MSCS 264: Homework #8

Due Fri, March 10

Al Ashir Intisar

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Part 1: Spotify

This dataset includes information about over 30,000 songs on spotify. We will investigate the relationship of various characteristics to the genre of the song. The variable `playlist_genre` indicates the genre associated with the playlist the song appears in.

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

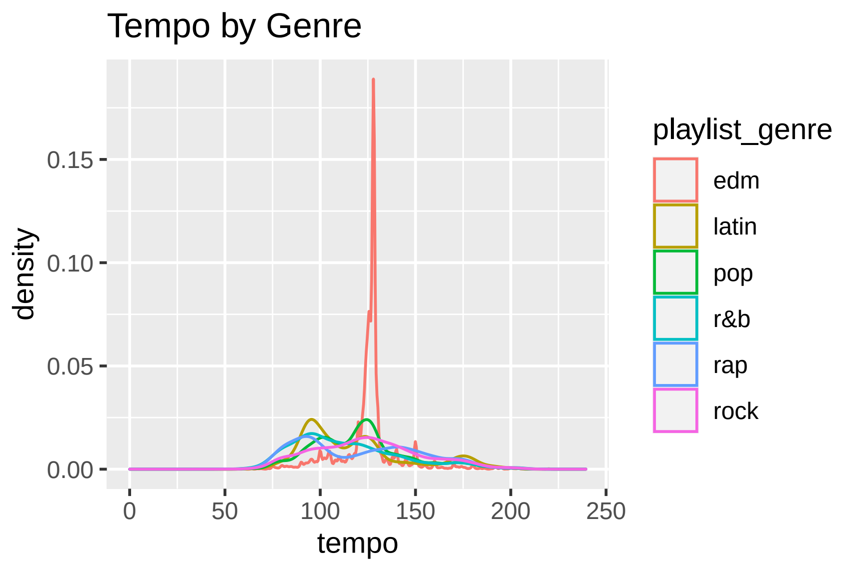
```
## Rows: 32833 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): track_id, track_name, track_artist, track_album_id, track_album_na...
## dbl (13): track_popularity, danceability, energy, key, loudness, mode, speec...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
bigspotify
```

```
## # A tibble: 32,833 x 23
##    track_id      track~1 track~2 track~3 track~4 track~5 track~6 playl~7 playl~8
##    <chr>         <chr>   <chr>     <dbl> <chr>   <chr>   <chr>   <chr>   <chr>
##  1 6f807x0ima9a~ I Don'~ Ed She~      66 2oCs0D~ I Don'~ 2019-0~ Pop Re~ 37i9dQ~
##  2 0r7CVbZTWZgb~ Memori~ Maroon~      67 63rPSO~ Memori~ 2019-1~ Pop Re~ 37i9dQ~
##  3 1z1Hg7Vb0AhH~ All th~ Zara L~      70 1HoSmj~ All th~ 2019-0~ Pop Re~ 37i9dQ~
##  4 75FpbthrwQmz~ Call Y~ The Ch~      60 1nqYsO~ Call Y~ 2019-0~ Pop Re~ 37i9dQ~
##  5 1e8PAfcKUYoK~ Someon~ Lewis ~      69 7m7vv9~ Someon~ 2019-0~ Pop Re~ 37i9dQ~
##  6 7fvUMiyapMsR~ Beauti~ Ed She~      67 2yiy9c~ Beauti~ 2019-0~ Pop Re~ 37i9dQ~
##  7 2OAylPUDDfwR~ Never ~ Katy P~      62 7INHYS~ Never ~ 2019-0~ Pop Re~ 37i9dQ~
##  8 6b1RNvAcJjQH~ Post M~ Sam Fe~      69 6703SR~ Post M~ 2019-0~ Pop Re~ 37i9dQ~
##  9 7bF6tCO3gFb8~ Tough ~ Avicii       68 7CvAfG~ Tough ~ 2019-0~ Pop Re~ 37i9dQ~
## 10 1IXGILkPm0tO~ If I C~ Shawn ~      67 4Qxzbf~ If I C~ 2019-0~ Pop Re~ 37i9dQ~
## # ... with 32,823 more rows, 14 more variables: playlist_genre <chr>,
## #   playlist_subgenre <chr>, danceability <dbl>, energy <dbl>, key <dbl>,
## #   loudness <dbl>, mode <dbl>, speechiness <dbl>, acousticness <dbl>,
## #   instrumentalness <dbl>, liveness <dbl>, valence <dbl>, tempo <dbl>,
## #   duration_ms <dbl>, and abbreviated variable names 1: track_name,
## #   2: track_artist, 3: track_popularity, 4: track_album_id,
## #   5: track_album_name, 6: track_album_release_date, 7: playlist_name, ...
## # i Use 'print(n = ...)' to see more rows, and 'colnames()' to see all variable names
```

