

```

# dataset courtesy of user 'gregorut' via Kaggle:
# https://www.kaggle.com/gregorut/videogamesales

sales <- tibble::as_tibble(read.csv("vgsales.csv"))
head(sales)

## # A tibble: 6 x 11
##   Rank Name Platform Year  Genre Publisher NA_Sales EU_Sales JP_Sales
##   <int> <chr>  <chr>    <chr> <chr>    <dbl>    <dbl>    <dbl>
## 1     1 Wii ~ Wii    2006 Spor~ Nintendo    41.5    29.0    3.77
## 2     2 Supe~ NES    1985 Plat~ Nintendo    29.1    3.58    6.81
## 3     3 Mari~ Wii    2008 Raci~ Nintendo    15.8    12.9    3.79
## 4     4 Wii ~ Wii    2009 Spor~ Nintendo    15.8    11.0    3.28
## 5     5 Poke~ GB     1996 Role~ Nintendo    11.3    8.89    10.2
## 6     6 Tetr~ GB     1989 Puzz~ Nintendo    23.2    2.26    4.22
## # ... with 2 more variables: Other_Sales <dbl>, Global_Sales <dbl>

colnames(sales)

## [1] "Rank"          "Name"          "Platform"       "Year"          "Genre"
## [6] "Publisher"     "NA_Sales"      "EU_Sales"       "JP_Sales"      "Other_Sales"
## [11] "Global_Sales"

for(i in c(2, 3, 4, 5, 6)){
  sales[[i]] <- as.factor(sales[[i]])
}

# missing values
dim(sales)

## [1] 16598     11

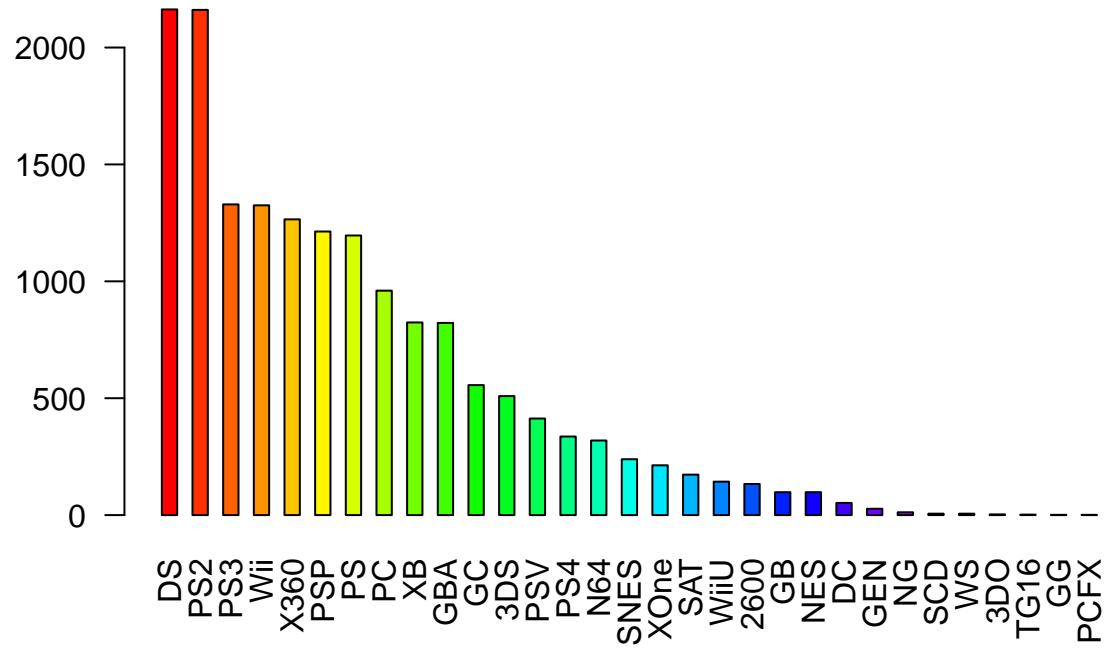
sales <- sales[!(is.na(sales$Year)), ]
dim(sales)

## [1] 16598     11

# bar plot of counts by platform
par(mfrow=c(1, 1))
barplot(sort(table(sales$Platform), decreasing=T),
        col=rainbow(length(table(sales$Platform))),
        space=1,
        las=2,
        main="Popular publishers")

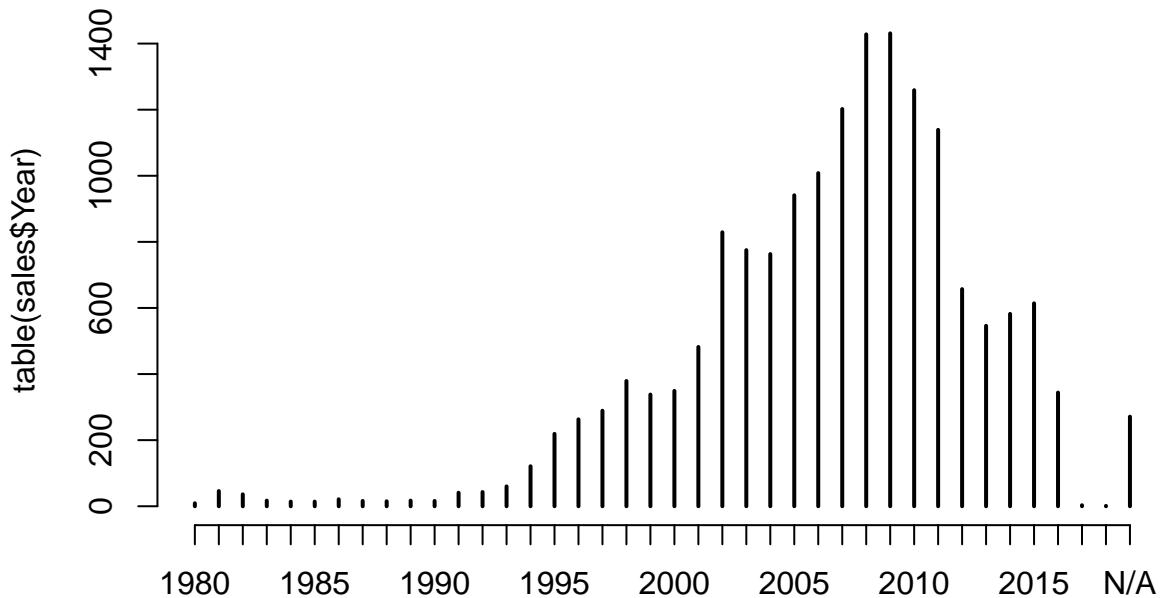
```

