ggplot exercises

Task #1 Load Data

You are given the `**netlix_titles.csv**` file.

Load the dataframe into the working directory and file.

Inspect it quickly.

Task #1 Load Data - Solution

```r
file <- "netflix_titles.csv"
netflix_titles <- read_csv(
   file,
   col_names = TRUE,
   skip = 0
)
 glimpse(netflix_titles)
```

Task #2 Filter and mutate data

- Create `**tv_shows**` dataframe by filtering original dataframe by type, keeping only entries after year 2000, and by transforming release_year to a factor column;
- Create `**movies**` dataframe by filtering original dataframe by type, keeping only entries after year 2000, and by transforming release_year to a factor column;
- Create `**tv_shows_agg**` by aggregating `tv_shows` (from before) and counting number of shows by year;
- Create `**movies_agg**` by aggregating `movies` (from before) and counting number of movies by year.

Task #2 Filter and mutate data - Solution

```
tv_shows <- netflix_titles %>%
  filter (
    type == 'TV Show' &
    release_year > 2000
  ) %>%
  mutate(
    release_year = factor(release_year)
   )
```

Task #2 Filter and mutate data - Solution

```
movies <- netflix_titles %>%
  filter (
    type == 'Movie' &
      release_year > 2000
  ) %>%
  mutate(
    release_year = factor(release_year)
  )
```

Task #2 Filter and mutate data - Solution

```
tv_shows_agg <- tv_shows %>%
  dplyr::group_by(release_year) %>%
  summarise(n_of_tv_shows = n()) %>%
  ungroup()
```

Task #2 Filter and mutate data - Solution

```
movies_agg <- movies %>%
  dplyr::group_by(release_year) %>%
  summarise(n_of_tv_shows = n()) %>%
  ungroup()
```

Task #3 Simple Barplot

- Create a simple barplot using the `**tv_shows**` for displaying the number of tv shows per year.

Task #3 Simple Barplot - Solution

```
ggplot(
  data = tv_shows,
  aes(x = release_year, fill = release_year)
) +
geom_bar()
```

Task #4 Simple Barplot

- Create a simple barplot using the `**movies_agg**` for displaying the number of movies per year.

Task #4 Simple Barplot - Solution

```
ggplot(
  data = movies_agg,
  aes(x = release_year, y = n_of_movies, fill =
release_year)
) +
geom_bar(stat="identity")
```

Task #5 Barplot

Create a barplot using the `**movies**` dataframe:
- Rotate the plot by 90 degrees
- Remove the legend
- Add custom text for x and y axes
- Add custom title
- Center the title
- Add custom ticks for y axis (from 0 to 800, by 50)

Task #5 Barplot - Solution

```
ggplot(
  movies,
  aes (x = release_year, fill = release_year)
) +
  geom_bar() +
  coord_flip() +
  theme(legend.position="none") +
  xlab("Year") + ylab("Number of movies") +
  ggtitle("Number of Neflix movies since 2000, by year") +
  theme(
    plot.title = element_text(hjust = 0.5)
  ) +
  scale_y_continuous(breaks = seq(0, 800, 50))
```

Task #6 Stacked Barplot

Create a stacked barplot using the `**tv_shows**` dataframe, stacking the shows by rating:

- Transform data by filtering null values for `**rating**` and mutating it to factor type
- Rotate the plot by 90 degrees
- Add custom labels for x and y axes
- Add a custom title
- Rotate the ticks on the bottom (flipped!) by 45 degrees
- Put the legend on the bottom

Task #6 Stacked Barplot - Solution

```
ggplot(
  data = tv_shows %>%
    filter(!is.na(rating)) %>%
    mutate(rating = factor(rating)),
  aes(x = release_year, fill = rating)
) +
  geom_bar() +
  coord_flip() +
  xlab("Year") + ylab("Number of TV shows") +
  ggtitle("Structure of the TV shows") +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)
  ) +
  theme(legend.position="bottom")
```

Task #7 Side-by-side Barplot

Adapt the plot from Task #6 by putting columns side by side instead of staking them

- Keep the original rotation

7. Side-by-side Barplot - Solution

```
ggplot(
  data = tv_shows %>%
    filter(!is.na(rating)) %>%
    mutate(rating = factor(rating)),
  aes(x = release_year, fill = rating)
) +
  geom_bar(position=position_dodge()) +
  xlab("Year") + ylab("Number of TV shows") +
  ggtitle("Structure of the TV shows") +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)
  ) +
  theme(legend.position="bottom")
```

