GLM model on Film data

Import Necessary packages

```
library(tidyverse)
library(lmtest)
library(caTools)
library(pander) #Table format
library(performance)
library(car) # check vif of the model
library(lubridate)
```

The dataset can be located from this github repo

```
# Import data
Merged_data <- read_csv('Cleaned_data.csv')
#Merged_data <- read_csv('Merged_data.csv')
names(Merged_data)
```

```
##  [1] "...1"              "primary_title"     "runtime_minutes"
##  [4] "genres"            "averagerating"     "original_language"
##  [7] "original_title"    "popularity"        "vote_average"
## [10] "release_date"      "production_budget" "worldwide_gross"
## [13] "studio"            "domestic_gross"    "foreign_gross"
```

Data Cleaning/Preparation

```
month(Merged_data$release_date)
```

```
##   [1]  3  3  3 12  9  6 10  4 11  3  7  6  8 10  8  6 11 10  6  6  6  6  6  6
##  [25] 11  2  9  7  8  4 11  7  1  2 11 11 11 11 11 11 11 11  5  7  1  4  4  2
##  [49]  3  1  5  9  4  7  2  3  9  6  2  9  4  3 11 12  7  5  6 10 12 12 12 12
##  [73] 12 12  1  1 11  3 11 11  6  2  5 10  9  9  7  5  2  2  3  3 10  4  8  2
##  [97]  1  8  5  3  5 12  7 12  7  7  7  4 12  3  7  6  5 11  5 11  5 11  5 11
## [121]  5 11  5 11  5 11  5 11  5 11  7 12  4 11 11 12 11  3 12 12  9  2  1  4
## [145] 12 11  1  2  2  4  4  3  3  1  8 10 10 12 11 11  2 10  1  3 11  1 11 12
## [169] 10  3 10  7  4  7  2 11  6  5  2  7  7  5 12  3  3  3  3  3  3  3  3 10
## [193] 10 12 12  6  6  1  4 12  4 12 12  2 10  6 12 11 11  2  6  8  2 10  3 10
## [217]  4  4  4  4  6  6  1  1 12  9 12  9  3  9 10  3  3  8 12  4  3  5 11  1
## [241] 10  1 12  9  7 11 10 10  4  8 10 10  8  6  1  1  1  1  6  6  3  3  3  9
## [265]  9  7  6  1  1  8  8  5 10 10 10 10 12  2 11  3  6  3  8 10  4  4 10  1
## [289] 12  5  1 10 10  5  5  5  5  4 11 11  4  1  5  4  2  8  6  8  1 12 12 12
## [313]  9  6  4  9  6 10  7  9  3  7  5  5  5 12  3  3  9  9  6  8  9  8  8  8
##  [337] 12  5  1 12  9 11 11 11 11 11 11  7  6  4  6  6  1  9 12  6 11  9  7  3
##  [361]  8  9  9  7  3 12 12  9 10 11  1  1  2  3  3  7  6  3  2  8  8 12 12  9
##  [385] 12  3  5  6  9  9 10  2  8  6  6  2 11  2 11 11 12  3  3 11  5  5  1  1
##  [409]  1  9  5  4  5  3  1  8  8  8  8  8  8  9  1 12 10 10 10 10  7 11  7 11
##  [433]  4  2  2 12 12 12 12 12 12 11  8  8  8  8  4  4  4  7  5  5  6  9 10  1
##  [457] 11  6  9  7  6  8 10  5  5  5  5  5  5  5  5  7 11  9  4  9  9  8  2 12
##  [481] 12 12 12  5  5 10 10 10 10 10 12  5  7  4  4 12  3  4  4 12  8  1  1  1
##  [505]  1  1  1  1  1  1  1  1  1  6 10  7  2  8  7  1 10 12 10  1  7  7 11 11
##  [529] 12  5  6  9  8 10  8 10  8  9  7  7 11  9  9  8  7  1 12 12 12  9  9  9
##  [553]  9  1  8  8  8  8  8  8  9 11  1 12 12 11 12  1  1  7  3  4  5  