Analysis of Worldwide Video Games Sales

Alejandro Arellano

4/30/2021

```r
#Reading Data File
vgsales  <- read.csv("C:/Users/alex0/Desktop/Stat 495 - R/STAT 495 WD/vgsales.csv")
glimpse(vgsales)
```

```
## Rows: 16,598
## Columns: 11
## $ Rank         <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17~
## $ Name         <chr> "Wii Sports", "Super Mario Bros.", "Mario Kart Wii", "Wii~
## $ Platform     <chr> "Wii", "NES", "Wii", "Wii", "GB", "GB", "DS", "Wii", "Wii~
## $ Year         <chr> "2006", "1985", "2008", "2009", "1996", "1989", "2006", "~
## $ Genre        <chr> "Sports", "Platform", "Racing", "Sports", "Role-Playing",~
## $ Publisher    <chr> "Nintendo", "Nintendo", "Nintendo", "Nintendo", "Nintendo~
## $ NA_Sales     <dbl> 41.49, 29.08, 15.85, 15.75, 11.27, 23.20, 11.38, 14.03, 1~
## $ EU_Sales     <dbl> 29.02, 3.58, 12.88, 11.01, 8.89, 2.26, 9.23, 9.20, 7.06, ~
## $ JP_Sales     <dbl> 3.77, 6.81, 3.79, 3.28, 10.22, 4.22, 6.50, 2.93, 4.70, 0.~
## $ Other_Sales  <dbl> 8.46, 0.77, 3.31, 2.96, 1.00, 0.58, 2.90, 2.85, 2.26, 0.4~
## $ Global_Sales <dbl> 82.74, 40.24, 35.82, 33.00, 31.37, 30.26, 30.01, 29.02, 2~
```

```r
#Getting rid of invalid observations
vgsales <- vgsales[!(vgsales$Year %in% c("N/A", "2017", "2020")),]
```

```r
# #Printing first 10 row
head(vgsales, 10 )
```

```
##    Rank                      Name Platform Year        Genre Publisher NA_Sales
## 1     1                Wii Sports      Wii 2006       Sports  Nintendo    41.49
## 2     2         Super Mario Bros.      NES 1985     Platform  Nintendo    29.08
## 3     3            Mario Kart Wii      Wii 2008       Racing  Nintendo    15.85
## 4     4         Wii Sports Resort      Wii 2009       Sports  Nintendo    15.75
## 5     5  Pokemon Red/Pokemon Blue       GB 1996 Role-Playing  Nintendo    11.27
## 6     6                    Tetris       GB 1989       Puzzle  Nintendo    23.20
## 7     7     New Super Mario Bros.       DS 2006     Platform  Nintendo    11.38
## 8     8                  Wii Play      Wii 2006         Misc  Nintendo    14.03
## 9     9 New Super Mario Bros. Wii      Wii 2009     Platform  Nintendo    14.59
## 10   10                 Duck Hunt      NES 1984      Shooter  Nintendo    26.93
##    EU_Sales JP_Sales Other_Sales Global_Sales
## 1     29.02     3.77        8.46        82.74
## 2      3.58     6.81        0.77        40.24
## 3     12.88     3.79        3.31        35.82
## 4     11.01     3.28        2.96        33.00
## 5      8.89    10.22         1.00         31.37
## 6      2.26     4.22         0.58         30.26
## 7      9.23     6.50         2.90         30.01
## 8      9.20     2.93         2.85         29.02
## 9      7.06     4.70         2.26         28.62
## 10     0.63     0.28         0.47         28.31
```

```
#Creating summary stats
summary(vgsales)
```

```
##       Rank           Name             Platform             Year          
##  Min.   :    1   Length:16323       Length:16323       Length:16323      
##  1st Qu.: 4136   Class :character   Class :character   Class :character  
##  Median : 8294   Mode  :character   Mode  :character   Mode  :character  
##  Mean   : 8292                                                           
##  3rd Qu.:12440                                                           
##  Max.   :16600                                                           
##     Genre            Publisher            NA_Sales          EU_Sales      
##  Length:16323       Length:16323       Min.   : 0.0000   Min.   : 0.0000  
##  Class :character   Class :character   1st Qu.: 0.0000   1st Qu.: 0.0000  
##  Mode  :character   Mode  :character   Median : 0.0800   Median : 0.0200  
##                                        Mean   : 0.2655   Mean   : 0.1476  
##                                        3rd Qu.: 0.2400   3rd Qu.: 0.1100  
##                                        Max.   :41.4900   Max.   :29.0200  
##     JP_Sales        Other_Sales       Global_Sales   
##  Min.   : 0.00000   Min.   : 0.00000   Min.   : 0.0100  
##  1st Qu.: 0.00000   1st Qu.: 0.00000   1st Qu.: 0.0600  
##  Median : 0.00000   Median : 0.01000   Median : 0.1700  
##  Mean   : 0.07868   Mean   : 0.04834   Mean   : 0.5403  
##  3rd Qu.: 0.04000   3rd Qu.: 0.04000   3rd Qu.: 0.4800  
##  Max.   :10.22000   Max.   :10.57000   Max.   :82.7400  
```

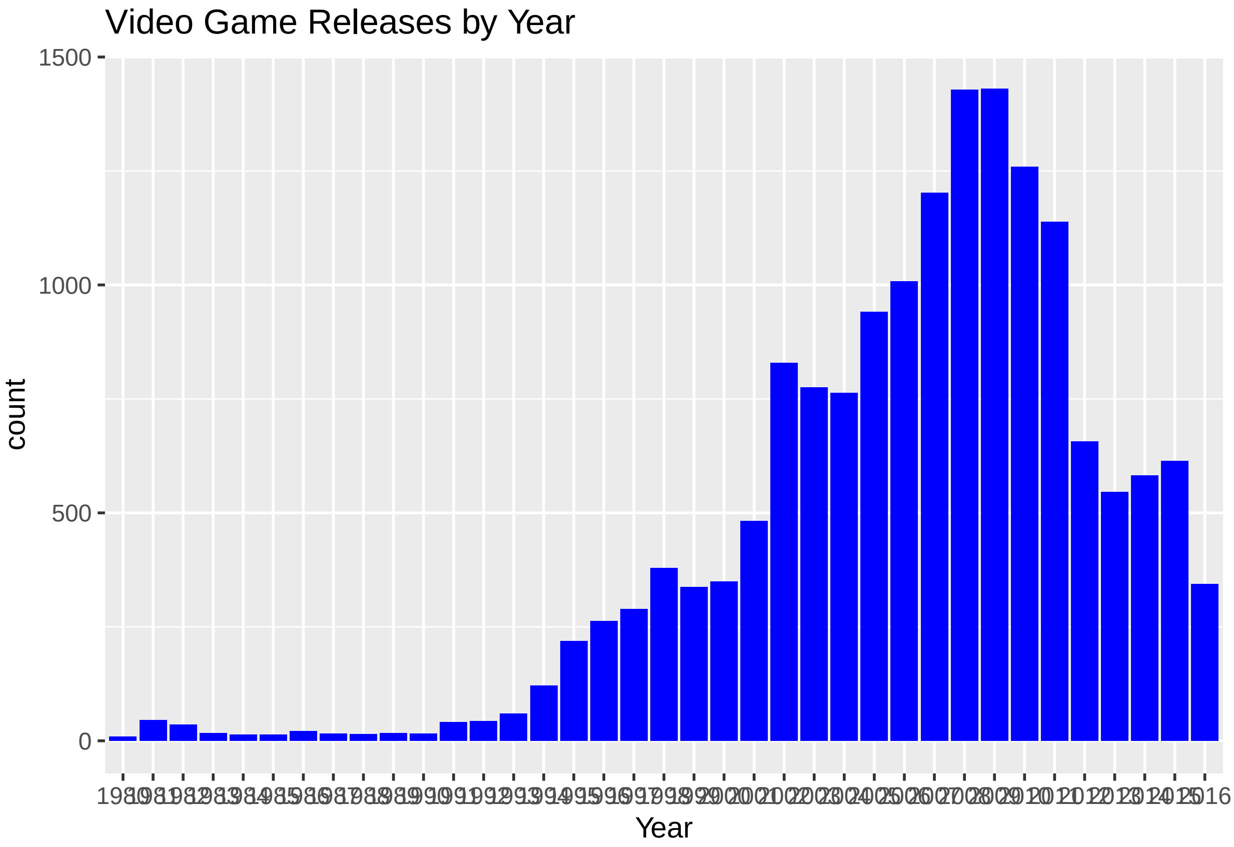
```
glimpse(vgsales)
```

```
## Rows: 16,323
## Columns: 11
## $ Rank         <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17~
## $ Name         <chr> "Wii Sports", "Super Mario Bros.", "Mario Kart Wii", "Wii~
## $ Platform     <chr> "Wii", "NES", "Wii", "Wii", "GB", "GB", "DS", "Wii", "Wii~
## $ Year         <chr> "2006", "1985", "2008", "2009", "1996", "1989", "2006", "~
## $ Genre        <chr> "Sports", "Platform", "Racing", "Sports", "Role-Playing",~
## $ Publisher    <chr> "Nintendo", "Nintendo", "Nintendo", "Nintendo", "Nintendo~
## $ NA_Sales     <dbl> 41.49, 29.08, 15.85, 15.75, 11.27, 23.20, 11.38, 14.03, 1~
## $ EU_Sales     <dbl> 29.02, 3.58, 12.88, 11.01, 8.89, 2.26, 9.23, 9.20, 7.06, ~
## $ JP_Sales     <dbl> 3.77, 6.81, 3.79, 3.28, 10.22, 4.22, 6.50, 2.93, 4.70, 0.~
## $ Other_Sales  <dbl> 8.46, 0.77, 3.31, 2.96, 1.00, 0.58, 2.90, 2.85, 2.26, 0.4~
## $ Global_Sales <dbl> 82.74, 40.24, 35.82, 33.00, 31.37, 30.26, 30.01, 29.02, 2~
```

```
#Summary stats for numeric variables
vgsales_summary <- vgsales %>%
  select (.,NA_Sales,EU_Sales,JP_Sales,Other_Sales,Global_Sales) %>%
  describe(.)
vgsales_summary
```

```
##              vars     n mean   sd median trimmed  mad  min   max range  skew
## NA_Sales        1 16323 0.27 0.82   0.08    0.13 0.12 0.00 41.49 41.49 18.75
## EU_Sales        2 16323 0.15 0.51   0.02    0.06 0.03 0.00 29.02 29.02 18.79
## JP_Sales        3 16323 0.08 0.31   0.00    0.02 0.00 0.00 10.22 10.22 11.13
## Other_Sales     4 16323 0.05 0.19   0.01    0.02 0.01 0.00 10.57 10.57 24.12
## Global_Sales    5 16323 0.54 1.57   0.17    0.28 0.21 0.01 82.74 82.73 17.32
##              kurtosis   se
## NA_Sales       643.73 0.01
## EU_Sales       747.49 0.00
## JP_Sales       191.48 0.00
## Other_Sales   1013.34 0.00
## Global_Sales   596.80 0.01
```

```
#Barplot for releases by year
ggplot(vgsales, aes(Year)) +
  geom_bar(fill = "blue") +
  ggtitle("Video Game Releases by Year")
```



```
#Table with year and number of releases sorted in descending order by releases
game_release_count <- vgsales %>%
  count(Year) %>%
  arrange(desc(n))
game_release_count
```