Popular publishers



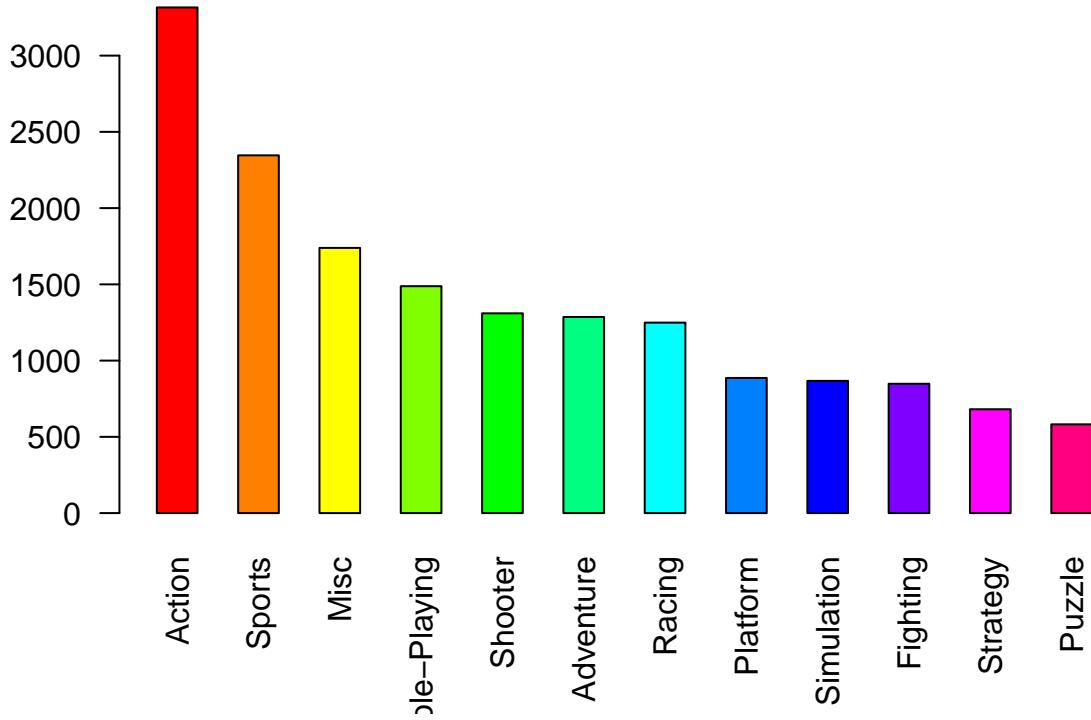
```
# " year
plot(table(sales$Year), main="Sales per year")
```