9  5  2
##  [577]  3  1  4  2  4  4  4  2 12  2  9  2  9  9  9  9  9  1 12  7 12  4  9  9
##  [601]  9  9  5  2  2  2  6 10 10 10 12 11  2 10  3 11  4  8  8  8 12  6 10 10
##  [625]  6 11 11 11  7  9 10  9  3  3 10  6  5  6  6  6  6  6  6  6  6 10  5  6
##  [649]  3  5 10  8 12  3  6  8 11  4 11 11  2  8  9 12  5 10 12 12  7  7  7  7
##  [673]  7  8  3  7  5  5  3  6  9  3 11  3 10 10 10  7 12 10  8  2  2  6  6  6
##  [697]  1 12 10  2 11  8 12 12 12 12 12 12  7 10 11 11  6  6  9  5 10 10 10  9
##  [721] 11  8  3  9  3  7  7  7  7  7  7  7  7 11 10 11  8 11 11  2  5  9 12  8
##  [745]  8 12  3  7 12  1  7  1  1  1  1  1  2  2  7  7  7  7  7  7  7  7 12  5
##  [769]  3 11  1  3  2  2  5  5 12  8 12  8 12  8 12  8 12  8 12  3 10  3  9  4
##  [793] 10 10 12 12 12  8  9  9 10  6  1  7  9 10  5  5 10 10 10 10  5 11 11 11
##  [817] 11 11  1 11  3  3  3 11  6  9  2 10 10  4 10  4  4  9  9  9  9  6  6  5
##  [841] 11 11 11  7  9  6  6  6  6  5  5 12 12  1  6  6  6  6  6  6  7  5 10 11
##  [865] 11 11 10  9  5  9  5  9 10  5  5  8 11  8  7  8  2  2  2 10  1  2  2 12
##  [889]  4 10 11  9 11  6 11  2  3  3  7  7 10  6 11  8  8  8  8  8  8  6  4  4
##  [913]  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  5
##  [937] 12 12 10  9  9 11 10  2  2 12 12 10 10  7  6  7 10 11  3  4  4  4  4  4
##  [961]  4  4  5  7  4  9  8  8  6  7  6  5  1 11  7  1  1  1 10  7  6  3 11  2
##  [985]  3 12  7  6  3  3 11  1  7 12 11 11  5 11  2  8  7  7  6  6  8  4  3 10
## [1009]  3 11 11 11 11 11 11 11  2 11 10 10 10 10  8  4  9  5  4 12  8 11  2  4
## [1033]  6  7  6  6  6  6 11  1  1 11  7 10  1 12 11 11 11 10  9  6  3  4 11 11
## [1057]  9 10  3  3  5  6 11  8 10 10  1  7  6  2  3  1  1  1  1  1  1  1  1  1
## [1081]  1  1  1  1  1  1  1  1  1  1  1  7 12  6  8  5  5 12  4  2  7  1  4  1
## [1105]  8  3 12  2  4  7  4  5  1  7 10  7  7  3 10  2  2 12  9  3  5 12  7  3
## [1129]  9 11  8  4  1  4 10  1  8  6  7 10  1  1 12  9 12  3  8  1 10 10 10 10
## [1153]  6  5 10  9  2 12  1  7  4  7 12  6 12 12 12 12 12  8  5 11 12 11 11  8
## [1177] 11 11  9  4 11  4 11  4 11  4 11  7  3  3  2  6  6 10  7  8  4 11 10 10
## [1201]  7 11 11  4  1  9  1  4  4  3 11 11 12  3  7  9  5 12  8  5  6 12  9  3
## [1225]  9  9  9  6  2  6  1 11  9  6  3  3  8  7 11  8  4  7  3  2  8  9 10  5
## [1249]  2  2 12  9 11 11 11  8 12 11  9  4 11  9  7  8  1 12 12 12  2  7 12  8
## [1273]  6 12  4  3  7  6 11  9 10  5  7 11 12  2  2  6 12 11  7  2  8  3  7 10
## [1297]  8  6  5 11 12 10 12 10  8  8 12  1 11  7  6 11  5  7  9  8 12  4 11  4
## [1321]  4 11  1  6  4  2  9 11  9 12 12  3  6  8  3 11  6
```

```
# Check for duplicates
sum(duplicated(Merged_data))
```

```
## [1] 0
```

```
# Check missing values
pander::pander(colSums(is.na(Merged_data)))
```

