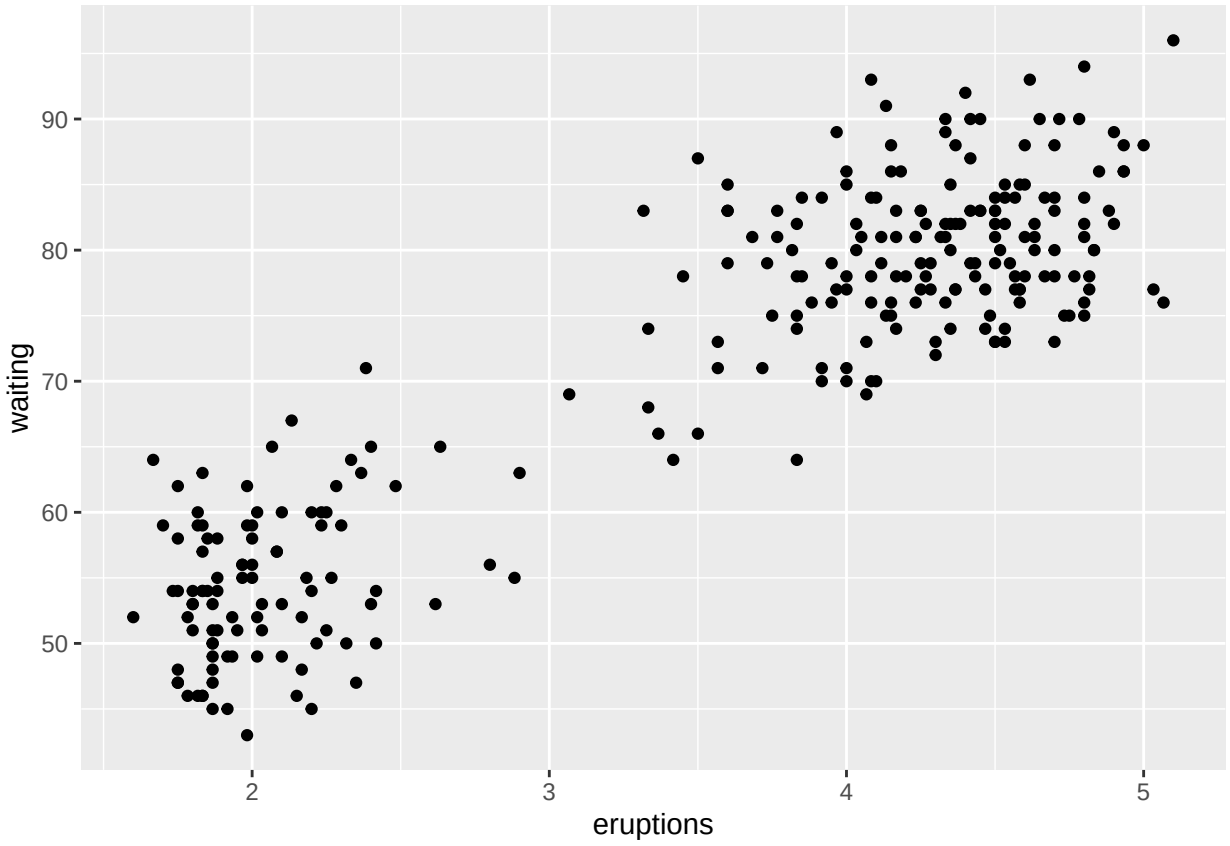
Summary

Viewing patterns in data via plots can be a useful way to identify systematic relationships. In addition to identifying relationships, we can use these plots to identify the strength of the relationship(s) and impact of other variables on the relationship(s) of interest.