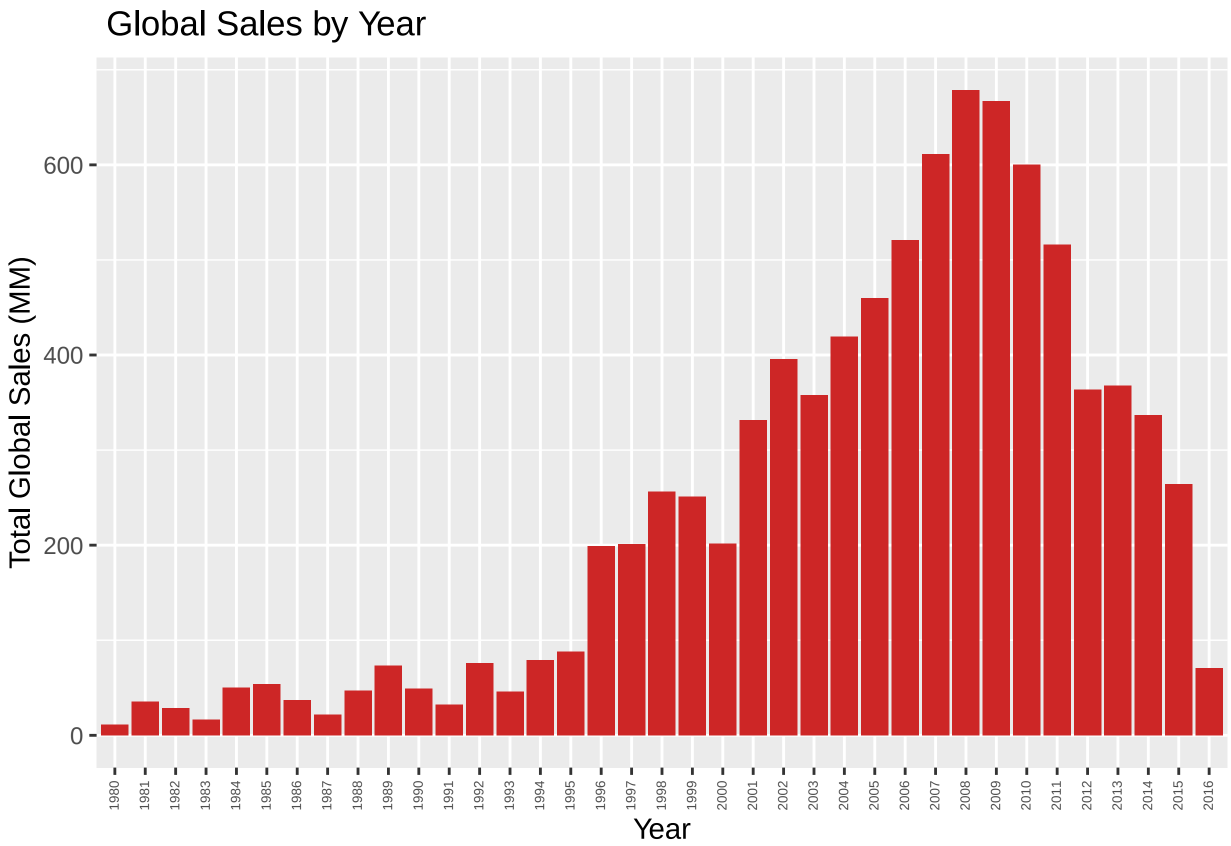
```
##     Year    n
## 1  2009 1431
## 2  2008 1428
## 3  2010 1259
## 4  2007 1202
## 5  2011 1139
## 6  2006 1008
## 7  2005  941
## 8  2002  829
## 9  2003  775
## 10 2004  763
## 11 2012  657
## 12 2015  614
## 13 2014  582
## 14 2013  546
## 15 2001  482
## 16 1998  379
## 17 2000  349
## 18 2016  344
## 19 1999  338
## 20 1997  289
## 21 1996  263
## 22 1995  219
## 23 1994  121
## 24 1993   60
## 25 1981   46
## 26 1992   43
## 27 1991   41
## 28 1982   36
## 29 1986   21
## 30 1983   17
## 31 1989   17
## 32 1987   16
## 33 1990   16
## 34 1988   15
## 35 1984   14
## 36 1985   14
## 37 1980    9
```

```r
#Sorting and arranging by years with the highest global sales
global_sales_by_year <- vgsales %>%
                    group_by(Year) %>%
                    summarize(total_global_sales = sum(Global_Sales)) %>%
                    arrange(desc(total_global_sales))
```

```r
global_sales_by_year
```

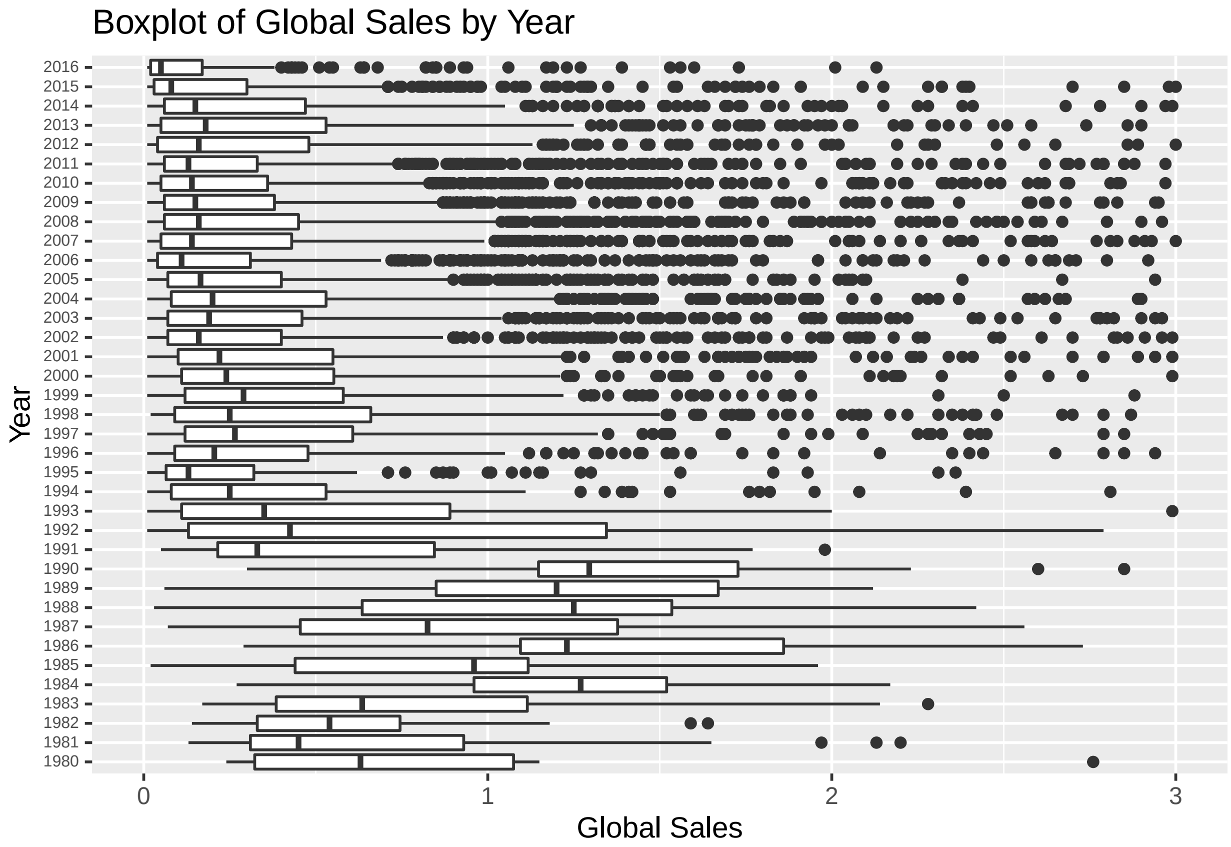
```
## # A tibble: 37 x 2
##    Year  total_global_sales
##    <chr>              <dbl>
##  1 2008                679.
##  2 2009                667.
##  3 2007                611.
##  4 2010                600.
##   5 2006                521.
##   6 2011                516.
##   7 2005                460.
##   8 2004                419.
##   9 2002                396.
##  10 2013                368.
## # ... with 27 more rows
```

```
#Barplot of years and their respective global sales
ggplot(global_sales_by_year, aes(Year, total_global_sales)) +
  geom_bar(fill = "firebrick3", stat = "identity") +
  ggtitle("Video Game Revenue by Year") +
  theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4)) +
  ggtitle(" Global Sales by Year") +
  ylab('Total Global Sales (MM)')
```