Table 1: Table continues below

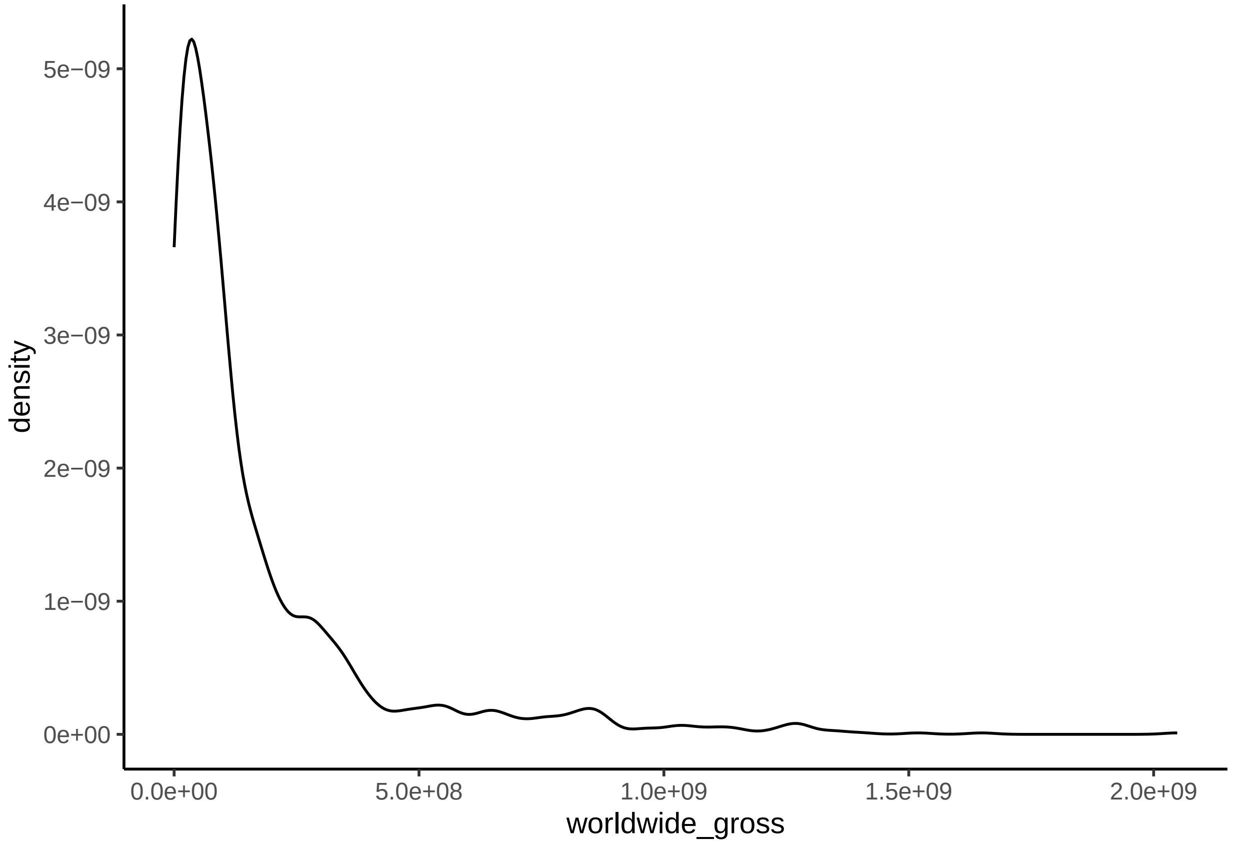
…1	primary_title	runtime_minutes	genres	averagerating
0	0	0	0	0