Sales per year



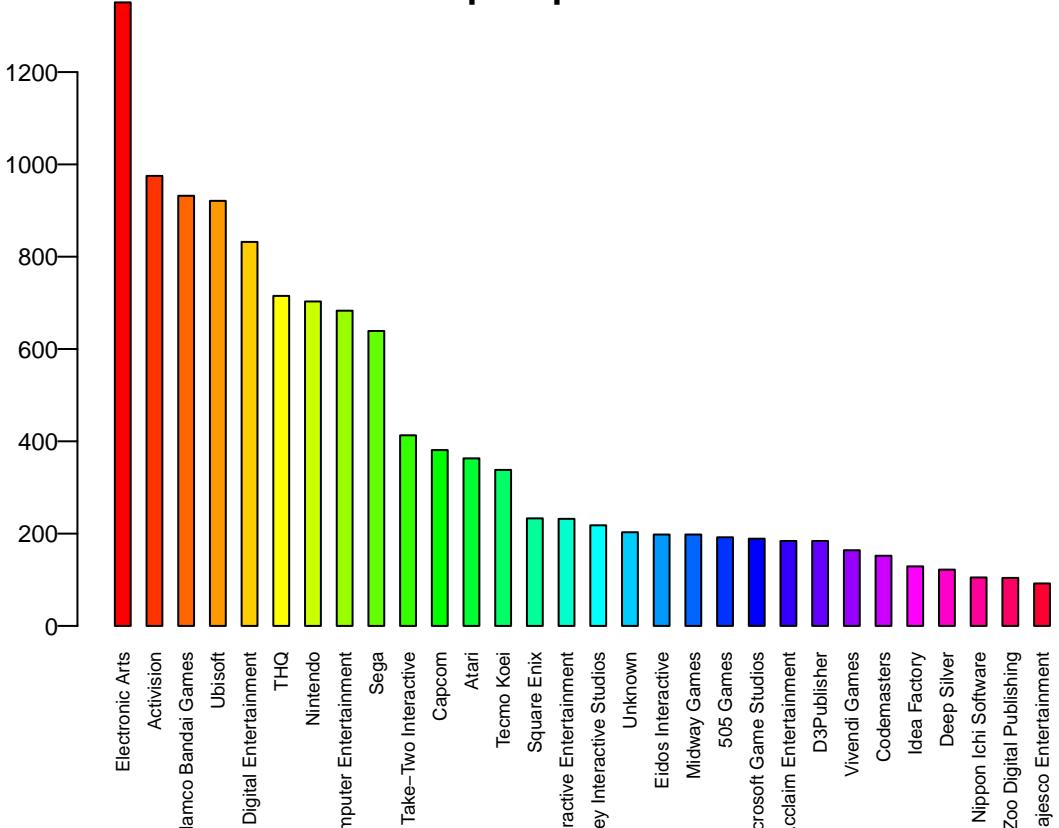
```
# " genre
barplot(sort(table(sales$Genre), decreasing=T),
       col=rainbow(length(table(sales$Genre))),
       space=1,
       las=2,
       main="Popular genres")
```

Popular genres



```
# " publisher (top 30)
par(mar=c(5.1, 4.1, 1, 2.1), mgp=c(3, 0.5, 0))
barplot(sort(table(sales$Publisher), decreasing=T)[1:30],
       col=rainbow(30),
       space=1,
       las=2,
       cex.names=0.55,
       cex.axis=0.75,
       main="Top 30 publishers")
```

Top 30 publishers



```
# going to create new sales df sales_new; dropping legacy platforms (only a few obs)
# inspect them...
dim(sales)

## [1] 16598      11

sort(table(sales$Platform)) # drop: GG, PCFX, TG16, 3DO, SCD, WS, NG, GEN, DC

##
##   GG PCFX TG16 3DO  SCD   WS   NG   GEN   DC   GB   NES 2600 WiiU   SAT XOne SNES
##   1    1    2    3    6    6   12   27   52   98   98  133  143  173  213  239
##   N64 PS4  PSV 3DS   GC  GBA   XB   PC   PS  PSP X360   Wii  PS3  PS2     DS
## 319 336  413 509  556  822  824  960 1196 1213 1265 1325 1329 2161 2163

# remove, if not removed it will cause issues with estimating test error since the random selection of
items <- c("GG", "PCFX", "TG16", "3DO", "SCD", "WS", "NG", "GEN", "DC")
for(item in items){
  sales <- sales[sales[, 3] != item, ]
}
```

