Final Project 11/21

Akhil Sreekumar Nair, Hanxi Hu, Paras Pandey, Sumedha Anupindi, Wei Yuan

11/18/2019

```
#install.packages("ggplot2")
#install.packages("dplyr")
#install.packages("DT")
#install.packages("tidyr")
#install.packages("wesanderson")
#install.packages("ggthemes")
#install.packages("viridis")
#install.packages("lubridate")
#install.packages("wordcloud")
#install.packages("qdap")
#install.packages("tm")
#install.packages("ngram")
#install.packages("RColorBrewer")
#install.packages("gridExtra")
#install.packages("tidyverse")
#install.packages("sf")
#install.packages("raster")
#install.packages("spData")
#install.packages("tmap")
#install.packages("leaflet")
#install.packages("mapview")
#install.packages("shiny")
#install.packages("plotly")
#install.packages("gridExtra")
#install.packages("fmsb")
#install.packages("corrplot")
#install.packages("corrgram")
#install.packages("GGally")
#install.packages("caTools")
#install.packages("psych")
#install.packages("neuralnet")
library(ggplot2)
library(dplyr)
```

```
##
## 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
```

```
library(DT)
library(tidyr)
library(wesanderson)
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────────────────
──────── tidyverse 1.2.1 ──
```

```
## ✔ tibble  2.1.3     ✔ purrr   0.3.3
## ✔ readr   1.3.1     ✔ stringr 1.4.0
## ✔ tibble  2.1.3     ✔ forcats 0.4.0
```

```
## ── Conflicts ────────────────────────────────────────────────────────────
── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library(ggthemes)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(tm)
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(ngram)
library(RColorBrewer)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(sf)
```

```
## Linking to GEOS 3.7.2, GDAL 2.4.2, PROJ 5.2.0
```

```
library(raster)
```

```
## Loading required package: sp
```

```
##
## Attaching package: 'raster'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(spData)
```

```
## To access larger datasets in this package, install the spDataLarge
## package with: `install.packages('spDataLarge',
## repos='https://nowosad.github.io/drat/', type='source')`
```

```
library(tmap)
library(leaflet)
library(mapview)
library(shiny)
```

```
##
## Attaching package: 'shiny'
```

```
## The following objects are masked from 'package:DT':
##
##     dataTableOutput, renderDataTable
```

```
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:raster':
##
##     select
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(gridExtra)
library(fmsb)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(corrgram)
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus
```

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
library(caTools)
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(neuralnet)
```

```
##
## Attaching package: 'neuralnet'
```

```
## The following object is masked from 'package:dplyr':
##
##     compute
```

Synopsis

```
#The purpose of this document is to clean and do an exploratory analysis of the video game sales data. We examine how various factors influence the sale of video games and figure out what factors contribute the most to the sales of video games.
```

Data

```
#The data titled "video games sales" was used to do the analysis. After cleaning, the dataset contains around 11 columns and 16325 rows. The field names include - Rank - Ranking of overall sales Name -The game name Platform - Platform is the gaming console on which these games were released Year - Year in which the game was released Genre - Genre of the game Publisher - Who published the game NA_Sales - Sales in North America (in millions) EU_Sales - Sales in Europe (in millions) JP_Sales - Sales in Japan (in millions) Other_Sales - Sales in the rest of the world (in millions) Global_Sales - Total worldwide sales.
```

Read Data

```
vgsales <- data.frame(read.csv("/Users/iris/Downloads/vgsalesCleaned.csv"),sheet = 1, header = TRUE)
```

Understand data

```
head(vgsales)
```

```
##   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
##   EU_Sales JP_Sales Other_Sales Global_Sales sheet header
## 1    29.02     3.77        8.46        82.74     1   TRUE
## 2     3.58     6.81        0.77        40.24     1   TRUE
## 3    12.88     3.79        3.31        35.82     1   TRUE
## 4    11.01     3.28        2.96        33.00     1   TRUE
## 5     8.89    10.22        1.00        31.37     1   TRUE
## 6     2.26     4.22        0.58        30.26     1   TRUE
```

```
tail(vgsales)
```

```
##        Rank                                          Name Platform Year
## 16320 16595                                      Plushees       DS 2008
## 16321 16596                   Men in Black II: Alien Escape       GC 2003
## 16322 16597             Woody Woodpecker in Crazy Castle 5      GBA 2002
## 16323 16598                                   Know How 2       DS 2010
## 16324 16599 SCORE International Baja 1000: The Official Game      PS2 2008
## 16325 16600                                Spirits & Spells      GBA 2003
##            Genre  Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales
## 16320 Simulation  Destineer     0.01     0.00        0           0         0.01
## 16321    Shooter Infogrames     0.01     0.00        0           0         0.01
## 16322   Platform      Kemco     0.01     0.00        0           0         0.01
## 16323     Puzzle   7G//AMES     0.00     0.01        0           0         0.01
## 16324     Racing Activision     0.00     0.00        0           0         0.01
## 16325   Platform    Wanadoo     0.01     0.00        0           0         0.01
##       sheet header
## 16320     1   TRUE
## 16321     1   TRUE
## 16322     1   TRUE
## 16323     1   TRUE
## 16324     1   TRUE
## 16325     1   TRUE
```

```
str(vgsales)
```

```
## 'data.frame':    16325 obs. of  13 variables:
##  $ Rank        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Name        : Factor w/ 11359 levels "¡Shin Chan Flipa en colores!",..: 10864 9
243 5463 10866 7281 9601 6572 10862 6575 2558 ...
##  $ Platform    : Factor w/ 31 levels "2600","3DO","3DS",..: 26 12 26 26 6 6 5 26 2
6 12 ...
##  $ Year        : int  2006 1985 2008 2009 1996 1989 2006 2006 2009 1984 ...
##  $ Genre       : Factor w/ 12 levels "Action","Adventure",..: 11 5 7 11 8 6 5 4 5
9 ...
##  $ Publisher   : Factor w/ 577 levels "10TACLE Studios",..: 369 369 369 369 369 36
9 369 369 369 369 ...
##  $ NA_Sales    : num  41.5 29.1 15.8 15.8 11.3 ...
##  $ EU_Sales    : num  29.02 3.58 12.88 11.01 8.89 ...
##  $ JP_Sales    : num  3.77 6.81 3.79 3.28 10.22 ...
##  $ Other_Sales : num  8.46 0.77 3.31 2.96 1 0.58 2.9 2.85 2.26 0.47 ...
##  $ Global_Sales: num  82.7 40.2 35.8 33 31.4 ...
##  $ sheet       : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ header      : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
```

```
summary(vgsales)
```

```
##       Rank                                  Name          Platform
##  Min.   :    1   Need for Speed: Most Wanted:   12   DS     :2132
##  1st Qu.: 4136   FIFA 14                     :    9   PS2    :2123
##  Median : 8294   LEGO Marvel Super Heroes    :    9   PS3    :1309
##  Mean   : 8293   Ratatouille                 :    9   Wii    :1286
##  3rd Qu.:12442   Angry Birds Star Wars       :    8   X360   :1239
##  Max.   :16600   Cars                        :    8   PSP    :1198
##                  (Other)                     :16270   (Other):7038
##       Year               Genre                                Publisher
##  Min.   :1980   Action      :3250   Electronic Arts             : 1339
##  1st Qu.:2003   Sports      :2304   Activision                  :  966
##  Median :2007   Misc        :1710   Namco Bandai Games          :  928
##  Mean   :2006   Role-Playing:1471   Ubisoft                     :  918
##  3rd Qu.:2010   Shooter     :1282   Konami Digital Entertainment:  823
##  Max.   :2020   Adventure   :1276   THQ                         :  712
##                 (Other)     :5032   (Other)                     :10639
##     NA_Sales          EU_Sales          JP_Sales         Other_Sales
##  Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.00000   Min.   : 0.00000
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.00000   1st Qu.: 0.00000
##  Median : 0.0800   Median : 0.0200   Median : 0.00000   Median : 0.01000
##  Mean   : 0.2654   Mean   : 0.1476   Mean   : 0.07867   Mean   : 0.04833
##  3rd Qu.: 0.2400   3rd Qu.: 0.1100   3rd Qu.: 0.04000   3rd Qu.: 0.04000
##  Max.   :41.4900   Max.   :29.0200   Max.   :10.22000   Max.   :10.57000
##
##   Global_Sales         sheet    header
##  Min.   : 0.0100   Min.   :1   Mode:logical
##  1st Qu.: 0.0600   1st Qu.:1   TRUE:16325
##  Median : 0.1700   Median :1
##  Mean   : 0.5403   Mean   :1
##  3rd Qu.: 0.4800   3rd Qu.:1
##  Max.   :82.7400   Max.   :1
##
```

Data preparation

```
vgsales$Year = as.numeric(as.character(vgsales$Year))
vgsales$Global_Sales = as.numeric(as.character(vgsales$Global_Sales))
vgsales$NA_Sales = as.numeric(as.character(vgsales$NA_Sales))
vgsales$JP_Sales = as.numeric(as.character(vgsales$JP_Sales))
vgsales$Other_Sales = as.numeric(as.character(vgsales$Other_Sales))
vgsales$EU_Sales = as.numeric(as.character(vgsales$EU_Sales))
vgsales <- vgsales[!(vgsales$Year %in% c("N/A", "2017", "2020")),]

#The data was primarily cleaned in Excel. The filter function within excel was used t
o remove the anomalies in each column. For instance, there were N/A's and non-numeric
values for the year of release. These data were removed. Similarly, all the data whic
h was not in order was cleansed within excel. For example, in row 16267, "2007" is su
pposed to be in "Year" instead of "Genre." Then we manually put all data back to thei
r place.
```

Statistical Hypotheses Testing

```
#Hypothesis 1:The Global Sales is not associated with Platform of the Video Game
chisq.test(table(vgsales$Global_Sales , vgsales$Platform), correct=FALSE)
```

```
## Warning in chisq.test(table(vgsales$Global_Sales, vgsales$Platform), correct =
## FALSE): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(vgsales$Global_Sales, vgsales$Platform)
## X-squared = 22931, df = 18600, p-value < 2.2e-16
```

```
#P-value is less than 0.05, so the null hypothesis is rejected. We can conclude that
they are associate.
```

```
#Hypothesis 2:The Global Sales is not associated with Genre of the Video Game
chisq.test(table(vgsales$Global_Sales , vgsales$Genre), correct=FALSE)
```

```
## Warning in chisq.test(table(vgsales$Global_Sales, vgsales$Genre), correct =
## FALSE): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(vgsales$Global_Sales, vgsales$Genre)
## X-squared = 7862.7, df = 6820, p-value < 2.2e-16
```

```
#P-value is less than 0.05, so the null hypothesis is rejected. We can conclude that they are associate.
```

```
#Hypothesis 3:The Global Sales is not associated with Publisher of the Video Game
chisq.test(table(vgsales$Global_Sales , vgsales$Publisher), correct=FALSE)
```

```
## Warning in chisq.test(table(vgsales$Global_Sales, vgsales$Publisher), correct =
## FALSE): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(vgsales$Global_Sales, vgsales$Publisher)
## X-squared = 119612, df = 357120, p-value = 1
```

```
#P-value is bigger than 0.05, so the null hypothesis cannot be rejected. We can concl
ude that they are not associate.
```

```
#Hypothesis 4:The Global Sales is not associated with Year of the Video Game
chisq.test(table(vgsales$Global_Sales , vgsales$Year), correct=FALSE)
```

```
## Warning in chisq.test(table(vgsales$Global_Sales, vgsales$Year), correct =
## FALSE): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(vgsales$Global_Sales, vgsales$Year)
## X-squared = 49642, df = 22320, p-value < 2.2e-16
```

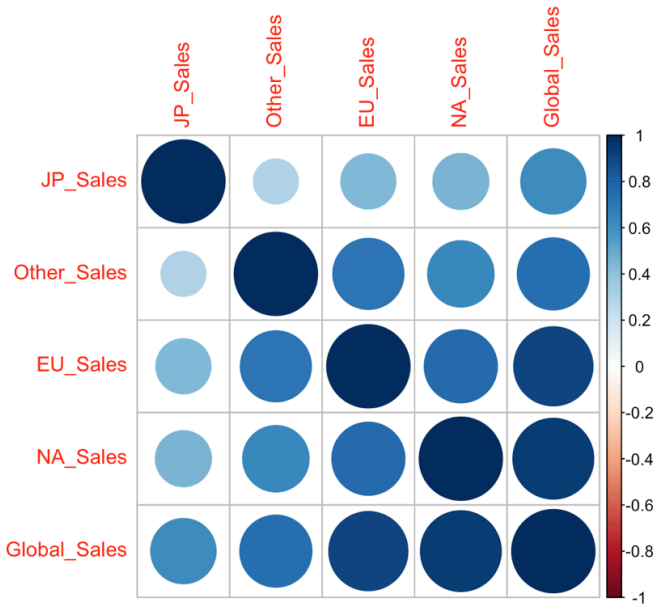
```
#P-value is bigger than 0.05, so the null hypothesis cannot be rejected. We can concl
ude that they are associate.
```

Correlation of sales in different Region.

