Video Game EDA and Visualization

Yang Fei(Sophia)

2020_8

Table of Contents


```
## Warning in extract(path, exdir = path.expand(dirname(default_inst()))):
error 1
## in extracting from zip file

## Warning in system2("tlmgr", args, ...): '"tlmgr"' not found

## Warning in system2("texhash"): '"texhash"' not found

## Warning in system2(if (usermode) "fmtutil-user" else "fmtutil-sys", "--
all", :
## '"fmtutil-sys"' not found

## Warning in system2(if (usermode) "updmap-user" else "updmap-sys"):
'"updmap-
## sys"' not found
```

```
## Warning in system2("fc-cache", args): '"fc-cache"' not found

## NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use
these themes.

##       Please use hrbrthemes::import_roboto_condensed() to install Roboto
Condensed and

##       if Arial Narrow is not on your system, please see
https://bit.ly/arialnarrow

## Loading required package: ggplot2

## No renderer backend detected. gganimate will default to writing frames to
separate files
## Consider installing:
## - the `gifski` package for gif output
## - the `av` package for video output
## and restarting the R session

##
## ********************************************************

## Note: As of version 1.0.0, cowplot does not change the

##   default ggplot2 theme anymore. To recover the previous

##   behavior, execute:
##   theme_set(theme_cowplot())

## ********************************************************

##
## Attaching package: 'cowplot'

## The following object is masked from 'package:ggthemes':
##
##     theme_map

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Loading required package: pacman

## WARNING: Rtools is required to build R packages, but is not currently
installed.
```

```
##
## Please download and install Rtools 4.0 from https://cran.r-
project.org/bin/windows/Rtools/.

## Skipping install of 'bbplot' from a github remote, the SHA1 (82af5952) has
not changed since last install.
##   Use `force = TRUE` to force installation
```

Data Cleaning

import the dataset

import the dataset. check the size of the dataframe

```
#import the dataset
raw_data<-read.csv('E:/DS/repo/My_Project/Video_Game_EDA_R/vgsales.csv')
head(raw_data, 5)

##   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
##   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

dim(raw_data)

## [1] 16598    11
```

Deal with the missing values

compute and drop the “N/A” number

```
#compute the "N/A" number
sum(raw_data$Year == "N/A")

## [1] 271

#drop the "N/A" rows
w<-which(raw_data$Year=="N/A")
```

```
raw_data2<-raw_data[-w,]
dim(raw_data2)

## [1] 16327    11
```

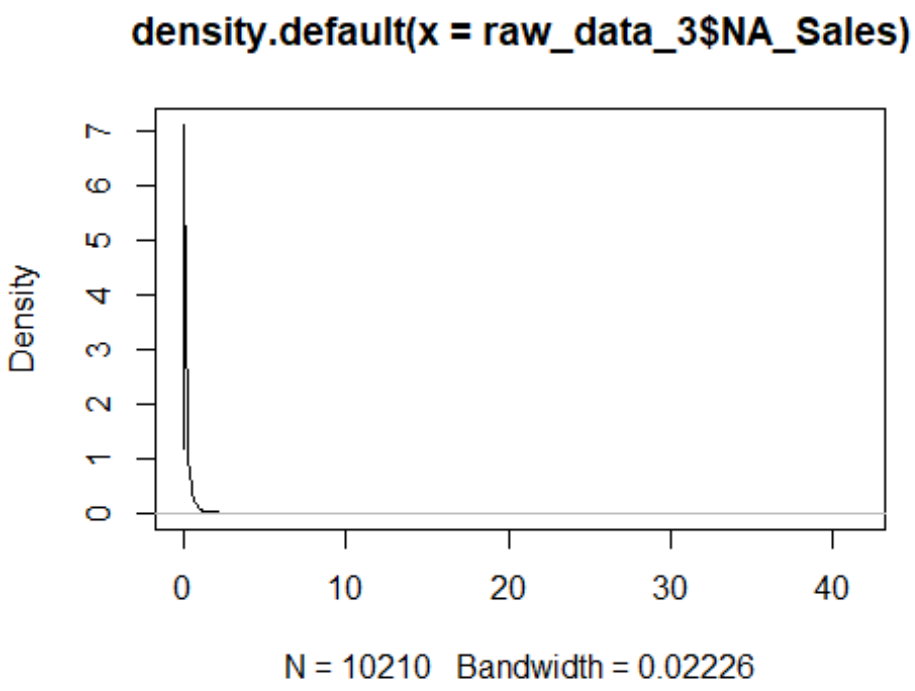
Change the analysis size

change the the period to recent 12 years

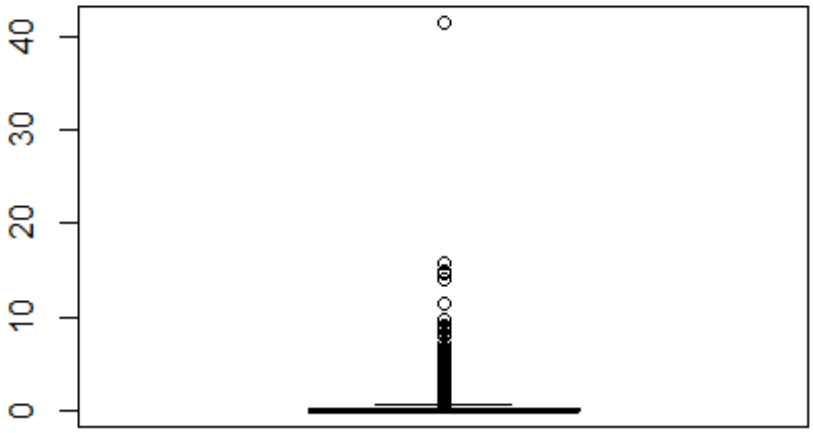
```
#select the recent 5 years
raw_data_3<-filter(raw_data2,Year>2005,Year<2017)
dim(raw_data_3)

## [1] 10210    11
```

