VIDEO GAME SALES FORECASTING

Assignment A3: Business Insight Report

Student Name:

Abi Joshua George

(46656697)

Executive Summary

The analysis was able to provide the audience with predictors of success or failure of video games. It also provided a baseline for predicting future sales with the use of classification and time series models. Success of video games was defined as sales of over 0.5 million units globally. Random Forest had the best overall results based on accuracy, sensitivity and interpretability at 99.17% accuracy, therefore Random Forest was used as our champion model. Neural Network and Logistic Regression had high accuracy (99.83% accuracy) but neural net was deemed the challenger model since it had a longer processing time and less interpretability than Random Forest. All models used found that NA_Sales contributed the most important predictor from which to determine success in the north American market. Time series analysis using an ARIMA(2,1,0) model on monthly average sales showed a modest upward trend but with wide confidence intervals, signaling high uncertainty. These results support a strategic focus on North American markets and the use of Random Forest for future predictive tasks.

Part I: Business Insight Report

Background and Industry Context:

The video game industry has changed markedly in the last 40 years, growing and changing as technology has evolved, consumers' behaviors have shifted, and global digital platforms have emerged. With annual revenues of over $180 billion globally in recent years, the video game industry ranks as one of the most profitable entertainment industries we have today. Understanding the components that contribute to a game's success - whether it's the game's genre, platform, publisher or regional sales trends - can provide important information on how to maximize profitability for upcoming product launches

Dataset Overview:

This analysis uses a dataset that contains sales records for 11,470 video games that were released from the 1980s to 2010s. The dataset has 11 variables: Rank, Name, Platform, Year, Genre, Publisher, and regional sales amounts for North America, Europe, Japan, and Other, as well as Global_Sales. The variables present a range of ways to examine performance, by game genre and game platform and game publisher.

Business Objective:

The main objective of this analysis is to identify the strongest predictors of commercial success for a video game and provide recommendations for game developers/publishers for the future. We define business success in a binary way, based on whether or not a game has successful sales levels (over 0.5 million global sales). In addition to the classification task, we perform a time series forecast to assess sales trends over time. This is important for our stakeholders, since it assists them in identifying upcoming changes in market performance and gives them incentives to allocate for those shifts.

Exploratory Data Analysis (EDA)

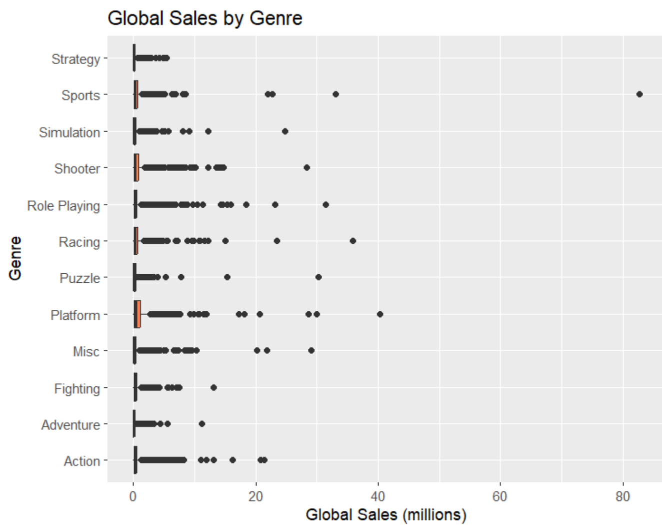
Descriptive Statistics:

```
> # Descriptive Statistics
> summary(df_clean[, c("NA_Sales", "EU_Sales", "JP_Sales", "Other_Sales", "Global_Sales")])
    NA_Sales          EU_Sales          JP_Sales         Other_Sales        Global_Sales   
 Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.00000   Min.   : 0.010  
 1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.00000   1st Qu.: 0.060  
 Median : 0.0700   Median : 0.0200   Median : 0.0000   Median : 0.01000   Median : 0.170  
 Mean   : 0.2864   Mean   : 0.1589   Mean   : 0.1052   Mean   : 0.05122   Mean   : 0.602  
 3rd Qu.: 0.2400   3rd Qu.: 0.1100   3rd Qu.: 0.0600   3rd Qu.: 0.03000   3rd Qu.: 0.510  
 Max.   :41.4900   Max.   :29.0200   Max.   :10.2200   Max.   :10.57000   Max.   :82.740  
```

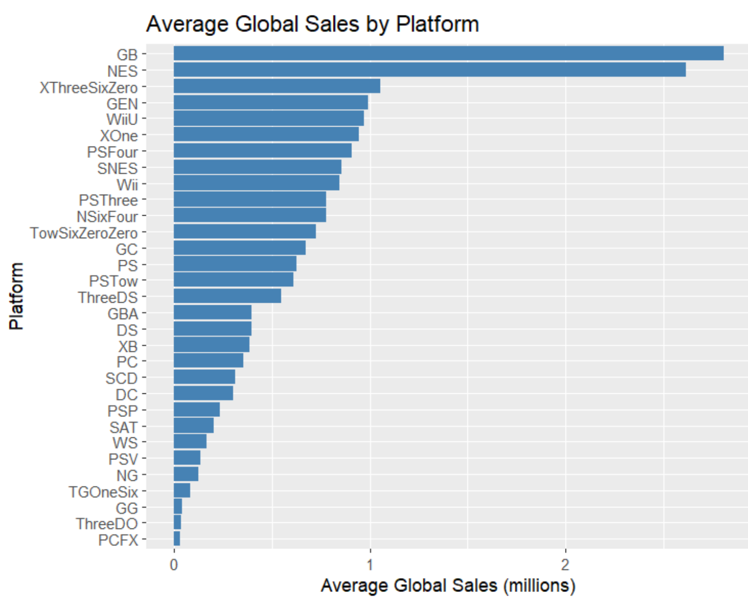
The summary of the numerical sales columns highlights substantial regional differences. On average, games sold 0.60 million units globally, with the breakdown showing North America leading (0.29 million), Europe (0.16 million), Japan (0.11 million), and other (0.05 million). However, the median values were a much lower value (0.17 million units globally) which suggests that there are a handful of games that generate most of the meaning and excess commercial value, therefore using inspection and numeric had shown that a small number of games accounted for an exchange of excess commercial success. The extreme maximum global sales value of 82.74 million further substantiate these distinct outliers. This distortion in reported means and medians is expected as limited titles likely drove the observed blockbuster effect within these categories, therefore the following observations indicate and support the rationale associated with implementing classification models to assess what determines success.

Visual Insights: Genre, Platform, Publisher, Sales Trends:

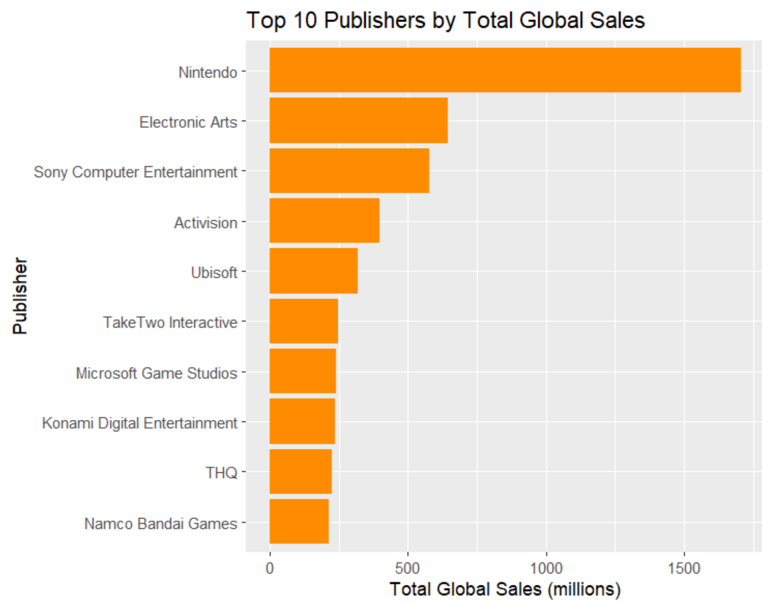
Genre: Boxplots illustrate the most outlying and, importantly, the highest sales range were Action, Sports, and Shooter games — most of the other genres stuck to the low sales figures supporting the highly skewed sales data.