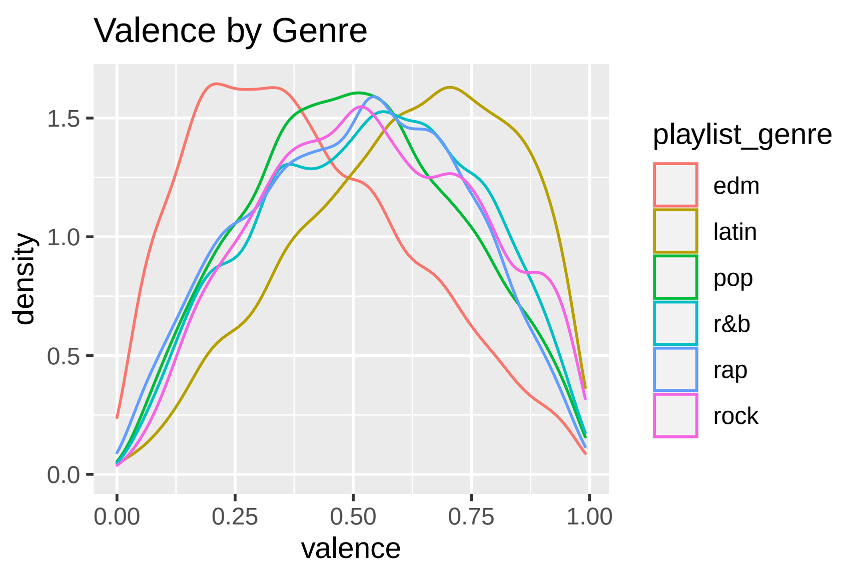
1. The three plots below examine the characteristics `tempo` (the speed of the beat of the song, higher is faster),`valence` (the emotionality … higher is more positive, lower is more negative), and `energy` (higher means the song feels more active and intense).

```r
ggplot(bigspotify) +
  geom_density(aes(x = tempo, color = playlist_genre)) +
  labs(title = "Tempo by Genre")
```



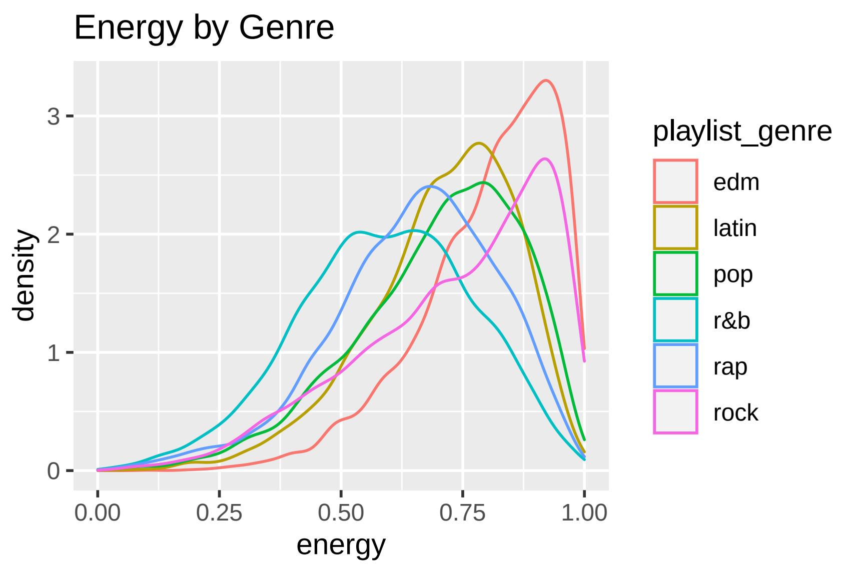
Ans: From the graph we can see that the edm has highest center and least spread and the shape is more like a bell curve. The other songs have almost similar spread and center but has two bumps (not like a bell curve).

```r
ggplot(bigspotify) +
  geom_density(aes(x = valence, color = playlist_genre))+
  labs(title = "Valence by Genre")
```