drop the outliers

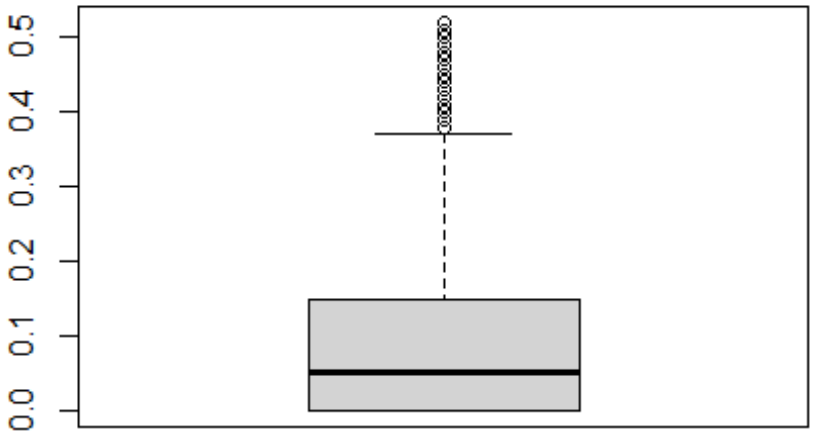
```
#select and delete outliers
plot(density(raw_data_3$NA_Sales))
```




```
boxplot(raw_data_3$NA_Sales)
## get the outliers
out=boxplot(raw_data_3$NA_Sales)$out
```




```
## get the outliers index
x<-which(raw_data_3$NA_Sales %in% out)
## get the clean data
clean_data<-raw_data_3[-x,]
## check the clean data
boxplot(clean_data$NA_Sales)
```




```
dim(clean_data)
```

```
## [1] 9211   11
```

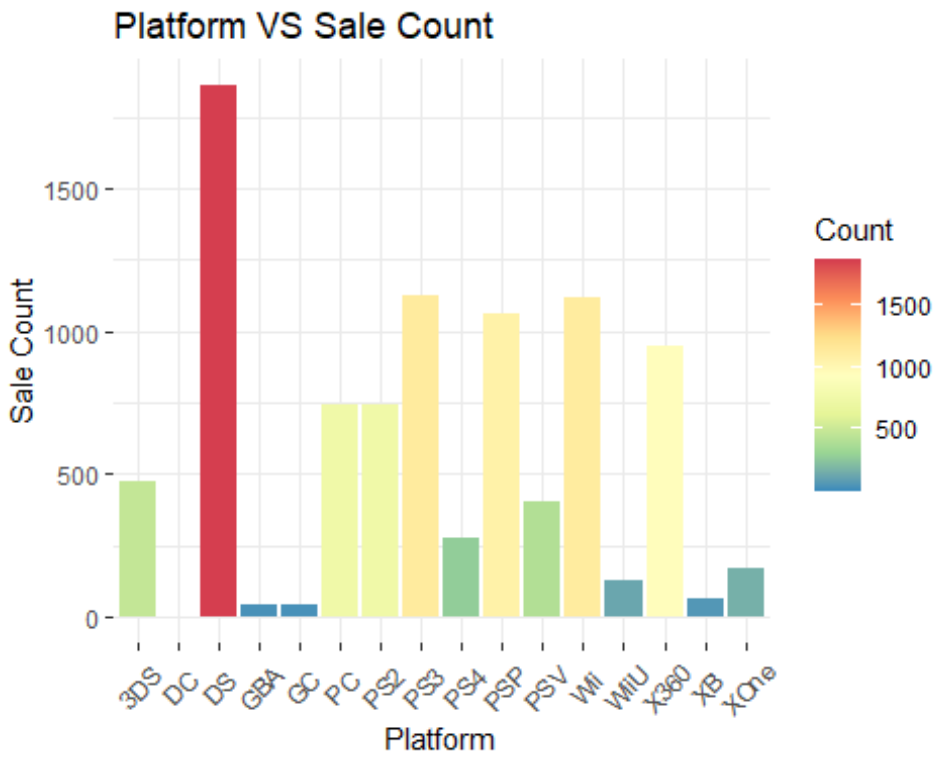
EDA

generate the the relationship between platform and sale count

```
platform <-clean_data %>%
  group_by(Platform)%>%
  summarise(Count = n())
```