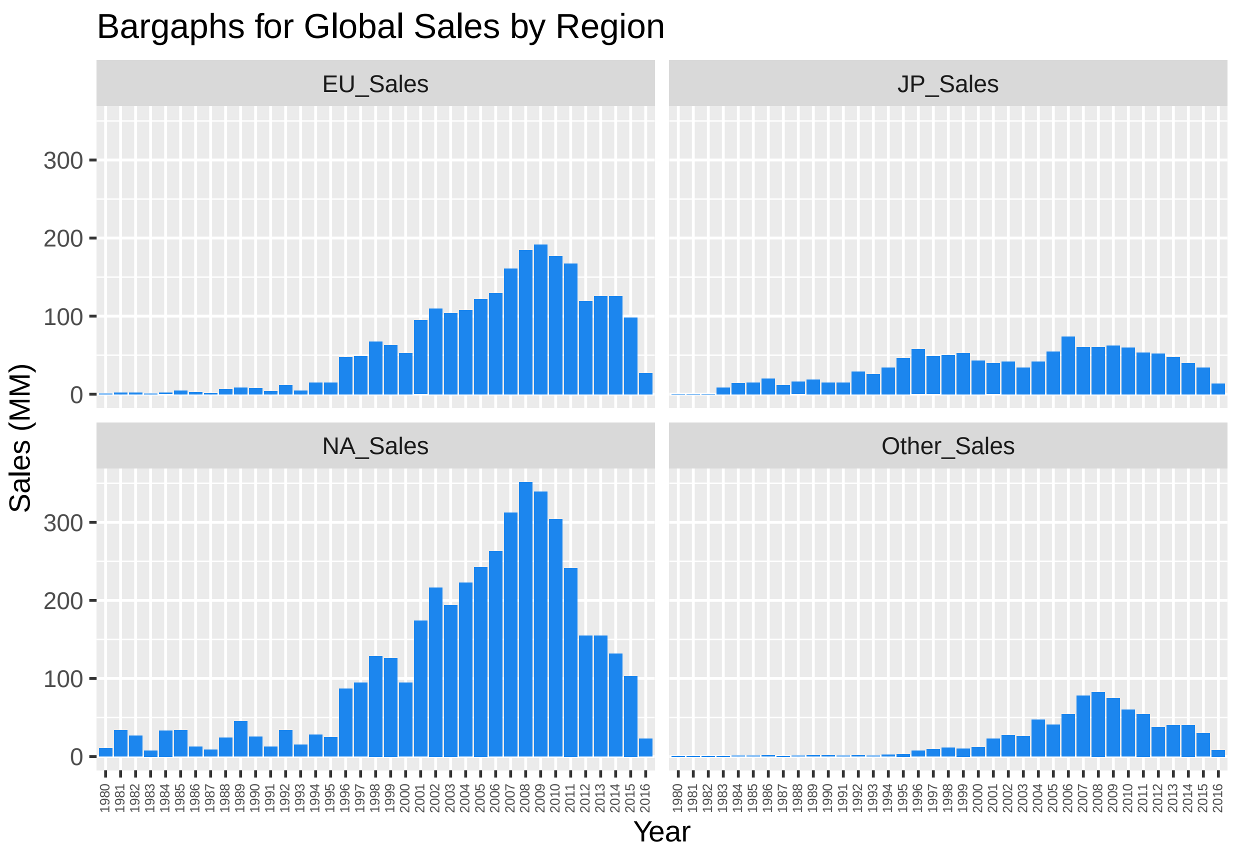
```
#Boxplot of global sales per year
ggplot(data = vgsales,
mapping = aes(x = factor(Year), y = Global_Sales)) +
geom_boxplot()+
theme(axis.text.y=element_text(angle=0, size = 6,vjust=0.4)) +
  ylim(0,3) +
  coord_flip() +
  xlab("Year") +
  ylab("Global Sales") +
  ggtitle("Boxplot of Global Sales by Year ")
```

```
## Warning: Removed 464 rows containing non-finite values (stat_boxplot).
```



```
#Faceted bar graph for sales per region
region_concatenated <- gather(vgsales, key="measure", value="value", c("NA_Sales","EU_Sales","JP_Sales"

ggplot(region_concatenated, aes(x= Year, y=value))+
  geom_bar(stat=’identity’, fill="dodgerblue2")+
  facet_wrap(~measure) +
  theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4)) +
  xlab("Year") +
  ylab("Sales (MM)") +
  ggtitle("Bargaphs for Global Sales by Region")
```



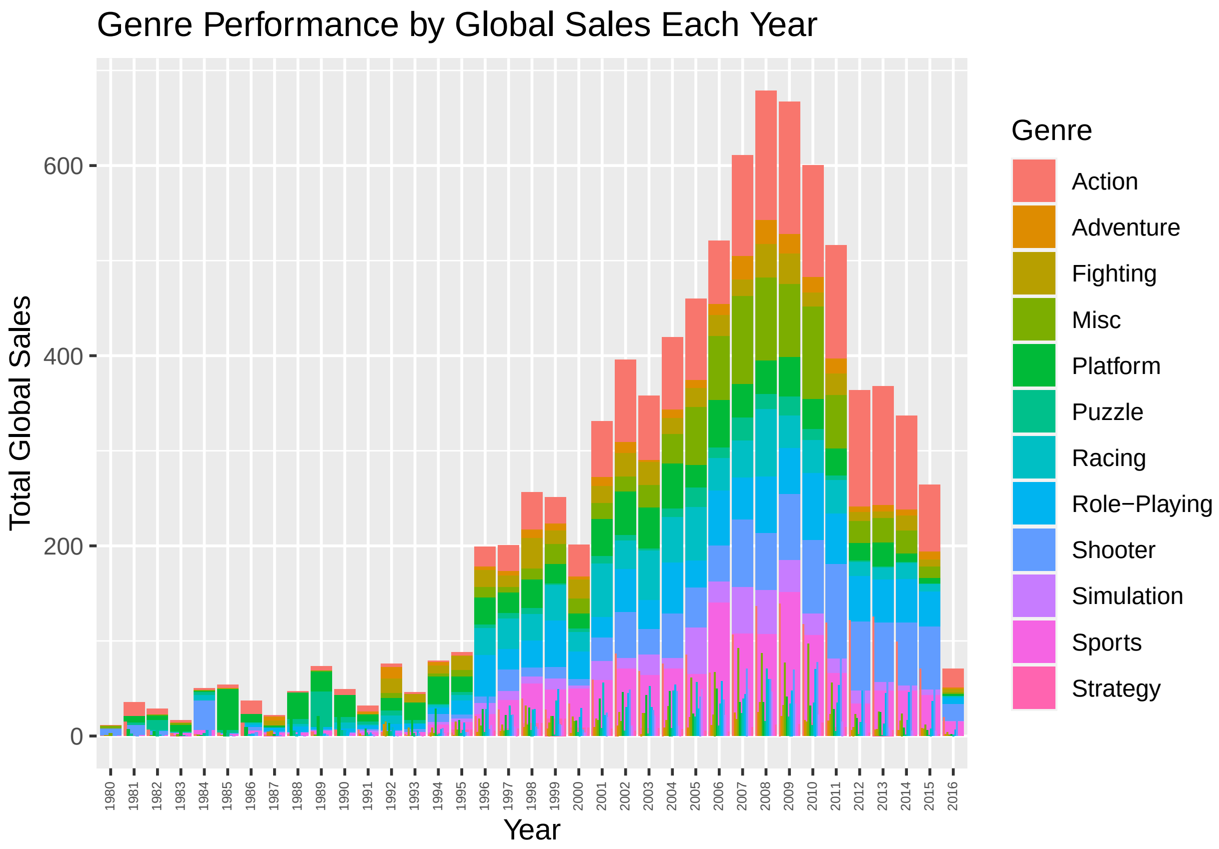
```
#Sorting by year and genre
sales_by_genre <- vgsales %>%
         group_by(Year, Genre) %>%
         summarize(total_global_sales = sum(Global_Sales))
```

```
## ‘summarise()‘ has grouped output by ’Year’. You can override using the ‘.groups‘ argument.
```

```
head(sales_by_genre)
```

```
## # A tibble: 6 x 3
## # Groups:   Year [2]
##   Year  Genre    total_global_sales
##   <chr> <chr>                 <dbl>
## 1 1980  Action                 0.34
## 2 1980  Fighting               0.77
## 3 1980  Misc                   2.71
## 4 1980  Shooter                7.07
## 5 1980  Sports                 0.49
## 6 1981  Action                14.8
```

```
#Barplot of erforamce of each genre per year
ggplot(sales_by_genre, aes(Year, total_global_sales, fill = Genre)) +
  geom_bar(stat = "identity") +
  geom_bar(position = ’dodge’, stat=’identity’) +
  ggtitle("Genre Performance by Global Sales Each Year") +
  ylab('Total Global Sales') +
  theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4))
```



```
#Sorting by year, genre, and showing the genre that was the most popular for that year
sales_by_genre <- vgsales %>%
         group_by(Year, Genre) %>%
         summarize(total_global_sales = sum(Global_Sales)) %>%
        arrange(desc(total_global_sales)) %>%
         top_n(1)
```