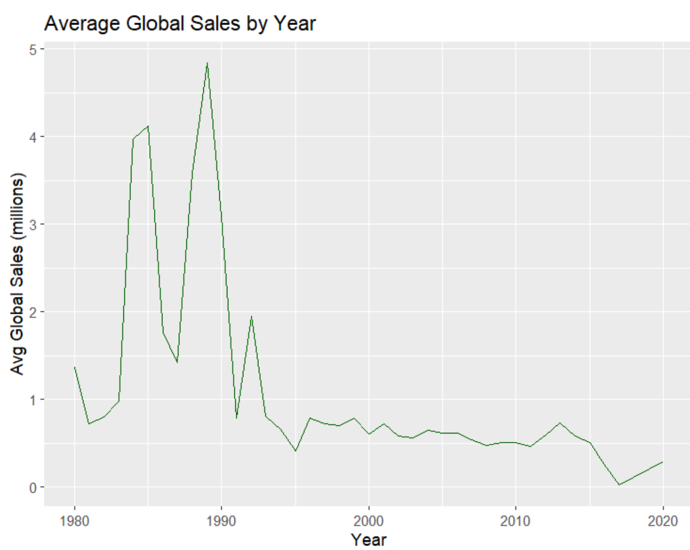
Platform: Overall, legacy platforms like Game Boy and NES outsold modern consoles in total global sales -- this is very likely attributed to reduced total games wing in the market and less competition and dominance during their existences.



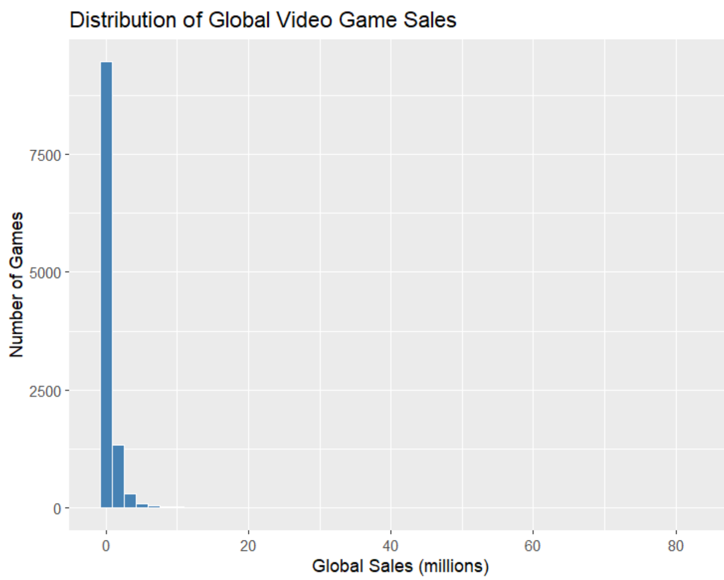
Publisher: There is a clear divide between the first place in total global sales, Nintendo, and concurrently 3rd place Electronic Arts and Sony -- the 'big three' publishers are impactful on the market and represent a strong predictor of success.



Sales Over Time: Average sales peaked in the early 1990s but ebbed after 2000 at a steady pace, with a decline indicating the growth of market fragmentation or shifting distribution models.



Sales Distribution: Most games did not sell more than 1 million units, but there were more than two extremely successful games. This limited information further supports the classification based approach for modelling.

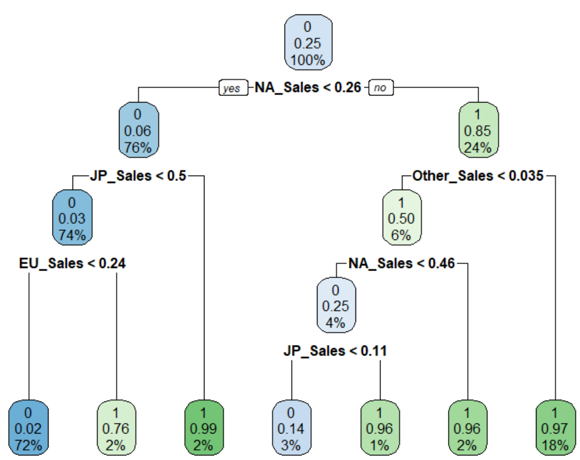


Predictive Modeling Frameworks

Framework 1: Gini Decision Tree:

We applied the Gini-based Decision Tree using both raw and normalized variables and observed nearly identical performance. The raw model achieved 96.86% accuracy, with strong sensitivity (97.76%) and specificity (94.00%). The most important variable was NA_Sales, followed by JP and EU sales. The tree revealed clear thresholds that distinguish successful from unsuccessful games, making it easy to interpret.

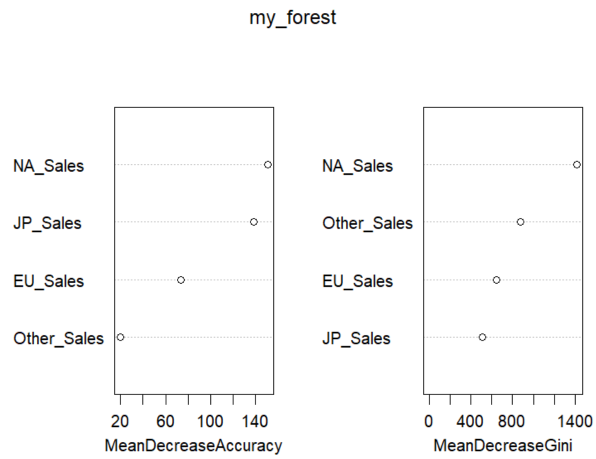
```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1705   33
         1   39  517

               Accuracy : 0.9686
                 95% CI : (0.9606, 0.9754)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9142

 Mcnemar's Test P-Value : 0.5557

            Sensitivity : 0.9776
            Specificity : 0.9400
```



Framework 2: Random Forest:

The Random Forest model produced highly accurate results, achieving an overall accuracy of 99.17%, sensitivity of 99.60%, and specificity of 97.82%. Just like the decision tree, we tested the model on both raw and normalized variables, and the outcomes were nearly identical. Among all features, NA_Sales was again the most influential predictor, followed by JP_Sales, EU_Sales, and Other_Sales. The model also showed strong generalizability with minimal overfitting, making it a reliable tool for classification tasks in this dataset.