Task #8 Side-by-side Barplot, Faceted

Adapt the plot from Task #7 by displaying the plots in a faceted manner instead of putting them side by side

Task #8 Side-by-side Barplot, Faceted - Solution

```
ggplot(
  data = tv_shows %>%
    filter(!is.na(rating)) %>%
    mutate(rating = factor(rating)),
  aes(x = release_year, fill = rating)
) +
  geom_bar(position = position_dodge()) +
  coord_flip() +
  theme_bw() +
  xlab("Year") + ylab("Number of TV shows") +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1 )
  ) +
  ggtitle("Rating of tv shows\nBy Year") +
  facet_wrap( ~ rating) +
  theme(legend.position="none")
```

Task #9 Data Filtering for better visuals

```
filtered_movies <- movies %>%
  filter(
    as.numeric(as.character(release_year)) > 2010 &
      rating %in% c("TV-14", "TV-MA", "TV-PG", "TV-Y", "TV-Y7") &
      !is.na(rating) &
      !is.na(listed_in)
  ) %>%
  mutate(
    rating = factor(rating),
    genre = factor(str_split(listed_in, ",") %>% sapply(`[`, 1))
  ) %>%
  filter(
    genre %in% c("Dramas", "Comedies", "Horror Movies",
"Documentaries", "Thrillers")
  )
```

Task #9 Barplot for 3 variables, wrap

Using the `**filtered_movies**` dataframe, create a barplot that will display number of movies based on three columns: year, rating and genre:

- Choose another theme
- Add custom x and y axis labels
- Rotate the x axis ticks by 45 degrees
- Add a custom title
- Using a **facet wrap**, add both the genre and rating
- Remove the legend

Task #9 Barplot for 3 variables, wrap - Solution

```
ggplot(
  data = filtered_movies,
  aes(x = release_year, fill = rating)
) +
  geom_bar(position = position_dodge()) +
  theme_bw() +
  xlab("Year") + ylab("Number of movies") +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1 )
  ) +
  ggtitle("Rating of movies\nBy Year") +
  facet_wrap( genre ~ rating, labeller = label_both) +
  theme(legend.position="none")
```

Task #10 Barplot for 3 variables, grid

Using the `**filtered_movies**` dataframe, create a barplot that will display number of movies based on three columns: year, rating and genre:

- Choose another theme
- Add custom x and y axis labels
- Rotate the x axis ticks by 45 degrees
- Add a custom title
- Using a **facet grid**, add both the genre and rating
- Remove the legend

Task #10 Barplot for 3 variables, grid - Solution

```
ggplot(
  data = filtered_movies,
  aes(x = release_year, fill = rating)
) +
  geom_bar(position = position_dodge()) +
  coord_flip() +
  xlab("Year") + ylab("Number of Movies") +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1 )
  ) +
  ggtitle("Rating of movies\nBy Year") +
  facet_grid( genre ~ rating, labeller = label_both) +
  theme(legend.position="none")
```

Task #11 Load video games data

Download and load video games dataset into the current working directory.

Task #11 Load video games data - Solution

```
file <- "vgsales.csv"
video_game_sales <- read_csv(file, col_names = TRUE, skip = 0)
glimpse(video_game_sales)
table(video_game_sales$Platform)
```

Task #12 Simple histogram

Create a histogram for `**NA_Sales**` for “Sony Computer Entertainment” Publisher:

- Add a custom title
- Use a different color for bins outline

Task #12 Simple histogram - Solution

```
ggplot(
  video_game_sales %>%
    filter (
      !is.na(NA_Sales) &
      Publisher == "Sony Computer Entertainment"
      ),
  aes(x = NA_Sales)
) +
  geom_histogram(color = "white") +
  ggtitle("Histogram of NA Sales for Sony Copmuter Entertainment")
```

Task #13 Customized histogram

Create a histogram for `**EU_Sales**` for “Fighting” Genre:

- Add a custom title
- Use a different color for bins outline
- Change color for bins
- Add opacity to bins
- Change the bin width to cover intervals of 0.2
- Add a dotted red line that will represent the mean of the data
-