Creating a linear model

```
set.seed(1)
train <- sample(dim(sales)[1], round(0.8*dim(sales)[1]))
lm_model <- lm(data=sales, Global_Sales ~ Platform + Genre + NA_Sales, subset=train)
summary(lm_model)
```

```

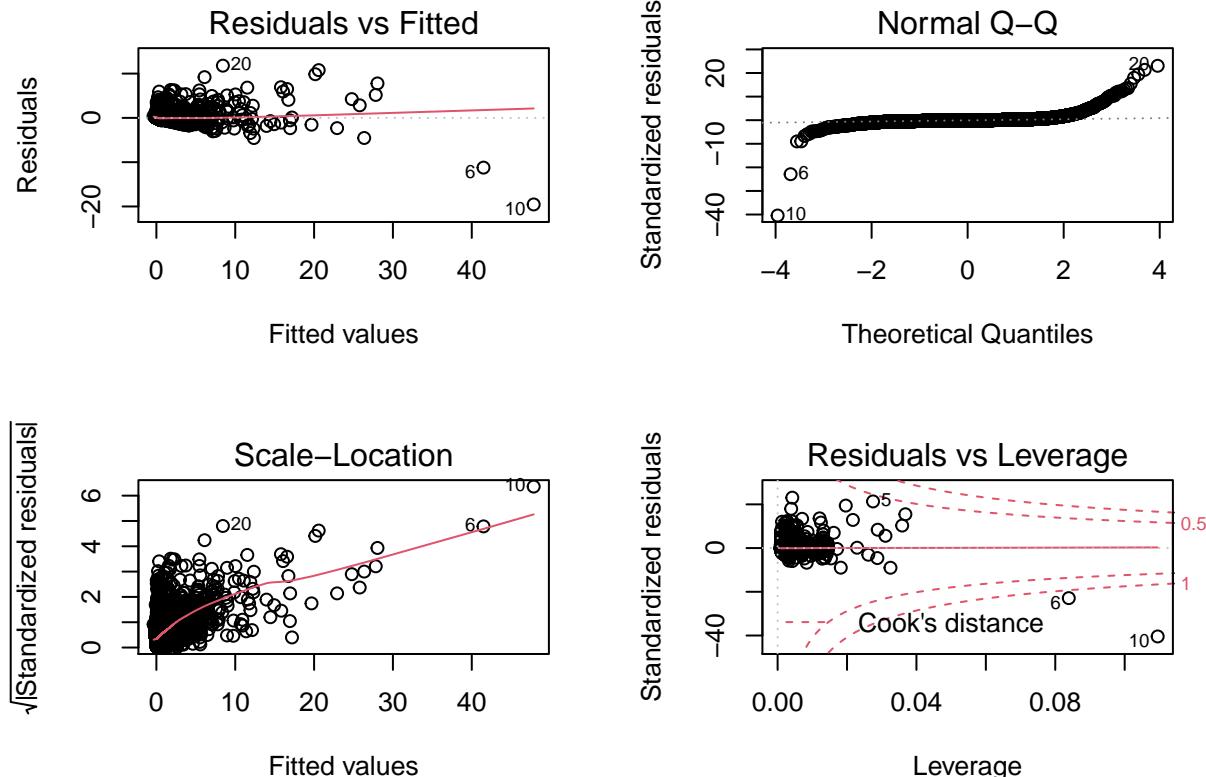
##
## Call:
## lm(formula = Global_Sales ~ Platform + Genre + NA_Sales, data = sales,
##      subset = train)
##
## Residuals:
##    Min      1Q  Median      3Q     Max 
## -19.5426 -0.1157 -0.0445  0.0476 11.7852 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)           -0.471308  0.051353 -9.178 < 2e-16 ***
## Platform3DS            0.651887  0.057030 11.431 < 2e-16 ***
## PlatformDS              0.524804  0.052565  9.984 < 2e-16 ***
## PlatformGB              1.019783  0.076938 13.255 < 2e-16 ***
## PlatformGBA             0.433211  0.054745  7.913 2.71e-15 ***
## PlatformGC              0.395830  0.056476  7.009 2.52e-12 ***
## PlatformN64              0.394515  0.060302  6.542 6.28e-11 ***
## PlatformNES              0.818459  0.075979 10.772 < 2e-16 ***
## PlatformPC              0.552767  0.054517 10.139 < 2e-16 ***
## PlatformPS              0.576985  0.053564 10.772 < 2e-16 ***
## PlatformPS2             0.571243  0.052512 10.878 < 2e-16 ***
## PlatformPS3             0.661965  0.053291 12.422 < 2e-16 ***
## PlatformPS4             0.795479  0.059503 13.369 < 2e-16 ***
## PlatformPSP             0.538660  0.053739 10.024 < 2e-16 ***
## PlatformPSV             0.530389  0.058564  9.057 < 2e-16 ***
## PlatformSAT             0.659959  0.067908  9.718 < 2e-16 ***
## PlatformSNES             0.820278  0.063634 12.890 < 2e-16 ***
## PlatformWii              0.490804  0.053371  9.196 < 2e-16 ***
## PlatformWiiU             0.560760  0.070216  7.986 1.51e-15 ***
## PlatformX360             0.405130  0.053385  7.589 3.44e-14 ***
## PlatformXB              0.377499  0.054755  6.894 5.66e-12 ***
## PlatformXOne             0.438647  0.063613  6.896 5.61e-12 ***
## GenreAdventure          -0.035393  0.019115 -1.852  0.0641 .  
## GenreFighting            -0.005202  0.023003 -0.226  0.8211
## GenreMisc                -0.004555  0.017283 -0.264  0.7921
## GenrePlatform            0.009581  0.022263  0.430  0.6669
## GenrePuzzle              -0.018269  0.025944 -0.704  0.4813
## GenreRacing              0.038870  0.019473  1.996  0.0459 * 
## GenreRole-Playing         0.142458  0.018165  7.842 4.76e-15 ***
## GenreShooter             -0.037803  0.018967 -1.993  0.0463 * 
## GenreSimulation          -0.002182  0.022183 -0.098  0.9216
## GenreSports              -0.003326  0.015743 -0.211  0.8327
## GenreStrategy            -0.005082  0.024539 -0.207  0.8359
## NA_Sales                  1.765439  0.006186 285.403 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5122 on 13156 degrees of freedom
## Multiple R-squared:  0.8691, Adjusted R-squared:  0.8688
## F-statistic:  2647 on 33 and 13156 DF,  p-value: < 2.2e-16