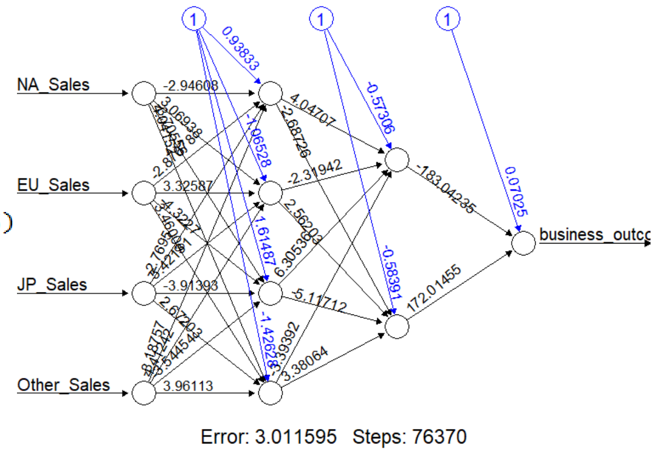
```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1737   12
         1    7  538

               Accuracy : 0.9917
                 95% CI : (0.9871, 0.995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9772

 Mcnemar's Test P-Value : 0.3588

            Sensitivity : 0.9960
            Specificity : 0.9782
```



Framework 3: Neural Network:

A (4,2) neural network architecture was used, with four input neurons representing regional sales and two hidden neurons to capture nonlinear relationships. This structure balanced complexity with training time and interpretability. The model achieved 99.8% accuracy, with 100% sensitivity and 99.3% specificity, outperforming all other models in both precision and generalizability. The model was also attempted to train the model with normalized variables, but it was computationally intensive and impractical within the timeframe. Despite this, the raw model was highly effective, demonstrating that even without normalization, the neural net captured patterns that simpler models may have missed.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1744    4
         1    0  546

               Accuracy : 0.9983
                 95% CI : (0.9955, 0.9995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9952

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9927
```



Framework 4: Logistic Regression:

Logistic regression was performed with both the raw and normalized variables. The results using both forms of the variables were virtually identical. The final model using raw variables produced 99.8% accuracy, with a sensitivity of 100% and specificity of 99.3, comparable to the neural network's results. In our case, logistic regression worked well to classify successful games despite using a more simplistic method. The results support a strong prediction using the chosen sales features. Therefore, when speed and transparency are more important than a complicated model, logistic regression can be a simple, interpretable alternative.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1744    4
         1    0  546

               Accuracy : 0.9983
                 95% CI : (0.9955, 0.9995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9952

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9927
```

Model Performance Comparison:

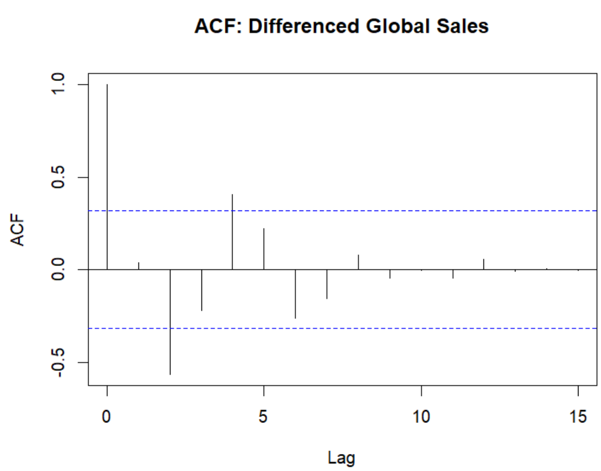
Both the Neural Network and Logistic Regression achieved the highest accuracy (99.83%), but the Neural Network had perfect sensitivity and was more robust across thresholds. Hence, **Neural Network is our challenger model**, with **Random Forest (99.17%) as the champion model** due to its balance of accuracy and interpretability. Gini Decision Tree delivered solid performance but ranked slightly lower. Across all models, NA_Sales consistently emerged as the most important predictor.

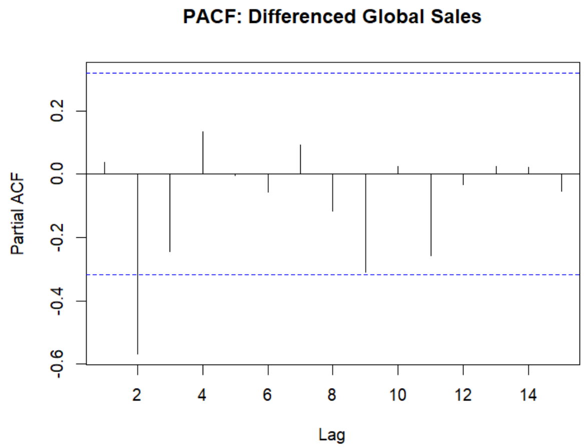
Time Series Forecasting

Stationarity Testing (ADF, ACF, PACF):

In order to complete the forecasting, first Global Sales data was converted into monthly averages. The series also proved to be non-stationary, and first order differencing has been applied. With a p-value from the Augmented Dickey-Fuller test of 0.017, the differenced series is stationary. The ACF and PACF plots exhibited short spikes with rapid decay, further giving support for forecasting with an ARIMA model.

```
        Augmented Dickey-Fuller Test

data:  diff_sales_ts
Dickey-Fuller = -4.0759, Lag order = 3, p-value = 0.01742
alternative hypothesis: stationary
```

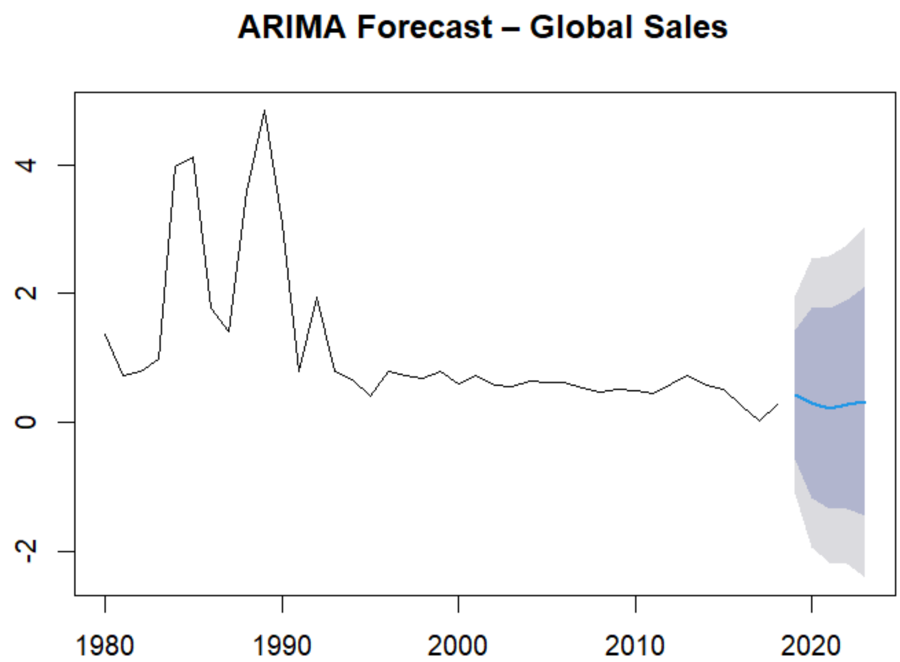




Framework 5: ARIMA Model and Forecast Results:

From the stationary series and ACF and PACF info, `auto.arima` found an ARIMA (2,1,0) model. This was chosen as the model because it includes a limited number of autoregressive effects, enough to do some limited short-term forecasting, and does not overfit the model to the dataset. The training set error metrics (RMSE of 0.75, MAE of 0.40) indicated reasonable prediction accuracy. The forecast indicates increasing average global sales, but with wide confidence intervals which again highlights the uncertainty of the industry.

```
Series: avg_sales_ts
ARIMA(2,1,0)

Coefficients:
         ar1      ar2
      0.0622  -0.5474
s.e.  0.1337   0.1286

sigma^2 = 0.6161:  log likelihood = -44.05
AIC=94.1   AICc=94.8   BIC=99.01

Training set error measures:
                     ME      RMSE       MAE      MPE     MAPE      MASE       ACF1
Training set -0.0388768 0.7541407 0.4046993 -52.4915 75.1254 0.7891611 -0.1342948
```



Business Insight and Recommendations

Key Insights:

- **North American sales dominate worldwide performance:** In all the models - Gini Tree, Random Forest, and Logistic Regression - NA_Sales was the most important variable in explaining a game's business success. This was demonstrated by being the root split in the decision tree and having the best ranking in Random Forest importance metrics (MeanDecreaseAccuracy and MeanDecreaseGini). This suggests that North America is a critical market driver for the success of video game sales.