Table 2: Table continues below

original_language	original_title	popularity	vote_average	release_date	production_budget	worldwide_gross	studio	domestic_gross	foreign_gross
0	0	0	0	0	0	0	0	0	0

For this regression model we are interested in understanding the relationship between the worldwide gross for movie production against various variables in our models. The Dependent variable is **Worldwide_gross**

```r
# Invetsigate the properties of dependent variable
ggplot(data = Merged_data,aes(x=worldwide_gross))+geom_density()+theme_classic()
```



```
# There is alot of skewness in the data (positive skewness)

# Test for normality in the data
ks.test(Merged_data$worldwide_gross,"pnorm")
```

```
## Warning in ks.test.default(Merged_data$worldwide_gross, "pnorm"): ties should
## not be present for the one-sample Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  Merged_data$worldwide_gross
## D = 0.99701, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
# Normality test violated a glm model that uses the gamma distribution would be effective to try and mo
```

```
# remove rows where the gross returns is zero
Merged_data %>% filter(!worldwide_gross <= 0) -> Merged_data
```

GLM models

```
#
Merged_data %>% select(runtime_minutes,vote_average,popularity
                    ,production_budget,domestic_gross,foreign_gross
                    ,worldwide_gross,release_date) %>% mutate(month_release =month(release_date)) %>% s

# Split the data for Train and test data
Y = Model_data$worldwide_gross
sample_sl =sample.split(Y,SplitRatio = 3/4)

# Create Train and test data
train_data =subset(Model_data,sample_sl==TRUE)

test_data =subset(Model_data,sample_sl==FALSE)

# Fit GLM gamma model
Fit_model=glm(worldwide_gross~. -runtime_minutes,family = Gamma(link = "log"),data = train_data)

# Model 2
model2 =glm(worldwide_gross~ domestic_gross + foreign_gross,family = Gamma(link = "log"),data = train_da

# Model 3
model3 =glm(worldwide_gross~ production_budget,family = Gamma(link = "log"),data = train_data)

# Model 4
model4=glm(worldwide_gross~. -runtime_minutes,data = train_data)# using Gaussian model


summary(train_data$production_budget)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##    175000  14250000  32500000  54888952  75000000 330600000
```

```
summary.glm(model4)
```

```
##
## Call:
## glm(formula = worldwide_gross ~ . - runtime_minutes, data = train_data)
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -3.541e+07  1.381e+07  -2.563  0.01051 *
## vote_average       2.930e+06  2.259e+06   1.297  0.19499
## popularity         7.529e+05  2.880e+05   2.614  0.00908 **
## production_budget  4.295e-01  5.404e-02   7.948 5.14e-15 ***
## domestic_gross     1.580e+00  3.746e-02  42.163  < 2e-16 ***
## foreign_gross      5.498e-01  2.395e-02  22.961  < 2e-16 ***
## month_release     -1.138e+06  5.847e+05  -1.946  0.05193 .
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for gaussian family taken to be 4.106603e+15)
##
##     Null deviance: 6.6878e+19  on 998  degrees of freedom
## Residual deviance: 4.0737e+18  on 992  degrees of freedom
## AIC: 38759
##
## Number of Fisher Scoring iterations: 2
```

```
# Summary of the model
#pander::pander(summary.glm(Fit_model))
summary.glm(Fit_model)
```

```
##
## Call:
## glm(formula = worldwide_gross ~ . - runtime_minutes, family = Gamma(link = "log"),
##     data = train_data)
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.760e+01  1.291e-01 136.318  < 2e-16 ***
## vote_average      -2.799e-03  2.111e-02  -0.133  0.89454
## popularity         1.001e-02  2.691e-03   3.720  0.00021 ***
## production_budget  4.013e-09  5.050e-10   7.947 5.18e-15 ***
## domestic_gross     5.553e-09  3.501e-10  15.863  < 2e-16 ***
## foreign_gross      2.371e-09  2.238e-10  10.594  < 2e-16 ***
## month_release     -1.631e-02  5.464e-03  -2.986  0.00290 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for Gamma family taken to be 0.3586122)
##
##     Null deviance: 1714.74  on 998  degrees of freedom
## Residual deviance:  611.13  on 992  degrees of freedom
## AIC: 38799
##
## Number of Fisher Scoring iterations: 11
```