```

From the above model summary, it is worth noting the highly significant p value of RPG's (Role-Playing Games).

Linear model assumptions

```
# diagnostic plots
par(mar=c(4.7, 4.5, 3.4, 2.2), mfrow=c(2, 2), mgp=c(3, 1, 0))
plot(lm_model)
```



1) Residuals vs Fitted:

No discernible evidence of non-linearity. It appears that a linear model captures the model sufficiently.

2) Normal Q-Q:

This plot shows a ‘heavy-tailed’ distribution. Most notably at lower theoretical quantile values. It can be seen that the residuals deviate significantly at higher theoretical quantile values. This could be explained by the unpredictability of extremely high or unexpectedly low sales of some games.

3) Scale-Location:

There is convincing evidence of a non-random spread and therefore heteroscedasticity. I think this is to be expected due to the nature of the data being analysed. Unfortunately this violates an assumption of OLS regression to the degree of heteroscedasticity present.

4) Residuals vs Leverage:

It appears that there are a few influential cases, with the majority being well within the Cook’s distance lines. It is worth removing observations 6 and 10 as seen on the plot in order to improve R210 from analysis and check R2.

Current R-squared:

```
summary(lm_model)$r.squared
```

```
## [1] 0.8691054
```

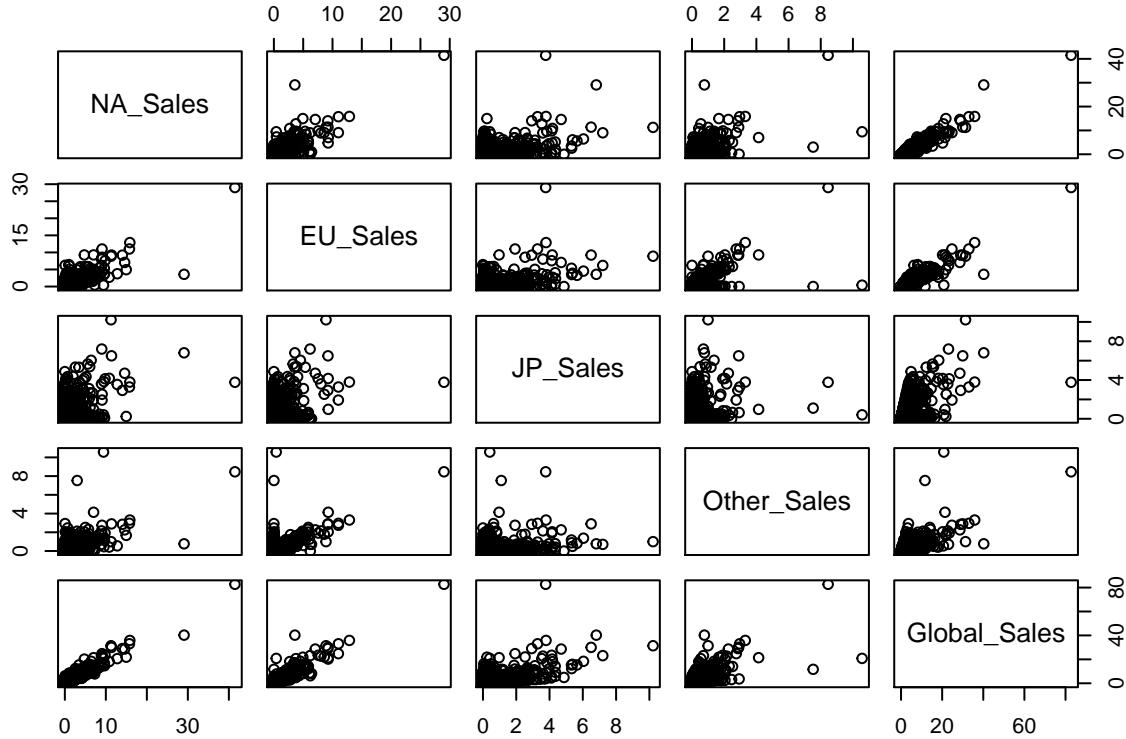
New R-squared:

```
sales <- sales[-c(6, 10), ]
lm_model <- lm(data=sales, Global_Sales ~ Platform + Genre + NA_Sales, subset=train)
summary(lm_model)$r.squared

## [1] 0.8887784

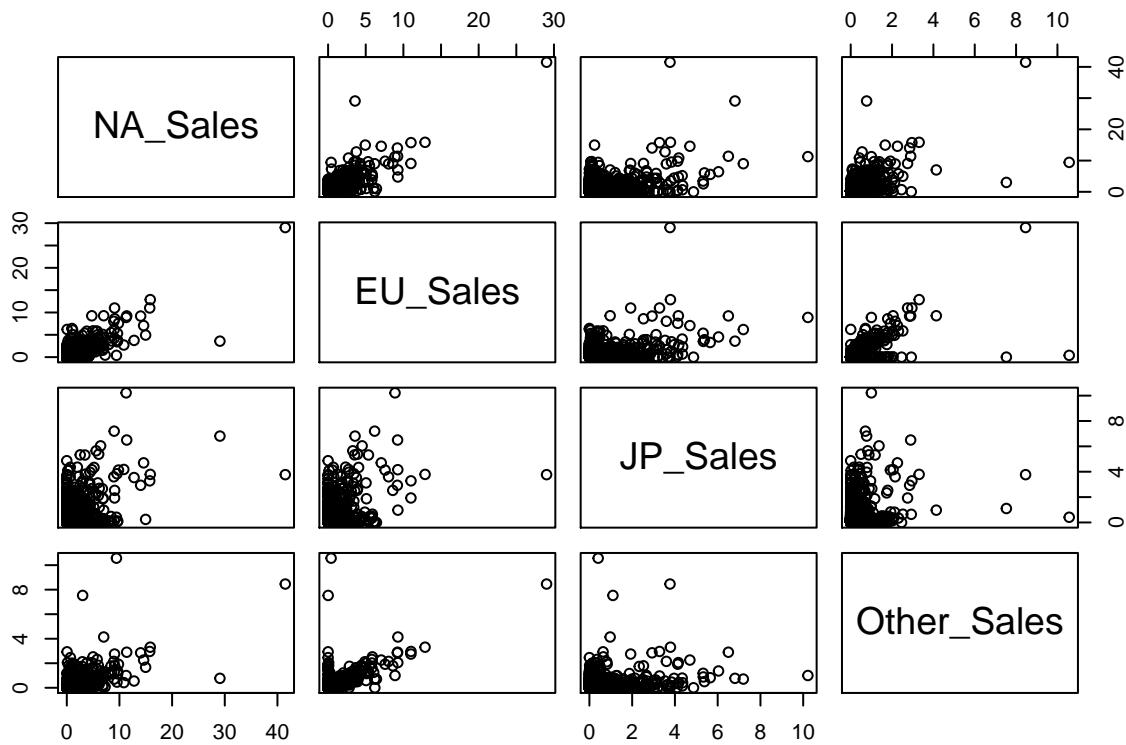
# Check linearity, excluding categorical variables
sales_small <- sales[, -which(names(sales) %in% c("Rank", "Name", "Year", "Platform", "Genre", "Published"))

pairs(sales_small)
```



```
# Check collinearity
sales_small <- sales[, -which(names(sales) %in% c("Rank", "Name", "Year", "Platform", "Genre", "Published"))

pairs(sales_small)
```