- **Low regional sales lead to low business outcome:** The Gini Decision Tree displayed that games with NA_Sales lower than 0.26 along with low JP_Sales and EU_Sales predominantly did not achieve business success. This suggests a threshold effect: if a game does not achieve a minimum level of engagement in one, it will likely underperform in all global markets.

- **Random Forest is the best model:** Given the highest accuracy (99.17%) and an excellent trade-off between sensitivity (99.60%) and specificity (97.82%), Random

Forest produced the strongest overall predictive performance. Random Forest is a steady, consistent, and interpretable approach to identifying the main drivers of success.

- **Neural Network Matches Logistic Regression Performance, at a Higher Cost:** Neural nets were able to match the accuracy of logistic regression with an accuracy of 99.83%, however it required many more training steps and did not add any interpretability options. The logistic model was able to provide the same outcome with greater efficiency. However, it also tested the validity of the model in a situation when interpretability or computational resources may be limited.

- **Forecasts Suggest Modest Growth Trend with High Uncertainty:** The ARIMA time series modeling applied to monthly average global sales post-differencing produced a stationary pattern and reasonable fit (RMSE = 0.75). While the forecasts suggest a modest uptick, the confidence interval remains wide, suggesting high volatility within the market, and the need for scenario-based accommodating.

Actionable Recommendation:

- **Focus Strategic Investment in North America:** Future game launches should focus marketing, partners and distribution channel resources into North America where sales are the best predictor of global success. Resource and marketing campaigns should be allocated and tested in North America so that product-market fit can be established early.

- **Establish Inline Sales Thresholds as Go/No Go Gates:** For internal benchmarks, NA_Sales > 0.26, and Global_Sales > 0.6 should serve useful internal KPIs. Initial evaluations could be flagged for cost containment, or for strategic pivots if a title, based on these thresholds, have not been met within a few months.

- **Use Random Forest as Future Prediction Pipeline:** The Random Forest model is to be adopted as the champion modelling framework for any pre-launch evaluations or portfolio simulations. The Random Forest Model consistently captures complex interactions, and indicates which regional markets deserve the most focus.

- **Plan Around Uncertainty While Considering Forecasting Confidence Bands:** Business planning should consider best-case, base-case and worst-case for the future given the large intervals in the ARIMA forecast. These future confidence bands will allow for more flexible budgeting, more agile marketing responses, and better planning of inventory or server capacity around launches.

Part II (Appendix): Full R Code w/ Outputs

```r
###############################################################################
### Assignment A3: Video Game Sales - Business Case Modeling and Forecast ####
### Goal: Predict success of video games and forecast future sales ###########
### Student Name: Abi Joshua George (46656697) ###############################
###############################################################################

# --- Load Required Libraries ---
library(readr)
library(dplyr)
library(stringr)
library(lubridate)
library(ggplot2)
library(caret)
library(rpart)
library(rpart.plot)
library(randomForest)
library(neuralnet)
library(tseries)
library(forecast)

# -------------------------------|
# STEP 1: LOAD AND INITIAL CLEAN  |
# -------------------------------|

df <- read_csv("C:/Users/abijo/OneDrive/Desktop/Assignment A3/Video_Games_Sales.csv")
summary(df)
names(df)
```

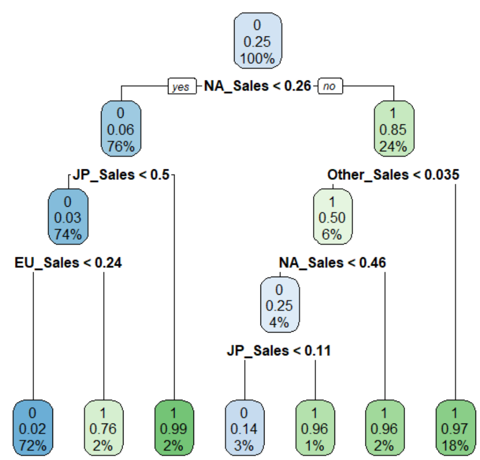
```
> summary(df)
      Rank           Name              Platform              Year        Genre
 Min.   :    1   Length:11470       Length:11470       Min.   :   0   Length:11470
 1st Qu.: 3963   Class :character   Class :character   1st Qu.:2002   Class :character
 Median : 8326   Mode  :character   Mode  :character   Median :2007   Mode  :character
 Mean   : 8293                                         Mean   :1977
 3rd Qu.:12645                                         3rd Qu.:2010
 Max.   :16599                                         Max.   :2020
  Publisher            NA_Sales          EU_Sales          JP_Sales         Other_Sales
 Length:11470       Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.00000
 Class :character   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.00000
 Mode  :character   Median : 0.0700   Median : 0.0200   Median : 0.0000   Median : 0.01000
                    Mean   : 0.2855   Mean   : 0.1577   Mean   : 0.1042   Mean   : 0.05087
                    3rd Qu.: 0.2400   3rd Qu.: 0.1100   3rd Qu.: 0.0600   3rd Qu.: 0.03000
                    Max.   :41.4900   Max.   :29.0200   Max.   :10.2200   Max.   :10.57000
  Global_Sales
 Min.   : 0.0100
 1st Qu.: 0.0600
 Median : 0.1700
 Mean   : 0.5984
 3rd Qu.: 0.5000
 Max.   :82.7400
> names(df)
 [1] "Rank"         "Name"         "Platform"     "Year"         "Genre"        "Publisher"
 [7] "NA_Sales"     "EU_Sales"     "JP_Sales"     "Other_Sales"  "Global_Sales"
>
```

```r
# ----------------------------------------|
# STEP 2: CLEANING AND FEATURE ENGINEERING |
# ----------------------------------------|

# Remove rows with NA Year or Global_Sales
df_clean <- df %>%
  filter(!is.na(Year), !is.na(Global_Sales), Global_Sales > 0)

# Convert Year to numeric
df_clean$Year <- as.numeric(df_clean$Year)

# Business outcome: Success if global sales > 0.5M units
df_clean$business_outcome <- ifelse(df_clean$Global_Sales > 0.5, 1, 0)

# Normalize numeric sales columns
rescale <- function(x) (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
df_clean$NA_Sales_norm     <- rescale(df_clean$NA_Sales)
df_clean$EU_Sales_norm     <- rescale(df_clean$EU_Sales)
df_clean$JP_Sales_norm     <- rescale(df_clean$JP_Sales)
df_clean$Other_Sales_norm  <- rescale(df_clean$Other_Sales)
df_clean$Global_Sales_norm <- rescale(df_clean$Global_Sales)

# Convert categorical columns
df_clean$Genre     <- as.factor(df_clean$Genre)
df_clean$Platform  <- as.factor(df_clean$Platform)
df_clean$Publisher <- as.factor(df_clean$Publisher)

summary(df_clean)

# ------------------------------|
# STEP 3: TRAIN-TEST SPLIT      |
# ------------------------------|

set.seed(123)
indx <- sample(1:nrow(df_clean), size = 0.8 * nrow(df_clean))
game_train <- df_clean[indx, ]
game_test  <- df_clean[-indx, ]
```