```
#Before implementing statistical modeling, it is important to understand the corelati
on between the numeric variables

cor(vgsales[7:11])
```

```
##                NA_Sales  EU_Sales  JP_Sales Other_Sales Global_Sales
## NA_Sales      1.0000000 0.7689325 0.4512792   0.6345028    0.9412669
## EU_Sales      0.7689325 1.0000000 0.4364037   0.7262615    0.9032687
## JP_Sales      0.4512792 0.4364037 1.0000000   0.2906447    0.6127886
## Other_Sales   0.6345028 0.7262615 0.2906447   1.0000000    0.7479704
## Global_Sales 0.9412669 0.9032687 0.6127886   0.7479704    1.0000000
```

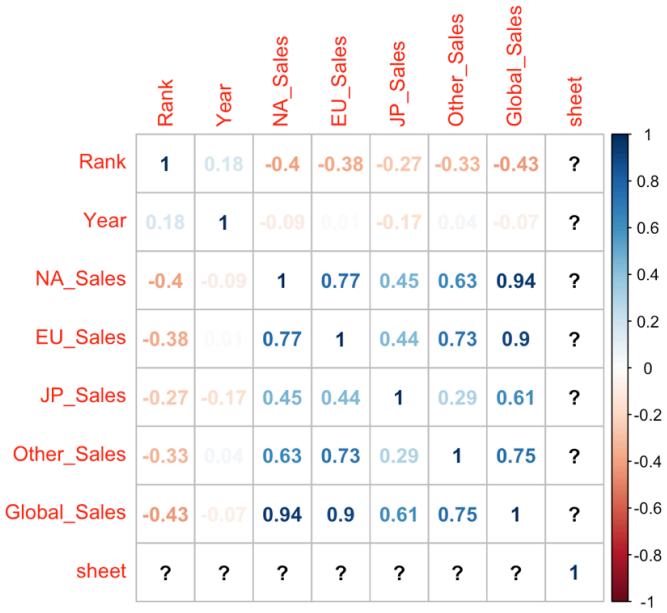
```
corrplot(cor(vgsales[7:11]) , order="hclust")
```




```
num.cols <- sapply(vgsales, is.numeric)
cor.data <- cor(vgsales[,num.cols])
```

```
## Warning in cor(vgsales[, num.cols]): the standard deviation is zero
```

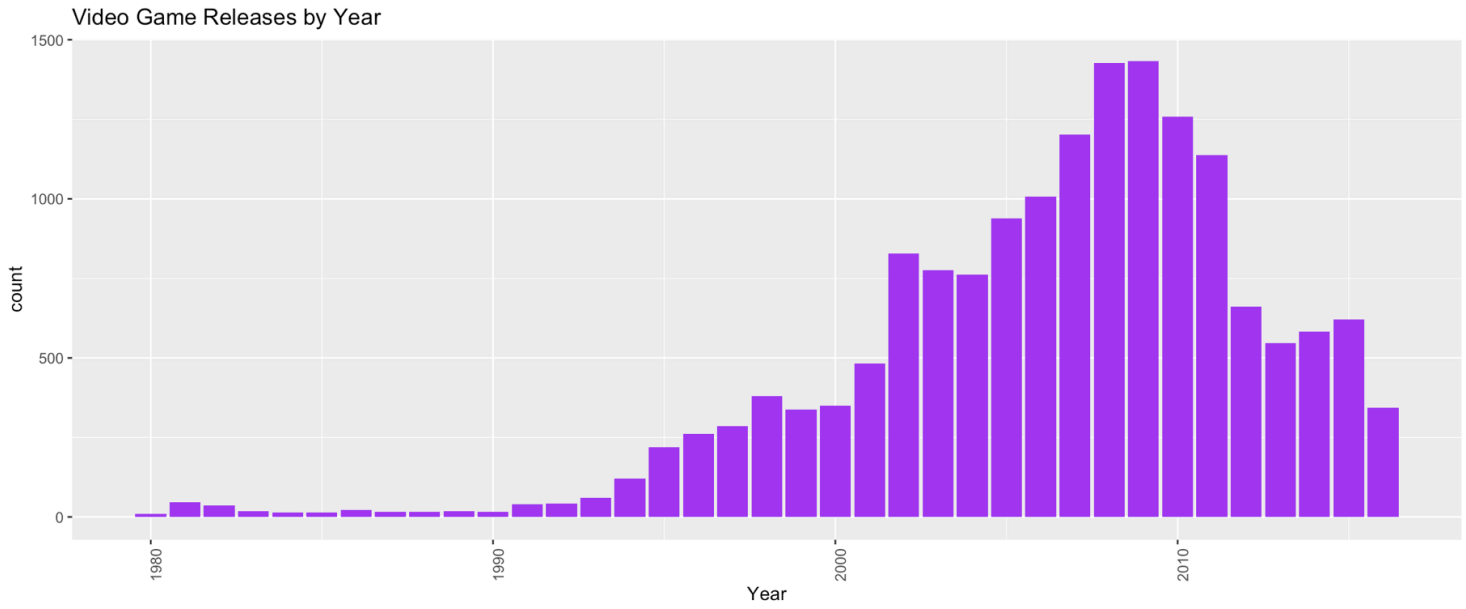
```
corrPLOT<-corrplot(cor.data,method='number')
```



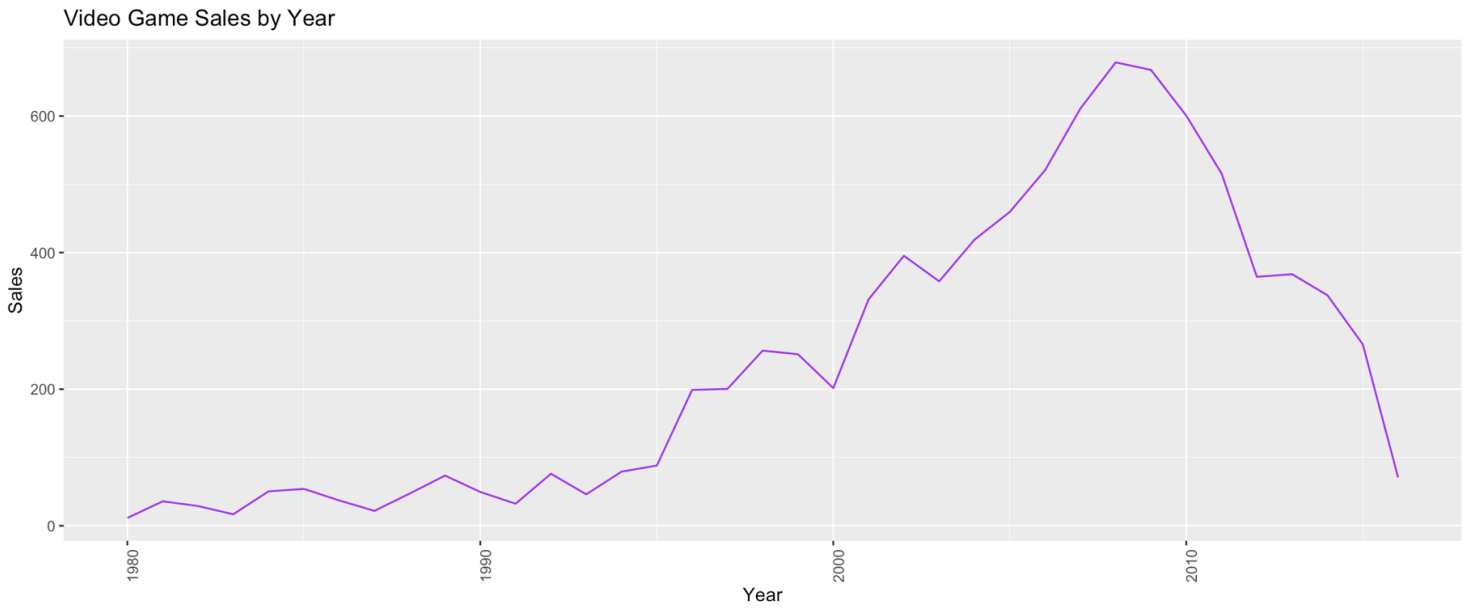

```
#From the correlation table it is observed that the NA_Sales(0.94), EU_Sales(0.9) and Other_Sales(0.75) are highly positive correlated with the Global_Sales. Although Global_Sales is correlated with the all Sales regions.
```

Game release and sales by year

```
ggplot(vgsales, aes(Year)) +
  geom_bar(fill="purple") + theme(axis.text.x = element_text(angle = 90))+
  ggtitle("Video Game Releases by Year")
```



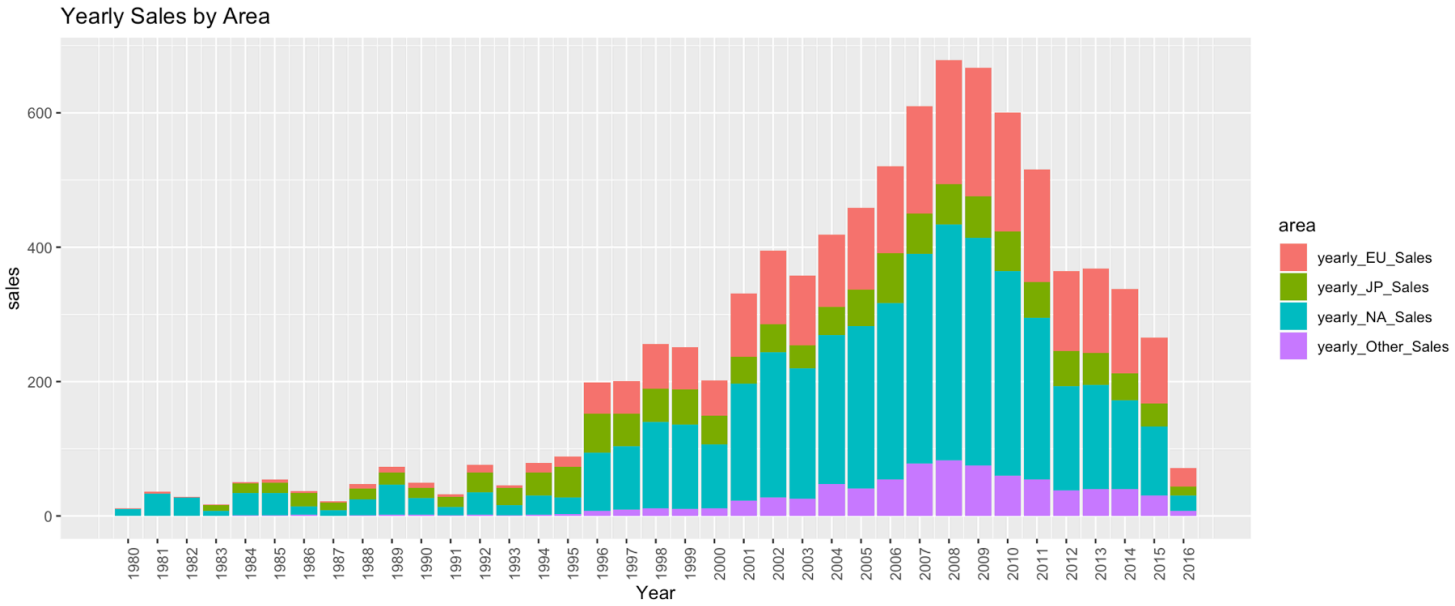
```
Sales_by_year <- vgsales %>%
                    group_by(Year) %>%
                      summarize(Sales = sum(Global_Sales))

ggplot(Sales_by_year, aes(Year, Sales)) +
  geom_line(color = "purple", stat = "identity") +theme(axis.text.x = element_text(an
gle = 90))+
  ggtitle("Video Game Sales by Year")
```



#There is a significant increse in the realease of video game since 1993, and it peaked in 2008 and 2009. The sales of the video games witnessed a similar trend, which means the more games were released, the more sales there were.

Yearly Sales by Area

```
yearly_vgsales_by_area <- vgsales %>% filter(!is.na(Year)) %>% group_by(Year) %>%
summarize(yearly_NA_Sales = sum(NA_Sales),
yearly_EU_Sales = sum(EU_Sales),
yearly_JP_Sales = sum(JP_Sales),
yearly_Other_Sales = sum(Other_Sales),
yearly_Global_Sales = sum(Global_Sales))

# Yearly sales by area as stacked barplot
yearly_vgsales_by_area_long <- gather(yearly_vgsales_by_area, area, sales, yearly_NA_
Sales:yearly_Other_Sales)
ggplot(yearly_vgsales_by_area_long, aes(x=Year, y=sales, fill = area)) +
  geom_bar(stat="identity") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  scale_x_continuous(breaks=(unique(yearly_vgsales_by_area$Year))) +
  ggtitle("Yearly Sales by Area")
```



#Following precious barcharts, this stacked barplot provide a direct vidualization of how the game marketing changed along the time globally and regionally. In general, th e sales of video game started to surge with a little fluctuation between 1995 and 200 3, while the sales drrped greatlly after 2008, which might cause by the financial cri sis in 2008. NA sales outweights the sales in other regions until 2008 when it witnes sed a major fall.

Top 2 Publisher, Genre, Platform and games by Sales each Year

```
#Exploratory Data Analysis and Visulasition
vgsales <- vgsales %>% gather(Region, Sales, 7:10)
vgsales$Region <- factor(vgsales$Region)

top2_publisher_year <- vgsales %>%
         group_by(Year, Publisher) %>%
         summarize(Sales = sum(Global_Sales)) %>%
         top_n(2)
```

```
## Selecting by Sales
```

```
top2_Genre_year <- vgsales %>%
         group_by(Year, Genre) %>%
         summarize(Sales = sum(Global_Sales)) %>%
         top_n(2)
```

```
## Selecting by Sales
```

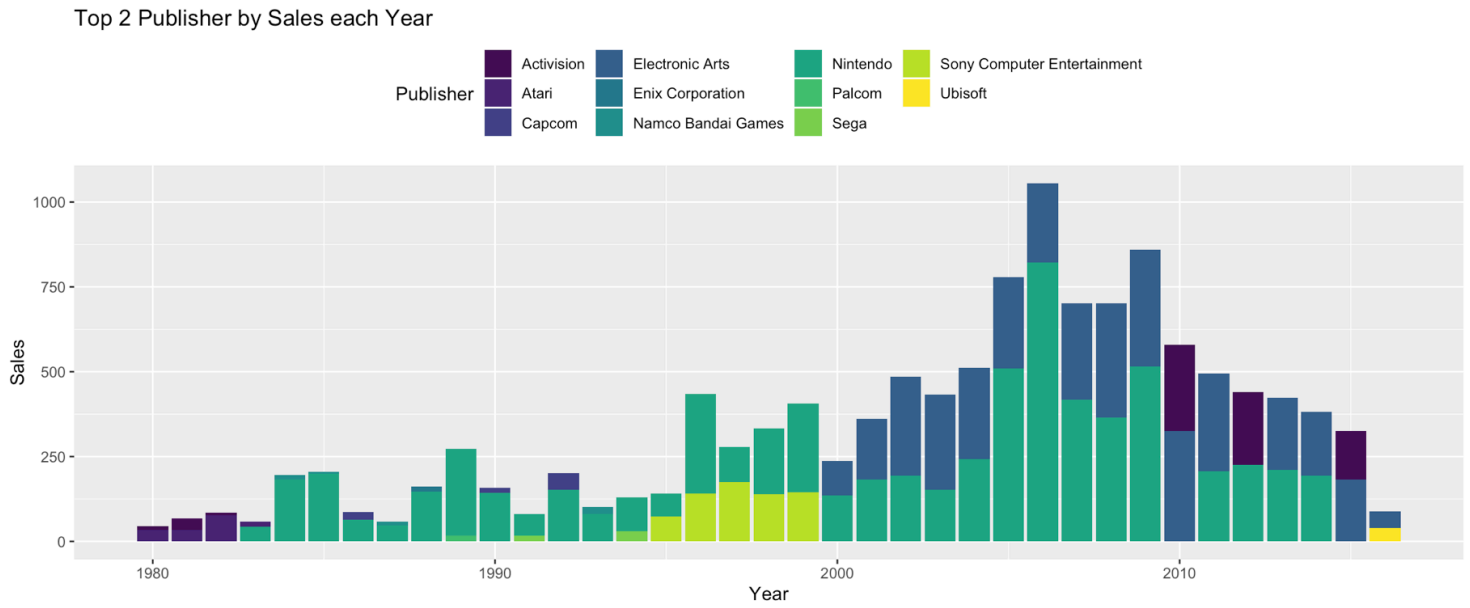
```
top2_platforms_year <- vgsales %>%
             group_by(Year, Platform) %>%
             summarize(Sales = sum(Global_Sales)) %>%
             arrange(desc(Sales)) %>%
             top_n(2)
```