Ans: From this we can see that edm has the lowest center and latin has the highest center. The rest of the genres have similar center. All the genres have similar spread but edm and latin are not so bell curve like.

```
ggplot(bigspotify) +
  geom_density(aes(x = energy, color = playlist_genre))+
  labs(title = "Energy by Genre")
```



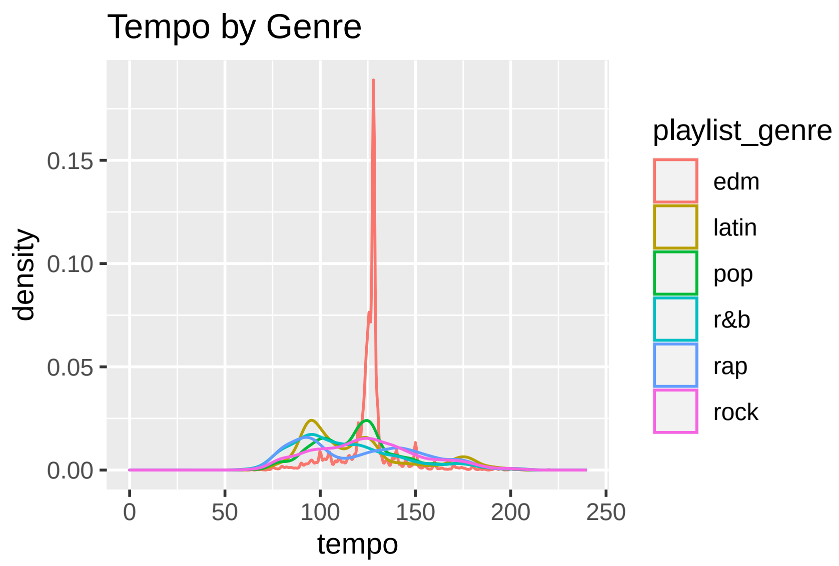
Ans: From the graph above we can see that edm and rock has the highest center and edm has probably the lowest spread. Th rest of the genres look similar with somewhat bell curve like shape.

Write 1-2 sentences for each plot, describing what each tells us about how characteristics are related to genres. Be sure to mention things like shape, center and spread!

2. Choose one of the three characteristics plotted in #1 (tempo, valence, energy), and create a different type of visualization (e.g. a different geom_) which allows you to compare the distribution of that characteristic across the different genres. Write 1 - 2 sentences of the pros and cons of each type of