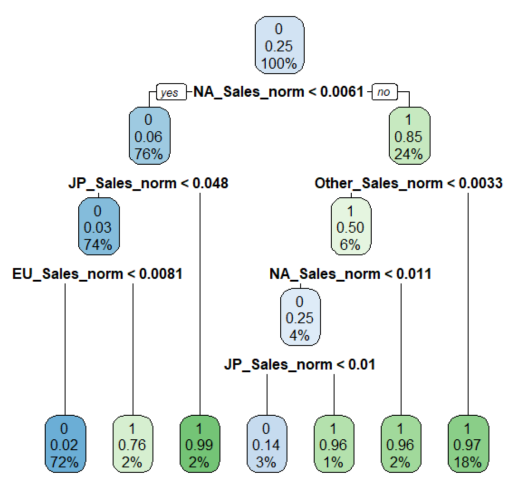
```
> summary(df_clean)
      Rank            Name              Platform        Year               Genre     
 Min.   :    1   Length:11470       PSTow  :1855   Min.   :   0   Action      :1917  
 1st Qu.: 3963   Class :character   DS     :1805   1st Qu.:2002   Sports      :1375  
 Median : 8326   Mode  :character   PS     :1121   Median :2007   Misc        :1327  
 Mean   : 8293                      Wii    : 948   Mean   :1977   Role Playing:1218  
 3rd Qu.:12645                      PSP    : 904   3rd Qu.:2010   Adventure   :1047  
 Max.   :16599                      PSThree: 705   Max.   :2020   Shooter     : 815  
                                    (Other):4132                  (Other)     :3771  
                         Publisher       NA_Sales          EU_Sales          JP_Sales      
 Namco Bandai Games          : 756   Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.0000  
 Nintendo                    : 660   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.0000  
 Konami Digital Entertainment: 633   Median : 0.0700   Median : 0.0200   Median : 0.0000  
 Sony Computer Entertainment : 612   Mean   : 0.2855   Mean   : 0.1577   Mean   : 0.1042  
 Electronic Arts             : 593   3rd Qu.: 0.2400   3rd Qu.: 0.1100   3rd Qu.: 0.0600  
 Ubisoft                     : 559   Max.   :41.4900   Max.   :29.0200   Max.   :10.2200  
 (Other)                     :7657                                                        
  Other_Sales        Global_Sales     business_outcome NA_Sales_norm      EU_Sales_norm     
 Min.   : 0.00000   Min.   : 0.0100   Min.   :0.0000   Min.   :0.000000   Min.   :0.0000000  
 1st Qu.: 0.00000   1st Qu.: 0.0600   1st Qu.:0.0000   1st Qu.:0.000000   1st Qu.:0.0000000  
 Median : 0.01000   Median : 0.1700   Median :0.0000   Median :0.001687   Median :0.0006892  
 Mean   : 0.05087   Mean   : 0.5984   Mean   :0.2494   Mean   :0.006881   Mean   :0.0054347  
 3rd Qu.: 0.03000   3rd Qu.: 0.5000   3rd Qu.:0.0000   3rd Qu.:0.005785   3rd Qu.:0.0037905  
 Max.   :10.57000   Max.   :82.7400   Max.   :1.0000   Max.   :1.000000   Max.   :1.0000000  
                                                                                             
 JP_Sales_norm      Other_Sales_norm    Global_Sales_norm  
 Min.   :0.000000   Min.   :0.0000000   Min.   :0.0000000  
 1st Qu.:0.000000   1st Qu.:0.0000000   1st Qu.:0.0006044  
 Median :0.000000   Median :0.0009461   Median :0.0019340  
 Mean   :0.010191   Mean   :0.0048125   Mean   :0.0071128  
 3rd Qu.:0.005871   3rd Qu.:0.0028382   3rd Qu.:0.0059229  
 Max.   :1.000000   Max.   :1.0000000   Max.   :1.0000000  
```

```r
# -------------------------------------------|
# Framework 1A: Decision Tree - RAW VARIABLES |
# -------------------------------------------|

my_tree <- rpart(business_outcome ~ NA_Sales + JP_Sales + EU_Sales + Other_Sales,
                 data = game_train, method = "class", cp = 0.01)
rpart.plot(my_tree)

tree_pred <- predict(my_tree, game_test)
confusionMatrix(
  data = factor(as.numeric(tree_pred[, 2] > 0.5), levels = c(0, 1)),
  reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
)

# -------------------------------------------------|
# Framework 1B: Decision Tree - NORMALIZED VARIABLES |
# -------------------------------------------------|

my_tree_norm <- rpart(business_outcome ~ NA_Sales_norm + JP_Sales_norm + EU_Sales_norm + Other_Sales_norm,
                      data = game_train, method = "class", cp = 0.01)
rpart.plot(my_tree_norm)

tree_pred_norm <- predict(my_tree_norm, game_test)
confusionMatrix(
  data = factor(as.numeric(tree_pred_norm[, 2] > 0.5), levels = c(0, 1)),
  reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1705   33
         1   39  517

               Accuracy : 0.9686
                 95% CI : (0.9606, 0.9754)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9142

 Mcnemar's Test P-Value : 0.5557

            Sensitivity : 0.9776
            Specificity : 0.9400
         Pos Pred Value : 0.9810
         Neg Pred Value : 0.9299
             Prevalence : 0.7602
         Detection Rate : 0.7432
   Detection Prevalence : 0.7576
      Balanced Accuracy : 0.9588

       'Positive' Class : 0
```

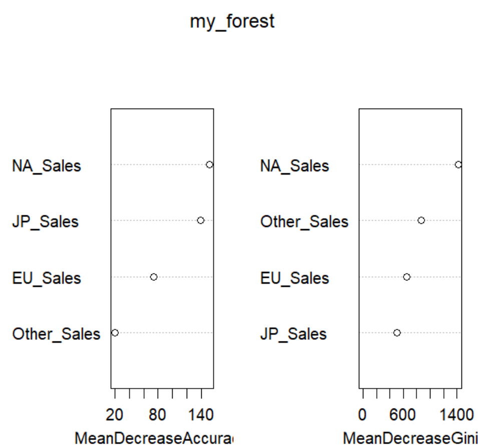



```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1705   33
         1   39  517

               Accuracy : 0.9686
                 95% CI : (0.9606, 0.9754)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9142

 Mcnemar's Test P-Value : 0.5557

            Sensitivity : 0.9776
            Specificity : 0.9400
         Pos Pred Value : 0.9810
         Neg Pred Value : 0.9299
             Prevalence : 0.7602
         Detection Rate : 0.7432
   Detection Prevalence : 0.7576
      Balanced Accuracy : 0.9588

       'Positive' Class : 0
```