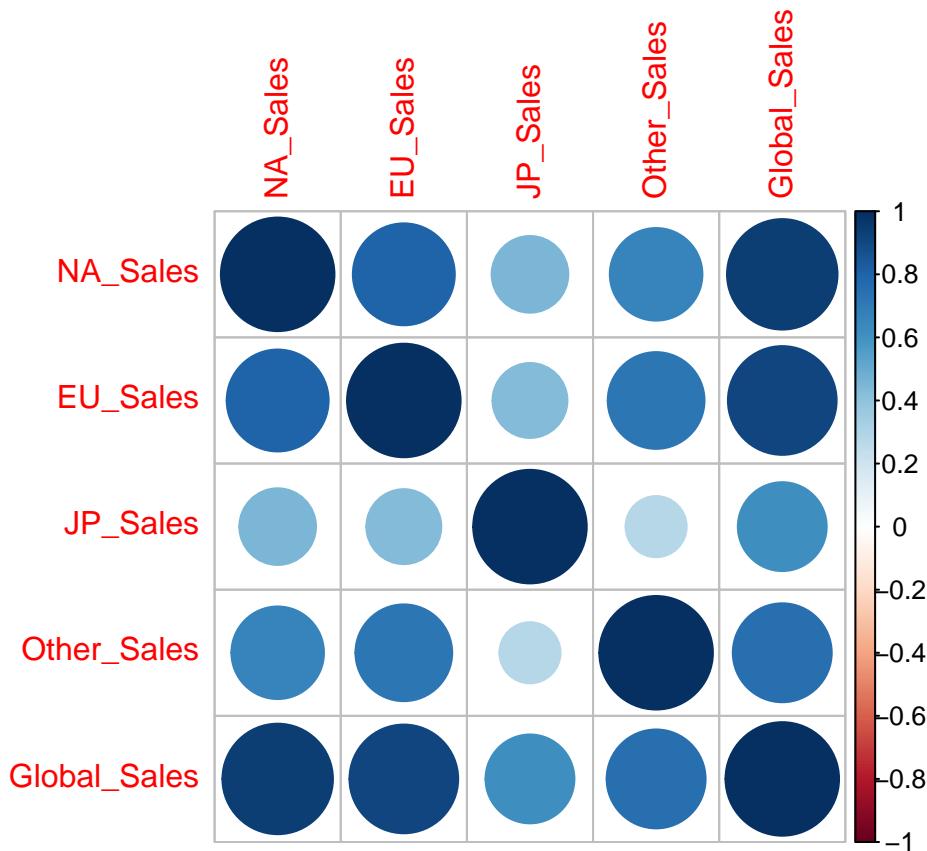


Above it can be seen that regional sales are highly collinear with global sales. As a result, only one major region should be included in model to prevent overfitting. Ideally this dataset would have more useful predictors. It should also be acknowledged that having to use any sale metric as a predictor for predicting global sales is not ideal, however due to the exploratory nature of this project, NA_Sales included. If sales predictors were included in the model, it would be unrealistic to predict global sales with high accuracy due to the poor predictive power of the non sales-related predictors.

```
# Pearson correlation matrix

library(corrplot)

## Warning: package 'corrplot' was built under R version 4.0.4
## corrplot 0.84 loaded
correlations <- cor(sales[, 7:11])
corrplot(correlations, method="circle")
```