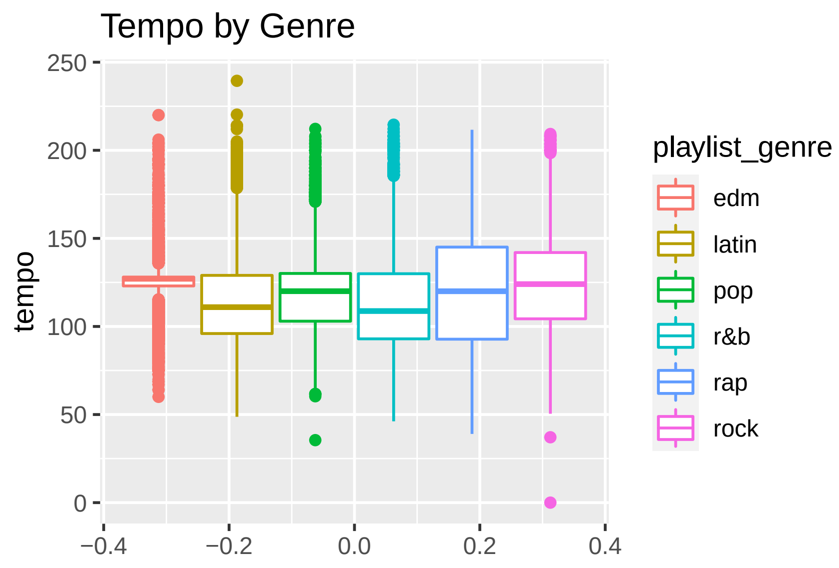
visualization.

```r
ggplot(bigspotify) +
  geom_density(aes(x = tempo, color = playlist_genre)) +
  labs(title = "Tempo by Genre")
```



Ans: This graph helps us to see the spread of the data better but only gives a idea about the median value/center.

```r
ggplot(bigspotify) +
  geom_boxplot(aes(y = tempo, color = playlist_genre)) +
  labs(title = "Tempo by Genre")
```

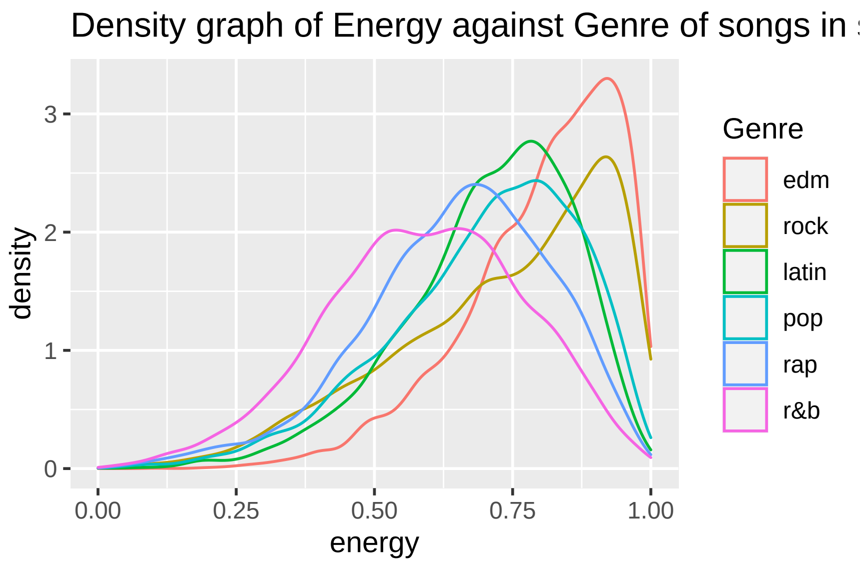


Ans: Whereas the geom_boxplot helps us visualize the median value in a better was and the outliers. But it is not as good as the geom_density for visualizing the skewedness of the distribution.

3. Consider the density graph for energy in #1. Copy and modify the code so that the legend order is by median energy level, with the highest energy listed first and lowest energy last. (e.g. edm should be