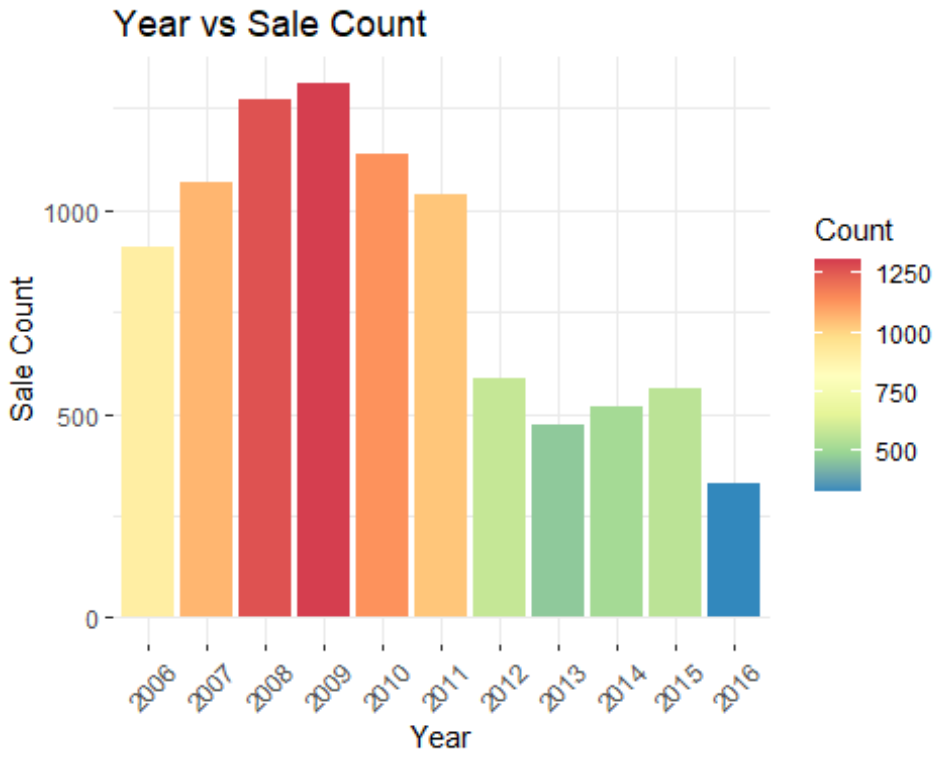
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
ggplot(platform,aes(x = Platform , y = Count,fill=Count)) +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  geom_col() +
  ggtitle('Platform VS Sale Count')+
  scale_fill_distiller(palette = 'Spectral') +
  ylab('Sale Count')
```



generate the the relationship between different years and count

```
years <-clean_data %>%
  group_by(Year)%>%
  summarise(Count = n())

## `summarise()` ungrouping output (override with `.groups` argument)

ggplot(years,aes(x = Year , y = Count,fill=Count)) +
  theme_bw()+
  geom_col() +
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  ggtitle('Year vs Sale Count')+
  scale_fill_distiller(palette = 'Spectral') +
  ylab('Sale Count')+
  xlab('Year')
```



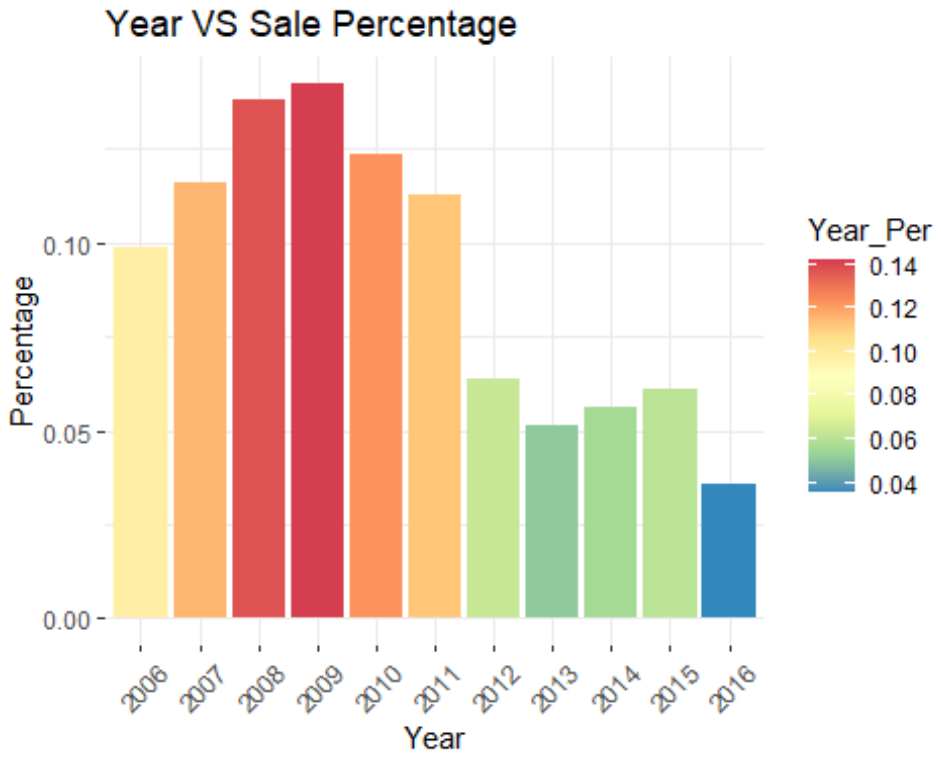
generate the the relationship between different years and percentage

```
Year_freq<-table(clean_data$Year)
Year_Per<- prop.table(table(clean_data$Year) * 100)
year_df<-data.frame(cbind(Year_freq,Year_Per))
year_df
```