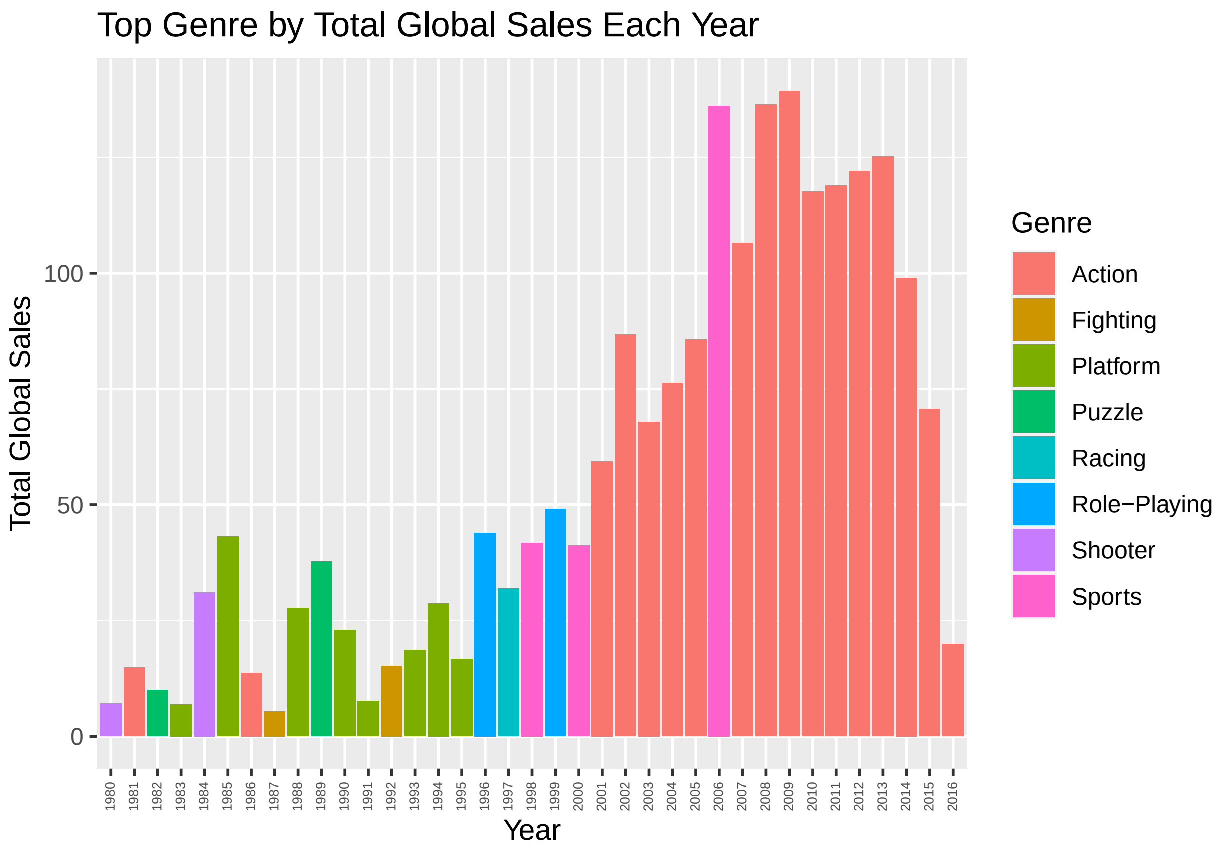
```
## ‘summarise()‘ has grouped output by ’Year’. You can override using the ‘.groups‘ argument.
```

```
## Selecting by total_global_sales
```

```
head(sales_by_genre)
```

```
## # A tibble: 6 x 3
## # Groups:   Year [6]
##   Year  Genre  total_global_sales
##   <chr> <chr>               <dbl>
## 1 2009  Action               139.
## 2 2008  Action               136.
## 3 2006  Sports                136.
## 4 2013  Action                125.
## 5 2012  Action                122.
## 6 2011  Action                119.
```

```
#Barplot of the most popular genre per year
ggplot(sales_by_genre, aes(Year, total_global_sales, fill = Genre)) +
  geom_bar(stat = "identity") +
  geom_bar(position = ’dodge’, stat=’identity’) +
  ggtitle("Top Genre by Total Global Sales Each Year") +
  ylab(’Total Global Sales’) +
  theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4))
```



```
#Sorting by year and platform and arranging by the most successful platform that year
top_platforms <- vgsales %>%
             group_by(Year, Platform) %>%
             summarize(total_global_sales = sum(Global_Sales)) %>%
             arrange(desc(total_global_sales)) %>%
            top_n(1)
```

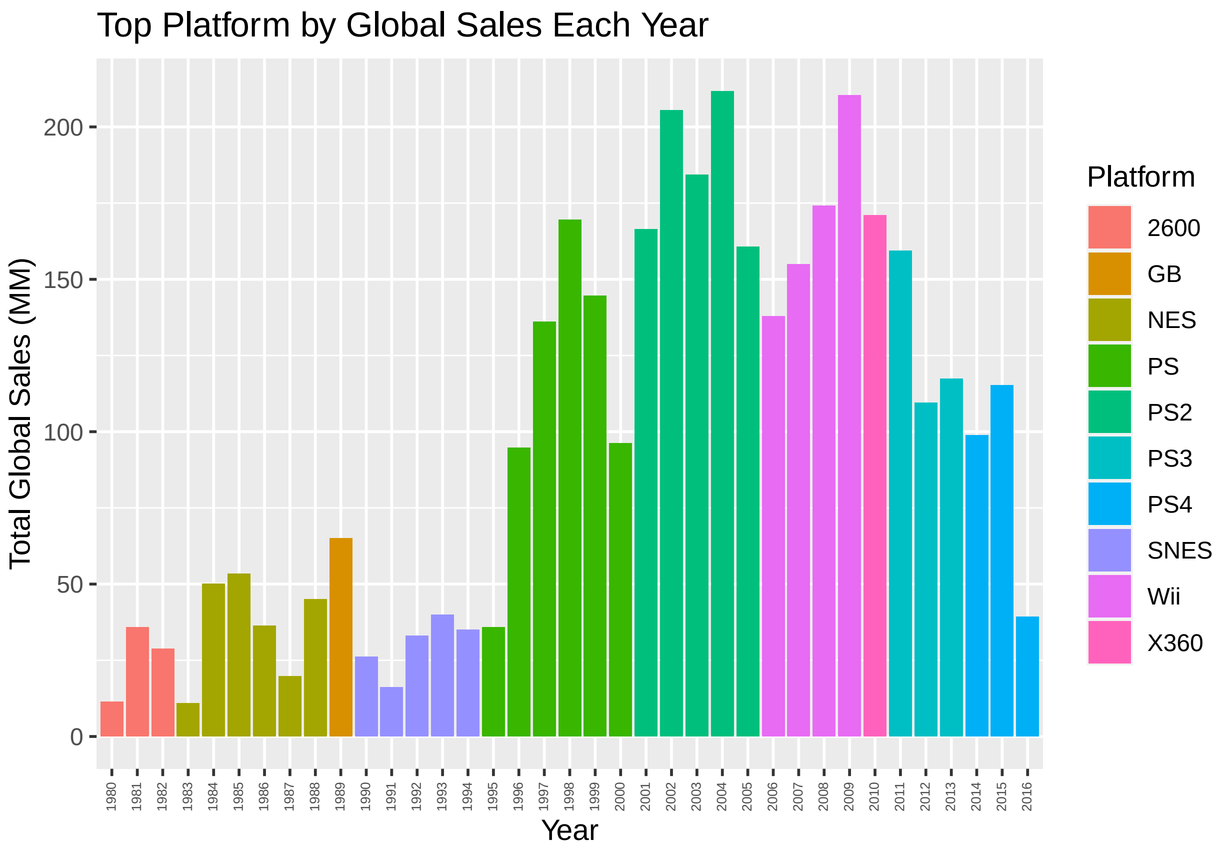
```
## ‘summarise()‘ has grouped output by ’Year’. You can override using the ‘.groups‘ argument.

## Selecting by total_global_sales
```

```r
top_platforms
```

```
## # A tibble: 37 x 3
## # Groups:   Year [37]
##    Year  Platform total_global_sales
##    <chr> <chr>                 <dbl>
##  1 2004  PS2                    212.
##  2 2009  Wii                    210.
##  3 2002  PS2                    205.
##  4 2003  PS2                    184.
##  5 2008  Wii                    174.
##  6 2010  X360                   171.
##  7 1998  PS                     170.
##  8 2001  PS2                    166.
##  9 2005  PS2                    161.
## 10 2011  PS3                    159.
## # ... with 27 more rows
```

```r
#Barplot of the most successful platform that year
ggplot(top_platforms, aes(Year, total_global_sales, fill = Platform)) +
  geom_bar(stat = "identity") +
  theme(legend.position = "right") +
  ggtitle("Top Platform by Global Sales Each Year") +
  ylab('Total Global Sales (MM)') +
  theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4))
```