```
## Selecting by Sales
```

```
top_games_year <- vgsales %>%
             group_by(Year, Name) %>%
             summarize(Sales = sum(Global_Sales)) %>%
             arrange(desc(Sales)) %>%
             top_n(1)
```

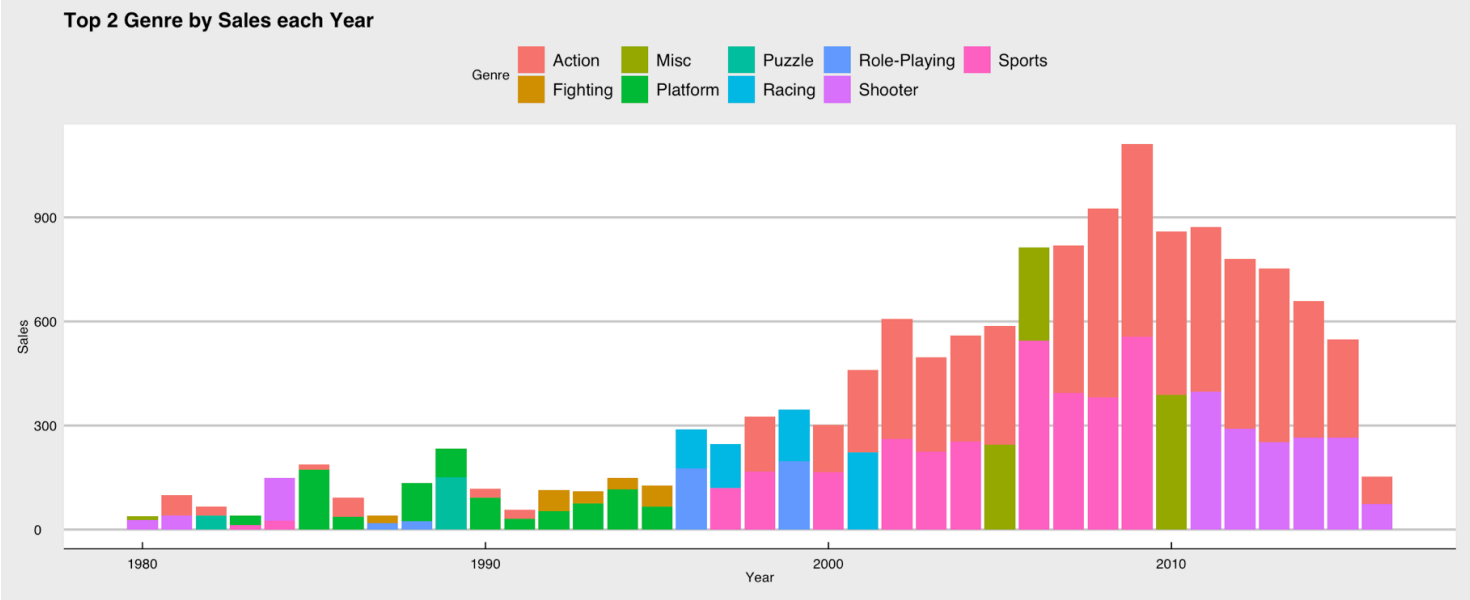
```
## Selecting by Sales
```

```
ggplot(top2_publisher_year, aes(Year, Sales, fill = Publisher)) +
  geom_bar(stat = "identity") +
  scale_color_viridis(discrete = TRUE, option = "D")+
  scale_fill_viridis(discrete = TRUE) +
  ggtitle("Top 2 Publisher by Sales each Year") +
  theme(legend.position = "top")
```



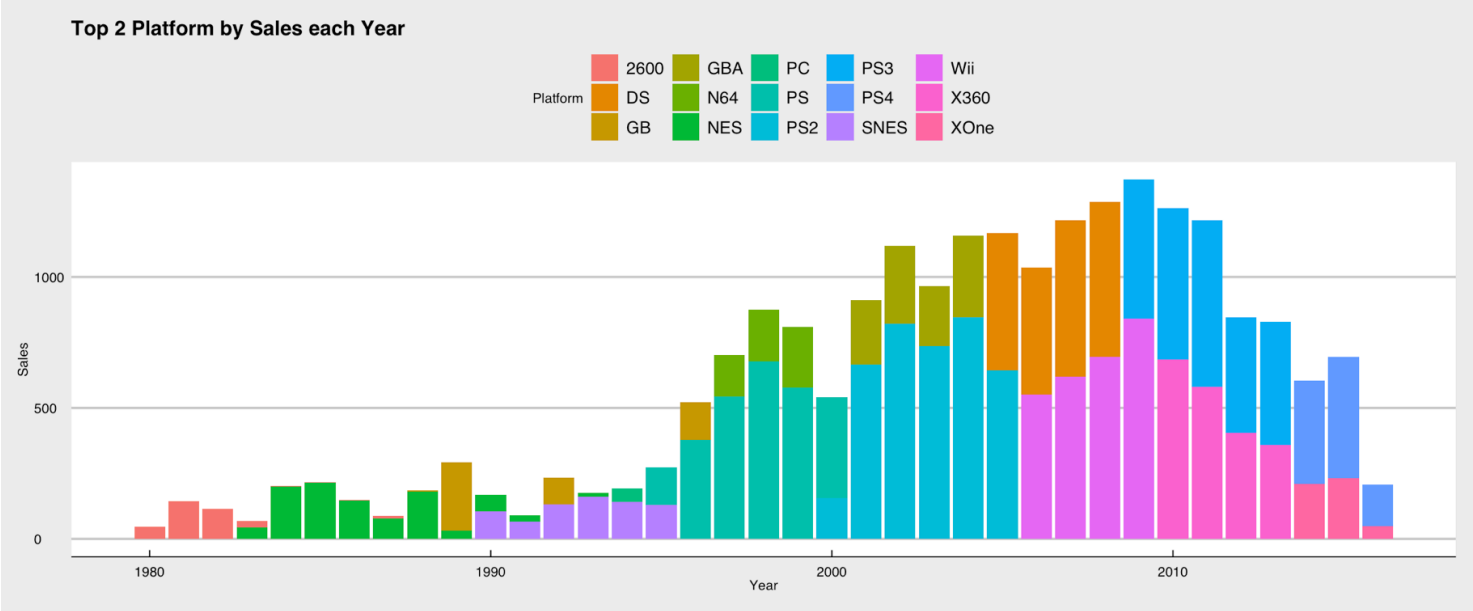

```
#According to sales, Nintendo and EA are the most dominating in the video game industry.
```

```
ggplot(top2_Genre_year, aes(Year, Sales, fill = Genre)) +
  geom_bar(stat = "identity") +
  theme_economist_white(base_size=8) +
  ggtitle("Top 2 Genre by Sales each Year")  +
  theme(legend.position = "top")
```



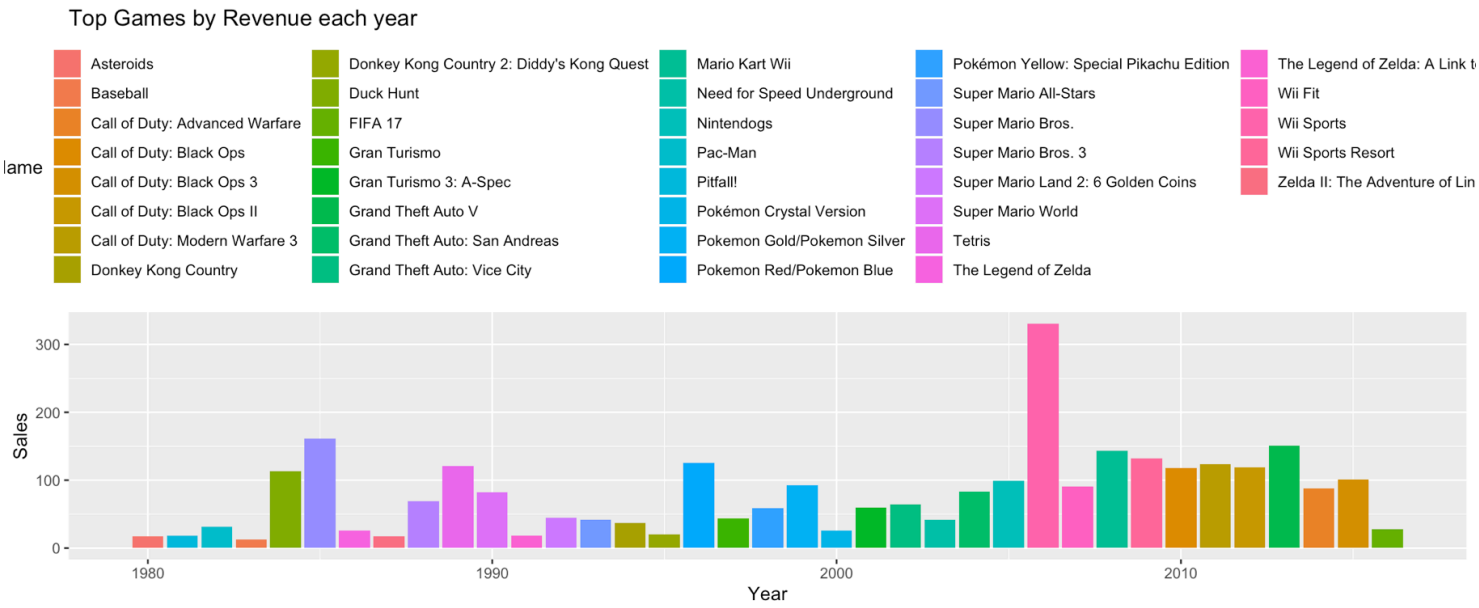

```
#According to sales, action games generated the most sales from 2001 to 2016 which could also be seen from the rise of games like league of legends and DOTA in real life, followed by sports game before 2010 and shooter games after 2010.
```

```
ggplot(top2_platforms_year, aes(Year, Sales, fill = Platform)) +
  geom_bar(stat = "identity") +
  theme_economist_white(base_size=8) +
  scale_color_viridis(discrete = TRUE, option = "B") +
  ggtitle("Top 2 Platform by Sales each Year")  +
  theme(legend.position = "top")
```




```
#The most popular platform witnessed a quite similar trend across the years. Even tho
ugh players' preference towards platforms changes over time, the popularily would rem
ian the same for a period of time.
#One thing worth noticing is that Nintendo in troduced the SNES around 1990, which wa
s known for moving the video games industry forward, starting the uptrend of the sale
s and relase of games in the industry as shown in the chart of "Game release and sale
s by year".
```

```
ggplot(top_games_year, aes(Year, Sales, fill = Name)) +
  geom_bar(stat = "identity") +

  ggtitle("Top Games by Revenue each year") +
  theme(legend.position = "top")
```

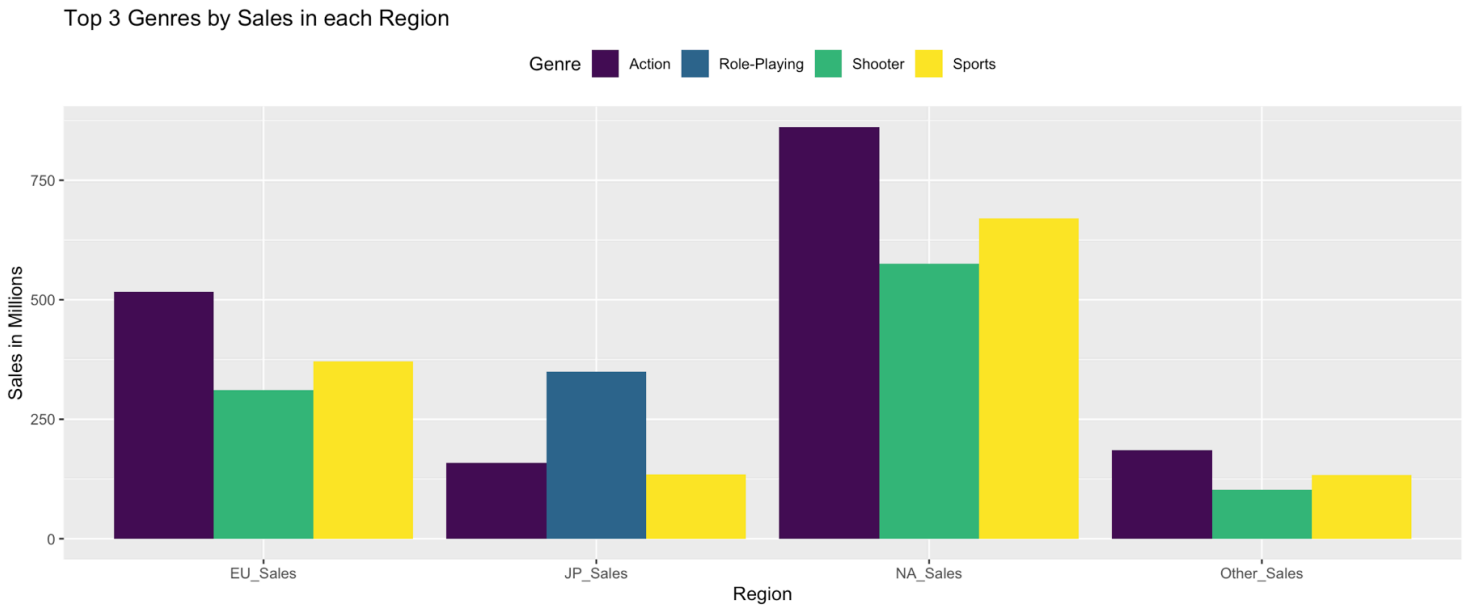


#Players' preferance towards games changes over time. In arround 1990s,Tetris and Sup per Marro world were introduce, which marked the rise of video games industry. In 200 8, when the sales reaching the peaking point, Grand theft auto attrcted the most atte ntion.