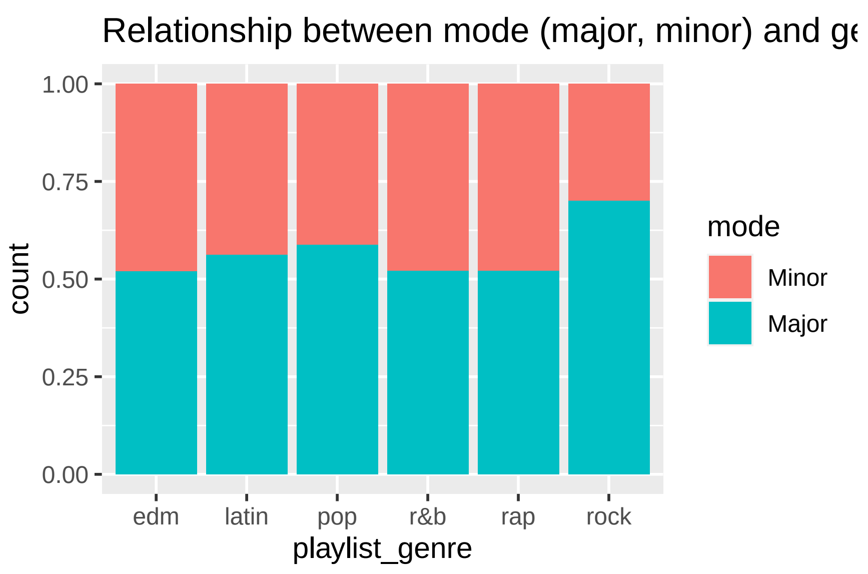
the first genre listed, and r&b the last genre). Change the legend label to say "Genre" and give the plot a meaningful title. *For full credit, use an fct_ function OTHER THAN manually releveling wtih fct_relevel.*

```
bigspotify|>
  mutate(playlist_genre = fct_reorder(playlist_genre, energy, .fun = median, .desc = TRUE))|>
  ggplot() +
  geom_density(aes(x = energy, color = playlist_genre))+
  labs(title = "Density graph of Energy against Genre of songs in spotify playlist", color = "Genre")
```



4. Create a graph that shows the relationship between mode (major, minor) and genre. Write 1-2 sentences describing what the plot shows.

```
bigspotify|>
  mutate(mode = as.factor(mode))|>
  mutate(mode = fct_recode(mode, "Minor" = "0", "Major" = "1"))|> #assuming 1 is major and 0 is minor
  ggplot() +
  geom_bar(mapping = aes(playlist_genre, fill = mode),  position = "fill")+
  labs(title = "Relationship between mode (major, minor) and genre.")
```

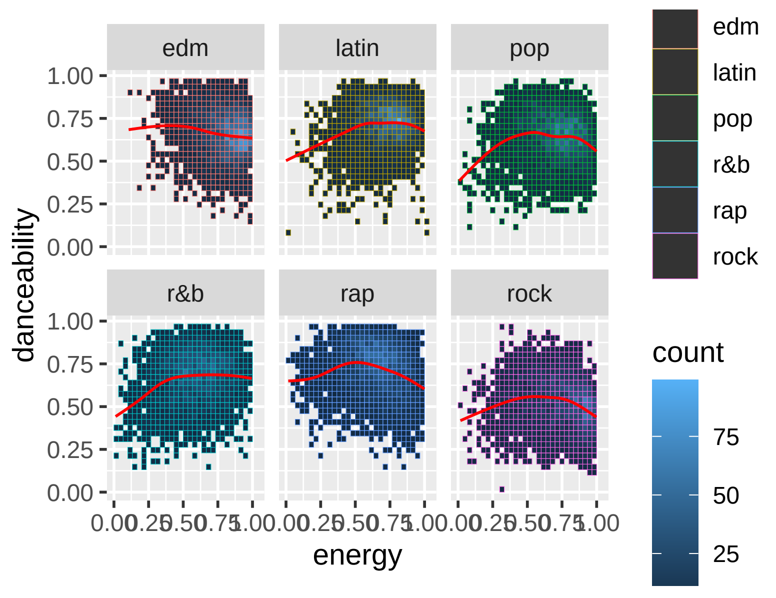


Ans: This geom_bar graph shows what portion of each genre songs is either minor or major in terms of mode. I assumed the value 1 for mode is major and value 0 for mode is minor.