```
##      Year_freq   Year_Per
## 2006       909 0.09868635
## 2007      1069 0.11605689
## 2008      1274 0.13831289
## 2009      1310 0.14222126
## 2010      1137 0.12343937
## 2011      1037 0.11258278
## 2012       587 0.06372815
## 2013       471 0.05113451
## 2014       520 0.05645424
## 2015       565 0.06133970
## 2016       332 0.03604386
```

```
ggplot(year_df,aes(x = row.names(year_df) , y = Year_Per,fill=Year_Per)) +
  geom_col()+
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  ggtitle('Year VS Sale Percentage')+
  scale_fill_distiller(palette = 'Spectral') +
```

```
  ylab('Percentage')+
  xlab('Year')
```



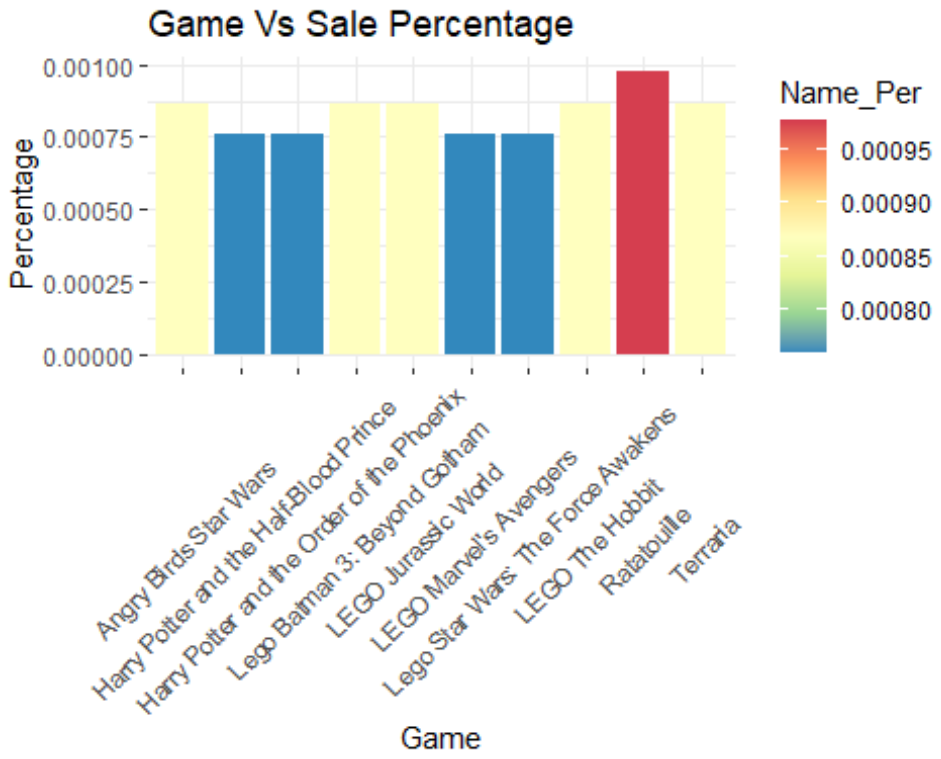
generate the the relationship between different game and percentage

```
Name_freq<-table(clean_data$Name)
Name_Per<- prop.table(table(clean_data$Name) * 100)
Name_df<-data.frame(cbind(Name_freq,Name_Per))
head(Name_df,10)
```

```
##                                           Name_freq     Name_Per
## .hack//G.U. Vol.1//Rebirth                        1 0.0001085658
## .hack//G.U. Vol.2//Reminisce                      1 0.0001085658
## .hack//G.U. Vol.2//Reminisce (jp sales)           1 0.0001085658
## .hack//G.U. Vol.3//Redemption                     1 0.0001085658
## .hack//Link                                       1 0.0001085658
## .hack: Sekai no Mukou ni + Versus                 1 0.0001085658
## [Prototype 2]                                     3 0.0003256975
## 007: Quantum of Solace                            5 0.0005428292
## 1 vs. 100                                         1 0.0001085658
## 1/2 Summer +                                      1 0.0001085658
```

```
Name_df<- head(Name_df[order(Name_df$Name_freq, decreasing = T), ], 10)

ggplot(Name_df,aes(x = row.names(Name_df) , y = Name_Per,fill=Name_Per)) +
  geom_col()+
  bbc_style() +
  theme_bw()+
```