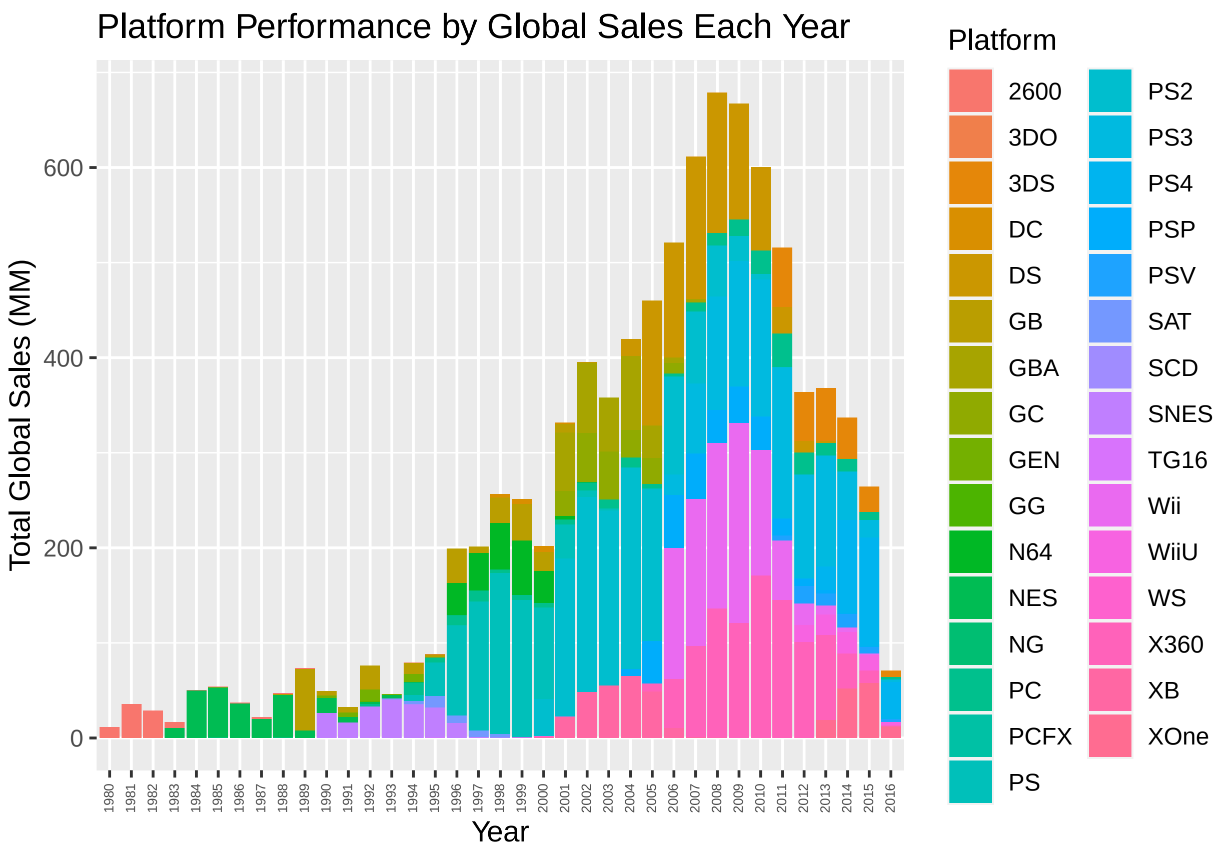
```
#Sorting by year and platform
top_platforms <- vgsales %>%
             group_by(Year, Platform) %>%
             summarize(total_global_sales = sum(Global_Sales)) %>%
             arrange(desc(total_global_sales))
```

```
## ‘summarise()‘ has grouped output by ’Year’. You can override using the ‘.groups‘ argument.
```

```
top_platforms
```

```
## # A tibble: 238 x 3
## # Groups:   Year [37]
##    Year  Platform total_global_sales
##    <chr> <chr>                 <dbl>
##  1 2004  PS2                    212.
##  2 2009  Wii                    210.
##  3 2002  PS2                    205.
##  4 2003  PS2                    184.
##  5 2008  Wii                    174.
##  6 2010  X360                   171.
##  7 1998  PS                     170.
##  8 2001  PS2                    166.
##  9 2005  PS2                    161.
## 10 2011  PS3                    159.
## # ... with 228 more rows
```

```
#Barplot of the performance of each platform per year
ggplot(top_platforms, aes(Year, total_global_sales, fill = Platform)) +
  geom_bar(stat = "identity") +
  theme(legend.position = "right") +
  ggtitle("Platform Performance by Global Sales Each Year") +
  ylab(’Total Global Sales (MM)’) +
  theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4))
```



```
#Bootstrap and CI for Action Genre

#Sorting for years with Action as the best selling genre
#Genre_bootstrap_dataframe <- vgsales %>%
  #group_by(Year, Genre) %>%
  #summarize(total_global_sales = sum(Global_Sales)) %>%
  #arrange(desc(total_global_sales)) %>%
  #top_n(1) %>%
  #filter(Genre=="Action")
#Genre_bootstrap_dataframe

#Sorting by Games that are in the action category
Genre_bootstrap_dataframe <- vgsales %>%
  group_by(Name,Year, Genre) %>%
  summarize(total_global_sales = sum(Global_Sales)) %>%
  arrange(desc(total_global_sales)) %>%
  filter(Genre=="Action")
```

```
## `summarise()` has grouped output by 'Name', 'Year'. You can override using the `.groups` argument.
```

```
Genre_bootstrap_dataframe
```

```
## # A tibble: 2,037 x 4
## # Groups:   Name, Year [2,037]
##    Name                              Year  Genre  total_global_sales
##    <chr>                              <chr> <chr>                 <dbl>
##  1 Grand Theft Auto V                 2013  Action                 37.8
##  2 Grand Theft Auto IV                2008  Action                 22.5
##  3 Grand Theft Auto: San Andreas      2004  Action                 20.8
##  4 Grand Theft Auto V                 2014  Action                 17.1
##  5 FIFA Soccer 13                     2012  Action                 16.2
##  6 Grand Theft Auto: Vice City        2002  Action                 16.2
##  7 LEGO Star Wars: The Complete Saga 2007  Action                 15.8
##  8 Assassin's Creed IV: Black Flag    2013  Action                 13.2
##  9 Assassin's Creed III               2012  Action                 13.1
## 10 Grand Theft Auto III               2001  Action                 13.1
## # ... with 2,027 more rows
```

```
#Specifying the formula we want
Genre_bootstrap_dataframe %>%
specify(formula = total_global_sales  ~ NULL)
```

```
## Response: total_global_sales (numeric)
## # A tibble: 2,037 x 1
##    total_global_sales
##                 <dbl>
##  1               37.8
##  2               22.5
##  3               20.8
##  4               17.1
##  5               16.2
##  6               16.2
##  7               15.8
##  8               13.2
##  9               13.1
## 10               13.1
## # ... with 2,027 more rows
```

```
#Setting seed and reps
set.seed(1)
Genre_bootstrap_dataframe %>%
specify(response = total_global_sales ) %>%
generate(reps = 2000, type = "bootstrap")
```

```
## Response: total_global_sales (numeric)
## # A tibble: 4,074,000 x 2
## # Groups:   replicate [2,000]
##    replicate total_global_sales
##        <int>              <dbl>
##  1         1               0.22
##  2         1               0.02
##  3         1               0.45
##  4         1               3.38
##  5         1               0.26
##  6         1               0.06
##  7         1               0.8
##  8         1               1.45
##  9         1               1.62
## 10          1                 0.01
## # ... with 4,073,990 more rows
```

```
#Creating bootstrap distribution mean
bootstrap_distribution_2000_mean <- Genre_bootstrap_dataframe %>%
specify(response = total_global_sales) %>%
generate(reps = 2000) %>%
calculate(stat = "mean")
```

```
## Setting ‘type = "bootstrap"‘ in ‘generate()‘.
```

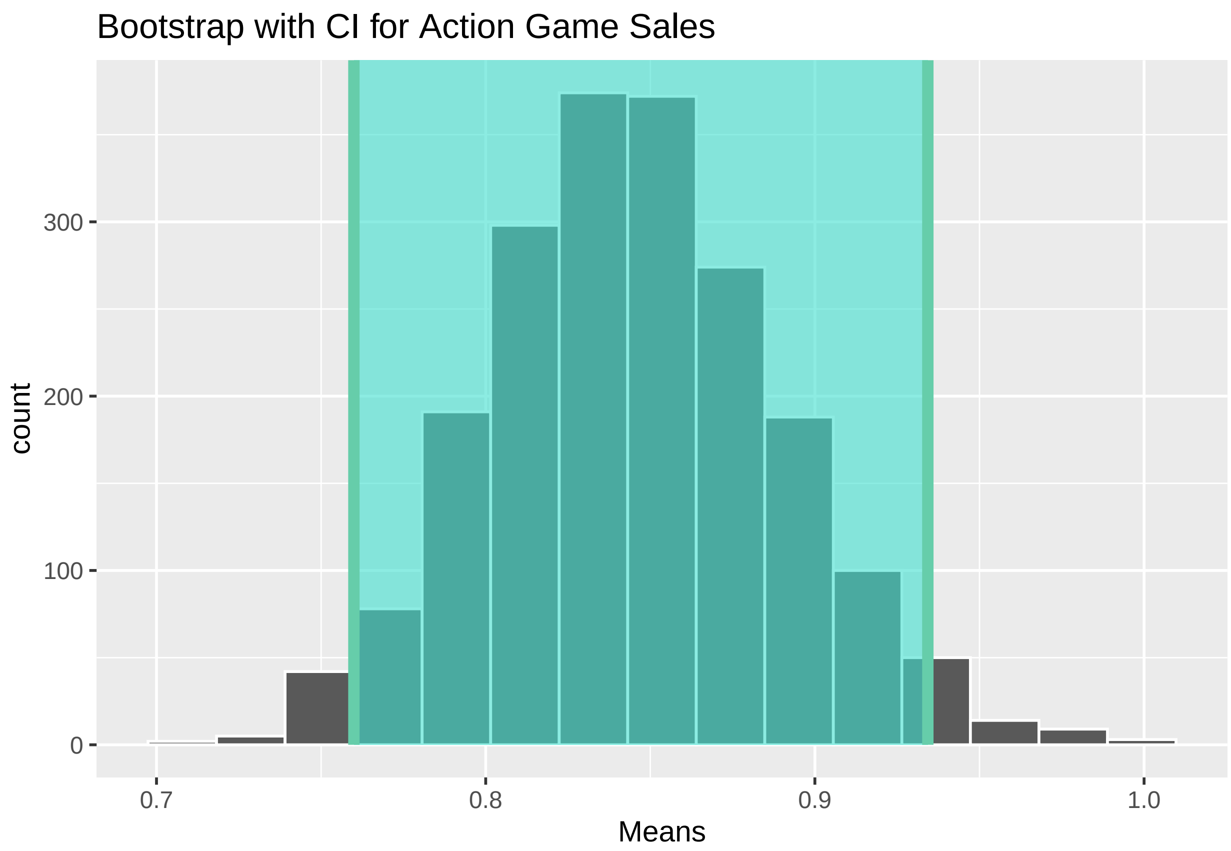
```
bootstrap_distribution_2000_mean
```

```
## Response: total_global_sales (numeric)
## # A tibble: 2,000 x 2
##    replicate  stat
##        <int> <dbl>
##  1         1 0.854
##  2         2 0.783
##  3         3 0.847
##  4         4 0.817
##  5         5 0.934
##  6         6 0.762
##  7         7 0.841
##  8         8 0.869
##  9         9 0.777
## 10        10 0.892
## # ... with 1,990 more rows
```

```
#Creating confidence interval
percentile_ci_2000 <- bootstrap_distribution_2000_mean %>%
get_confidence_interval(level = 0.95, type = "percentile")
percentile_ci_2000
```

```
## # A tibble: 1 x 2
##   lower_ci upper_ci
##      <dbl>    <dbl>
## 1    0.760    0.934
```

```
#visualizing bootstrap for 2000 replicates of the bootstrap
visualize(bootstrap_distribution_2000_mean) +
  shade_confidence_interval(endpoints = percentile_ci_2000) +
  ggtitle("Bootstrap with CI for Action Game Sales") +
  xlab('Means')
```