Interpreting the glm model

This generalized linear model (GLM) with a Gamma distribution and log link function is used to model the relationship between `worldwide_gross` and several predictors related to movie characteristics. Here's an interpretation of the results:

Residuals

The deviance residuals summary provides a quick look at the spread of residuals:

- **Min, 1Q, Median, 3Q, Max** values suggest that residuals are moderately centered around zero, indicating an acceptable fit. However, some residuals have relatively large negative and positive values, implying a few observations deviate notably from the model's predictions.

Coefficients

Each predictor in the model has an associated estimated coefficient, its standard error, and p-value:

- **(Intercept)**: The intercept is significant, with a large positive estimate (17.78), representing the baseline log of worldwide gross when all other predictors are at zero.
- **runtime_minutes**: This variable has a negative coefficient (-0.0007785), but it is not statistically significant ($p = 0.472$), meaning `runtime_minutes` does not appear to have a substantial impact on worldwide gross in this model.
- **averagerating**: This has a significant negative effect (-0.0446, $p = 0.027$), suggesting that, holding other factors constant, higher ratings are weakly associated with lower worldwide gross, though the effect is relatively small.
- **popularity**: This variable has a positive and statistically significant coefficient (0.009753, $p = 0.0005$), indicating a positive association with worldwide gross—higher popularity scores are associated with higher gross earnings.
- **production_budget**: A positive and significant effect (3.83e-09, $p < 0.001$), implying that larger production budgets lead to higher worldwide gross, though the impact is small per unit increase in budget.
- **domestic_gross** and **foreign_gross**: Both are highly significant ($p < 0.001$) and positively associated with `worldwide_gross`, with each having an estimated impact proportional to their gross values, underscoring their strong, direct contributions to worldwide gross.

Model Fit

- **Dispersion parameter**: The estimated dispersion parameter for the Gamma family is 0.399962, suggesting moderate variability around the fitted values.
- **Deviance**: The residual deviance (745.68) is much lower than the null deviance (1894.53), indicating that the model significantly reduces deviance (improves fit) compared to a null model without predictors.

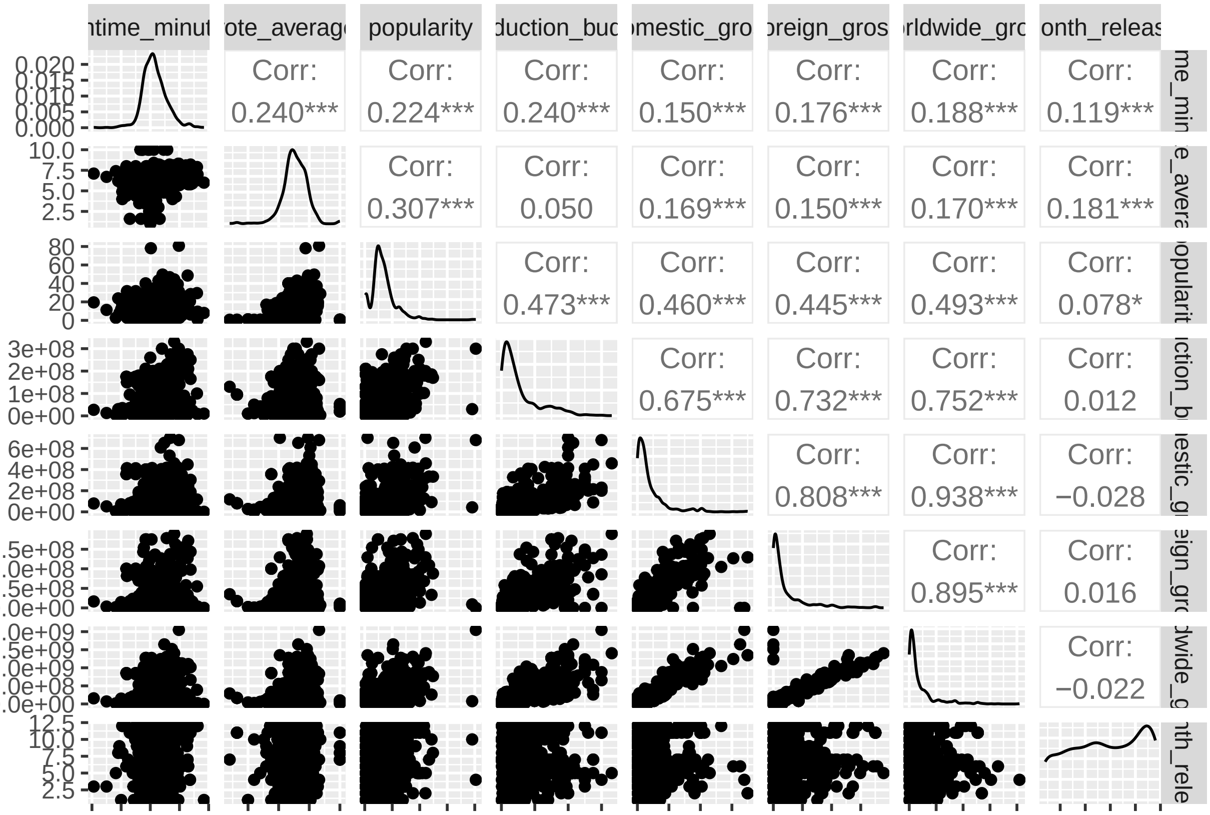
- **AIC**: The Akaike Information Criterion (AIC) of 38770 can be used to compare this model with other potential models, where a lower AIC generally indicates a better fit.