```
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  ggtitle('Game Vs Sale Percentage')+
  scale_fill_distiller(palette = 'Spectral') +
  ylab('Percentage')+
  xlab('Game')
```



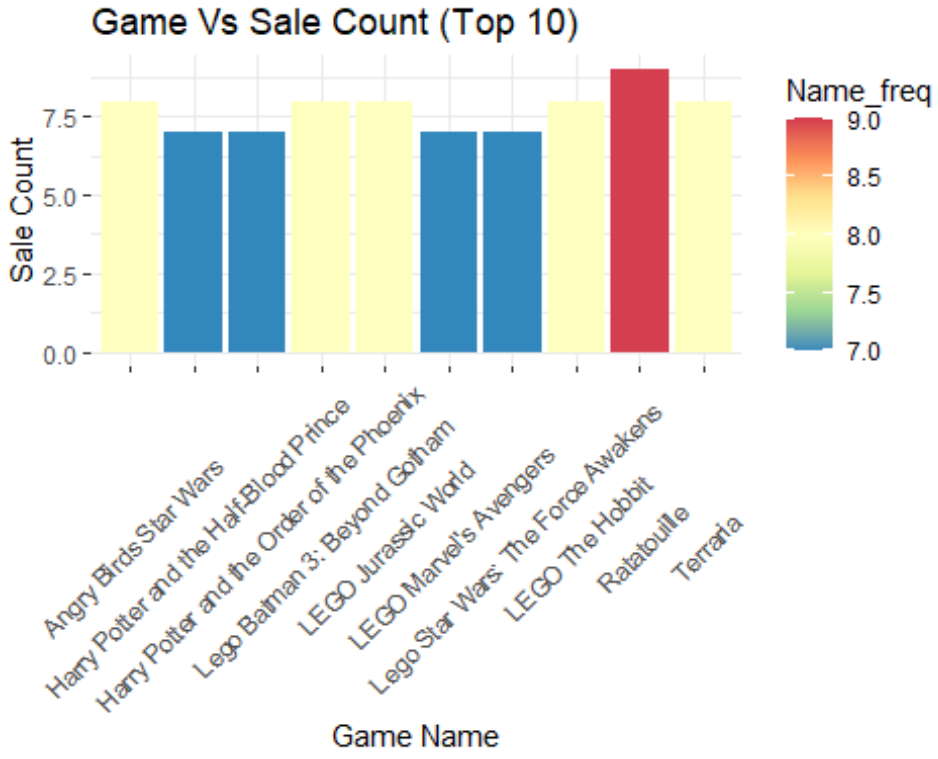
generate the the relationship between different game and Sale Count

```
Name_freq<-table(clean_data$Name)
Name_Per<- prop.table(table(clean_data$Name) * 100)
Name_df<-data.frame(cbind(Name_freq,Name_Per))
head(Name_df,10)
```

```
##                                          Name_freq     Name_Per
## .hack//G.U. Vol.1//Rebirth                       1 0.0001085658
## .hack//G.U. Vol.2//Reminisce                     1 0.0001085658
## .hack//G.U. Vol.2//Reminisce (jp sales)          1 0.0001085658
## .hack//G.U. Vol.3//Redemption                    1 0.0001085658
## .hack//Link                                      1 0.0001085658
## .hack: Sekai no Mukou ni + Versus                1 0.0001085658
## [Prototype 2]                                    3 0.0003256975
## 007: Quantum of Solace                           5 0.0005428292
## 1 vs. 100                                        1 0.0001085658
## 1/2 Summer +                                     1 0.0001085658
```

```
Name_df<- head(Name_df[order(Name_df$Name_freq, decreasing = T), ], 10)
```

```
ggplot(Name_df,aes(x = row.names(Name_df) , y = Name_freq,fill=Name_freq)) +
  geom_col()+
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  ggtitle('Game Vs Sale Count (Top 10)')+
  scale_fill_distiller(palette = 'Spectral') +
  ylab('Sale Count')+
  xlab('Game Name')
```



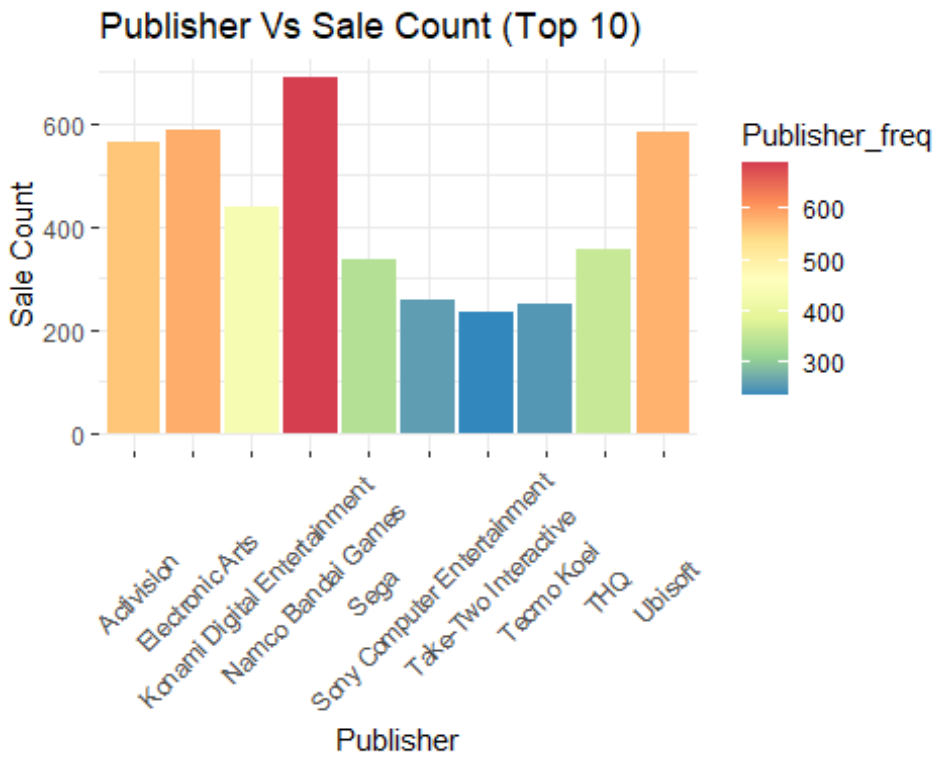
generate the the relationship between different Publisher and Sale Count