```
top3_genres_region <- vgsales %>%
             group_by(Region, Genre) %>%
             summarize(Sales = sum(Sales)) %>%
             arrange(desc(Sales)) %>%
             top_n(3)
```

```
## Selecting by Sales
```

```
library(wesanderson)
ggplot(top3_genres_region, aes(Region, Sales, fill = Genre)) +
  geom_bar(position = "dodge", stat = "identity")  +
  scale_color_viridis(discrete = TRUE, option = "D")+
  scale_fill_viridis(discrete = TRUE) +
  ggtitle("Top 3 Genres by Sales in each Region") +
  ylab("Sales in Millions") +
  xlab("Region") +
  theme(legend.position = "top")
```



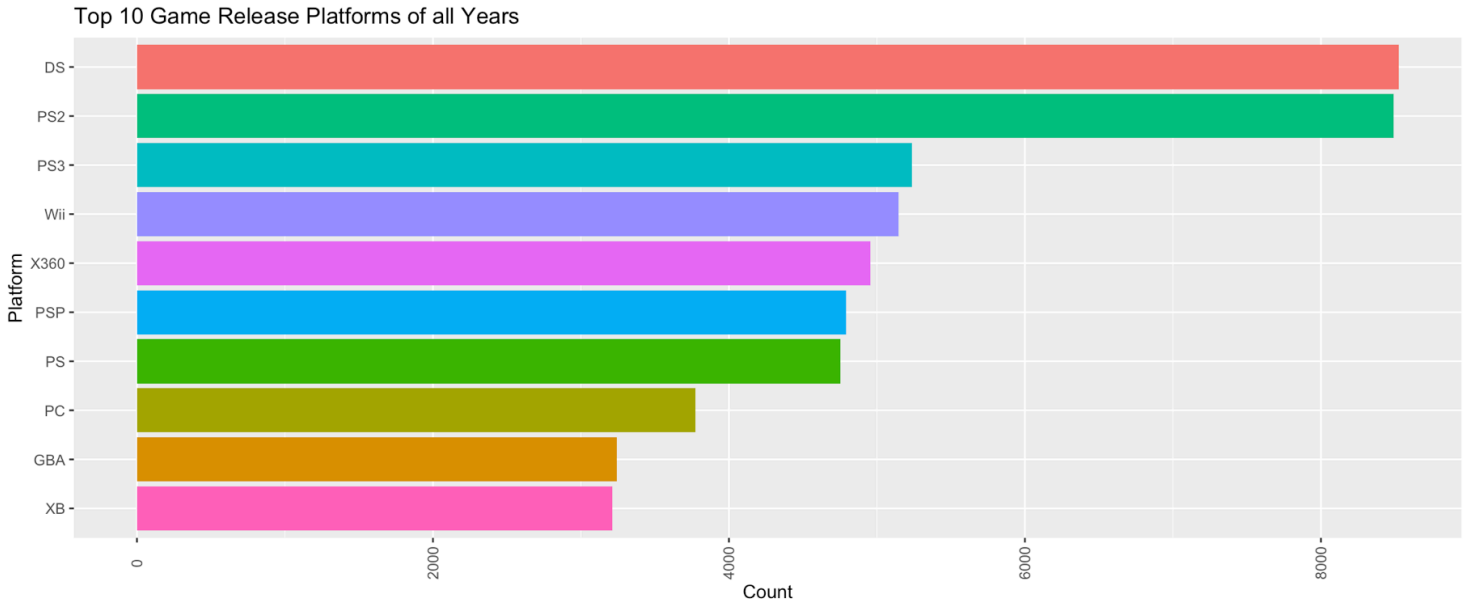
#Interstingly, the top 3 genres in North Amaerica, Europe and other regions were the same, which are action, shooter and sports. However, in Japan, role-playing was in th e dominating position, followed by action and sports.

Following the previous analysis of “Platform”“Publisher”“Genre” VS. “Sales”, supplement it with “Platform”“Publisher”“Genre” VS. “Number Released”

```
#Which platform has released most of the video games?
game_released_by_platform = subset(vgsales,select=c(Platform,Year,Name))
length(unique(game_released_by_platform$Platform))
```

```
## [1] 31
```

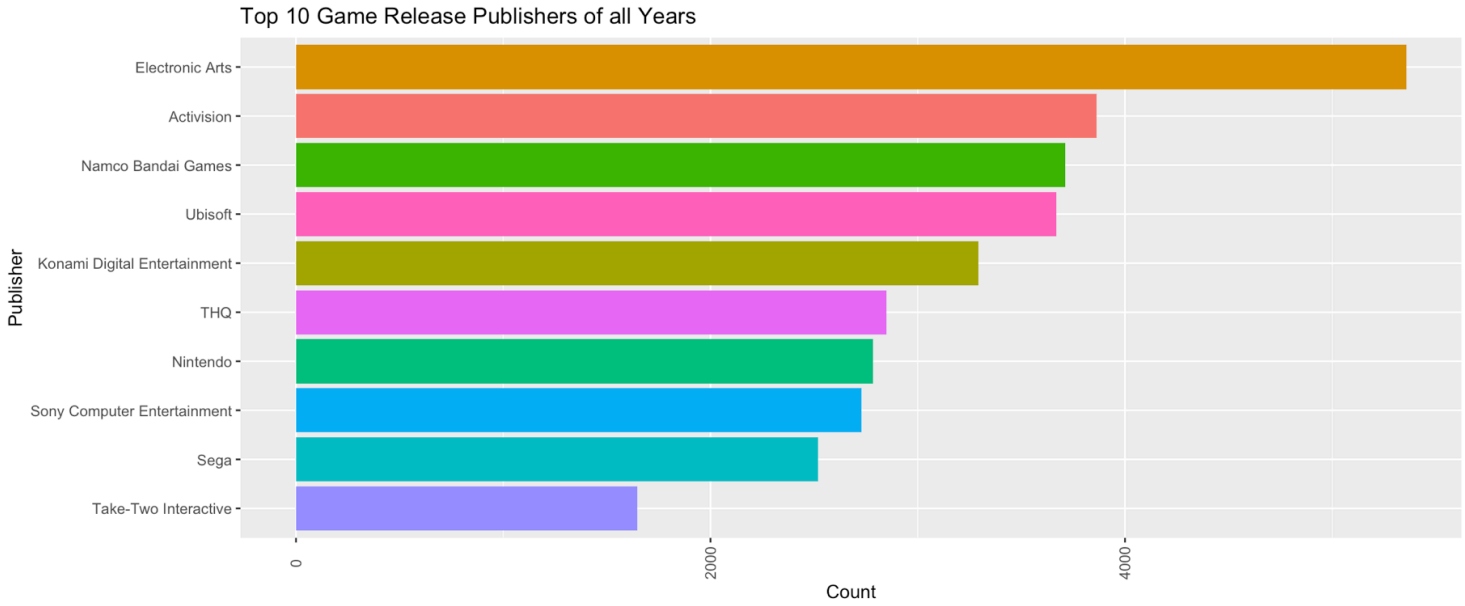
```
game_released_by_platform = game_released_by_platform  %>% group_by(Platform) %>% sum
marise(count=n()) %>% arrange(desc(count))
ggplot(head(game_released_by_platform,10),aes(reorder(Platform,count),count,fill=Plat
form))+geom_bar(stat="identity")+theme(axis.text.x = element_text(angle=90,vjust=0.5)
,legend.position="none")+ggtitle("Top 10 Game Release Platforms of all Years")+coord_
flip()+labs(x="Platform",y="Count")
```




```
#DS and PS2 have provided the most of the games. However, game players showed no loya
lty to these two platforms. As analyzed in the "Top Platform by Sales each year" char
t, the loyalty of game players is not come from the number of game released per year,
but the quality of games themselves or other factors.

#Which publisher has released most of the video games?
game_released_by_publisher = subset(vgsales,select=c(Publisher,Year,Name))
length(unique(game_released_by_publisher$Publisher))
```

```
## [1] 577
```

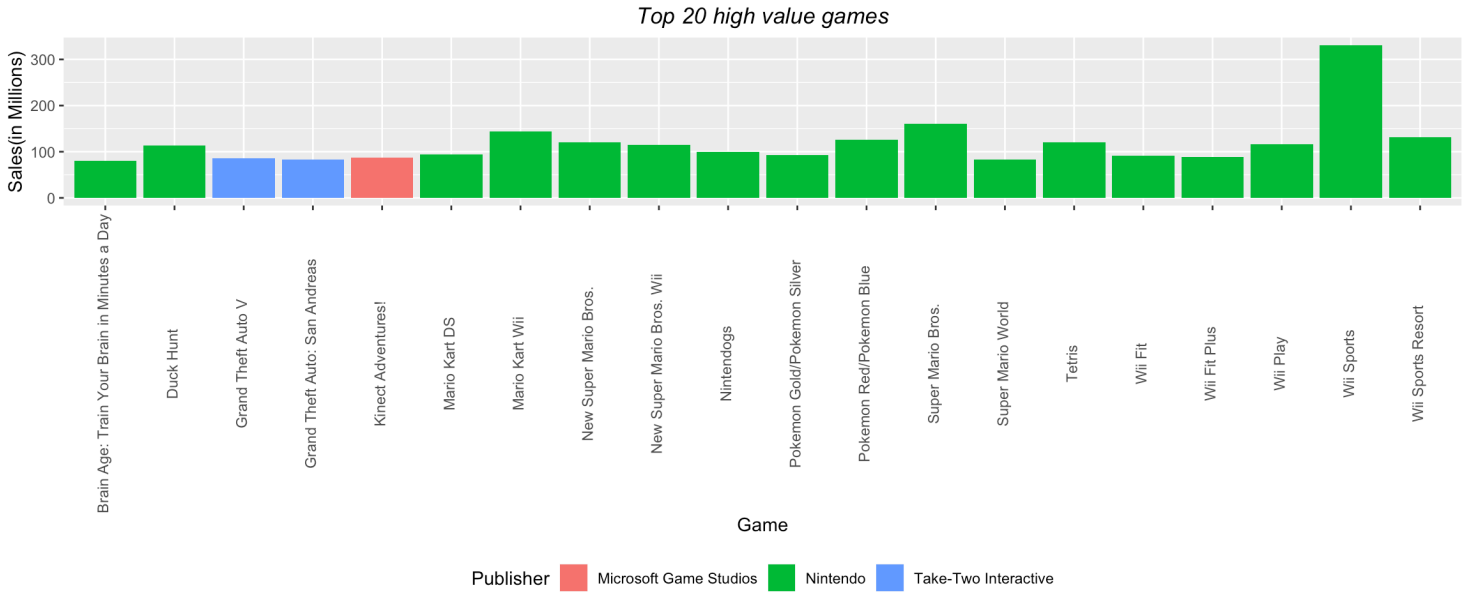
```
game_released_by_publisher = game_released_by_publisher  %>% group_by(Publisher) %>%
summarise(count=n()) %>% arrange(desc(count))
ggplot(head(game_released_by_publisher,10),aes(reorder(Publisher,count),count,fill=Pu
blisher))+geom_bar(stat="identity")+theme(axis.text.x = element_text(angle=90,vjust=0
.5),legend.position="none")+ggtitle("Top 10 Game Release Publishers of all Years")+co
ord_flip()+labs(x="Publisher",y="Count")
```



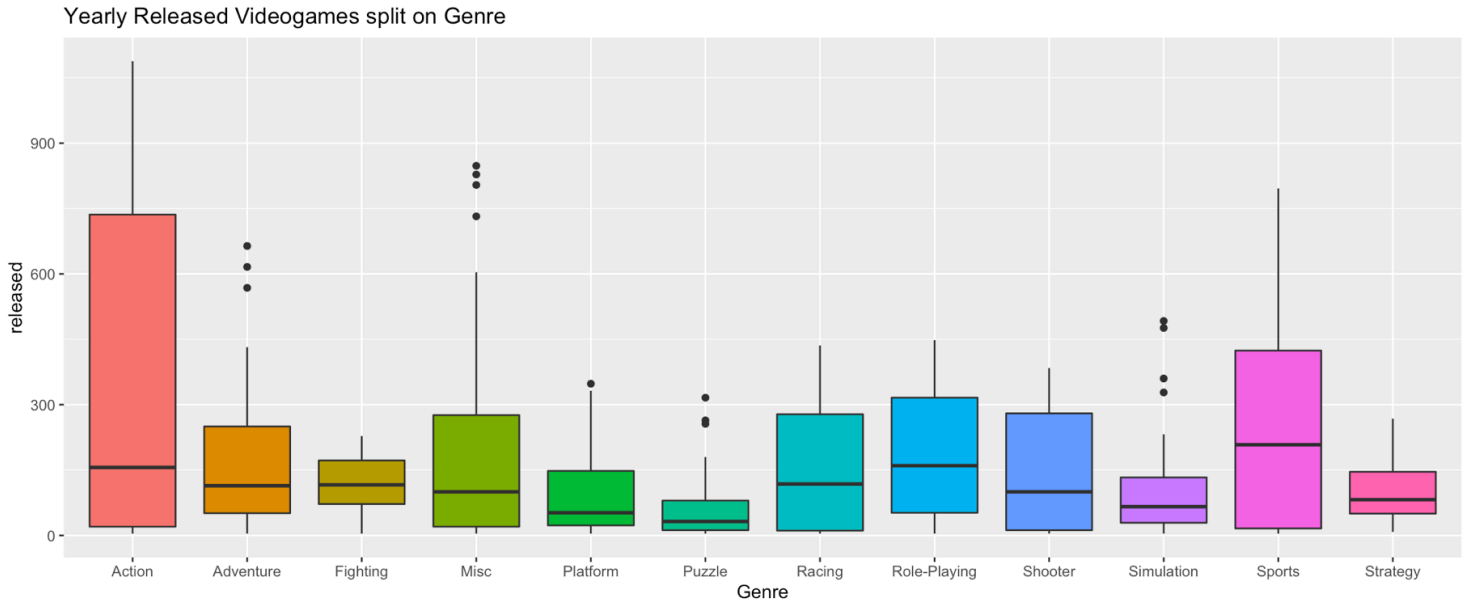
#EA has published the most games of all the years. Avtivison, NBG and Ubisoft rank 2nd,3rd and 4th respectively, with close volume. Combined with "Publisher" VS. "Sales" chart, it makes sense that EA has dominated the market share by quantity (the quality is also guaranteed). However, Nintendo ranks the 7th in number released but still dominates the game market by its quality and fame. As a result, if developing a new vedio game, we suggest publish it through Nintendo.

#Which game has generated the highest revenue (sales) ?

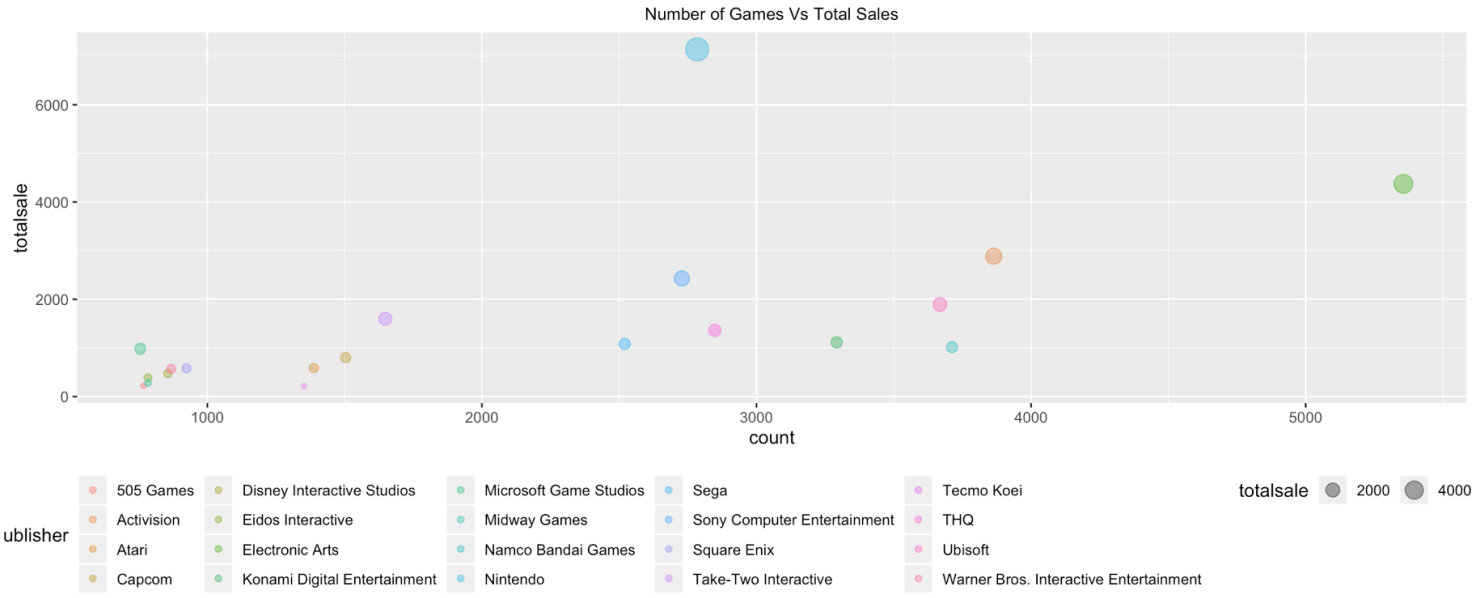
```
game_revenue = vgsales %>% select(Name,Global_Sales,Publisher) %>% arrange(desc(Global_Sales))
ggplot(head(game_revenue,80),aes(factor(Name),Global_Sales,fill=Publisher))+geom_bar(stat="identity")+theme(axis.text.x = element_text(angle=90,vjust=0.3),plot.title = element_text(hjust=0.5,face='italic'),legend.position="bottom")+ggtitle("Top 20 high value games")+labs(x="Game",y="Sales(in Millions)")
```




```
# To further prove the conclusion of Nintendo, we plotted the "Sales" VS. "Single Game" and fount out that Nintendo dominates the market with 17 of the 20 top games released including Duck Hunt, Mario.This is followed by Take-Two Interactive with the famous GTA game.