Conclusion

This model suggests that `popularity`, `production_budget`, `domestic_gross`, and `foreign_gross` are the primary predictors of `worldwide_gross`, with statistically significant positive associations. Although `averagerating` shows a significant but small negative impact, the impact of `runtime_minutes` does not appear statistically relevant in this context.

Assumptions

```
GGally::ggpairs(train_data)
```



Linearity

Based on the pair plot we can conclude that their is linearity between Dependent and Independent variables in the models

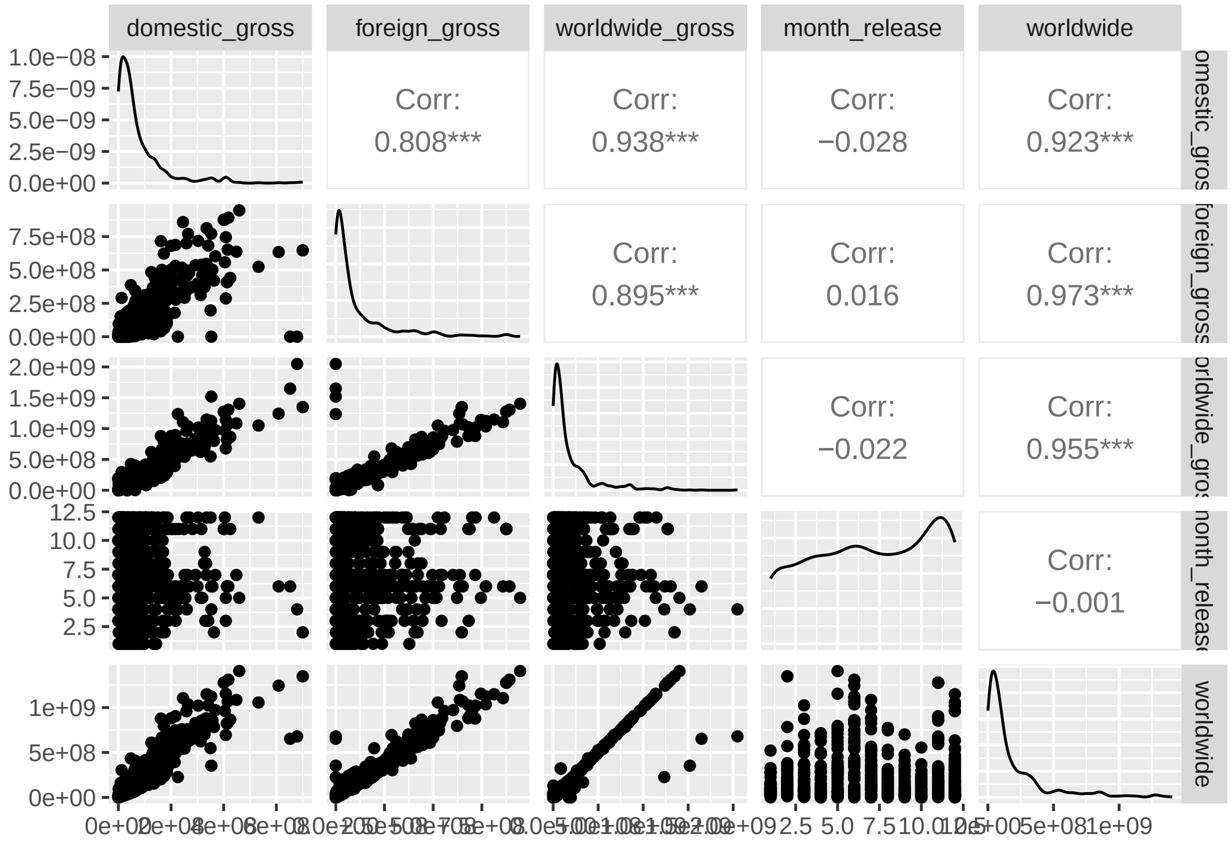
```
car::vif(Fit_model)
```

Multicollinearity

```
##       vote_average        popularity production_budget    domestic_gross
##           1.173827          1.482462          2.430127          3.124145
##      foreign_gross     month_release
##           3.560911          1.044981
```

Based on the VIF (Variance Inflation Factor) since no value is greater than 4 indicating low correlation between the independent variables but something worth looking is the relationship between domestic_gross and foreign_gross they tend to have moderate colinearity we can conclude there is no multicollinearity between the independent variables

```
# Create a column that contains the sum of the foreign and domestic returns
train_data %>% mutate(worldwide = foreign_gross + domestic_gross)%>% select(domestic_gross:worldwide)->
GGally::ggpairs(Corr_data)
```



Heteroscedasticity

```
bptest(Fit_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  Fit_model
## BP = 282.57, df = 6, p-value < 2.2e-16
```

The p-value is less than 0.05.This indicates that there is significant evidence of heteroscedasticity in the residuals of the regression model. Therefore, we can conclude that the assumption of homoscedasticity is violated for this model, meaning the variance of the error terms is constant across observations.

```
shapiro.test(Fit_model$residuals)
```