```
Publisher_freq<-table(clean_data$Publisher)
Publisher_Per<- prop.table(table(clean_data$Publisher) * 100)
Publisher_df<-data.frame(cbind(Publisher_freq,Publisher_Per))
head(Publisher_df,10)
```

```
##                 Publisher_freq Publisher_Per
## 10TACLE Studios              3  0.0003256975
## 1C Company                   3  0.0003256975
## 2D Boy                       1  0.0001085658
## 49Games                      1  0.0001085658
## 505 Games                  159  0.0172619694
## 5pb                         61  0.0066225166
## 7G//AMES                     4  0.0004342634
## Abylight                     1  0.0001085658
```

```
## Ackkstudios                  10  0.0010856585
## Acquire                      13  0.0014113560
```

```
Publisher_df<- head(Publisher_df[order(Publisher_df$Publisher_freq,
decreasing = T), ], 10)

ggplot(Publisher_df,aes(x = row.names(Publisher_df) , y =
Publisher_freq,fill=Publisher_freq)) +
  geom_col()+
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  ggtitle('Publisher Vs Sale Count (Top 10)')+
  scale_fill_distiller(palette = 'Spectral') +
  ylab('Sale Count')+
  xlab('Publisher')
```



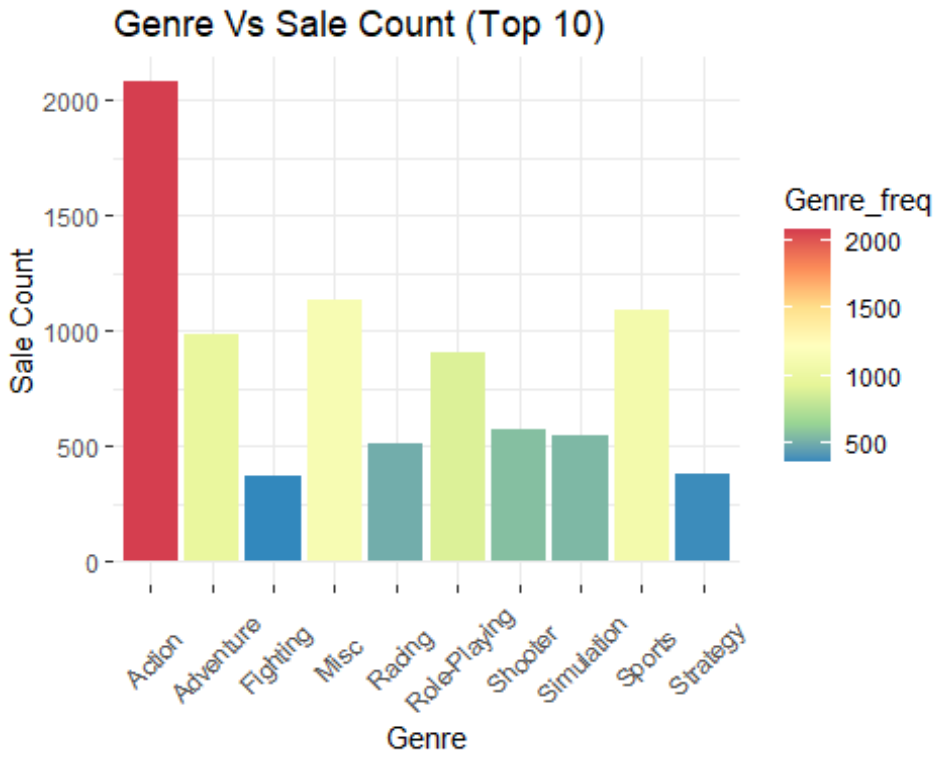
generate the the relationship between different Genre and Sale Count

```
Genre_freq<-table(clean_data$Genre)
Genre_Per<- prop.table(table(clean_data$Genre) * 100)
Genre_df<-data.frame(cbind(Genre_freq,Genre_Per))
head(Genre_df,10)
```

```
##              Genre_freq  Genre_Per
## Action             2082 0.22603409
## Adventure           984 0.10682879
```