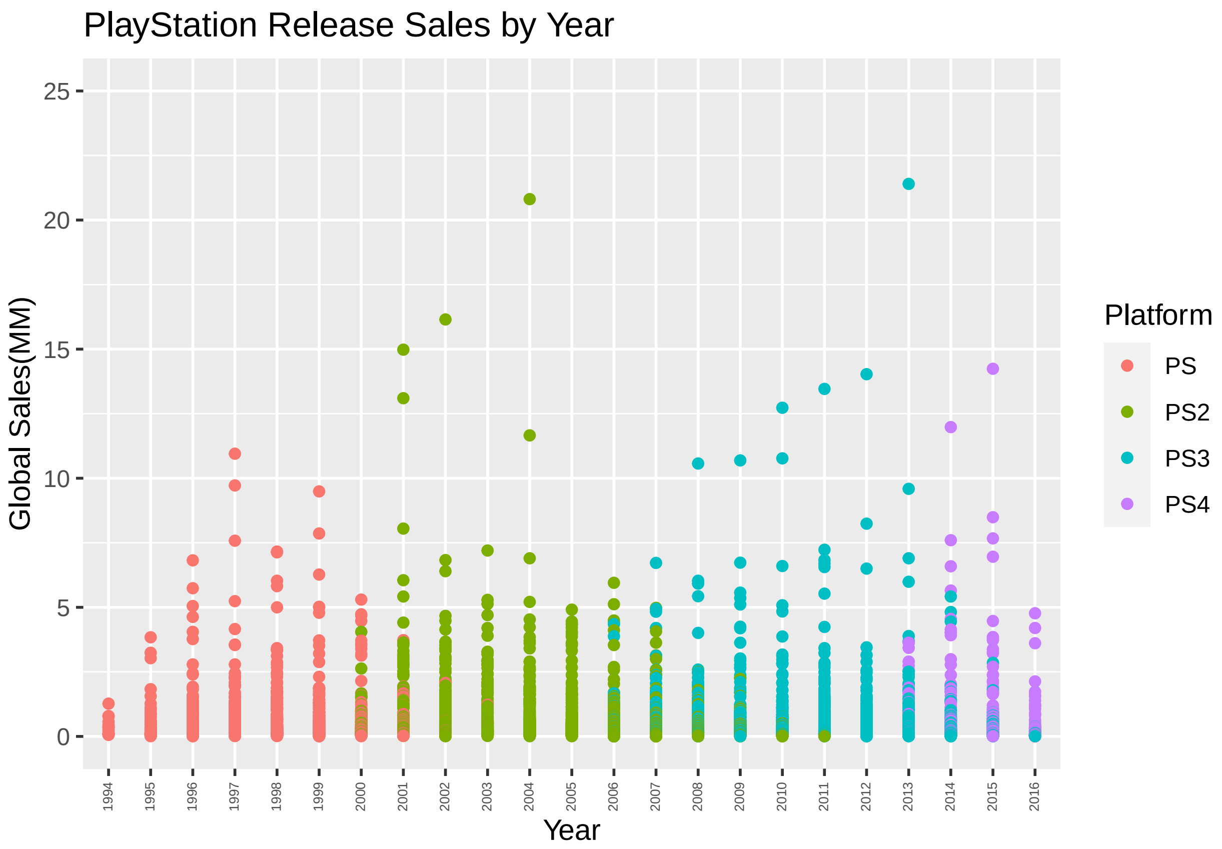
```
#Dotplot for game titles by platform
playstation_data<- vgsales %>%
             group_by(Name,Year, Platform) %>%
             summarize(total_global_sales = sum(Global_Sales)) %>%
             arrange(desc(total_global_sales)) %>%
              filter(Platform == ’PS’| Platform == ’PS2’|Platform == ’PS3’|Platform == ’PS4’)
```

```
## ‘summarise()‘ has grouped output by ’Name’, ’Year’. You can override using the ‘.groups‘ argument.
```

```
playstation_data
```

```
## # A tibble: 4,954 x 4
## # Groups:   Name, Year [4,638]
##    Name                           Year  Platform total_global_sales
##    <chr>                          <chr> <chr>                 <dbl>
##  1 Grand Theft Auto V             2013  PS3                    21.4
##  2 Grand Theft Auto: San Andreas  2004  PS2                    20.8
##  3 Grand Theft Auto: Vice City    2002  PS2                    16.2
##  4 Gran Turismo 3: A-Spec         2001  PS2                    15.0
##  5 Call of Duty: Black Ops 3      2015  PS4                    14.2
##  6 Call of Duty: Black Ops II     2012  PS3                    14.0
##  7 Call of Duty: Modern Warfare 3 2011  PS3                    13.5
##  8 Grand Theft Auto III           2001  PS2                    13.1
##  9 Call of Duty: Black Ops        2010  PS3                    12.7
## 10 Grand Theft Auto V             2014  PS4                    12.0
## # ... with 4,944 more rows
```

```
#Dotplot of game releases per PS console per year and global sales info
ggplot(playstation_data,
aes(x = Year, y = total_global_sales, color = Platform)) +
geom_point() +
  ggtitle('PlayStation Release Sales by Year')+
labs(x = "Year", y = "Global Sales(MM)", color = "Platform")  +
theme(axis.text.x=element_text(angle=90,size = 5,vjust=0.4))  +
ylim(0,25)
```



```
#------------------------------------------------------------------------------------------------
```

```
#Bootstrap for playstation data
playstation_data<- vgsales %>%
             group_by(Name,Year, Platform) %>%
             summarize(total_global_sales = sum(Global_Sales)) %>%
             arrange(desc(total_global_sales)) %>%
              filter(Platform == 'PS'| Platform == 'PS2'|Platform == 'PS3'|Platform == 'PS4')
```

```
## `summarise()` has grouped output by 'Name', 'Year'. You can override using the `.groups` argument.
```

```
playstation_data
```

```
## # A tibble: 4,954 x 4
## # Groups:   Name, Year [4,638]
##    Name                           Year  Platform total_global_sales
##    <chr>                          <chr> <chr>                 <dbl>
##  1 Grand Theft Auto V             2013  PS3                    21.4
##  2 Grand Theft Auto: San Andreas  2004  PS2                    20.8
##  3 Grand Theft Auto: Vice City    2002  PS2                    16.2
##  4 Gran Turismo 3: A-Spec         2001  PS2                    15.0
##  5 Call of Duty: Black Ops 3      2015  PS4                    14.2
##  6 Call of Duty: Black Ops II     2012  PS3                    14.0
##  7 Call of Duty: Modern Warfare 3 2011  PS3                    13.5
##  8 Grand Theft Auto III           2001  PS2                    13.1
##  9 Call of Duty: Black Ops        2010  PS3                    12.7
## 10 Grand Theft Auto V             2014  PS4                    12.0
## # ... with 4,944 more rows
```

```
#Specifying the formula we want
playstation_data %>%
specify(formula = total_global_sales ~ NULL)
```

```
## Response: total_global_sales (numeric)
## # A tibble: 4,954 x 1
##    total_global_sales
##                 <dbl>
##  1               21.4
##  2               20.8
##  3               16.2
##  4               15.0
##  5               14.2
##  6               14.0
##  7               13.5
##  8               13.1
##  9               12.7
## 10               12.0
## # ... with 4,944 more rows
```

```
#Setting seed and reps
set.seed(1)
playstation_data %>%
specify(response = total_global_sales ) %>%
generate(reps = 2000, type = "bootstrap")
```

```
## Response: total_global_sales (numeric)
## # A tibble: 9,908,000 x 2
## # Groups:   replicate [2,000]
##    replicate total_global_sales
##        <int>              <dbl>
##  1         1               0.82
##  2         1               0.02
##  3         1               0.32
##  4         1               0.52
##  5         1               0.03
##  6         1               0.28
##  7         1               2.39
##  8         1               0.06
##  9         1               0.13
## 10         1               0.06
## # ... with 9,907,990 more rows
```

```
#Creating bootstrap distribution mean
platform_bootstrap_distribution_2000_mean <- playstation_data %>%
specify(response = total_global_sales) %>%
generate(reps = 2000) %>%
calculate(stat = "mean")
```

```
## Setting `type = "bootstrap"` in `generate()`.
```

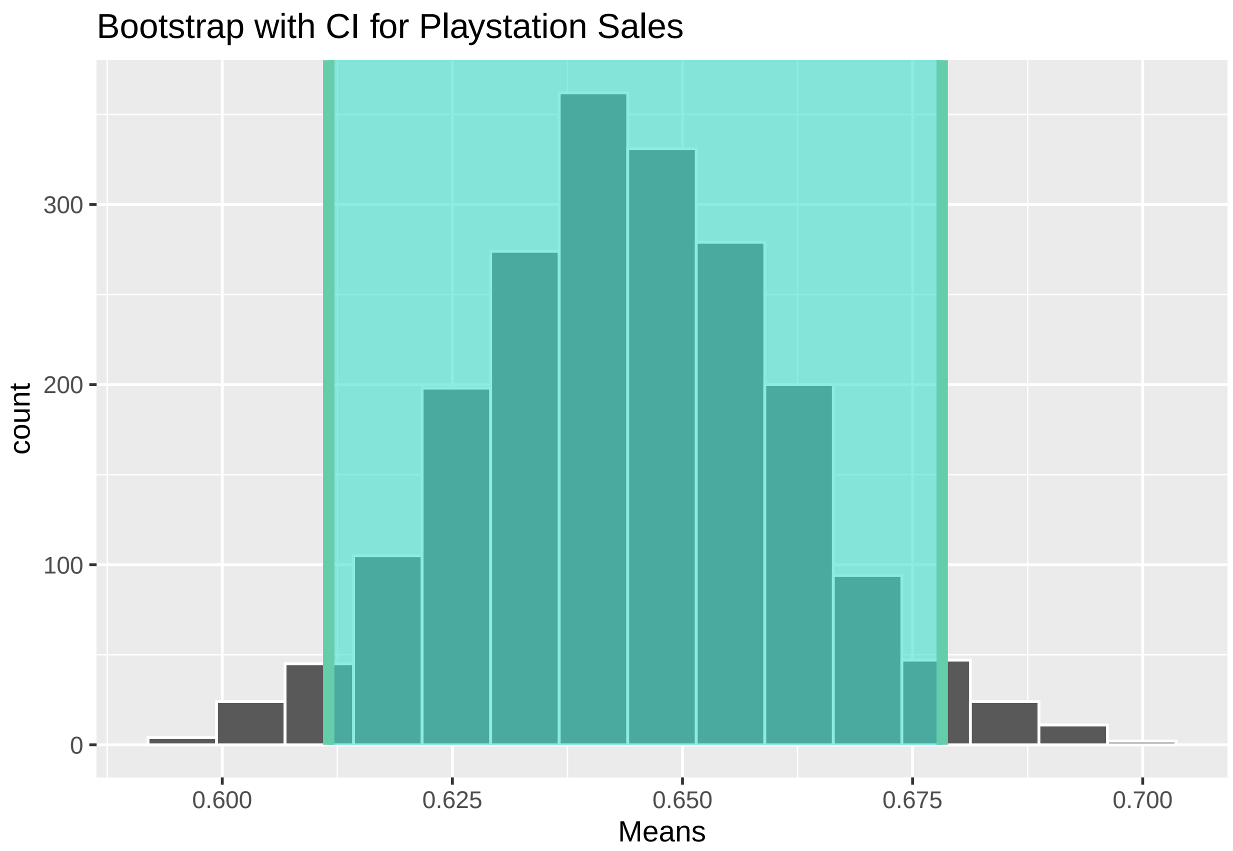
```
platform_bootstrap_distribution_2000_mean
```

```
## Response: total_global_sales (numeric)
## # A tibble: 2,000 x 2
##    replicate  stat
##        <int> <dbl>
##  1         1 0.637
##  2         2 0.649
##  3         3 0.636
##  4         4 0.618
##  5         5 0.661
##  6         6 0.639
##  7         7 0.644
##  8         8 0.623
##  9         9 0.668
## 10        10 0.638
## # ... with 1,990 more rows
```

```
#Creating confidence interval
platform_percentile_ci_2000 <- platform_bootstrap_distribution_2000_mean %>%
get_confidence_interval(level = 0.95, type = "percentile")
platform_percentile_ci_2000
```

```
## # A tibble: 1 x 2
##   lower_ci upper_ci
##      <dbl>    <dbl>
## 1    0.612    0.678
```