#Yearly Release Videogames Split on Genre
yearly_videogame_by_genre <- vgsales %>% filter(!is.na(Year)) %>% group_by(Year, Genre) %>% summarize(released = n())

ggplot(yearly_videogame_by_genre, aes(x=Genre, y=released, fill=Genre)) +
  geom_boxplot() + theme(legend.position="none") +
  ggtitle("Yearly Released Videogames split on Genre")
```



```
#After analyzing the Top 3 Genres by Sales in each Region chart, we would like to fur
ther analyze the number of game released by year and genre, to see whether there is a
correlation between the number released each genre and the sales. According the the b
oxplot, the top 3 genres in terms of the released number are Action , Sports, followe
d by Role-Playing and Racing. Compared with the results in the Top 3 Genres by Sales
in each Region chart, Action, Sports and Role-Playing all rank high in both charts. W
e can reasonably assume that high released number will result in high sales volume, v
ice versa.

#How the sales are impacted by the number of release for different Publishers
top20pub=vgsales %>% group_by(Publisher) %>% summarise(count=n()) %>% arrange(desc(co
unt)) %>% head(20)
game_release_by_revenue= vgsales[vgsales$Publisher %in% top20pub$Publisher,]
game_release_by_revenue = game_release_by_revenue %>% group_by(Publisher) %>% summari
se(totalsale=sum(Global_Sales)) %>% arrange(desc(totalsale))
game_release_by_revenue = merge(top20pub,game_release_by_revenue,by="Publisher")
ggplot(game_release_by_revenue,aes(x=count,y=totalsale,col=factor(Publisher),size=tot
alsale))+
geom_point(alpha=0.4)+theme(legend.position="bottom",plot.title = element_text(size=1
0,hjust=0.5))+labs(title="Number of Games Vs Total Sales",col="Publisher")
```



#This chart proves that the impact of number released is not as high for the renowned publisher, Nintendo for example. They generated high sales even when less games were introduced, thank you their brand equity and the quality of the games.

Regression Model to Predict Sales in One Market based on the sales of other Markets

#We have alread seen that North america, Europe and Other countries(Except Japan) have a strong correlation with Global and eachother's sales. #Let's predict the sales in North American market based on the sales in remaining markets. ##Hypotheses: There is no significant change in NA_Sales with respect to sales in other countries

Consider the following regression model-

```
vgsales2 <- data.frame(read.csv("/Users/iris/Downloads/vgsalesCleaned.csv"),sheet = 1
, header = TRUE)
reg1 <- NA_Sales ~ EU_Sales + JP_Sales + Other_Sales
lm1 <- lm(reg1,data=vgsales2)
summary(reg1)
```

```
##  Length   Class    Mode
##       3 formula    call
```

```
#Critical Summary Statistics:
#p value for all countries is less than 0.05. Thus, we fail to accept the Null Hypoth
esis. Meaning, there is a relation between NA_Sales and sales of other countries in t
he decreasing order of  EU_Sales, Other_Sales and JP_Sales
```

##Regression Model to Predit Sales in Different Countries according to Genre and Platform ##1. To Analyse the change in NA_Sales w.r.t genre and platform

```
lm2 <- lm( NA_Sales ~ Genre , data = vgsales2)
summary(lm2)
```

```
##
## Call:
## lm(formula = NA_Sales ~ Genre, data = vgsales2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.509 -0.235 -0.154 -0.015 41.199
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.265132   0.014307  18.531  < 2e-16 ***
## GenreAdventure    -0.185148   0.026946  -6.871 6.60e-12 ***
## GenreFighting     -0.001089   0.031630  -0.034   0.9725
## GenreMisc         -0.029764   0.024367  -1.221   0.2219
## GenrePlatform      0.243989   0.031051   7.858 4.15e-15 ***
## GenrePuzzle       -0.051455   0.037011  -1.390   0.1645
## GenreRacing        0.026001   0.027337   0.951   0.3416
## GenreRole-Playing -0.043174   0.025631  -1.684   0.0921 .
## GenreShooter       0.183510   0.026900   6.822 9.30e-12 ***
## GenreSimulation   -0.051525   0.031408  -1.641   0.1009
## GenreSports        0.025705   0.022214   1.157   0.2472
## GenreStrategy     -0.164091   0.034564  -4.747 2.08e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8156 on 16313 degrees of freedom
## Multiple R-squared:  0.0152, Adjusted R-squared:  0.01454
## F-statistic: 22.89 on 11 and 16313 DF,  p-value: < 2.2e-16
```

```
#Factors with p value greater than 0.05 will have a significant change in NA_SAles. A
lso, the values of model coefficients determine the ranking of the effect.
#For example: Genre Racing, Sports and Shooter have the maximum impact on NA_Sales
```

##2. To Analyse the change in EA_Sales w.r.t genre and platform

```
lm3 <- lm( EU_Sales ~ Genre , data = vgsales2)
summary(lm3)
```

```
##
## Call:
## lm(formula = EU_Sales ~ Genre, data = vgsales2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2422 -0.1428 -0.1075 -0.0300 28.8588
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.158908   0.008884  17.887  < 2e-16 ***
## GenreAdventure    -0.108916   0.016732  -6.510 7.76e-11 ***
## GenreFighting     -0.039290   0.019641  -2.000 0.045465 *
## GenreMisc         -0.033867   0.015130  -2.238 0.025213 *
## GenrePlatform      0.070168   0.019281   3.639 0.000274 ***
## GenrePuzzle       -0.070414   0.022981  -3.064 0.002188 **
## GenreRacing        0.033849   0.016975   1.994 0.046160 *
## GenreRole-Playing -0.031389   0.015915  -1.972 0.048599 *
## GenreShooter       0.083253   0.016704   4.984 6.29e-07 ***
## GenreSimulation   -0.025888   0.019502  -1.327 0.184390
## GenreSports        0.002264   0.013793   0.164 0.869615
## GenreStrategy     -0.092018   0.021462  -4.287 1.82e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5065 on 16313 degrees of freedom
## Multiple R-squared:  0.009808,   Adjusted R-squared:  0.00914
## F-statistic: 14.69 on 11 and 16313 DF,  p-value: < 2.2e-16
```

```
#Using the same reasoning as above, Genres Sports has the highest effect on EA_Sales
```

##3. To Analyse the change in JP_Sales w.r.t genre and platform

```
lm4 <- lm( JP_Sales ~ Genre , data = vgsales2)
summary(lm4)
```

```
##
## Call:
## lm(formula = JP_Sales ~ Genre, data = vgsales2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2381 -0.0624 -0.0488 -0.0288  9.9819
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.048800   0.005375   9.079  < 2e-16 ***
## GenreAdventure    -0.008040   0.010123  -0.794 0.427078
## GenreFighting      0.055446   0.011883   4.666 3.09e-06 ***
## GenreMisc          0.013580   0.009154   1.484 0.137960
## GenrePlatform      0.100344   0.011665   8.602  < 2e-16 ***
## GenrePuzzle        0.050464   0.013904   3.629 0.000285 ***
## GenreRacing       -0.002625   0.010270  -0.256 0.798231
## GenreRole-Playing  0.189331   0.009629  19.662  < 2e-16 ***
## GenreShooter      -0.019018   0.010106  -1.882 0.059865 .
## GenreSimulation    0.025865   0.011799   2.192 0.028386 *
## GenreSports        0.009690   0.008345   1.161 0.245617
## GenreStrategy      0.024265   0.012985   1.869 0.061678 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3064 on 16313 degrees of freedom
## Multiple R-squared:  0.03348,    Adjusted R-squared:  0.03283
## F-statistic: 51.38 on 11 and 16313 DF,  p-value: < 2.2e-16
```

```
#Using the same reasoning as above, Genres Role Playing, Fighting and Platforms will have the highest effect on JP_Sales
```

##4. To Analyse the change in Global_Sales w.r.t genre and platform

```
lm5 <- lm( Global_Sales ~ Genre , data = vgsales2)
summary(lm5)
```

```
##
## Call:
## lm(formula = Global_Sales ~ Genre, data = vgsales2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.937 -0.461 -0.308 -0.038 82.172
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.530034   0.027308  19.409  < 2e-16 ***
## GenreAdventure    -0.346021   0.051431  -6.728 1.78e-11 ***
## GenreFighting      0.001126   0.060372   0.019   0.9851
## GenreMisc         -0.063589   0.046509  -1.367   0.1716
## GenrePlatform      0.416484   0.059266   7.027 2.19e-12 ***
## GenrePuzzle       -0.105831   0.070642  -1.498   0.1341
## GenreRacing        0.062764   0.052178   1.203   0.2290
## GenreRole-Playing  0.098002   0.048922   2.003   0.0452 *
## GenreShooter       0.270434   0.051344   5.267 1.40e-07 ***
## GenreSimulation   -0.071561   0.059948  -1.194   0.2326
## GenreSports        0.038213   0.042399   0.901   0.3675
## GenreStrategy     -0.271924   0.065972  -4.122 3.78e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.557 on 16313 degrees of freedom
## Multiple R-squared:  0.01215,    Adjusted R-squared:  0.01149
## F-statistic: 18.25 on 11 and 16313 DF,  p-value: < 2.2e-16
```

```
#Using the same reasoning as above, Genres Action, Sports and Shooter games will have the highest effect on JP_Sales
```

Forecasting

In [177]:
```
import statsmodels
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

In [178]:
```
data=pd.read_csv('vgsalesCleaned.csv')
```

In [179]:
```
data
```

Out[179]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_S
0	1	Wii Sports	Wii	2006	Sports	Nintendo	41.49	2
1	2	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	
2	3	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.85	1
3	4	Wii Sports Resort	Wii	2009	Sports	Nintendo	15.75	1
4	5	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	11.27	
5	6	Tetris	GB	1989	Puzzle	Nintendo	23.20	
6	7	New Super Mario Bros.	DS	2006	Platform	Nintendo	11.38	
7	8	Wii Play	Wii	2006	Misc	Nintendo	14.03	
8	9	New Super Mario Bros. Wii	Wii	2009	Platform	Nintendo	14.59	
9	10	Duck Hunt	NES	1984	Shooter	Nintendo	26.93	
10	11	Nintendogs	DS	2005	Simulation	Nintendo	9.07	1
11	12	Mario Kart DS	DS	2005	Racing	Nintendo	9.81	
12	13	Pokemon Gold/Pokemon Silver	GB	1999	Role-Playing	Nintendo	9.00	
13	14	Wii Fit	Wii	2007	Sports	Nintendo	8.94	
14	15	Wii Fit Plus	Wii	2009	Sports	Nintendo	9.09	
15	16	Kinect Adventures!	X360	2010	Misc	Microsoft Game Studios	14.97	
16	17	Grand Theft Auto V	PS3	2013	Action	Take-Two Interactive	7.01	

17	18	Grand Theft Auto: San Andreas	PS2	2004	Action	Take-Two Interactive	9.43
18	19	Super Mario World	SNES	1990	Platform	Nintendo	12.78
19	20	Brain Age: Train Your Brain in Minutes a Day	DS	2005	Misc	Nintendo	4.75
20	21	Pokemon Diamond/Pokemon Pearl	DS	2006	Role-Playing	Nintendo	6.42
21	22	Super Mario Land	GB	1989	Platform	Nintendo	10.83
22	23	Super Mario Bros. 3	NES	1988	Platform	Nintendo	9.54
23	24	Grand Theft Auto V	X360	2013	Action	Take-Two Interactive	9.63
24	25	Grand Theft Auto: Vice City	PS2	2002	Action	Take-Two Interactive	8.41
25	26	Pokemon Ruby/Pokemon Sapphire	GBA	2002	Role-Playing	Nintendo	6.06
26	27	Pokemon Black/Pokemon White	DS	2010	Role-Playing	Nintendo	5.57
27	28	Brain Age 2: More Training in Minutes a Day	DS	2005	Puzzle	Nintendo	3.44
28	29	Gran Turismo 3: A-Spec	PS2	2001	Racing	Sony Computer Entertainment	6.85
29	30	Call of Duty: Modern Warfare 3	X360	2011	Shooter	Activision	9.03
...	...	...	...	...	...	...	...
16295	16571	Fujiko F. Fujio Characters: Great Assembly! Sl...	3DS	2014	Action	Namco Bandai Games	0.00
16296	16572	Farming 2017 - The Simulation	PS4	2016	Simulation	UIG Entertainment	0.00
16297	16573	Grisaia no Kajitsu: La Fruit de la Grisaia	PSP	2013	Adventure	Prototype	0.00
16298	16574	Resident Evil 4 HD	XOne	2016	Shooter	Capcom	0.01
16299	16575	Scarlett: Nichijou no Kyoukaisen	PS2	2008	Adventure	Kadokawa Shoten	0.00
16300	16576	Mini Desktop Racing	Wii	2007	Racing	Popcorn Arcade	0.01
16301	16577	Yattaman Wii: BikkuriDokkiri Machine de Mou Ra...	Wii	2008	Racing	Takara Tomy	0.00