5. Modify the set of scatterplots below to make them easier to read. Write 1-2 sentences describing what the plot shows. (See EDA_CheatSheet or EDA_pros_and_cons Rmd files for ideas)

```
ggplot(bigspotify, aes(x = energy, y = danceability, color = playlist_genre)) +
  geom_bin2d() +
  geom_smooth(se = FALSE, color = "red", size = 0.5) +
  facet_wrap(~playlist_genre)
```

```
## ‘geom_smooth()‘ using method = ’gam’ and formula ’y ~ s(x, bs = "cs")’
```



Ans: The graph above uses geom_bin2d to show the overlapping of the points in a better way by using grids with certain number of counts represented by certain shade of the color blue.

Part 2: Look what you've learned!

Run this code to read in the basketball data

```
library(rvest)
```

```
##
## Attaching package: 'rvest'

## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
url <- glue::glue("http://www.basketball-reference.com/leagues/NBA_2022_totals.html")

bball <- read_html(url) %>%
  html_nodes("#totals_stats") %>%
  html_table() %>%
  data.frame(check.names = FALSE) %>%
  as_tibble() %>%
  mutate(across(G:PTS, parse_number),
         points_per_minute = PTS/MP,
         Pos = str_sub(Pos, 1,2),
         Pos = str_replace(Pos, "-", "")) %>%
  rename(FGpct = `FG%`,
         FTpct = `FT%`) %>%
  filter(MP > 10)
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number      G
##  50  -- a number      G
##  75  -- a number      G
## 104  -- a number      G
## 131  -- a number      G
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number     GS
##  50  -- a number     GS
##  75  -- a number     GS
## 104  -- a number     GS
## 131  -- a number     GS
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number     MP
##  50  -- a number     MP
##  75  -- a number     MP
## 104  -- a number     MP
## 131  -- a number     MP
## ... ... ........ ......
## See problems(...) for more details.
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number     FG
##  50  -- a number     FG
##  75  -- a number     FG
## 104  -- a number     FG
## 131  -- a number     FG
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    FGA
##  50  -- a number    FGA
##  75  -- a number    FGA
## 104  -- a number    FGA
## 131  -- a number    FGA
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    FG%
##  50  -- a number    FG%
##  75  -- a number    FG%
## 104  -- a number    FG%
## 131  -- a number    FG%
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number   eFG%
##  50  -- a number   eFG%
##  75  -- a number   eFG%
## 104  -- a number   eFG%
## 131  -- a number   eFG%
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number     FT
##  50  -- a number     FT
##  75  -- a number     FT
## 104  -- a number     FT
## 131  -- a number     FT
## ... ... ........ ......
## See problems(...) for more details.

## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    FTA
##  50  -- a number    FTA
##  75  -- a number    FTA
## 104  -- a number    FTA
## 131  -- a number    FTA
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    FT%
##  50  -- a number    FT%
##  75  -- a number    FT%
## 104  -- a number    FT%
## 131  -- a number    FT%
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    ORB
##  50  -- a number    ORB
##  75  -- a number    ORB
## 104  -- a number    ORB
## 131  -- a number    ORB
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    DRB
##  50  -- a number    DRB
##  75  -- a number    DRB
## 104  -- a number    DRB
## 131  -- a number    DRB
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    TRB
##  50  -- a number    TRB
##  75  -- a number    TRB
## 104  -- a number    TRB
## 131  -- a number    TRB
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    AST
##  50  -- a number    AST
##  75  -- a number    AST
## 104  -- a number    AST
## 131  -- a number    AST
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    STL
##  50  -- a number    STL
##  75  -- a number    STL
## 104  -- a number    STL
## 131  -- a number    STL
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    BLK
##  50  -- a number    BLK
##  75  -- a number    BLK
## 104  -- a number    BLK
## 131  -- a number    BLK
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    TOV
##  50  -- a number    TOV
##  75  -- a number    TOV
## 104  -- a number    TOV
## 131  -- a number    TOV
## ... ... ........ ......
## See problems(...) for more details.
```