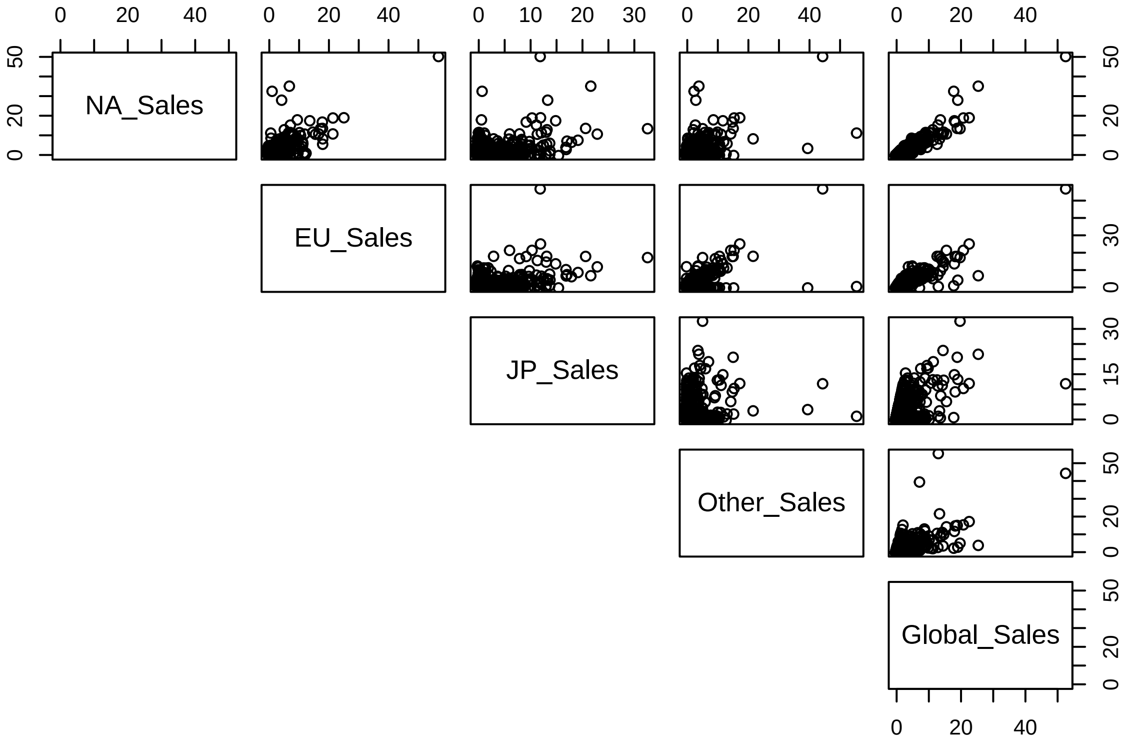
```
#visualizing bootstrap for 2000 replicates of the bootstrap
visualize(platform_bootstrap_distribution_2000_mean) +
  shade_confidence_interval(endpoints = platform_percentile_ci_2000) +
  ggtitle("Bootstrap with CI for Playstation Sales") +
  xlab('Means')
```



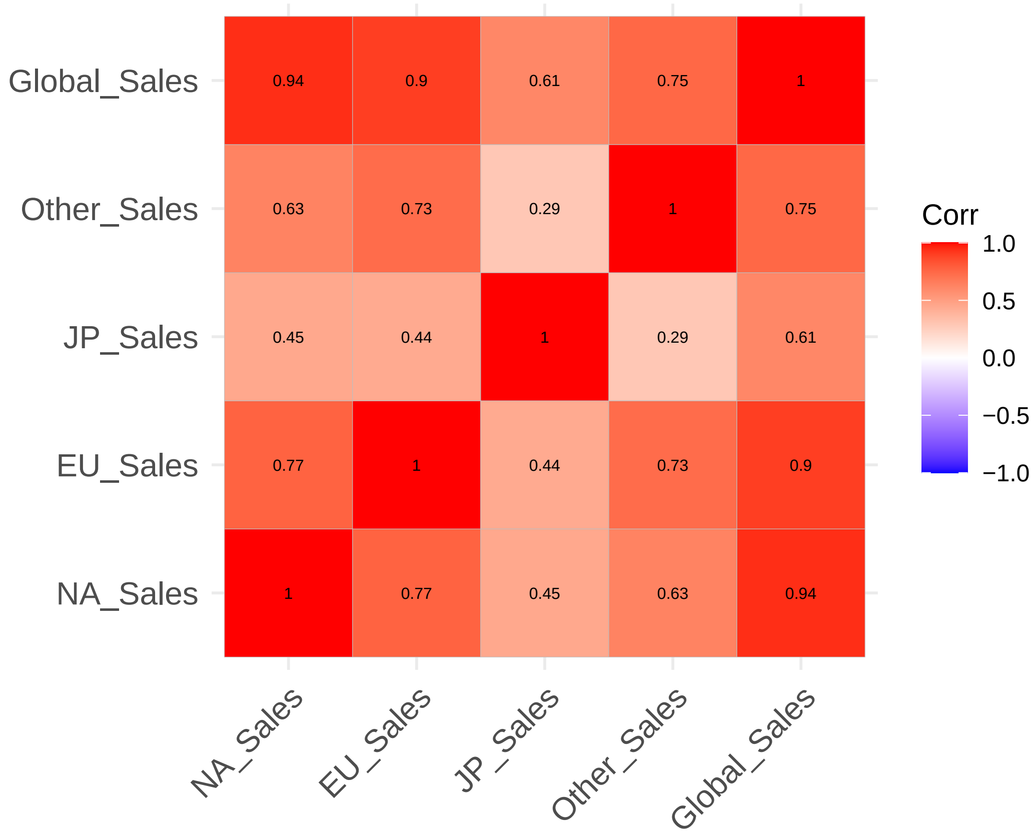
```
#Choosing sales only information
sales_only_data <- vgsales %>%
  select (.,NA_Sales,EU_Sales,JP_Sales,Other_Sales,Global_Sales)
head(sales_only_data)
```

```
##   NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales
## 1    41.49    29.02     3.77        8.46        82.74
## 2    29.08     3.58     6.81        0.77        40.24
## 3    15.85    12.88     3.79        3.31        35.82
## 4    15.75    11.01     3.28        2.96        33.00
## 5    11.27     8.89    10.22        1.00        31.37
## 6    23.20     2.26     4.22        0.58        30.26
```

```
pairs(scale(sales_only_data), lower.panel = NULL, cex = 1)
```



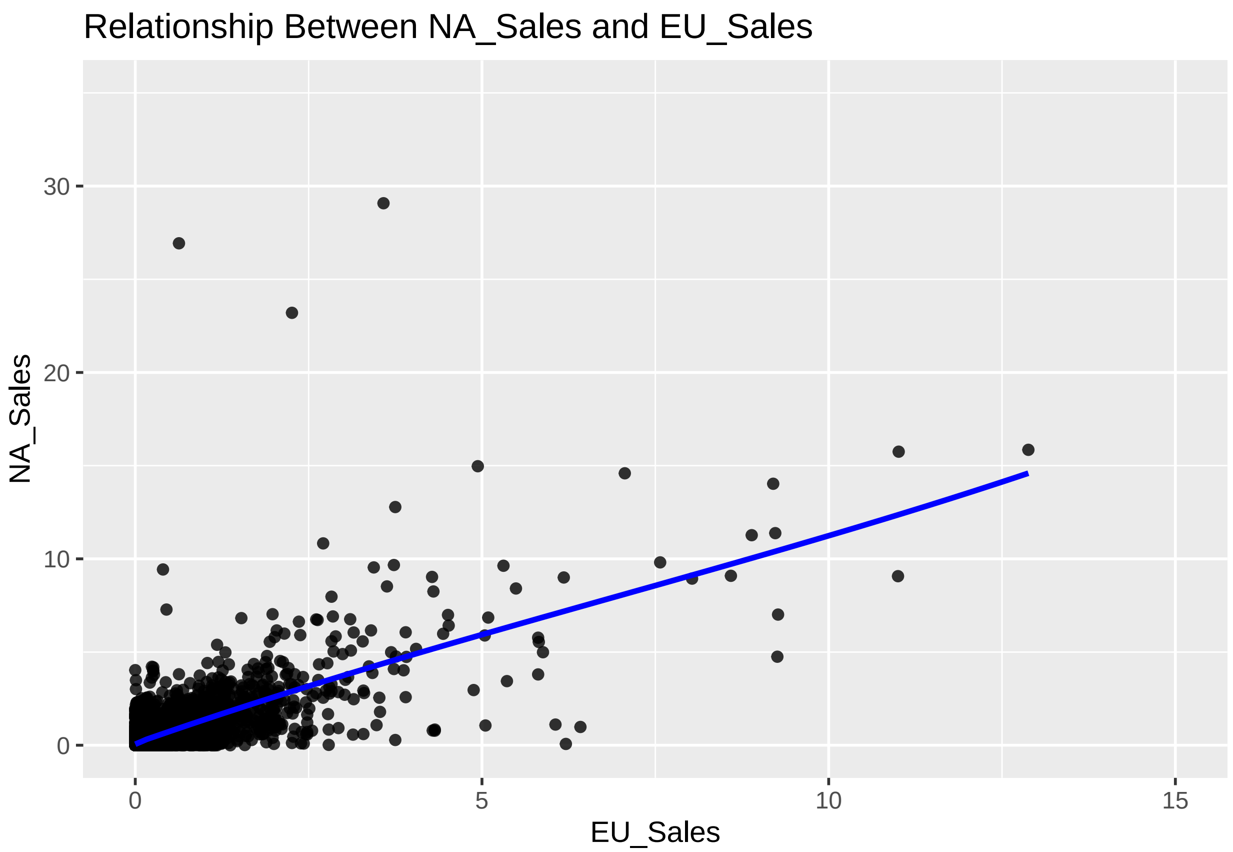
```r
#Making a correlation plot for our sales information
cor = cor(sales_only_data)
ggcorrplot(cor, lab_size = 2, lab= TRUE)
```



