Session IV Recap

N.Paterno

7/23/2021

This week we discussed the inner quartile range (IQR) and how to use it to identify outliers in data. Then we looked at how we can highlight outliers in a graph. Towards the end we briefly talked about covariance, which I explain in more detail below.

First, the data. We used the same `scoobydoo` dataset that we did last week.

```
scoobydoo <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   index = col_double(),
##   date_aired = col_date(format = ""),
##   run_time = col_double(),
##   monster_amount = col_double(),
##   unmask_other = col_logical(),
##   caught_other = col_logical(),
##   caught_not = col_logical(),
##   suspects_amount = col_double(),
##   culprit_amount = col_double(),
##   door_gag = col_logical(),
##   batman = col_logical(),
##   scooby_dum = col_logical(),
##   scrappy_doo = col_logical(),
##   hex_girls = col_logical(),
##   blue_falcon = col_logical()
## )
## i Use `spec()` for the full column specifications.
```

Let's filter our data to only look at the TV Series and change the `imdb` variable to be numeric.

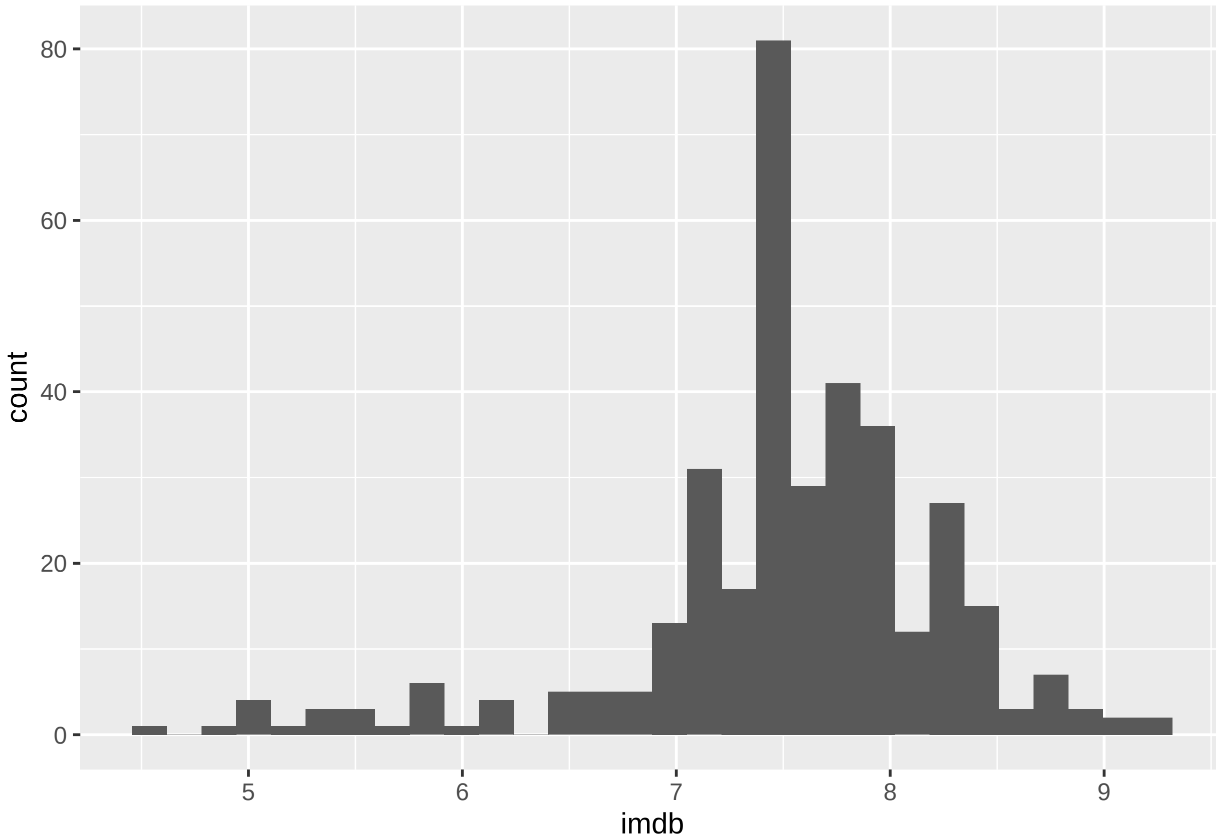
```
tv_data <- scoobydoo %>%
  filter(format == "TV Series") %>%
  mutate(imdb = as.numeric(imdb)) %>%
  filter(!is.na(imdb)) # This line removes all rows that have a value of NA for the imdb variable
```

```
## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion
```

Glancing at a histogram can give us an idea of if/how many outliers a dataset has.

```
ggplot(tv_data, aes(imdb))+
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



This isn't a perfect bell curve (which would be a normal distribution). Seeing how the left tail of the data is longer, I'd be willing to bet there are outliers there. We can determine this numerically using the IQR.