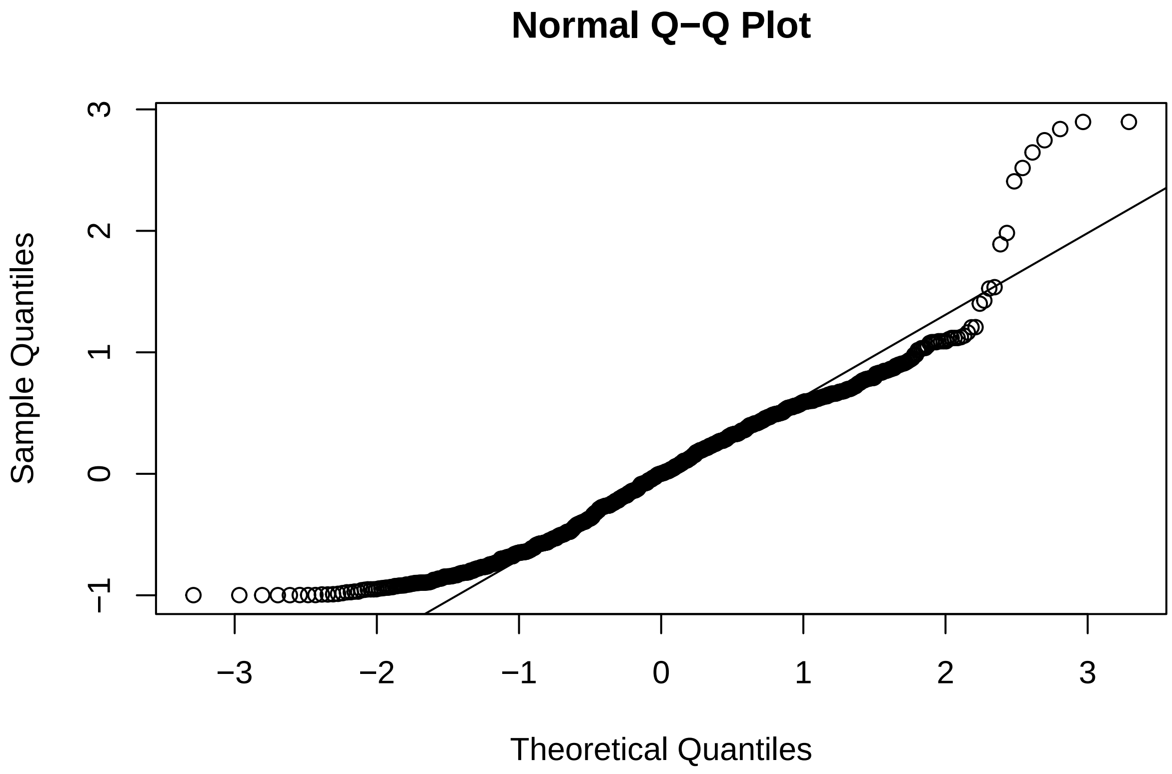
Normality of residuals

```
##
##  Shapiro-Wilk normality test
##
## data:  Fit_model$residuals
## W = 0.95461, p-value < 2.2e-16
```

```
check_normality(Fit_model$residuals)
```

```
## Warning: Non-normality of raw detected (p < .001).
```

```
qqnorm(Fit_model$residuals)
qqline(Fit_model$residuals)
```



Normality is violated the residuals are not normally distributed. From the qqplot we can observe that the residuals tend to move away from the residuals plot indicating non-normality in residuals.This can be confirmed by the statistical test from the performance package.

```r
check_autocorrelation(Fit_model)
```

Independence of observation

```
## Warning: Autocorrelated residuals detected (p < .001).
```

Autocorrelation is present Indicating there is correlation which is dependent on time. Implying that time influences the worldwide_gross which based on the data we concluded that time influences the data in terms of genres and the release date

Predicting the best model on the test data

```r
names(test_data)
```

```
## [1] "runtime_minutes"   "vote_average"      "popularity"
## [4] "production_budget" "domestic_gross"    "foreign_gross"
## [7] "worldwide_gross"   "month_release"
```

```r
test_data %>% select(-worldwide_gross)->test_data2
# Use the predict.glm to predict on our unseen data
predicted<-predict.glm(Fit_model,test_data2,type = "response")
# Add to the test data to be used while comparing to the actual value
test_data$Predicted = predicted
```

Compare the model metrics performance

```r
RMSE<-function(actual_val,Pred_val){
  return(sqrt((sum((actual_val-Pred_val)^2))/length(actual_val)))
}
RMSE(test_data$worldwide_gross,test_data$Predicted)
```

```
## [1] 611663347
```

```r
test_data %>% select(worldwide_gross,Predicted) %>% mutate(diff =worldwide_gross-Predicted)
```

```
## # A tibble: 334 x 3
##    worldwide_gross  Predicted         diff
##              <dbl>      <dbl>        <dbl>
##  1         9313302   50925031.   -41611729.
##  2       282778100  369388134.   -86610034.
##  3       177241171  140485333.    36755838.
##  4        82925064   62804328.    20120736.
##  5        66540205   61757840.     4782365.
##  6       821133378 1919556337. -1098422959.
##  7       192903624  134918933.    57984691.
##  8       165720921  117199052.    48521869.
##  9        61721826   60918979.      802847.
## 10        91126600   76187875.    14938725.
## # i 324 more rows
```

Very High MSE implying that our model does not predict well