Estimating test error

```
# 1) Validation set
test_set <- subset(sales[-train, ], select=c(Platform, Genre, NA_Sales))
dim(test_set)

## [1] 3298     3

sale_preds_test_set <- c()
for(i in 1:nrow(test_set)){
  temp_row <- test_set[i, ]
  pred <- predict(lm_model, newdata=temp_row)
  sale_preds_test_set <- c(sale_preds_test_set, pred)
}

predictions <- data.frame(Predicted=sale_preds_test_set,
                           Actual=rep(NA, length(sale_preds_test_set)))
actual_test_set_sales <- subset(sales[-train, ], select=c(Global_Sales))
predictions[, 2] <- actual_test_set_sales

vs_mse <- mean((sales$Global_Sales - predict(lm_model, sales))[-train]^2)
test_estimates <- data.frame(VSA=vs_mse)

# 2) Leave-one-out cross-validation
library(boot)
```

```

glm_fit <- glm(Global_Sales ~ Platform + Genre + NA_Sales, data=sales)
coef(glm_fit)

##          (Intercept)      Platform3DS      PlatformDS      PlatformGB
## -0.543140248       0.718678671       0.574181088       1.075898113
##      PlatformGBA      PlatformGC      PlatformN64      PlatformNES
## 0.497730865       0.450665598       0.408477420       0.917264537
##      PlatformPC      PlatformPS      PlatformPS2      PlatformPS3
## 0.618586625       0.617085636       0.608756337       0.702208917
##      PlatformPS4      PlatformPSP      PlatformPSV      PlatformSAT
## 0.815537323       0.602198368       0.597918705       0.725465741
##      PlatformSNES      PlatformWii      PlatformWiiU      PlatformX360
## 0.876408923       0.521814795       0.613879272       0.421699464
##      PlatformXB      PlatformXOne      GenreAdventure      GenreFighting
## 0.431959189       0.468208889      -0.020371423      -0.010582701
##      GenreMisc      GenrePlatform      GenrePuzzle      GenreRacing
## -0.006413026      -0.041232215       0.001285357       0.032053371
##      GenreRole-Playing      GenreShooter      GenreSimulation      GenreSports
## 0.134098760      -0.038523229       0.013576959      -0.003385929
##      GenreStrategy      NA_Sales
## 0.006482942       1.882206057

cv_err <- cv.glm(sales, glm_fit)
cv_err$delta

## [1] 0.2298126 0.2298125

cv_te <- c(cv_err$delta[1])
test_estimates <- cbind(test_estimates, cv_te)

# 3) K-fold cross-validation
k_cv_err <- cv.glm(sales, glm_fit, K=10)
k_cv_err$delta

## [1] 0.2305986 0.2302456

k_cv_te <- c(k_cv_err$delta[1])
test_estimates <- cbind(test_estimates, k_cv_te)

# Resulting test error estimates using each approach
test_estimates

##          VSA      cv_te      k_cv_te
## 1 0.2836757 0.2298126 0.2305986

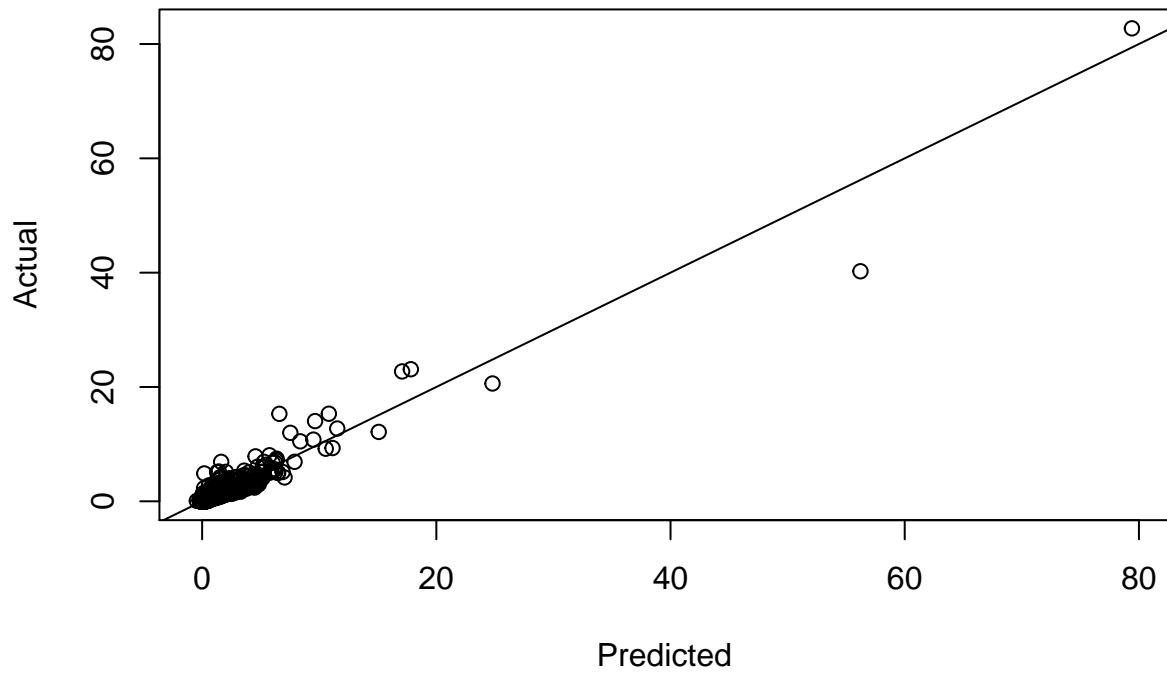
```

Visualising predicted vs actual

```

plot(predictions$Predicted, predictions$Actual, xlab="Predicted", ylab="Actual")
abline(a=0, b=1)

```



Concluding remarks

Overall this project was intended to test out the linear regression workflow in R based on my study of the textbook ‘Introduction to Statistical Learning’. While model accuracy was high and performed well on the test set, it requires more feature engineering to be robust - the obvious correlation between regional and global sales is a weak point, however, this project is not intended for model deployment and was just a fun exercise for myself. Ideally, after extensive feature engineering, the fit accuracy could be similar without using regional sales as a predictor.