16302	16578	Neo Angelique Special	PSP	2008	Adventure	Tecmo Koei	0.00
16303	16579	Rugby Challenge 3	XOne	2016	Sports	Alternative Software	0.00
16304	16580	Damnation	PC	2009	Shooter	Codemasters	0.00
16305	16581	Outdoors Unleashed: Africa 3D	3DS	2011	Sports	Mastiff	0.01
16306	16582	Real Rode	PS2	2008	Adventure	Kadokawa Shoten	0.00
16307	16583	Fit & Fun	Wii	2011	Sports	Unknown	0.00
16308	16584	Planet Monsters	GBA	2001	Action	Titus	0.01
16309	16585	Carmageddon 64	N64	1999	Action	Virgin Interactive	0.01
16310	16586	PGA European Tour	N64	2000	Sports	Infogrames	0.01
16311	16587	Bust-A-Move 3000	GC	2003	Puzzle	Ubisoft	0.01
16312	16588	Breach	PC	2011	Shooter	Destineer	0.01
16313	16589	Secret Files 2: Puritas Cordis	DS	2009	Adventure	Deep Silver	0.00
16314	16590	Mezase!! Tsuri Master DS	DS	2009	Sports	Hudson Soft	0.00
16315	16591	Mega Brain Boost	DS	2008	Puzzle	Majesco Entertainment	0.01
16316	16592	Chou Ezaru wa Akai Hana: Koi wa Tsuki ni Shiru...	PSV	2016	Action	dramatic create	0.00
16317	16593	Eiyuu Densetsu: Sora no Kiseki Material Collec...	PSP	2007	Role-Playing	Falcom Corporation	0.00
16318	16594	Myst IV: Revelation	PC	2004	Adventure	Ubisoft	0.01
16319	16595	Plushees	DS	2008	Simulation	Destineer	0.01
16320	16596	Men in Black II: Alien Escape	GC	2003	Shooter	Infogrames	0.01
16321	16597	Woody Woodpecker in Crazy Castle 5	GBA	2002	Platform	Kemco	0.01
16322	16598	Know How 2	DS	2010	Puzzle	7G//AMES	0.00
16323	16599	SCORE International Baja 1000: The Official Game	PS2	2008	Racing	Activision	0.00
16324	16600	Spirits & Spells	GBA	2003	Platform	Wanadoo	0.01

16325 rows × 11 columns

In [180]:
```
data=pd.DataFrame(data.groupby('Year').sum())
```

In [181]:
```
data
```

Out[181]:

Year	Rank	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1980	29827	10.59	0.67	0.00	0.12	11.38
1981	190488	33.40	1.96	0.00	0.32	35.77
1982	149186	26.92	1.65	0.00	0.31	28.86
1983	56761	7.76	0.80	8.10	0.14	16.79
1984	22910	33.28	2.10	14.27	0.70	50.36
1985	55507	33.73	4.74	14.56	0.92	53.94
1986	35989	12.50	2.84	19.81	1.93	37.07
1987	54701	8.46	1.41	11.63	0.20	21.74
1988	37181	23.87	6.59	15.76	0.99	47.22
1989	40158	45.15	8.44	18.36	1.50	73.45
1990	25110	25.46	7.63	14.88	1.40	49.39
1991	212826	12.76	3.95	14.78	0.74	32.23
1992	208926	33.87	11.71	28.91	1.65	76.16
1993	378761	15.12	4.65	25.33	0.89	45.98
1994	868887	28.15	14.88	33.99	2.20	79.17
1995	1879594	24.82	14.90	45.75	2.64	88.11
1996	1904364	86.64	47.18	57.44	7.68	198.94
1997	1874967	94.75	48.32	48.24	9.13	200.35
1998	2623489	128.36	66.90	50.04	11.03	256.47
1999	2179704	125.94	62.59	52.34	10.04	251.06
2000	2391361	94.50	52.76	42.77	11.62	201.58
2001	3386772	173.98	94.89	39.86	22.76	331.47
2002	6822852	216.16	109.72	41.61	27.27	395.31
2003	6110991	193.59	103.81	34.20	26.01	357.85
2004	5795716	222.37	107.16	41.65	47.23	418.89
2005	7713816	242.24	121.90	54.28	40.55	459.52
2006	9548942	262.75	129.20	73.73	54.40	520.62

2007	10337272	312.14	160.52	59.97	77.60	610.92
2008	12073311	351.14	184.32	60.26	82.34	678.48
2009	12427563	338.74	191.51	62.31	74.74	667.51
2010	11058989	304.22	176.68	59.57	59.89	600.45
2011	10010058	240.98	167.24	53.16	54.35	515.78
2012	5578997	155.53	119.14	51.68	37.97	364.53
2013	4405685	154.87	125.94	47.54	39.84	368.32
2014	4842688	132.31	125.78	39.46	40.00	337.47
2015	6059883	103.34	97.86	34.25	30.09	265.70
2016	3931650	22.66	26.76	13.70	7.75	70.93
2017	47078	0.00	0.00	0.05	0.00	0.05
2020	5959	0.27	0.00	0.00	0.02	0.29

In [182]:
```
data=data.drop(data.index[37])
```

In [183]:
```
data=data.drop(data.index[37])
```

In [184]:
```
data
```

Out[184]:

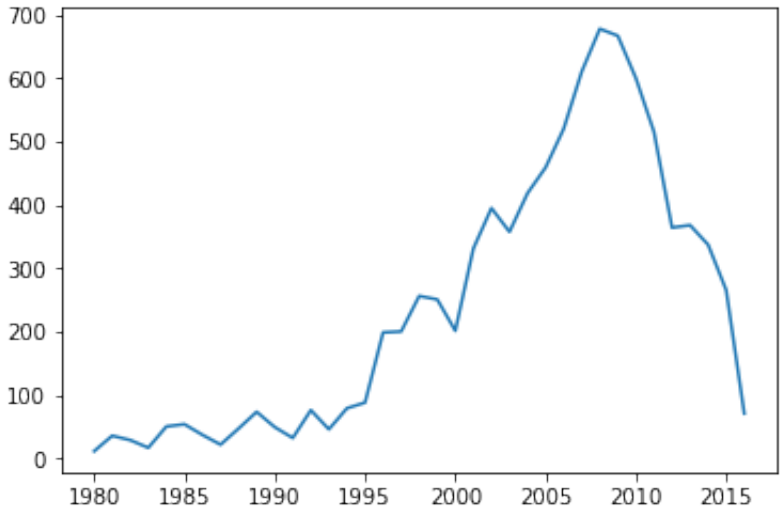
Year	Rank	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1980	29827	10.59	0.67	0.00	0.12	11.38
1981	190488	33.40	1.96	0.00	0.32	35.77
1982	149186	26.92	1.65	0.00	0.31	28.86
1983	56761	7.76	0.80	8.10	0.14	16.79
1984	22910	33.28	2.10	14.27	0.70	50.36
1985	55507	33.73	4.74	14.56	0.92	53.94
1986	35989	12.50	2.84	19.81	1.93	37.07
1987	54701	8.46	1.41	11.63	0.20	21.74
1988	37181	23.87	6.59	15.76	0.99	47.22
1989	40158	45.15	8.44	18.36	1.50	73.45
1990	25110	25.46	7.63	14.88	1.40	49.39
1991	212826	12.76	3.95	14.78	0.74	32.23
1992	208926	33.87	11.71	28.91	1.65	76.16
1993	378761	15.12	4.65	25.33	0.89	45.98

1994	868887	28.15	14.88	33.99	2.20	79.17
1995	1879594	24.82	14.90	45.75	2.64	88.11
1996	1904364	86.64	47.18	57.44	7.68	198.94
1997	1874967	94.75	48.32	48.24	9.13	200.35
1998	2623489	128.36	66.90	50.04	11.03	256.47
1999	2179704	125.94	62.59	52.34	10.04	251.06
2000	2391361	94.50	52.76	42.77	11.62	201.58
2001	3386772	173.98	94.89	39.86	22.76	331.47
2002	6822852	216.16	109.72	41.61	27.27	395.31
2003	6110991	193.59	103.81	34.20	26.01	357.85
2004	5795716	222.37	107.16	41.65	47.23	418.89
2005	7713816	242.24	121.90	54.28	40.55	459.52
2006	9548942	262.75	129.20	73.73	54.40	520.62
2007	10337272	312.14	160.52	59.97	77.60	610.92
2008	12073311	351.14	184.32	60.26	82.34	678.48
2009	12427563	338.74	191.51	62.31	74.74	667.51
2010	11058989	304.22	176.68	59.57	59.89	600.45
2011	10010058	240.98	167.24	53.16	54.35	515.78
2012	5578997	155.53	119.14	51.68	37.97	364.53
2013	4405685	154.87	125.94	47.54	39.84	368.32
2014	4842688	132.31	125.78	39.46	40.00	337.47
2015	6059883	103.34	97.86	34.25	30.09	265.70
2016	3931650	22.66	26.76	13.70	7.75	70.93

In [185]:
```
ts=data['Global_Sales']
```

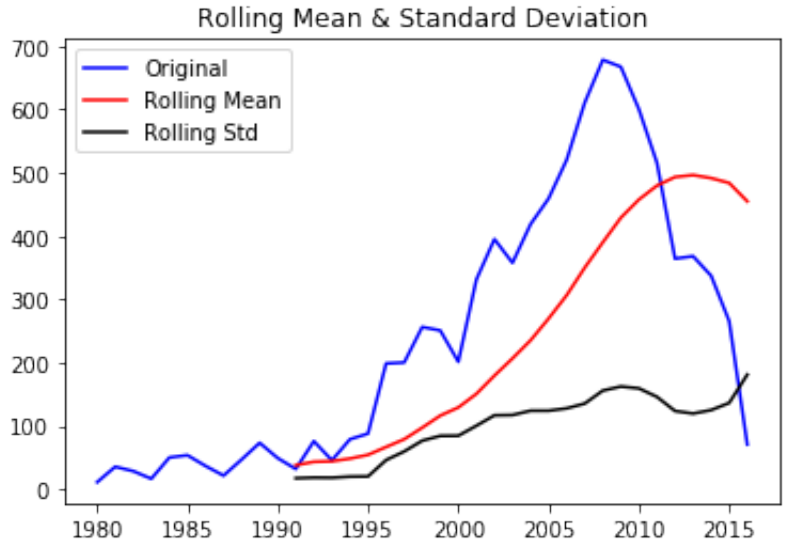
In [186]:
```
plt.plot(ts)
```

Out[186]:
```
[<matplotlib.lines.Line2D at 0x12537ada0>]
```



In [203]:
```
from statsmodels.tsa.stattools import adfuller
def test_stationarity(timeseries):
    rolmean = timeseries.rolling(12).mean()
    rolstd = timeseries.rolling(12).std()
    plt.plot(timeseries, color='blue',label='Original')
    plt.plot(rolmean, color='red', label='Rolling Mean')
    plt.plot(rolstd, color='black', label = 'Rolling Std')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.show()
    print ('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-va
lue','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print (dfoutput)
```

In [204]:
```
test_stationarity(ts)
```

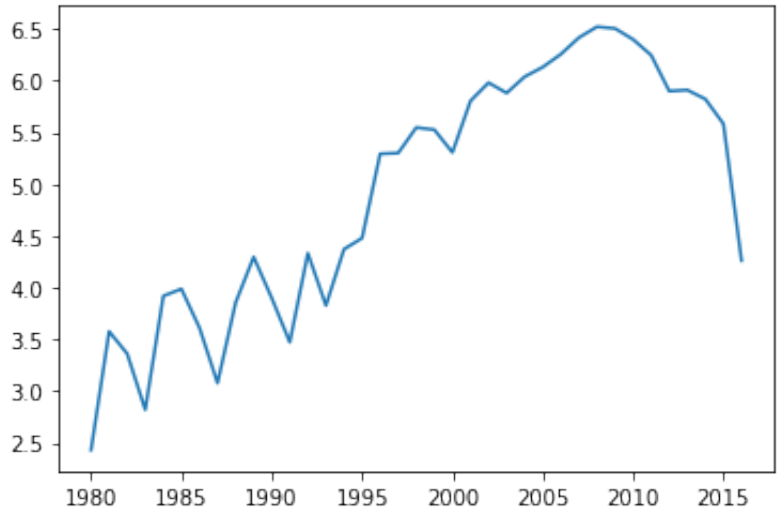



```
Results of Dickey-Fuller Test:
Test Statistic                  -1.211601
p-value                          0.668586
#Lags Used                       1.000000
Number of Observations Used     35.000000
Critical Value (1%)             -3.632743
Critical Value (5%)             -2.948510
Critical Value (10%)            -2.613017
dtype: float64
```

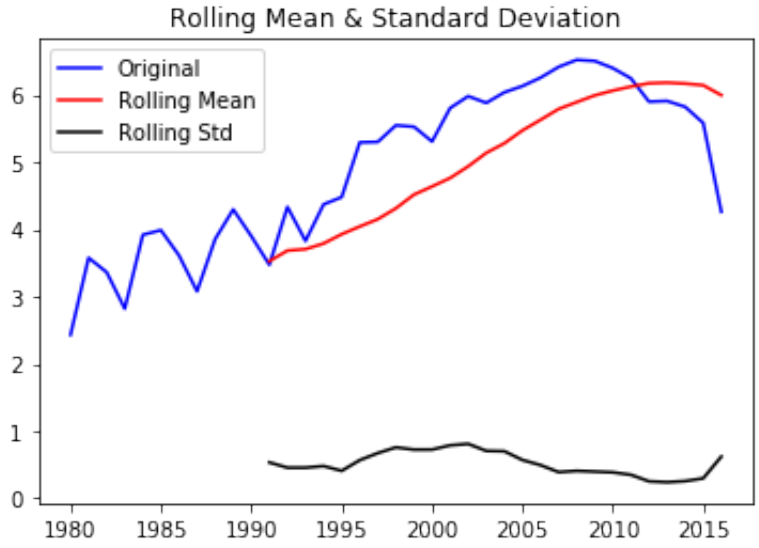
In [187]:
```
ts_log=np.log(ts)
```

In [188]:
```
plt.plot(ts_log)
```

Out[188]:
```
[<matplotlib.lines.Line2D at 0x125469c18>]
```



In [205]:
```
test_stationarity(ts_log)
```

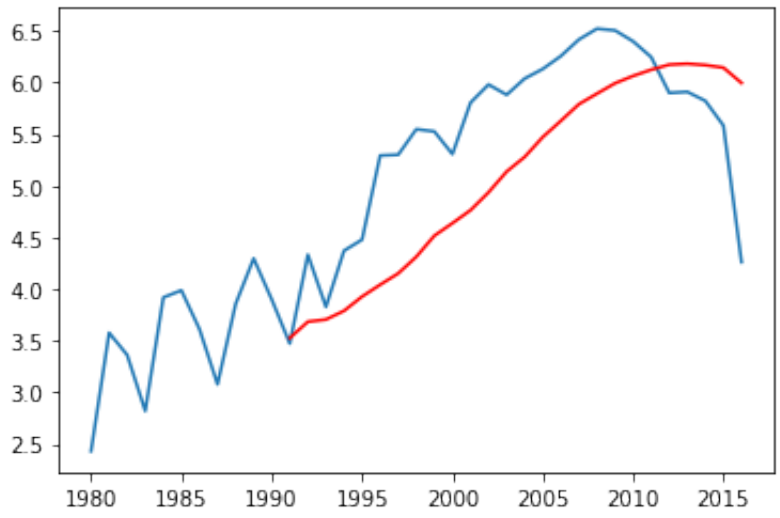



```
Results of Dickey-Fuller Test:
Test Statistic                  -2.146604
p-value                          0.226192
#Lags Used                       0.000000
Number of Observations Used     36.000000
Critical Value (1%)             -3.626652
Critical Value (5%)             -2.945951
Critical Value (10%)            -2.611671
dtype: float64
```