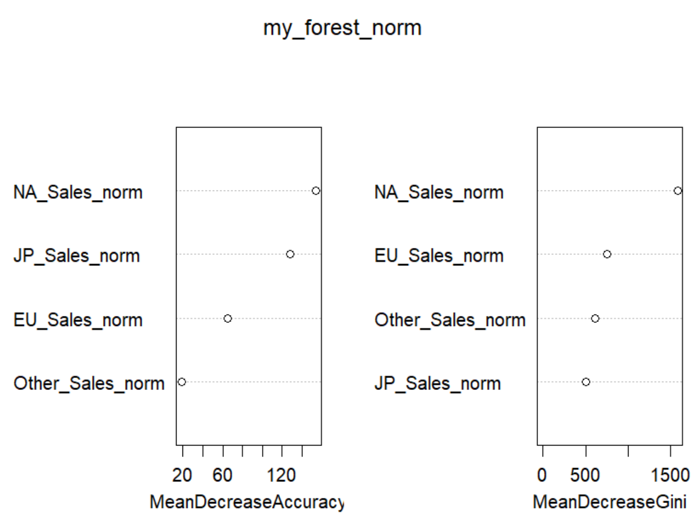


```r
# -----------------------------------------------|
# Framework 2A: Random Forest - RAW VARIABLES |
# -----------------------------------------------|

my_forest <- randomForest(as.factor(business_outcome) ~ NA_Sales + EU_Sales + JP_Sales + Other_Sales,
                          data = game_train, ntree = 100, mtry = 2, importance = TRUE)
varImpPlot(my_forest)

forest_pred <- predict(my_forest, game_test)
confusionMatrix(data = forest_pred, reference = factor(game_test$business_outcome))

# ------------------------------------------------------|
# Framework 2B: Random Forest - NORMALIZED VARIABLES |
# ------------------------------------------------------|

my_forest_norm <- randomForest(as.factor(business_outcome) ~ NA_Sales_norm + EU_Sales_norm +
                                 JP_Sales_norm + Other_Sales_norm,
                               data = game_train, ntree = 100, mtry = 2, importance = TRUE)
varImpPlot(my_forest_norm)

forest_pred_norm <- predict(my_forest_norm, game_test)
confusionMatrix(data = forest_pred_norm, reference = factor(game_test$business_outcome))
```

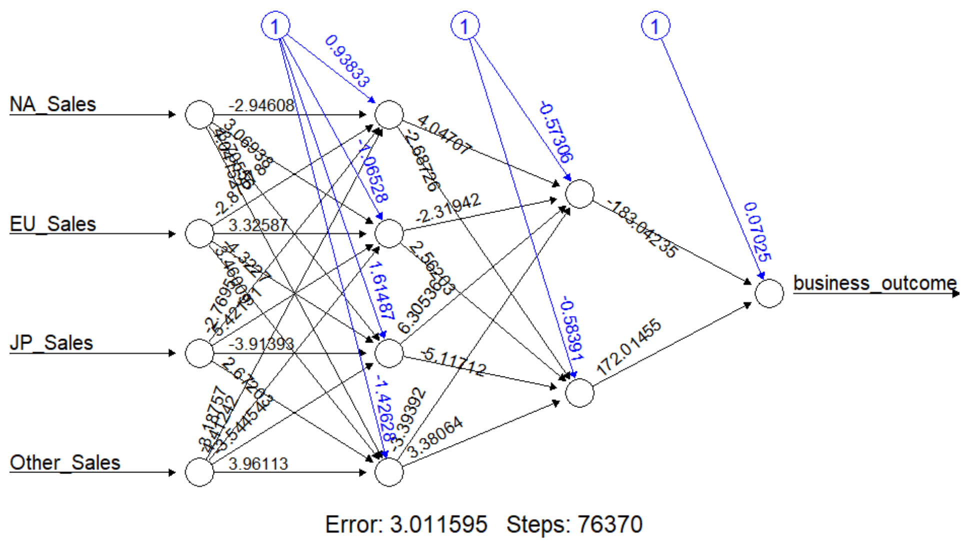
```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1737   12
         1    7  538

               Accuracy : 0.9917
                 95% CI : (0.9871, 0.995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9772

 Mcnemar's Test P-Value : 0.3588

            Sensitivity : 0.9960
            Specificity : 0.9782
         Pos Pred Value : 0.9931
         Neg Pred Value : 0.9872
             Prevalence : 0.7602
         Detection Rate : 0.7572
   Detection Prevalence : 0.7624
      Balanced Accuracy : 0.9871

       'Positive' Class : 0
```




```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1737   12
         1    7  538

               Accuracy : 0.9917
                 95% CI : (0.9871, 0.995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9772

 Mcnemar's Test P-Value : 0.3588

            Sensitivity : 0.9960
            Specificity : 0.9782
         Pos Pred Value : 0.9931
         Neg Pred Value : 0.9872
             Prevalence : 0.7602
         Detection Rate : 0.7572
   Detection Prevalence : 0.7624
      Balanced Accuracy : 0.9871

       'Positive' Class : 0
```



```r
# -------------------------------------------|
# Framework 3A: Neural Network - RAW VARIABLES |
# -------------------------------------------|

my_neural_raw <- neuralnet(business_outcome ~ NA_Sales + EU_Sales + JP_Sales + Other_Sales,
                           data = game_train, hidden = c(4, 2), linear.output = FALSE)
plot(my_neural_raw, rep = "best")

neural_pred_raw <- predict(my_neural_raw, game_test)
confusionMatrix(
  data = factor(as.numeric(neural_pred_raw > 0.5), levels = c(0, 1)),
  reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
)

# -----------------------------------------------------------------------|
# Framework 3B: Neural Network - NORMALIZED VARIABLES [TAKES TOO LONG TO RUN] |
# -----------------------------------------------------------------------|

my_neural_norm <- neuralnet(business_outcome ~ NA_Sales_norm + EU_Sales_norm +
                              JP_Sales_norm + Other_Sales_norm,
                            data = game_train, hidden = c(3, 2), linear.output = FALSE, stepmax = 1e6)
plot(my_neural_norm, rep = "best")

neural_pred_norm <- predict(my_neural_norm, game_test)
confusionMatrix(
  data = factor(as.numeric(neural_pred_norm > 0.5), levels = c(0, 1)),
  reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
)
```




```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1744    4
         1    0  546

               Accuracy : 0.9983
                 95% CI : (0.9955, 0.9995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9952

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9927
         Pos Pred Value : 0.9977
         Neg Pred Value : 1.0000
             Prevalence : 0.7602
         Detection Rate : 0.7602
   Detection Prevalence : 0.7620
      Balanced Accuracy : 0.9964

       'Positive' Class : 0
```

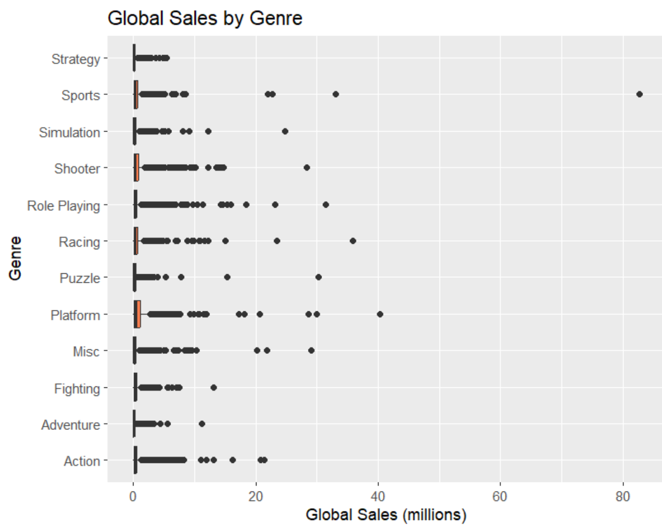
```r
# ------------------------------------------------|
# Framework 4A: Logistic Regression - RAW VARIABLES |
# ------------------------------------------------|

logit_model_raw <- glm(business_outcome ~ NA_Sales + EU_Sales + JP_Sales + Other_Sales,
                       data = game_train, family = binomial)
logit_pred_raw <- predict(logit_model_raw, game_test, type = "response")
confusionMatrix(
  data = factor(as.numeric(logit_pred_raw > 0.5), levels = c(0, 1)),
  reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
)

# -------------------------------------------------------|
# Framework 4B: Logistic Regression - NORMALIZED VARIABLES |
# -------------------------------------------------------|

logit_model_norm <- glm(business_outcome ~ NA_Sales_norm + EU_Sales_norm +
                          JP_Sales_norm + Other_Sales_norm,
                        data = game_train, family = binomial)
logit_pred_norm <- predict(logit_model_norm, game_test, type = "response")
confusionMatrix(
  data = factor(as.numeric(logit_pred_norm > 0.5), levels = c(0, 1)),
  reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1744    4
         1    0  546

               Accuracy : 0.9983
                 95% CI : (0.9955, 0.9995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9952

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9927
         Pos Pred Value : 0.9977
         Neg Pred Value : 1.0000
             Prevalence : 0.7602
         Detection Rate : 0.7602
   Detection Prevalence : 0.7620
      Balanced Accuracy : 0.9964

       'Positive' Class : 0
```