```
## Fighting             368 0.03995223
## Misc                1135 0.12322223
## Platform             288 0.03126696
## Puzzle               351 0.03810661
## Racing               508 0.05515145
## Role-Playing         902 0.09792639
## Shooter              575 0.06242536
## Simulation           549 0.05960265
```

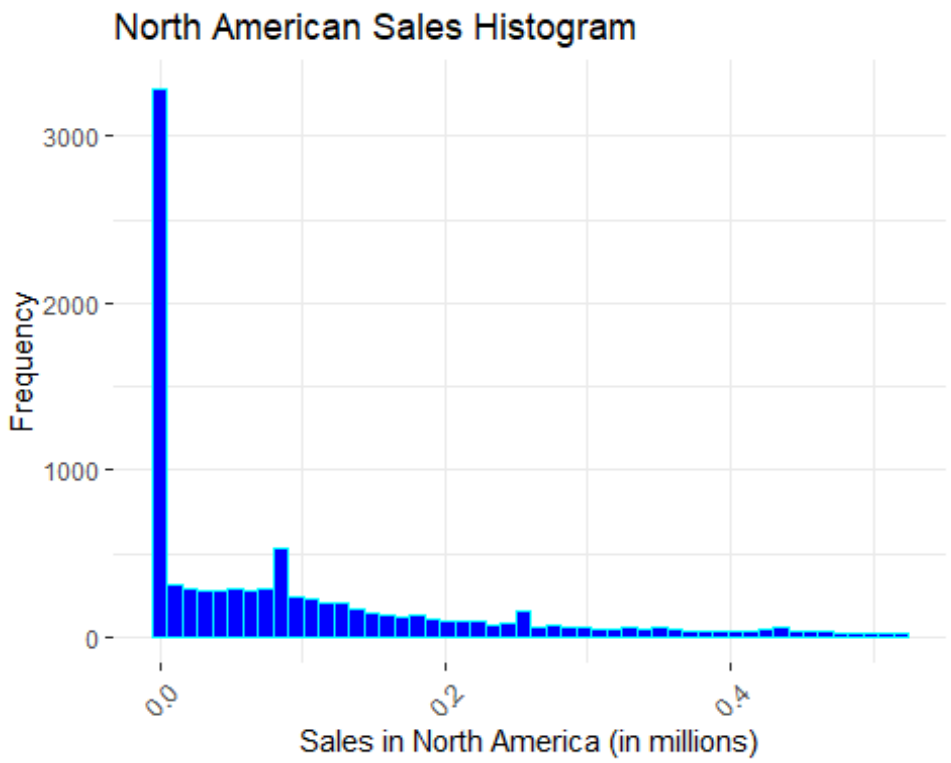
```
Genre_df<- head(Genre_df[order(Genre_df$Genre_freq, decreasing = T), ], 10)

ggplot(Genre_df,aes(x = row.names(Genre_df) , y =
Genre_freq,fill=Genre_freq)) +
  geom_col()+
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  ggtitle('Genre Vs Sale Count (Top 10)')+
  scale_fill_distiller(palette = 'Spectral') +
  ylab('Sale Count')+
  xlab('Genre')
```



generate the histogram of North America data

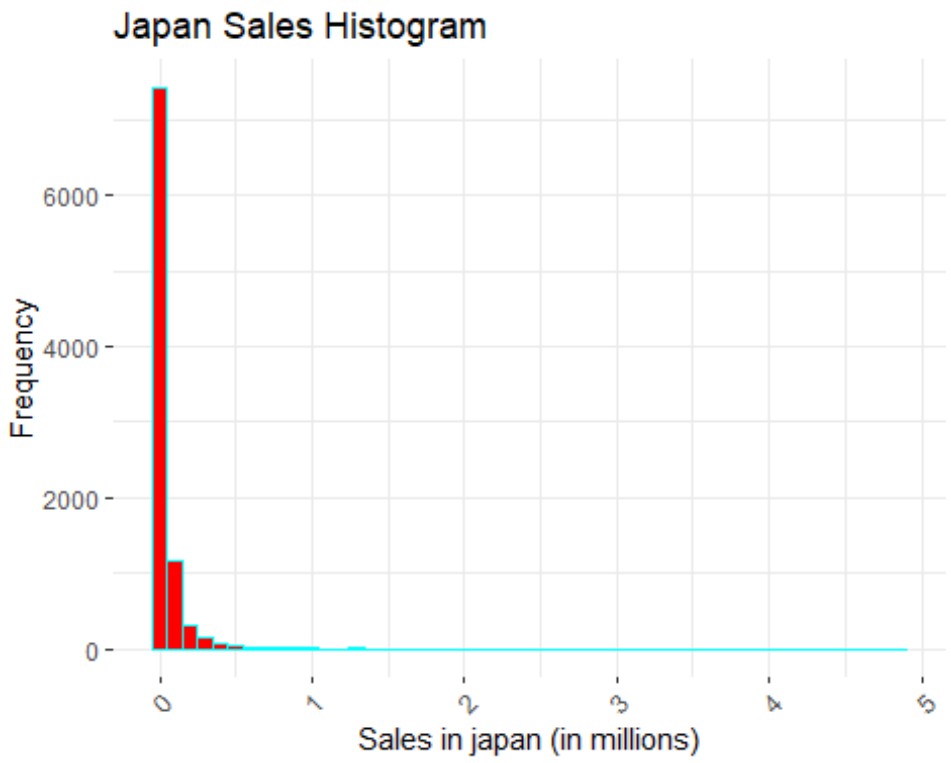
```
ggplot(data = clean_data, mapping = aes(x = NA_Sales)) +
  geom_histogram(bins = 50, fill = "blue", color = "cyan") +
```