```
tv_data_summary <- tv_data %>%
  summarize(iqr = IQR(imdb),
            q1 = fivenum(imdb)[2], # calculates Q1
            q3 = fivenum(imdb)[4]) # calculates Q3
```

The IQR is a range which contains the middle 50% of the data. It starts at the first quartile, `q1`, and ends at the third, `q3`.

Once we have this value, we can calculate the cutoff values for the outliers, called fences.

The lower fence is $Q_1 - 1.5 \cdot IQR$ and the upper is $Q_3 + 1.5 \cdot IQR$.

```
lower_fence <- tv_data_summary$q1-1.5*tv_data_summary$iqr
upper_fence <- tv_data_summary$q3+1.5*tv_data_summary$iqr
```

Now, we can add a variable to our `tv_data` dataframe that tells is if a data point is an outlier.

```
tv_data <- tv_data %>%
  mutate(outlier = case_when(
    imdb < 6.4 ~ "low",
    imdb > 8.8 ~ "high",
    TRUE ~ "not an outlier"
  ))
```

The `case_when()` function allows us to define a new variable based on a logical condition involving an existing variable. In this case, we compare the episodes `imdb` score to the `lower_fence` and `upper_fence` calculated above.

Note, we could also just define any outlier to be "outlier".

```
tv_data <- tv_data %>%
  mutate(is_outlier = case_when(
    imdb < 6.4 ~ "yes",
    imdb > 8.8 ~ "yes",
    TRUE ~ "no"
  ))
```

This definition may be more useful if we're only concerned about *if* a data point is an outlier and not what direction it is (high or low).

To make sure both definitions worked, we can `select` and `arrange` our dataset.

```
tv_test <- tv_data %>%
  select(c(imdb, outlier, is_outlier)) %>%
  arrange(imdb)
```

Let's make some plots to see how we can use color to id the outliers in a histogram.

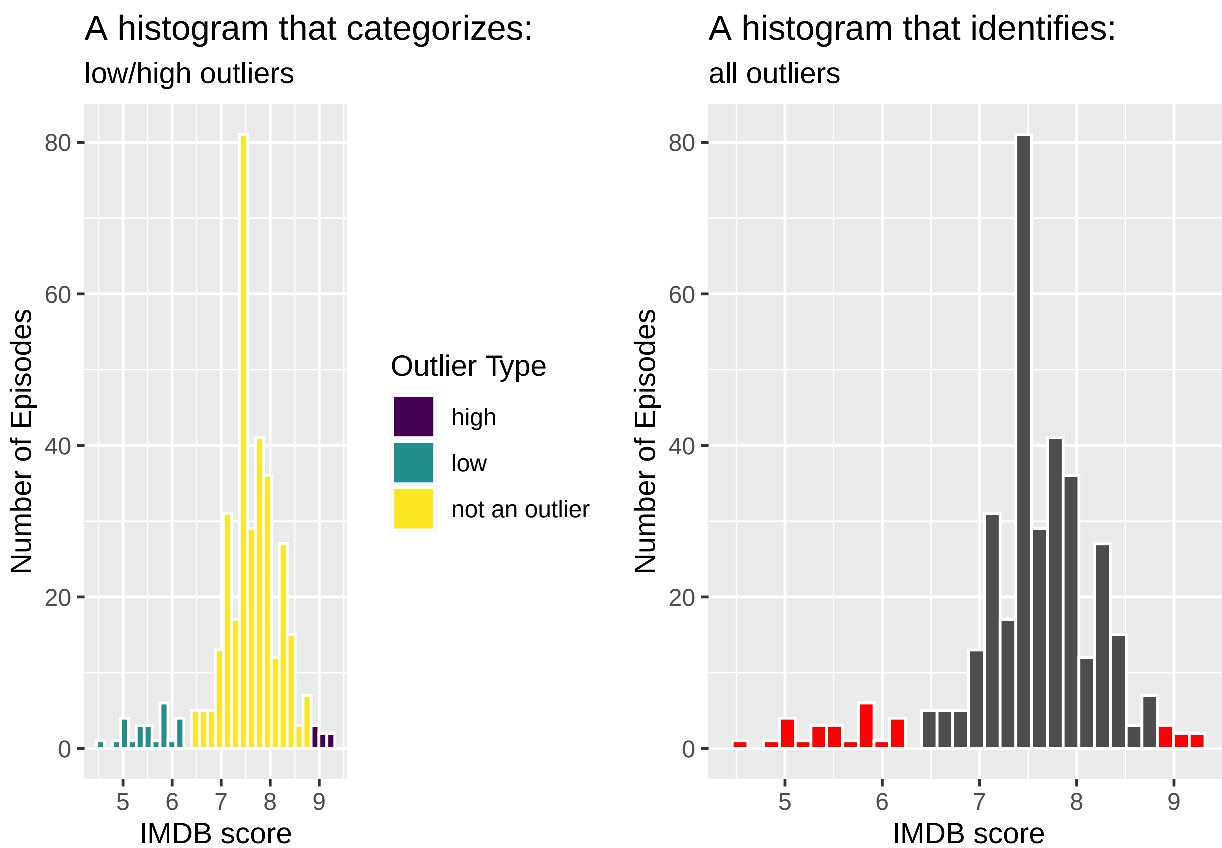
```
p1 <- ggplot(tv_data, aes(imdb, fill = outlier))+
  geom_histogram(color = "white")+
  labs(title = "A histogram that categorizes:",
       subtitle = "low/high outliers",
       x = "IMDB score",
       y = "Number of Episodes",
       fill = "Outlier Type")+
  scale_fill_viridis_d() # changes the colors of the bars