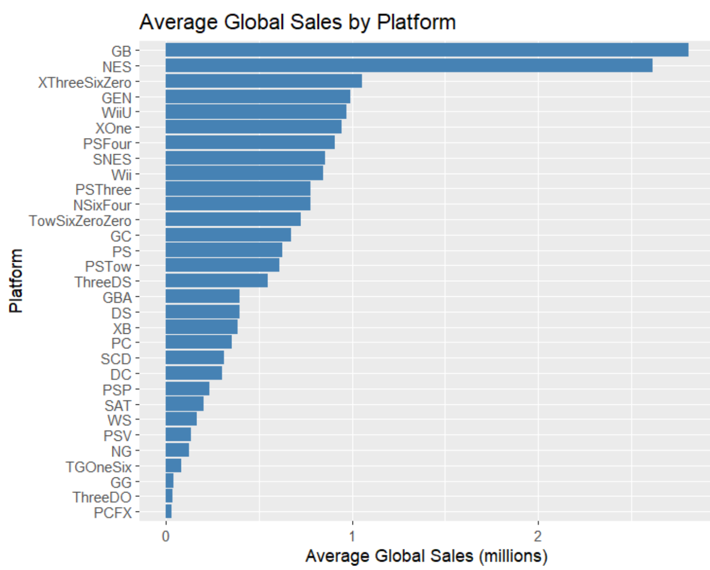
```
> logit_pred_norm <- predict(logit_model_norm, game_test, type = "response")
> confusionMatrix(
+   data = factor(as.numeric(logit_pred_norm > 0.5), levels = c(0, 1)),
+   reference = factor(as.numeric(game_test$business_outcome), levels = c(0, 1))
+ )
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1744    4
         1    0  546

               Accuracy : 0.9983
                 95% CI : (0.9955, 0.9995)
    No Information Rate : 0.7602
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9952

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 1.0000
            Specificity : 0.9927
         Pos Pred Value : 0.9977
         Neg Pred Value : 1.0000
             Prevalence : 0.7602
         Detection Rate : 0.7602
   Detection Prevalence : 0.7620
      Balanced Accuracy : 0.9964

       'Positive' Class : 0
```

```r
# ---------------------------------------------------------|
# Time Series Forecasting - Descriptive Statistics & Plots |
# ---------------------------------------------------------|

# Descriptive Statistics
summary(df_clean[, c("NA_Sales", "EU_Sales", "JP_Sales", "Other_Sales", "Global_Sales")])

# Histogram of Global Sales
ggplot(df_clean, aes(x = Global_Sales)) +
  geom_histogram(bins = 50, fill = "steelblue", color = "white") +
  labs(title = "Distribution of Global Video Game Sales",
       x = "Global Sales (millions)", y = "Number of Games")

# Year-wise average sales
df_clean <- df_clean %>%
  filter(Year >= 1980 & Year <= 2025)

df_clean %>%
  group_by(Year) %>%
  summarise(avg_sales = mean(Global_Sales, na.rm = TRUE)) %>%
  ggplot(aes(x = Year, y = avg_sales)) +
  geom_line(color = "darkgreen") +
  labs(title = "Average Global Sales by Year",
       x = "Year", y = "Avg Global Sales (millions)")

# Sales by Genre
ggplot(df_clean, aes(x = Genre, y = Global_Sales)) +
  geom_boxplot(fill = "coral") +
  coord_flip() +
  labs(title = "Global Sales by Genre", x = "Genre", y = "Global Sales (millions)")

# Average sales by Platform
df_clean %>%
  group_by(Platform) %>%
  summarise(avg_sales = mean(Global_Sales, na.rm = TRUE)) %>%
  ggplot(aes(x = reorder(Platform, avg_sales), y = avg_sales)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  labs(title = "Average Global Sales by Platform",
       x = "Platform", y = "Average Global Sales (millions)")
```

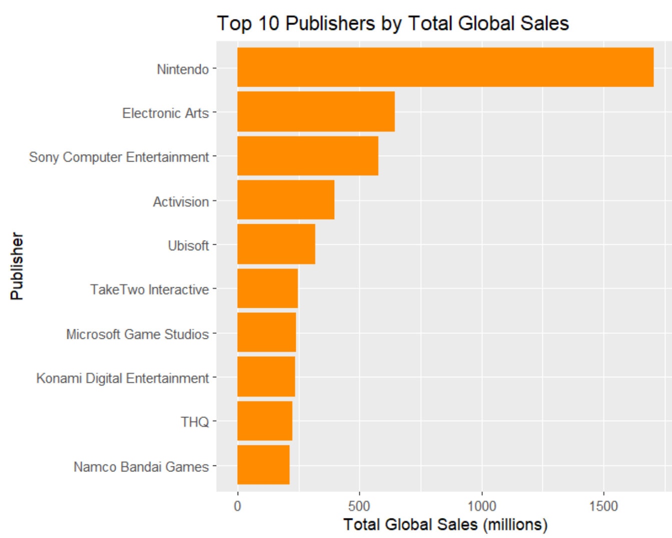
```r
# Summarize total global sales per publisher
publisher_sales <- df_clean %>%
  group_by(Publisher) %>%
  summarise(Total_Global_Sales = sum(Global_Sales, na.rm = TRUE)) %>%
  arrange(desc(Total_Global_Sales)) %>%
  top_n(10, Total_Global_Sales)

# Bar plot of top 10 publishers by total global sales
library(ggplot2)
ggplot(publisher_sales, aes(x = reorder(Publisher, Total_Global_Sales), y = Total_Global_Sales)) +
  geom_bar(stat = "identity", fill = "darkorange") +
  coord_flip() +
  labs(title = "Top 10 Publishers by Total Global Sales",
       x = "Publisher", y = "Total Global Sales (millions)")
```



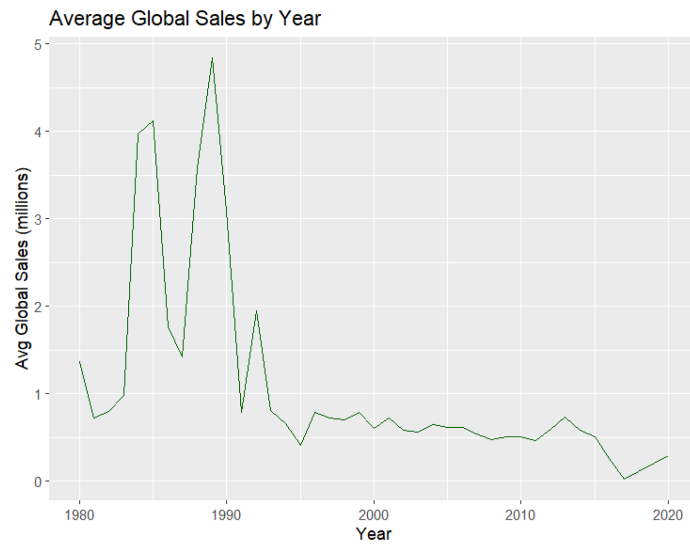
Global Sales by Genre
Strategy
Sports
Simulation
Shooter
Role Playing
Racing
Puzzle
Platform
Misc
Fighting
Adventure
Action
Genre
0
20
40
60
80
Global Sales (millions)