```
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  scale_fill_distiller(palette = 'Spectral') +
  xlab("Sales in North America (in millions)") +
  ylab("Frequency") +
  ggtitle("North American Sales Histogram")
```



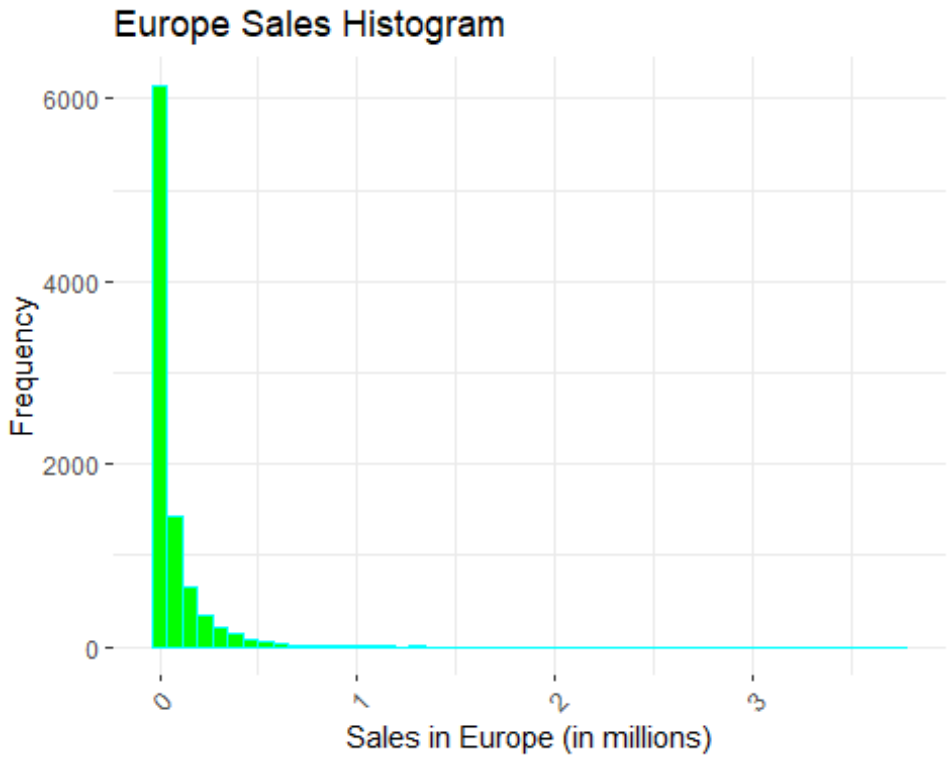
generate the histogram of Japan

```
ggplot(data = clean_data, mapping = aes(x = JP_Sales)) +
  geom_histogram(bins = 50, fill = "red", color = "cyan") +
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  scale_fill_distiller(palette = 'Spectral') +
  xlab("Sales in japan (in millions)") +
  ylab("Frequency") +
  ggtitle("Japan Sales Histogram")
```



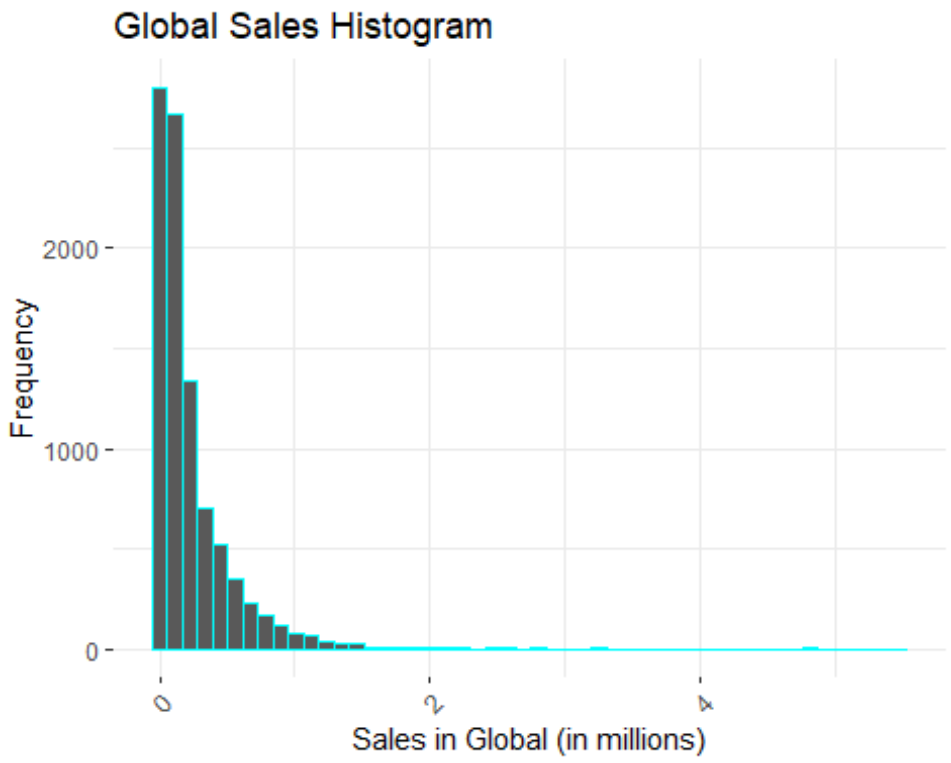
generate the histogram of Europe

```
ggplot(data = clean_data, mapping = aes(x = EU_Sales)) +
  geom_histogram(bins = 50, fill = "green", color = "cyan") +
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  scale_fill_distiller(palette = 'Spectral') +
  xlab("Sales in Europe (in millions)") +
  ylab("Frequency") +
  ggtitle("Europe Sales Histogram")
```



generate the histogram of global data

```
ggplot(data = clean_data, mapping = aes(x = Global_Sales)) +
  geom_histogram(bins = 50,  color = "cyan") +
  bbc_style() +
  theme_bw()+
  theme(panel.border = element_blank(),axis.text.x = element_text(angle = 45,
hjust = 0.5, vjust = 0.5))+
  scale_fill_distiller(palette = 'Spectral') +
  xlab("Sales in Global (in millions)") +
  ylab("Frequency") +
  ggtitle("Global Sales Histogram")
```


Global Sales Histogram
Frequency
2000
1000
0
0
2
4
Sales in Global (in millions)