# custom colors for plot 2
my_colors = c("grey30", "red")

p2 <- ggplot(tv_data, aes(imdb, fill = is_outlier))+
  geom_histogram(color = "white")+
  labs(title = "A histogram that identifies:",
       subtitle = "all outliers",
       x = "IMDB score",
       y = "Number of Episodes")+
  scale_fill_manual(values = my_colors)+
  guides(fill = FALSE)

cowplot::plot_grid(p1, p2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Now let's consider covariation. Variation is a way to measure how values for a single variable vary. This is typically measured using standard deviation, variance or the IQR.

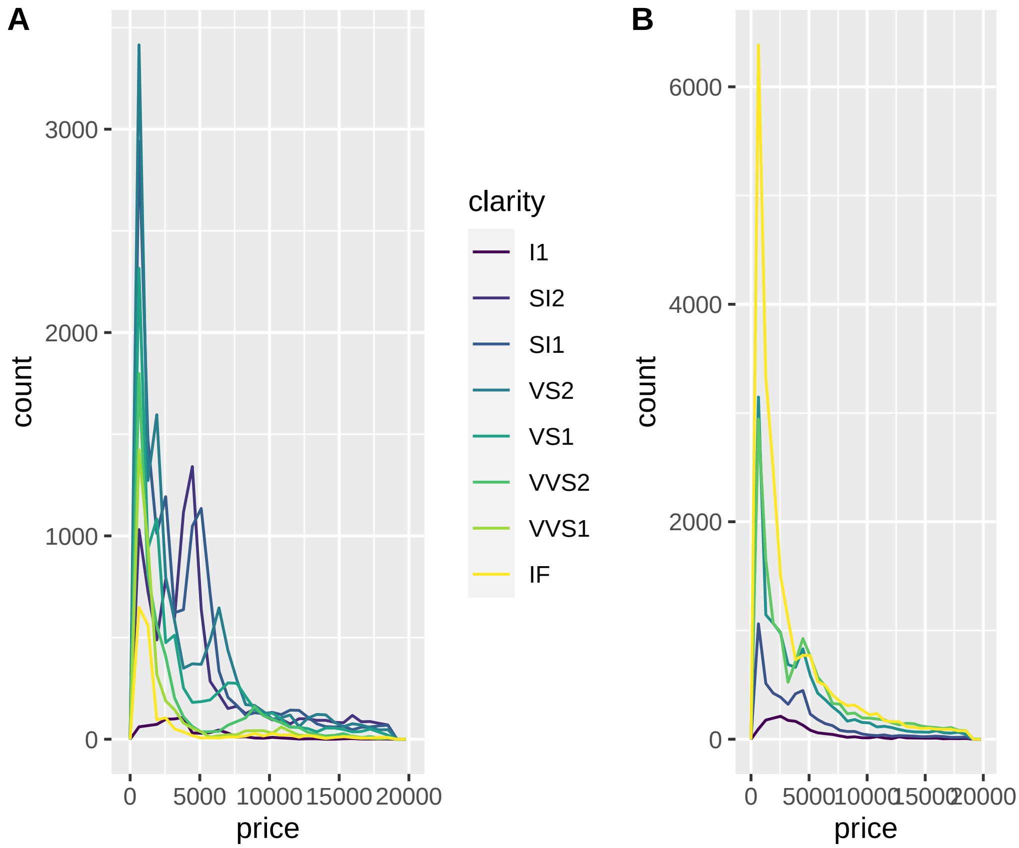
Covariation tells us it two variables tend to change together. For this we'll use the `diamonds` dataset. During our session I was using `geom_point` but I should have used (its easier to see with) `geom_freqpoly`.

```
p3 <- ggplot(diamonds, aes(price, color = clarity))+
  geom_freqpoly()

p4 <- ggplot(diamonds, aes(price, color = cut))+
  geom_freqpoly()

cowplot::plot_grid(p3, p4, labels = c("A", "B"), label_size = 12)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Here we can say the plot B has a higher covariation since its much easier to tell that the price of the diamonds changes with the cut.