Task #13 Customized histogram - Solution

```
ggplot(
  video_game_sales %>%
    filter ( !is.na(EU_Sales) & Genre == "Fighting" ),
  aes(x = EU_Sales)
) +
  geom_histogram(fill="lightblue",binwidth = 0.2,alpha = .6) +
  geom_vline(
    aes (xintercept = mean(EU_Sales, na.rm=T)),
    color="red",
    linetype="dashed",
    size= .5
  ) +
  ggtitle("Histogram of EU Sales for Fighting genre")
```

Task #14 Customized histogram

Create a histogram for `**Global_Sales**` for “Electronic Arts” Publisher:

- Add a custom title
- Add a custom text for x axis
- Use a different color for bins outline
- Change color for bins
- Add opacity to bins
- Add a dotted red line that will represent the mean of the data
- Add a line that will replicate the density estimation of the data

Task #14 Customized histogram - Solution

```
ggplot(
  video_game_sales %>%
    filter (!is.na(Global_Sales) & Publisher == "Electronic Arts"),
  aes(x = Global_Sales)
) +
  geom_histogram(
     fill="coral", colour="white",
     (aes(y = after_stat(density)), alpha = .5
  ) +
  geom_density(alpha = .5) +
  geom_vline(
    aes (xintercept = mean(Global_Sales, na.rm=T)),
    color="red", linetype="dashed", size=.5
  ) +
  ggtitle("Superimposed Histogram and Density Curve of sales") +
  xlab("sales in millions")
```

Task #15 Customized histogram

Create two super imposed histograms for `**NA_Sales**` and `**EU_Sales**` for “Activision” Publisher:

- Use red color for NA_Sales
- Use yellow color for EU_Sales
- Use opacity for bins
- Add custom title
- Add custom x axis label

Task #15 Customized histogram - Solution

```
ggplot(
  video_game_sales %>%
    filter (
      !is.na(NA_Sales) & !is.na(EU_Sales) & Publisher == "Activision"
    ),
  aes(x = NA_Sales)
) +    # global `aes`; used for first histogram
  geom_histogram(col = "red", fill = "red", alpha = 0.4) +
  geom_histogram(
    aes(x = EU_Sales),
    col = "yellow", fill = "yellow", alpha = 0.4
  ) +
  xlab("sales in millions") +
  ggtitle("Superimosed Histograms of NA Sales (red) and EU Sales (yellow) \nVideo 
Game Sales")
```

Task #16 Customized histogram

Create two super imposed histograms for `**NA_Sales**` and `**EU_Sales**` for “Activision” Publisher:

- Prior to that - transform data from wide to long format!
- Use opacity for bins
- Add custom title
- Add custom x axis label

Task #16 Customized histogram - Solution

```
eu_na_long <- video_game_sales %>%
  filter (
    !is.na(EU_Sales) & !is.na(NA_Sales) & Publisher == "Activision"
  ) %>%
  mutate (row_num = row_number()) %>%
  select (row_num, NA_Sales, EU_Sales) %>%
  gather (parameter, value, -row_num)

ggplot(
  eu_na_long,
  aes(x = value, fill = parameter, col = parameter)
) +
  geom_histogram(alpha = .4) +
  xlab("sales in millions") +
  ggtitle("Superimosed Histograms of NA Sales and EU Sales \nActivison Video Game
Sales")
```

Task #17 Customized histogram, facet

On the long data format from Task #16, use facet wrap to move from two histograms on one plot to 2 plots for each of the histograms.

Task #17 Customized histogram, facet - Solution

```
ggplot(
  eu_na_long,
  aes(x = value, fill = parameter, col = parameter)
) +
geom_histogram( alpha = 0.5) +
  xlab("sales in millions") +
  ggtitle("Superimosed Histograms of NA Sales and EU Sales \nActivison Video Game
Sales") +
  facet_wrap(~ parameter) +
  theme(legend.position = "none")
```

Task #18 Customized density plot

Create two super imposed density plots for `**EU_Sales**` and `**Other_Sales**` for “Racing” Genre:

- Use red color for Eu_Sales
- Use yellow color for Other_Sales
- Use opacity for bins
- Add custom title
- Add custom x axis label