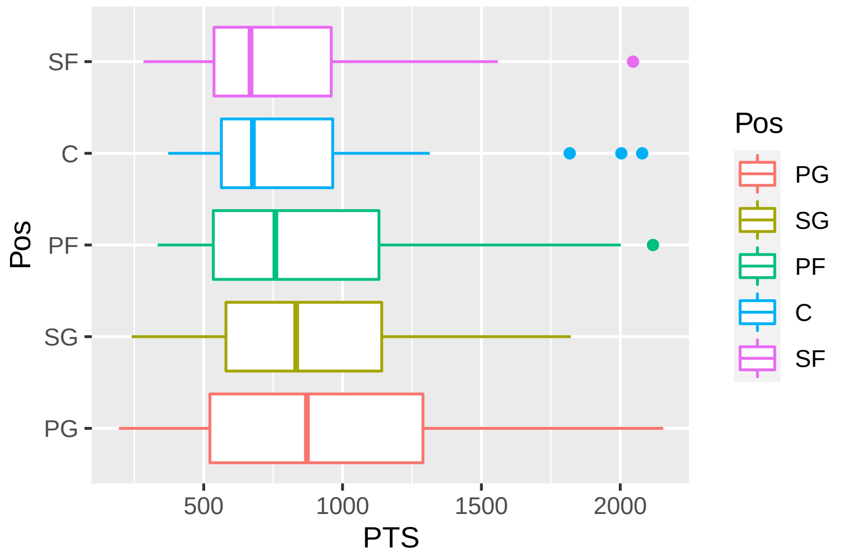
```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number     PF
##  50  -- a number     PF
##  75  -- a number     PF
## 104  -- a number     PF
## 131  -- a number     PF
## ... ... ........ ......
## See problems(...) for more details.
```

```
## Warning: 30 parsing failures.
## row col expected actual
##  27  -- a number    PTS
##  50  -- a number    PTS
##  75  -- a number    PTS
## 104  -- a number    PTS
## 131  -- a number    PTS
## ... ... ........ ......
## See problems(...) for more details.
```

6. On homework 3, we used the basketball data to make a boxplot of points scored by position. Modify the code below so that:

- You use pipes.
- The boxes are ordered by median points

```
bball|>
  filter(G >=58)|>
  mutate(Pos = fct_reorder(Pos, PTS, .fun = median, .desc = TRUE))|>
  ggplot(aes(x = PTS, y = Pos, color = Pos)) +
  geom_boxplot()
```



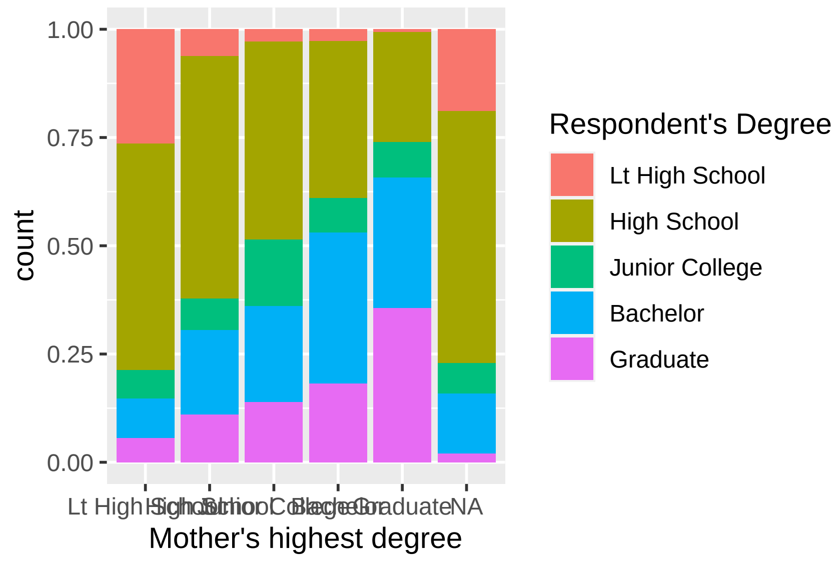
7. Also on homework 3, we use the General social survey data to create this bar chart.