Example

Below is an example of how a visual plot of Old Faithful eruption lengths versus the wait time between eruptions can be used to identify relationships that might otherwise be hard to identify.

```
# ?faithful
ggplot(data = faithful) +
    geom_point(mapping = aes(x = eruptions, y = waiting))
```

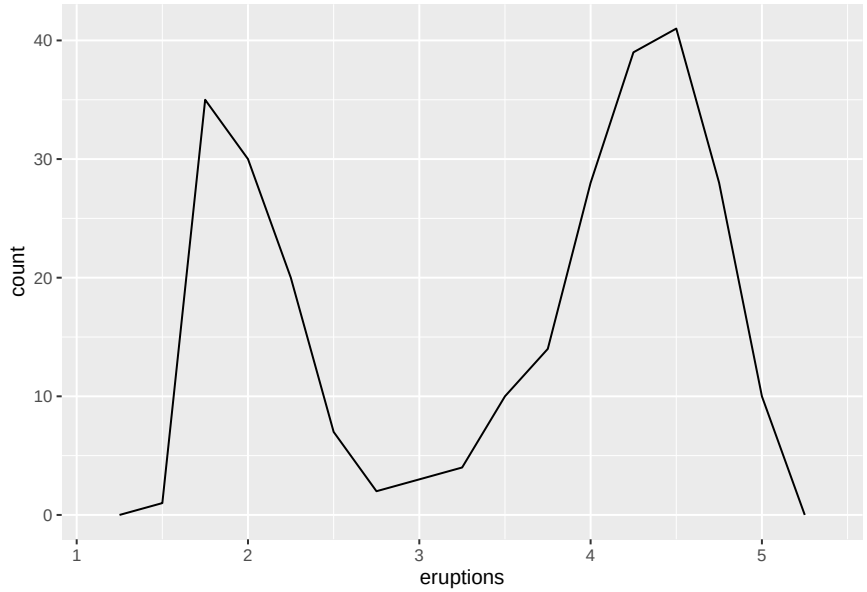


From the plot we can see that there are two clusters in the eruptions data that occur on the x-axis where centered where eruption time is equal to 2 and 4.25 min. This suggests most of the eruptions times are around these two times and that when that is the case the wait time between the eruptions is 55mins when eruption time is 2min and wait time is around 80min when the eruption time is 4min.

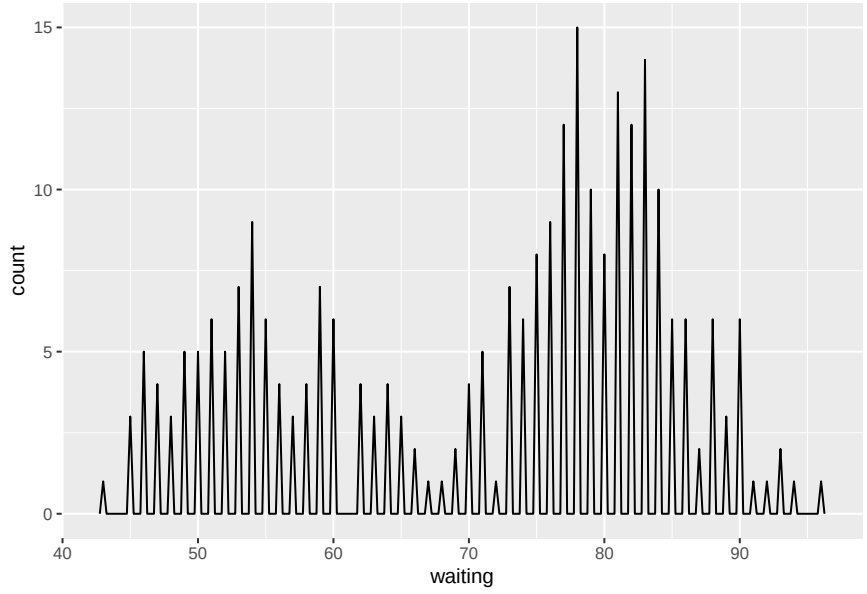
7.6: Patterns and models (continued)

We can further analyze this pattern by taking a closer look at occurrence of the eruptions times and wait times to next eruption.

```
ggplot(data = faithful, mapping = aes(x = eruptions)) +
    geom_freqpoly(binwidth = 0.25)
```




```
# or
# ggplot(faithful, aes(eruptions)) + geom_freqpoly(binwidth = 0.25)

ggplot(data = faithful, mapping = aes(x = waiting)) +
    geom_freqpoly(binwidth = 0.25)
```



7.6: Patterns and models (continued)

From this plot we can see that the greatest occurrence of eruptions occur with eruption time around 1.75min and 4.5min and the have greatest occurrence of wait times of 54min and 78min. We can see that this aligns with our observations from the the plot above, however the plot of eruption times and wait times allows up to see the correlation between these two variables.