In [192]:
```
moving_avg = ts_log.rolling(12).mean()
```

In [193]:
```
plt.plot(ts_log)
plt.plot(moving_avg,color='red')
```

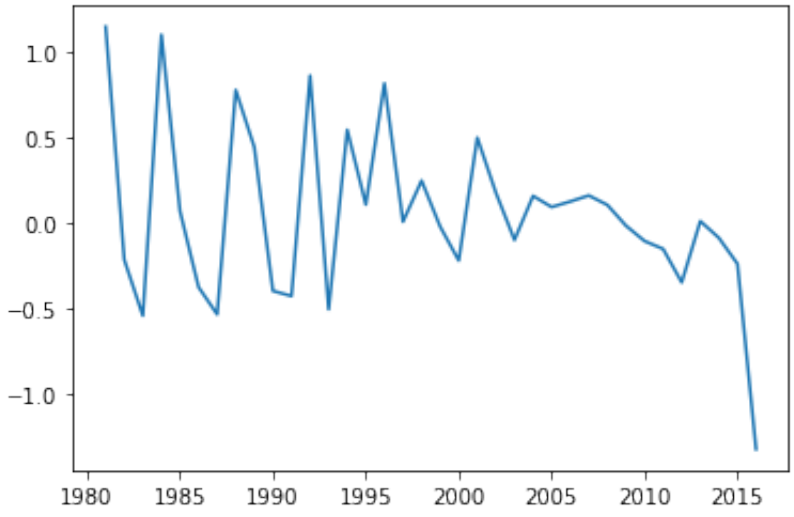
Out[193]:
```
[<matplotlib.lines.Line2D at 0x1254a32e8>]
```



In [194]:
```
ts_log_diff=ts_log-ts_log.shift()
```

In [195]:
```
plt.plot(ts_log_diff)
```

Out[195]:
```
[<matplotlib.lines.Line2D at 0x125db7ef0>]
```



In [206]:
```
from statsmodels.tsa.arima_model import ARIMA
```

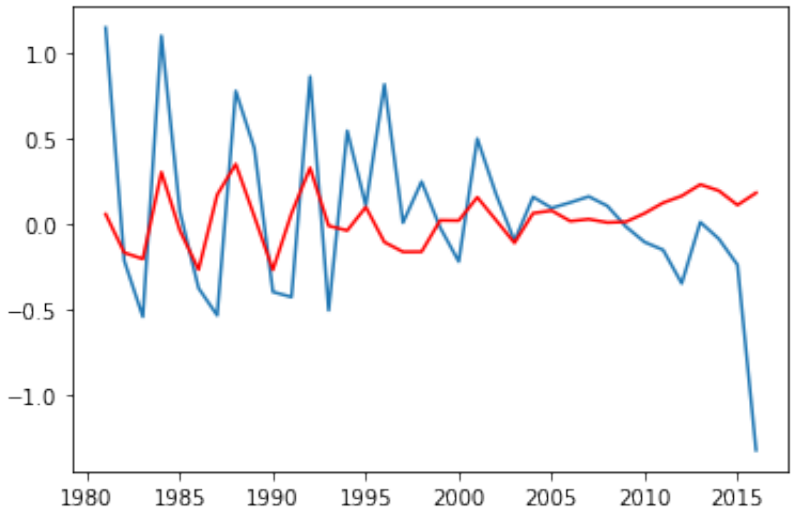
In [207]:
```
from statsmodels.tsa.stattools import acf,pacf
```

In [ ]:
```
##AR Model
```

In [209]:
```
model=ARIMA(ts_log,order=(2,1,0))
results=model.fit(disp=-1)
plt.plot(ts_log_diff)
plt.plot(results.fittedvalues, color='red')
```

```
/anaconda3/lib/python3.7/site-packages/statsmodels/tsa/base/tsa_mo
del.py:221: ValueWarning: An unsupported index was provided and wi
ll be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
/anaconda3/lib/python3.7/site-packages/statsmodels/tsa/base/tsa_mo
del.py:221: ValueWarning: An unsupported index was provided and wi
ll be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
```

Out[209]:
```
[<matplotlib.lines.Line2D at 0x131eafcf8>]
```

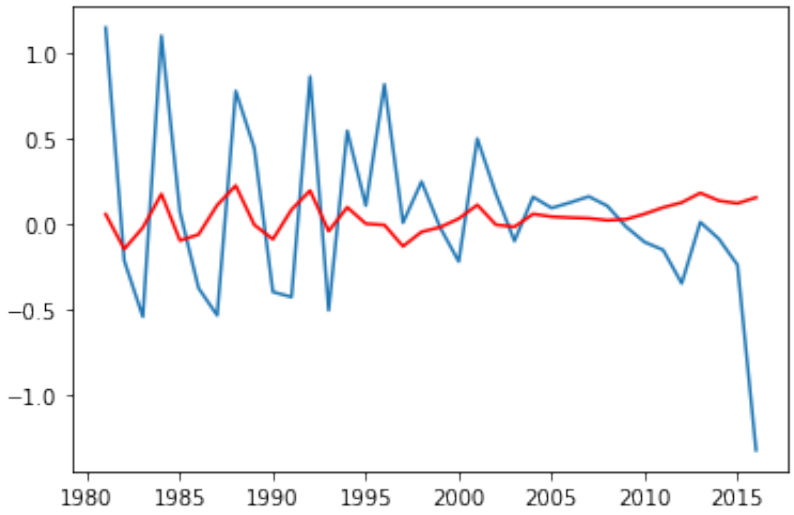


In [ ]:
```
##MA model
```

In [210]:
```
model=ARIMA(ts_log,order=(0,1,2))
results=model.fit(disp=-1)
plt.plot(ts_log_diff)
plt.plot(results.fittedvalues, color='red')
```

```
/anaconda3/lib/python3.7/site-packages/statsmodels/tsa/base/tsa_mo
del.py:221: ValueWarning: An unsupported index was provided and wi
ll be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
/anaconda3/lib/python3.7/site-packages/statsmodels/tsa/base/tsa_mo
del.py:221: ValueWarning: An unsupported index was provided and wi
ll be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
```

Out[210]:
```
[<matplotlib.lines.Line2D at 0x131fcc908>]
```

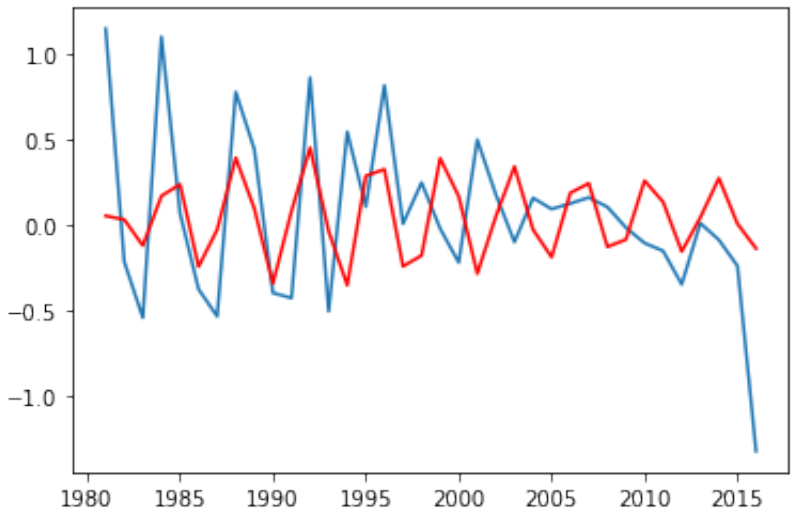


In [ ]:
```
##ARIMA model
```

In [211]:
```
model=ARIMA(ts_log,order=(2,1,2))
results=model.fit(disp=-1)
plt.plot(ts_log_diff)
plt.plot(results.fittedvalues, color='red')
```

```
/anaconda3/lib/python3.7/site-packages/statsmodels/tsa/base/tsa_mo
del.py:221: ValueWarning: An unsupported index was provided and wi
ll be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
/anaconda3/lib/python3.7/site-packages/statsmodels/tsa/base/tsa_mo
del.py:221: ValueWarning: An unsupported index was provided and wi
ll be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
/anaconda3/lib/python3.7/site-packages/statsmodels/base/model.py:4
88: HessianInversionWarning: Inverting hessian failed, no bse or c
ov_params available
  'available', HessianInversionWarning)
```

Out[211]:
```
[<matplotlib.lines.Line2D at 0x1320d3550>]
```



Game Sales Prediction and Modelling

In [63]:
```
import pandas as pd
import numpy as np
```

In [64]:
```
df=pd.read_csv('vgsalesCleaned.csv')
```

In [65]:
```
df.head()
```

Out[65]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales
0	1	Wii Sports	Wii	2006	Sports	Nintendo	41.49	29.02	3.77
1	2	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	3.58	6.81
2	3	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.85	12.88	3.79
3	4	Wii Sports Resort	Wii	2009	Sports	Nintendo	15.75	11.01	3.28
4	5	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	11.27	8.89	10.22

In [66]:
```
del(df['Name'])
```

In [67]:
```
del(df['Rank'])
del(df['Year'])
```

In [68]:
```
df.dtypes
```

Out[68]:
```
Platform         object
Genre            object
Publisher        object
NA_Sales        float64
EU_Sales        float64
JP_Sales        float64
Other_Sales     float64
Global_Sales    float64
dtype: object
```

In [69]:
```
df.shape
```

Out[69]:
```
(16325, 8)
```

In [70]:
```
from sklearn.preprocessing import StandardScaler
```

In [71]:
```
scaler=StandardScaler()
```

In [72]:
```
columns_object=[]
for i in df.columns:
    if df[i].dtypes=='object':
        columns_object.append(i)
```

In [73]:
```
columns_object
```

Out[73]:
```
['Platform', 'Genre', 'Publisher']
```

In [74]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [75]:
```
#df['Platform']=le.fit_transform(df['Platform'])
```

In [76]:
```
#df['Genre']=le.fit_transform(df['Genre'])
```

In [77]:
```
df
```

Out[77]:

	Platform	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Glo
0	Wii	Sports	Nintendo	41.49	29.02	3.77	8.46	
1	NES	Platform	Nintendo	29.08	3.58	6.81	0.77	
2	Wii	Racing	Nintendo	15.85	12.88	3.79	3.31	
3	Wii	Sports	Nintendo	15.75	11.01	3.28	2.96	
4	GB	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	
5	GB	Puzzle	Nintendo	23.20	2.26	4.22	0.58	
6	DS	Platform	Nintendo	11.38	9.23	6.50	2.90	
7	Wii	Misc	Nintendo	14.03	9.20	2.93	2.85	
8	Wii	Platform	Nintendo	14.59	7.06	4.70	2.26	
9	NES	Shooter	Nintendo	26.93	0.63	0.28	0.47	
10	DS	Simulation	Nintendo	9.07	11.00	1.93	2.75	
11	DS	Racing	Nintendo	9.81	7.57	4.13	1.92	
12	GB	Role-Playing	Nintendo	9.00	6.18	7.20	0.71	
13	Wii	Sports	Nintendo	8.94	8.03	3.60	2.15	
14	Wii	Sports	Nintendo	9.09	8.59	2.53	1.79	
15	X360	Misc	Microsoft Game Studios	14.97	4.94	0.24	1.67	

16	PS3	Action	Take-Two Interactive	7.01	9.27	0.97	4.14
17	PS2	Action	Take-Two Interactive	9.43	0.40	0.41	10.57
18	SNES	Platform	Nintendo	12.78	3.75	3.54	0.55
19	DS	Misc	Nintendo	4.75	9.26	4.16	2.05
20	DS	Role-Playing	Nintendo	6.42	4.52	6.04	1.37
21	GB	Platform	Nintendo	10.83	2.71	4.18	0.42
22	NES	Platform	Nintendo	9.54	3.44	3.84	0.46
23	X360	Action	Take-Two Interactive	9.63	5.31	0.06	1.38
24	PS2	Action	Take-Two Interactive	8.41	5.49	0.47	1.78
25	GBA	Role-Playing	Nintendo	6.06	3.90	5.38	0.50
26	DS	Role-Playing	Nintendo	5.57	3.28	5.65	0.82
27	DS	Puzzle	Nintendo	3.44	5.36	5.32	1.18
28	PS2	Racing	Sony Computer Entertainment	6.85	5.09	1.87	1.16
29	X360	Shooter	Activision	9.03	4.28	0.13	1.32
...	...	...	...	...	...	...	...
16295	3DS	Action	Namco Bandai Games	0.00	0.00	0.01	0.00
16296	PS4	Simulation	UIG Entertainment	0.00	0.01	0.00	0.00
16297	PSP	Adventure	Prototype	0.00	0.00	0.01	0.00
16298	XOne	Shooter	Capcom	0.01	0.00	0.00	0.00
16299	PS2	Adventure	Kadokawa Shoten	0.00	0.00	0.01	0.00
16300	Wii	Racing	Popcorn Arcade	0.01	0.00	0.00	0.00
16301	Wii	Racing	Takara Tomy	0.00	0.00	0.01	0.00
16302	PSP	Adventure	Tecmo Koei	0.00	0.00	0.01	0.00
16303	XOne	Sports	Alternative Software	0.00	0.01	0.00	0.00
16304	PC	Shooter	Codemasters	0.00	0.01	0.00	0.00
16305	3DS	Sports	Mastiff	0.01	0.00	0.00	0.00

16306	PS2	Adventure	Kadokawa Shoten	0.00	0.00	0.01	0.00
16307	Wii	Sports	Unknown	0.00	0.01	0.00	0.00
16308	GBA	Action	Titus	0.01	0.00	0.00	0.00
16309	N64	Action	Virgin Interactive	0.01	0.00	0.00	0.00
16310	N64	Sports	Infogrames	0.01	0.00	0.00	0.00
16311	GC	Puzzle	Ubisoft	0.01	0.00	0.00	0.00
16312	PC	Shooter	Destineer	0.01	0.00	0.00	0.00
16313	DS	Adventure	Deep Silver	0.00	0.01	0.00	0.00
16314	DS	Sports	Hudson Soft	0.00	0.00	0.01	0.00
16315	DS	Puzzle	Majesco Entertainment	0.01	0.00	0.00	0.00
16316	PSV	Action	dramatic create	0.00	0.00	0.01	0.00
16317	PSP	Role-Playing	Falcom Corporation	0.00	0.00	0.01	0.00
16318	PC	Adventure	Ubisoft	0.01	0.00	0.00	0.00
16319	DS	Simulation	Destineer	0.01	0.00	0.00	0.00
16320	GC	Shooter	Infogrames	0.01	0.00	0.00	0.00
16321	GBA	Platform	Kemco	0.01	0.00	0.00	0.00
16322	DS	Puzzle	7G//AMES	0.00	0.01	0.00	0.00
16323	PS2	Racing	Activision	0.00	0.00	0.00	0.00
16324	GBA	Platform	Wanadoo	0.01	0.00	0.00	0.00