Task #18 Customized density plot - Solution

```
ggplot(
  video_game_sales %>%
    filter (
      !is.na(Other_Sales) & !is.na(EU_Sales) & Genre == "Racing"
    ),
  aes(x = EU_Sales)
) +
  geom_density(col = "red", fill = "red", alpha = 0.4) +
  geom_density(
    aes(x = Other_Sales),
    col = "yellow", fill = "yellow", alpha = 0.4
  ) +
  xlab("sales in millions") +
  ggtitle("Superimosed Histograms of Other Sales (red) and EU Sales (yellow)
\nVideo Game Sales")
```

Task #19 Customized density plot

Create two super imposed density plots for `**EU_Sales**` and `**Other_Sales**` for “Racing” Genre:

- Prior to that - transform data from wide to long format!
- Use opacity for bins
- Add custom title
- Add custom x axis label

Task #19 Customized density plot - Solution

```
eu_other_long <- video_game_sales %>%
  filter (
    !is.na(Other_Sales) & !is.na(EU_Sales) & Genre == "Racing"
  ) %>%
  mutate (row_num = row_number()) %>%
  select (row_num, EU_Sales, Other_Sales) %>%
  gather (parameter, value, -row_num)
ggplot(
  eu_other_long,
  aes(x = value, fill = parameter, col = parameter)
) +
  geom_histogram(alpha = .4) +
  xlab("sales in millions") +
  ggtitle("Superimosed Histograms of EU Sales and Other Sales \nRacing Genre
Video Game Sales")
```

Task #20 Customized density plot

On the long data format from Task #19, use facet wrap to move from two density plots on one plot to 2 plots for each of the densities.

Task #20 Customized density plot - Solution

```
ggplot(
  eu_other_long,
  aes(x = value, fill = parameter, col = parameter)
) +
  geom_histogram( alpha = 0.5) +
  xlab("sales in millions") +
  ggtitle("Superimosed Histograms of Eu Sales and Other Sales \nRacing Genre
Video Game Sales") +
  facet_wrap(~ parameter) +
  theme(legend.position = "none")
```

Task #21 Customized boxplot

Filter data by “Shooter” Genre and “PS4” Platform. Create a boxplot of `**Global_Sales**`:

- Add a custom title
- Add custom text for y axis
- Remove x axis and text for x axis

Task #21 Customized boxplot - Solution

```
ggplot(
  video_game_sales %>%
    filter (!is.na(EU_Sales) & Genre == "Shooter" & Platform == "PS4"),
  aes(x = 0, y = EU_Sales)
) +
  geom_boxplot() +
  ggtitle("Boxplot of Global Sales") +
  ylab("sales in millions") + xlab("") +
  theme(axis.text.x = element_blank())
```

Task #22 Customized boxplots

- Create three boxplots for `**NA_Sales**`, `**EU_Sales**` and `**Other_Sales**`

Task #22 Customized boxplots - Solution

```
ggplot(
  video_game_sales %>%
    filter (
      !is.na(NA_Sales) &
      !is.na(EU_Sales) &
      !is.na(Other_Sales)
    ),
  aes(x = 'NA_Sales', y = NA_Sales)
) +  # global `aes`; used for first boxplot
  geom_boxplot() + # this box takes the settings from the global `aes`
  geom_boxplot(aes(x = 'EU_Sales', y = EU_Sales)) +  #  own `aes`
  geom_boxplot(aes(x = 'Other Sales', y = Other_Sales)) + #  own `aes`
  ylab("sales in millions") + xlab("region") +
  ggtitle("NA, EU, and Other region Sales")
```

Task #23 Correlation plot

Create a simple mixed correlation plot

Task #23 Correlation plot - Solution

```
video_game_sales %>%
  select (NA_Sales:Global_Sales) %>%
  filter(complete.cases(.)) %>% # remove all observations with NA values
  cor(.) %>%
  corrplot.mixed(., number.cex=0.75, tl.cex=0.6 )
```

Task #24 Correlation plot

Create a simple number correlation plot ot bottom part only.

Task #24 Correlation plot - Solution

```
video_game_sales %>%
  select (NA_Sales:Global_Sales) %>%
  filter(complete.cases(.)) %>% # remove all observations with NA values
  cor(.) %>%
  corrplot(., method = 'number', type = 'lower', number.cex=0.75, tl.cex=0.6)
```