Modify the code so that madeg and degree are in order: Lt High School, High School, Junior College, Bachelor, Graduate. Also drop any responses where degree is NA, but keep responses where madeg is NA.

```
gss_sm <- read_csv("~/Mscs 264 S23/Class/Data/gss_sm.csv")
```

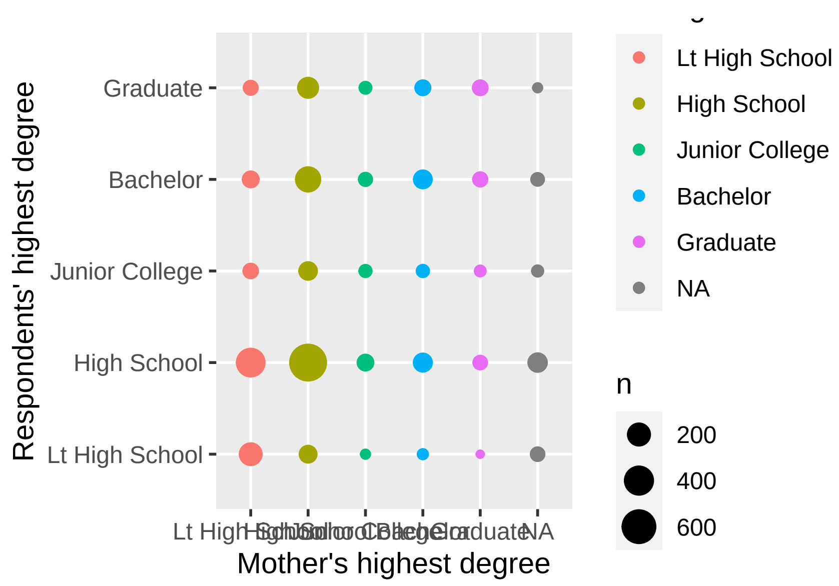
```
## Rows: 2867 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (23): degree, race, sex, region, income16, relig, marital, padeg, madeg,...
## dbl  (9): year, id, ballot, age, childs, sibs, pres12, wtssall, obama
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
gss_sm|>
  filter(!is.na(degree))|>
  mutate(madeg = fct_relevel(madeg, "Lt High School", "High School", "Junior College", "Bachelor", "Grad
  mutate(degree = fct_relevel(degree, "Lt High School", "High School", "Junior College", "Bachelor", "G
  ggplot(aes(x = madeg, fill = degree)) +
  geom_bar(position = "fill") +
  labs(x = "Mother's highest degree",
       fill = "Respondent's Degree")
```



8. Create an alternative plot type to show the relationship in 7. (See EDA_CheatSheet.Rmd in Class > Code for ideas). Write 1-2 sentences describing pros and cons of each plot type.

```
gss_sm|>
  filter(!is.na(degree))|>
  mutate(madeg = fct_relevel(madeg, "Lt High School", "High School", "Junior College", "Bachelor", "Grad
  mutate(degree = fct_relevel(degree, "Lt High School", "High School", "Junior College", "Bachelor", "G
  ggplot(aes(x = madeg, y = degree, color = madeg)) +
  geom_count() +
  labs(x = "Mother's highest degree", y = "Respondents' highest degree",
       fill = "Respondent's Degree")
```



Ans: The pros of the geom_bar is that it shows the proportion of the respondent's degree with respect to their mother's degrees very well. But it fails to whos the number of respondents that are available in each geoup. The pros for geom_count is that it shows the count of each group with respect to others and the somewhat the proportions as well in each degree groups of the respondents.