16325 rows × 8 columns

In [78]:
```
df.isna().sum()
```

Out[78]:
```
Platform         0
Genre            0
Publisher       36
NA_Sales         0
EU_Sales         0
JP_Sales         0
Other_Sales      0
Global_Sales     0
dtype: int64
```

In [79]:
```
df.dropna(inplace=True)
```

In [80]:
```
df.isna().sum()
```

Out[80]:
```
Platform        0
Genre           0
Publisher       0
NA_Sales        0
EU_Sales        0
JP_Sales        0
Other_Sales     0
Global_Sales    0
dtype: int64
```

In [81]:
```
#df['Publisher']=le.fit_transform(df['Publisher'])
```

In [82]:
```
x=pd.get_dummies(df[columns_object])
```

In [83]:
```
df=pd.concat([df,x],axis=1)
```

In [84]:
```
pd.set_option('display.max_columns', None)
```

In [85]:
```
df.head()
```

Out[85]:

	Platform	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	Wii	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	NES	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	Wii	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	Wii	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	GB	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

In [86]:
```
df.drop(df[columns_object],axis=1,inplace=True)
```

In [87]:
```
df.head()
```

Out[87]:

	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Platform_2600	Platform_3DO
0	41.49	29.02	3.77	8.46	82.74	0	0
1	29.08	3.58	6.81	0.77	40.24	0	0
2	15.85	12.88	3.79	3.31	35.82	0	0
3	15.75	11.01	3.28	2.96	33.00	0	0
4	11.27	8.89	10.22	1.00	31.37	0	0

In [88]:
```
y=df.Global_Sales
```

In [89]:
```
del(df['Global_Sales'])
```

In [90]:
```
df.head()
```

Out[90]:

	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Platform_2600	Platform_3DO	Platform_3DS
0	41.49	29.02	3.77	8.46	0	0	0
1	29.08	3.58	6.81	0.77	0	0	0
2	15.85	12.88	3.79	3.31	0	0	0
3	15.75	11.01	3.28	2.96	0	0	0
4	11.27	8.89	10.22	1.00	0	0	0

In [91]:
```
x=[]
for i in df.columns:
    if df[i].dtypes=='float64' or df[i].dtypes=='int64':
        df[[i]]=scaler.fit_transform(df[[i]])
```

In [92]:
```
df.head()
```

Out[92]:

	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Platform_2600	Platform_3DO	Platform_3I
0	50.123565	56.688258	11.834876	44.250812	0	0	
1	35.034576	6.738940	21.581952	3.795974	0	0	
2	18.948570	24.998714	11.899002	17.158170	0	0	
3	18.826983	21.327125	10.263802	15.316922	0	0	
4	13.379870	17.164682	32.515349	5.005936	0	0	

In [93]:
```
from sklearn.model_selection import train_test_split
```

In [94]:
```
X_train, X_test, y_train, y_test = train_test_split(df,y,test_size
=0.3)
```

In [95]:
```
from sklearn.linear_model import LinearRegression
```

In [96]:
```
lr = LinearRegression()
```

In [97]:
```
lr.fit(X_train,y_train)
```

Out[97]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, nor
malize=False)
```

In [98]:
```
from sklearn.metrics import r2_score
```

In [99]:
```
r2_score(y_test,lr.predict(X_test))
```

Out[99]:
```
-503512687488963.2
```

Since the R2 Score is coming out to be negative, that means that Linear Regression is not the model for prediction in this dataset

In [100]:
```
from sklearn.ensemble import RandomForestRegressor
```

In [101]:
```
rf = RandomForestRegressor()
```

In [102]:
```
X_train.columns
```

Out[102]:
```
Index(['NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales', 'Platfor
m_2600',
       'Platform_3DO', 'Platform_3DS', 'Platform_DC', 'Platform_DS
',
       'Platform_GB',
       ...
       'Publisher_Zushi Games', 'Publisher_bitComposer Games',
       'Publisher_dramatic create', 'Publisher_fonfun', 'Publisher
_iWin',
       'Publisher_id Software', 'Publisher_imageepoch Inc.',
       'Publisher_inXile Entertainment', 'Publisher_mixi, Inc',
       'Publisher_responDESIGN'],
      dtype='object', length=623)
```

In [103]:
```
rf.fit(X_train,y_train)
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/forest.py:
245: FutureWarning: The default value of n_estimators will change
from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[103]:
```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=N
one,
                      max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_spli
t=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=1
0,
                      n_jobs=None, oob_score=False, random_state=N
one,
                      verbose=0, warm_start=False)
```

In [104]:
```
r2_score(y_test,rf.predict(X_test))
```

Out[104]:
```
0.9777051517234282
```

Since the R2 Score is 97.7%, that means that Random Forest Regression is a very good model for prediction

In [105]:
```
from sklearn.linear_model import ElasticNet
```

In [106]:
```
en=ElasticNet()
```

In [107]:
```
en.fit(X_train,y_train)
```

Out[107]:
```
ElasticNet(alpha=1.0, copy_X=True, fit_intercept=True, l1_ratio=0.
5,
           max_iter=1000, normalize=False, positive=False, precomp
ute=False,
           random_state=None, selection='cyclic', tol=0.0001, warm
_start=False)
```

In [108]:
```
en.score(X_test,y_test)
```

Out[108]:
```
0.7592285716231674
```

Since the R2 Score is 75.9%, that means that Elastic Net is an alright model for prediction

In [109]:
```
from sklearn.neighbors import KNeighborsRegressor
```

In [110]:
```
kn=KNeighborsRegressor()
```

In [111]:
```
kn.fit(X_train,y_train)
```

Out[111]:
```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkow
ski',
                    metric_params=None, n_jobs=None, n_neighbors=5
, p=2,
                    weights='uniform')
```

In [112]:
```
kn.score(X_test,y_test)
```

Out[112]:
```
0.9449032379368333
```

Since the R2 Score is 94.4%, that means that KNN is a very good model for prediction

In [113]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [114]:
```
dg = DecisionTreeRegressor()
```

In [115]:
```
dg.fit(X_train,y_train)
```

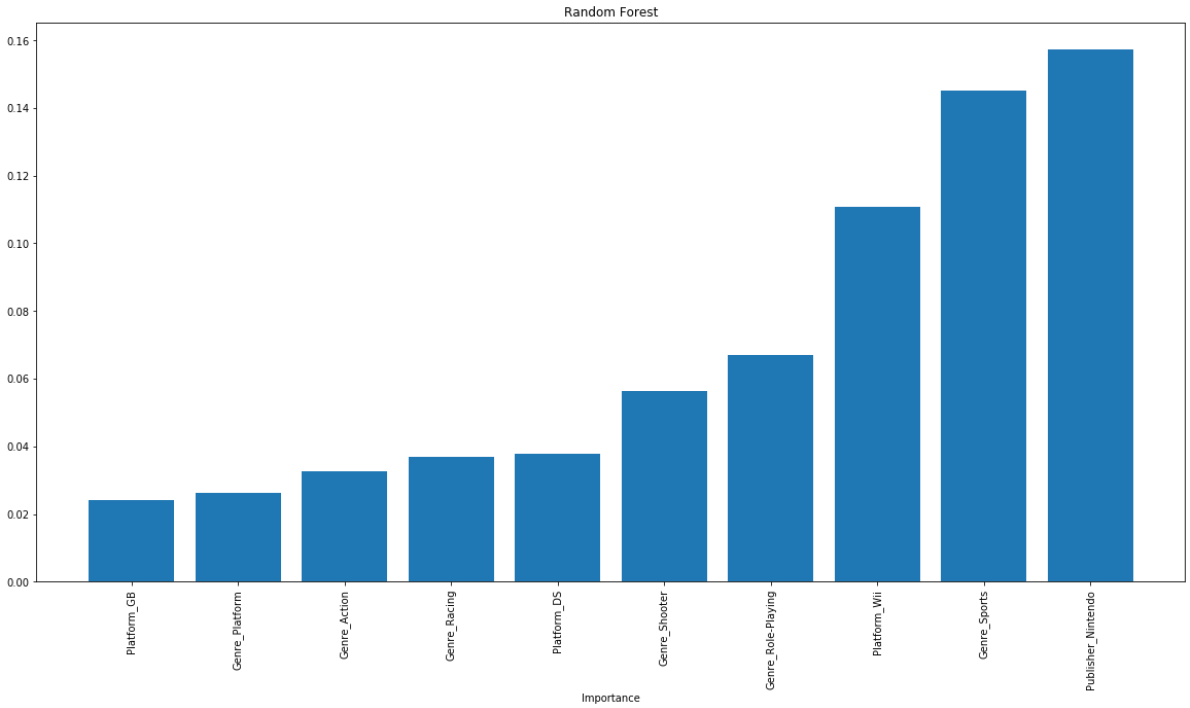
Out[115]:
```
DecisionTreeRegressor(criterion='mse', max_depth=None, max_feature
s=None,
                      max_leaf_nodes=None, min_impurity_decrease=0
.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_lea
f=0.0,
                      presort=False, random_state=None, splitter='
best')
```

In [116]:
```
dg.score(X_test,y_test)
```

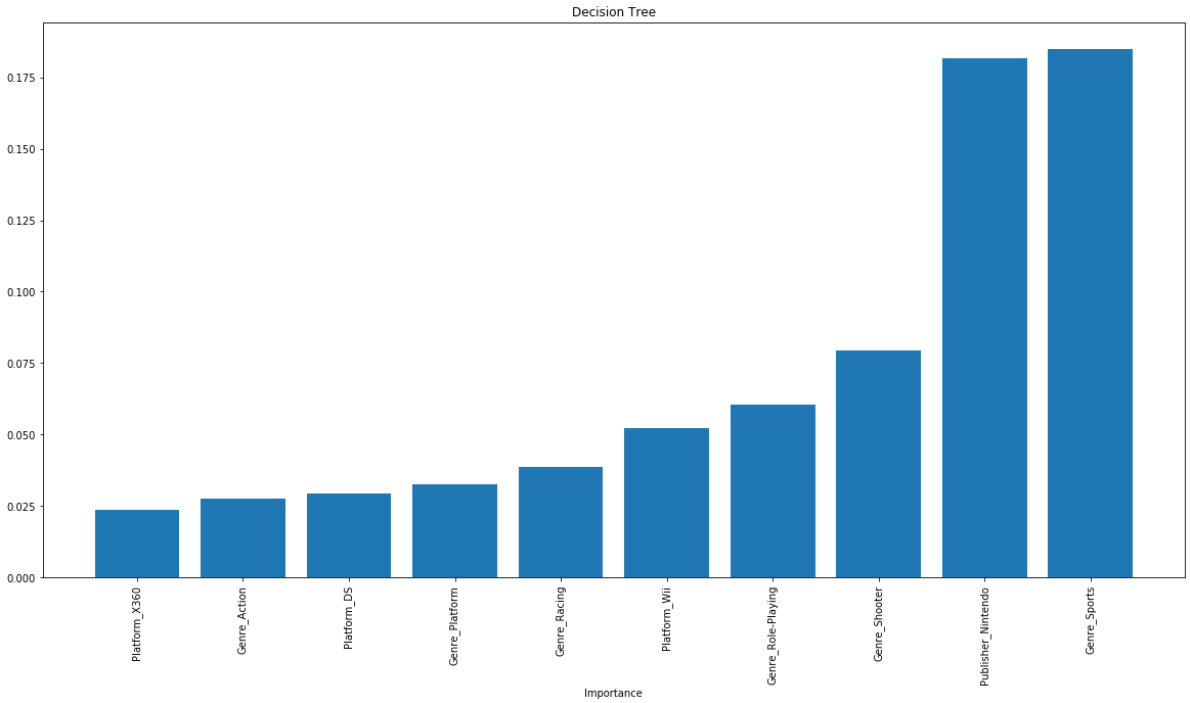
Out[116]:
```
0.9541075382850024
```

Since the R2 Score is 95.4%, that means that decision tree regressor is the best model for prediction

In [61]:
```
import matplotlib.pyplot as plt
feature_importance = rf.feature_importances_
sorted_idx = np.argsort(feature_importance)
sorted_idx = sorted_idx[len(feature_importance) - 10:]
pos = np.arange(sorted_idx.shape[0])

plt.figure(figsize=(20,10))
plt.bar(pos, feature_importance[sorted_idx])
plt.xticks(pos, X_train.columns[sorted_idx],rotation=90)
plt.xlabel('Importance')
plt.title('Random Forest')
plt.show()
```



In [62]:
```
import matplotlib.pyplot as plt
feature_importance = dg.feature_importances_
sorted_idx = np.argsort(feature_importance)
sorted_idx = sorted_idx[len(feature_importance) - 10:]
pos = np.arange(sorted_idx.shape[0])

plt.figure(figsize=(20,10))
plt.bar(pos, feature_importance[sorted_idx])
plt.xticks(pos, X_train.columns[sorted_idx],rotation=90)
plt.xlabel('Importance')
plt.title('Decision Tree')
plt.show()
```



Recommendations for Nintendo

1. Loyalty programs - because the test shows that there is very low loyalty. However, they do have a very recognizable brand.

2. Building platform agnostic games. We know sales is heavily dependent on the platform from the hypothesis tests - so if nintendo not just focus on their own platofrms but instead build games that can work on any platform

3. Introduce Loyalty programs - From our analysis of the data, it is clear that Nintendo was not able to create a loyal customer base in par with its brand equity. Therefore, we recommend Nintendo to focus their marketing efforts on creating a loyalty program to create a more loyal customer base.

4. Create platform-agnostic games - Based on the exploratory that we did, we fathom that platforms have a significant impact on the sales. Therefore, we recommend Nintendo to create platform-agnostic games that will improve their customer base and also profit (sales).

5. Strategy for Japan - Considering the difference in gaming culture in Japan, we recommend that, to increase the market share, Nintendo should focus on releasing more role-playing games in Japan.

6. Nintendo Global Strategy - Nintendo's global strategy should be to focus on racing, sports, and shooting games. Based on all the models (random forest, decision models, KNN, etc.), sports games are predicted to be the most preferred (according to sales) genre. Therefore, the focus should be on creating higher quality sports games.

7. Lower number of games/years: Nintendo should continue their differentiation strategy, where they create a fewer number of high-quality games per year. As seen from the analysis, Nintendo was able to lead the market even though they were seventh when it came to the number of games released per year.