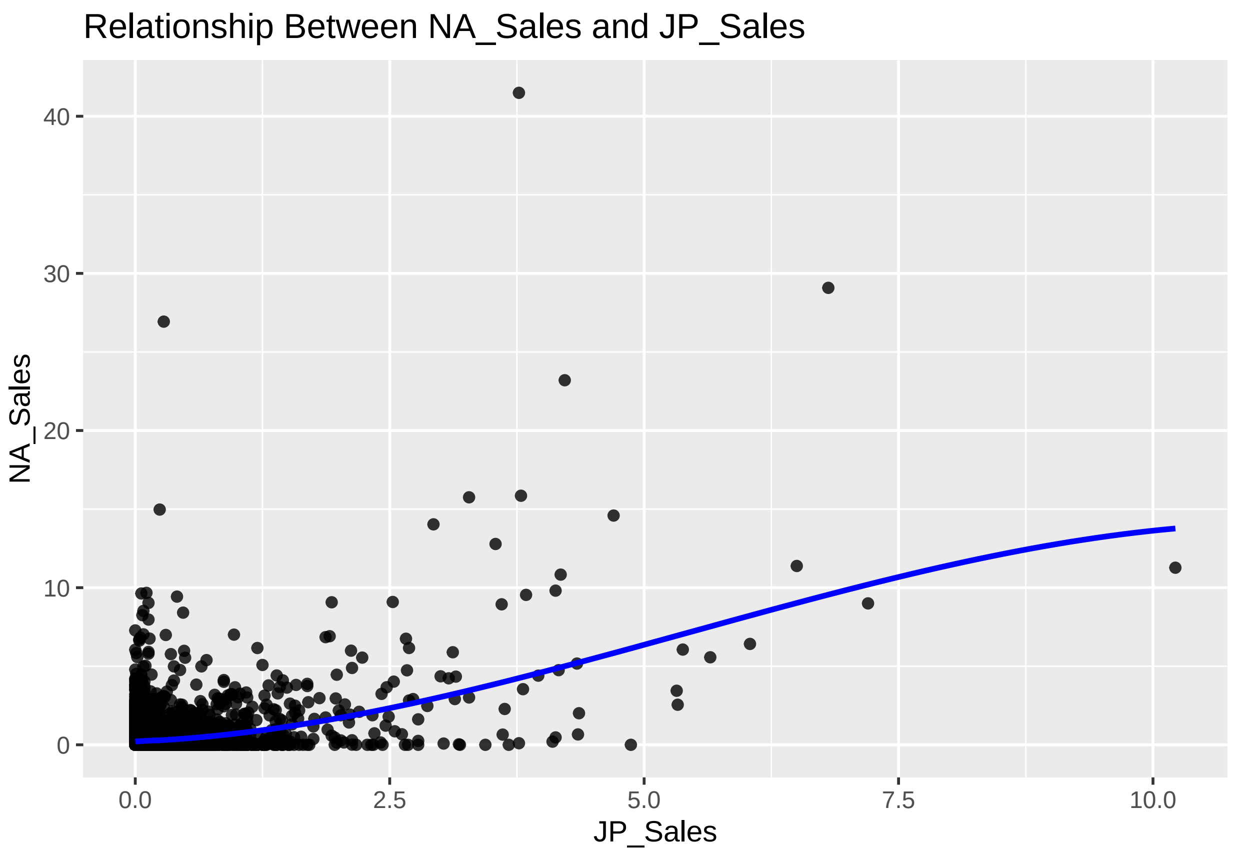
```
#Correlation plot between NA and EU
ggplot(sales_only_data, mapping =
       aes(x= EU_Sales, y= NA_Sales)) +
geom_point(col = "black", alpha = .8 ) +
  geom_smooth(method = "loess", formula = y ~ x, se=FALSE, col= 'blue') +
  ylim(0,35) +
  xlim(0,15) +
    ggtitle('Relationship Between NA_Sales and EU_Sales')+
labs(x = "EU_Sales", y = "NA_Sales")
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

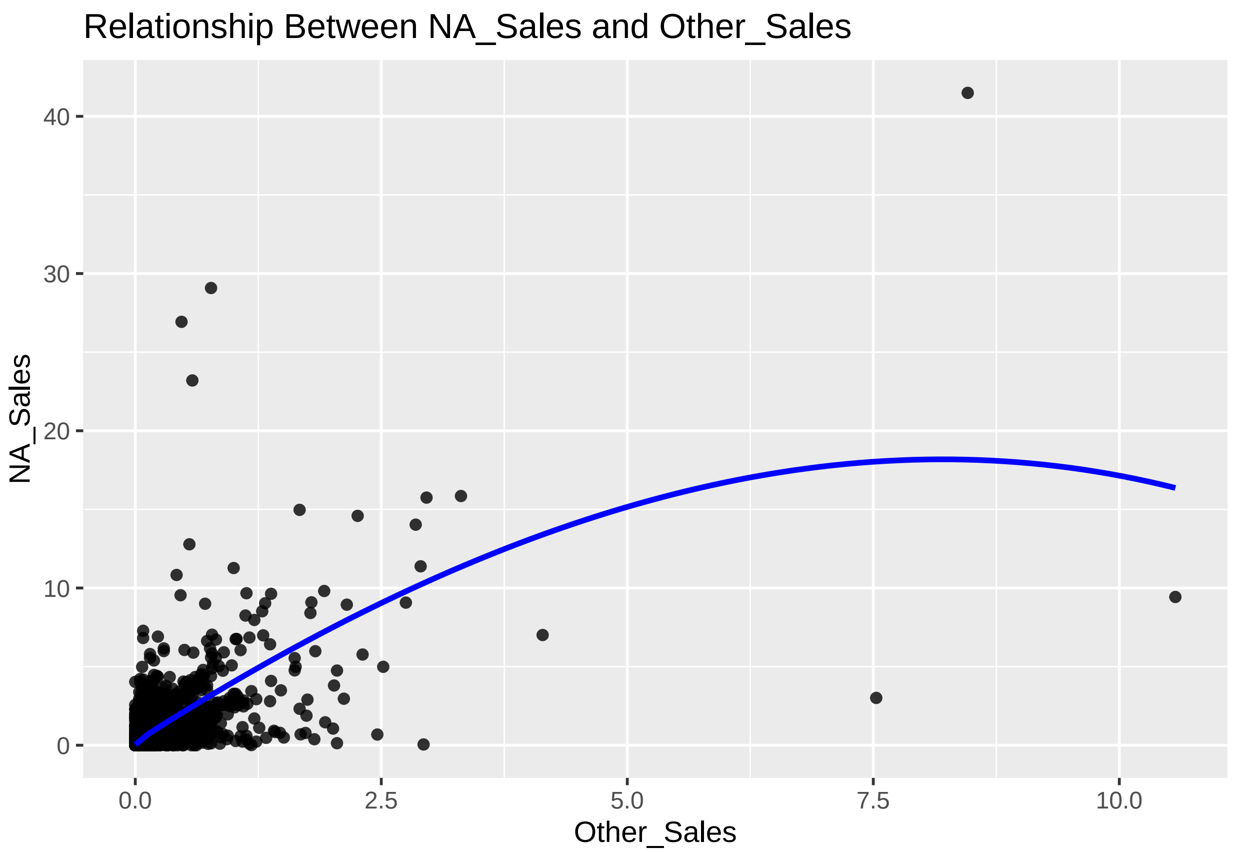
```
## Warning: Removed 1 rows containing missing values (geom_point).
```



```
#Correlation plot between NA and JP
ggplot(sales_only_data, mapping =
       aes(x= JP_Sales, y= NA_Sales)) +
geom_point(col = "black", alpha = .8 ) +
  geom_smooth(method = "loess", formula = y ~ x, se=FALSE, col= 'blue') +
    ggtitle('Relationship Between NA_Sales and JP_Sales')+
labs(x = "JP_Sales", y = "NA_Sales")
```



```
#Correlation plot between NA and Other
ggplot(sales_only_data, mapping =
       aes(x= Other_Sales, y= NA_Sales)) +
geom_point(col = "black", alpha = .8 ) +
  geom_smooth(method = "loess", formula = y ~ x, se=FALSE, col= ’blue’) +
    ggtitle(’Relationship Between NA_Sales and Other_Sales’)+
labs(x = "Other_Sales", y = "NA_Sales")
```



```
#Regression model for sales
sales_model <- lm(NA_Sales ~  EU_Sales + JP_Sales + Other_Sales, data= sales_only_data )
get_regression_table(sales_model)
```

```
## # A tibble: 4 x 7
##   term        estimate std_error statistic p_value lower_ci upper_ci
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
## 1 intercept      0.061     0.004      14.6       0    0.053    0.069
## 2 EU_Sales       0.939     0.012      77.9       0    0.915    0.962
## 3 JP_Sales       0.391     0.014      27.7       0    0.364    0.419
## 4 Other_Sales    0.732     0.03       24.1       0    0.673    0.792
```

```
regression_points <- get_regression_points(sales_model)
summary(sales_model)
```

```
##
## Call:
## lm(formula = NA_Sales ~ EU_Sales + JP_Sales + Other_Sales, data = sales_only_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1320 -0.0881 -0.0489  0.0319 25.8242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.060740   0.004168   14.57   <2e-16 ***
## EU_Sales    0.938670   0.012045   77.93   <2e-16 ***
## JP_Sales    0.391422   0.014131   27.70   <2e-16 ***
## Other_Sales 0.732136   0.030348   24.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5057 on 16319 degrees of freedom
## Multiple R-squared:  0.6213, Adjusted R-squared:  0.6212
## F-statistic:  8925 on 3 and 16319 DF,  p-value: < 2.2e-16
```

```
#In this case NA_Sales is response and the others are predictors
```