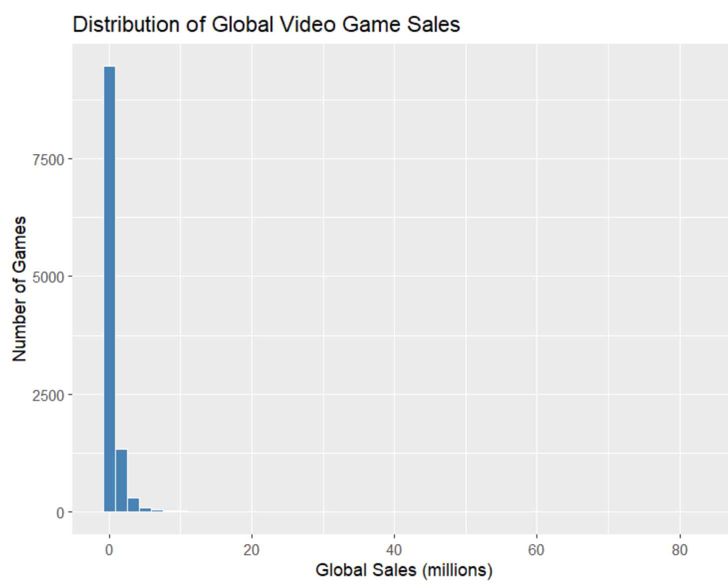


Average Global Sales by Platform
GB
NES
XThreeSixZero
GEN
WiiU
XOne
PSFour
SNES
Wii
PSThree
NSixFour
TowSixZeroZero
GC
PS
PSTow
ThreeDS
GBA
DS
XB
PC
SCD
DC
PSP
SAT
WS
PSV
NG
TGOneSix
GG
ThreeDO
PCFX
Platform
0
1
2
Average Global Sales (millions)



Top 10 Publishers by Total Global Sales
Nintendo
Electronic Arts
Sony Computer Entertainment
Activision
Ubisoft
TakeTwo Interactive
Microsoft Game Studios
Konami Digital Entertainment
THQ
Namco Bandai Games
Publisher
0
500
1000
1500
Total Global Sales (millions)

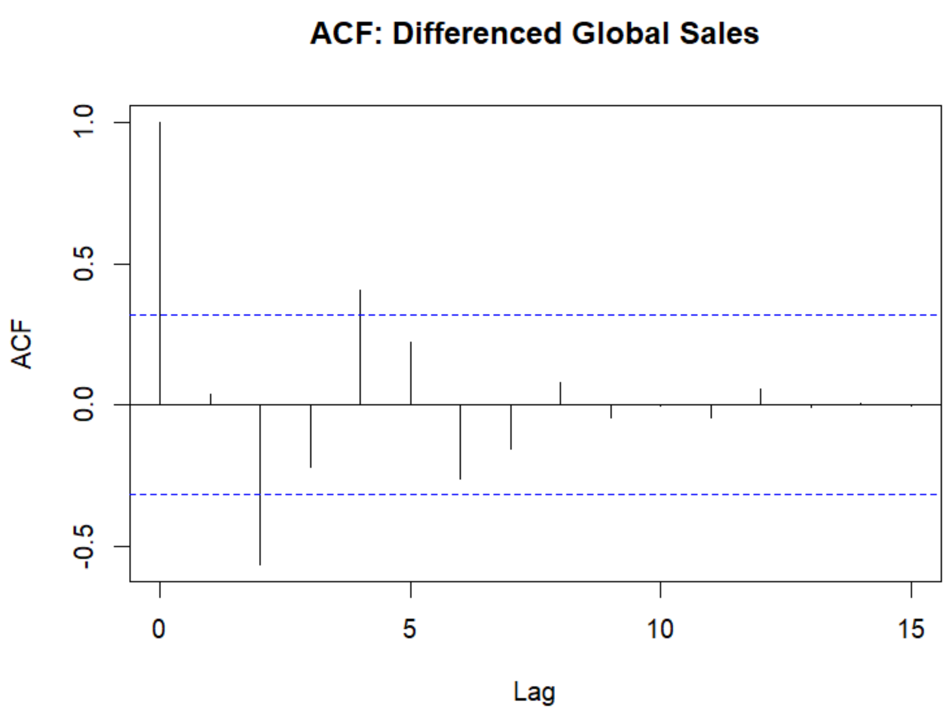


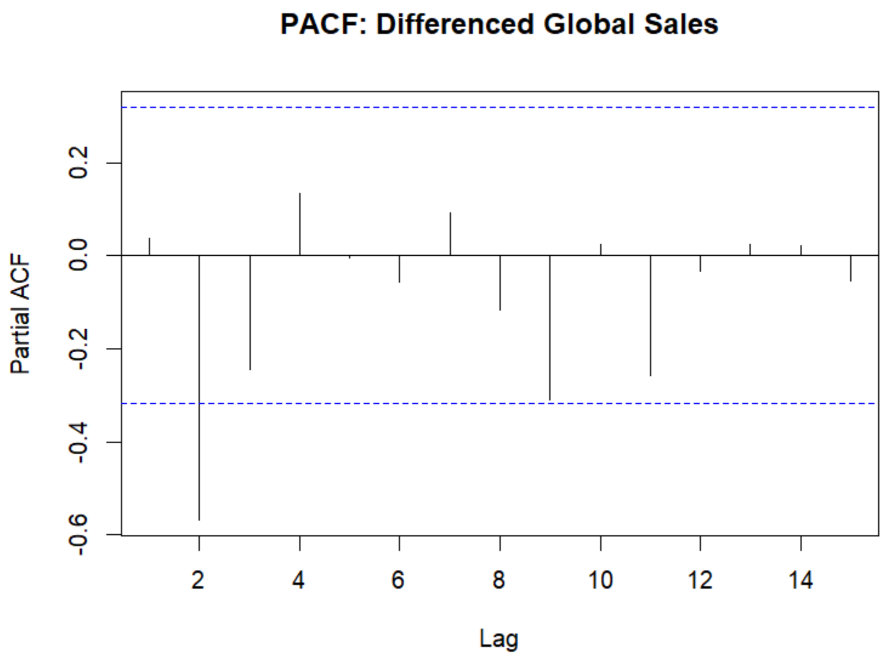



```
> summary(df_clean[, c("NA_Sales", "EU_Sales", "JP_Sales", "Other_Sales", "Global_Sales")])
    NA_Sales          EU_Sales          JP_Sales         Other_Sales     
 Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.00000  
 1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.00000  
 Median : 0.0700   Median : 0.0200   Median : 0.0000   Median : 0.01000  
 Mean   : 0.2864   Mean   : 0.1589   Mean   : 0.1052   Mean   : 0.05122  
 3rd Qu.: 0.2400   3rd Qu.: 0.1100   3rd Qu.: 0.0600   3rd Qu.: 0.03000  
 Max.   :41.4900   Max.   :29.0200   Max.   :10.2200   Max.   :10.57000  
  Global_Sales   
 Min.   : 0.010  
 1st Qu.: 0.060  
 Median : 0.170  
 Mean   : 0.602  
 3rd Qu.: 0.510  
 Max.   :82.740  
```

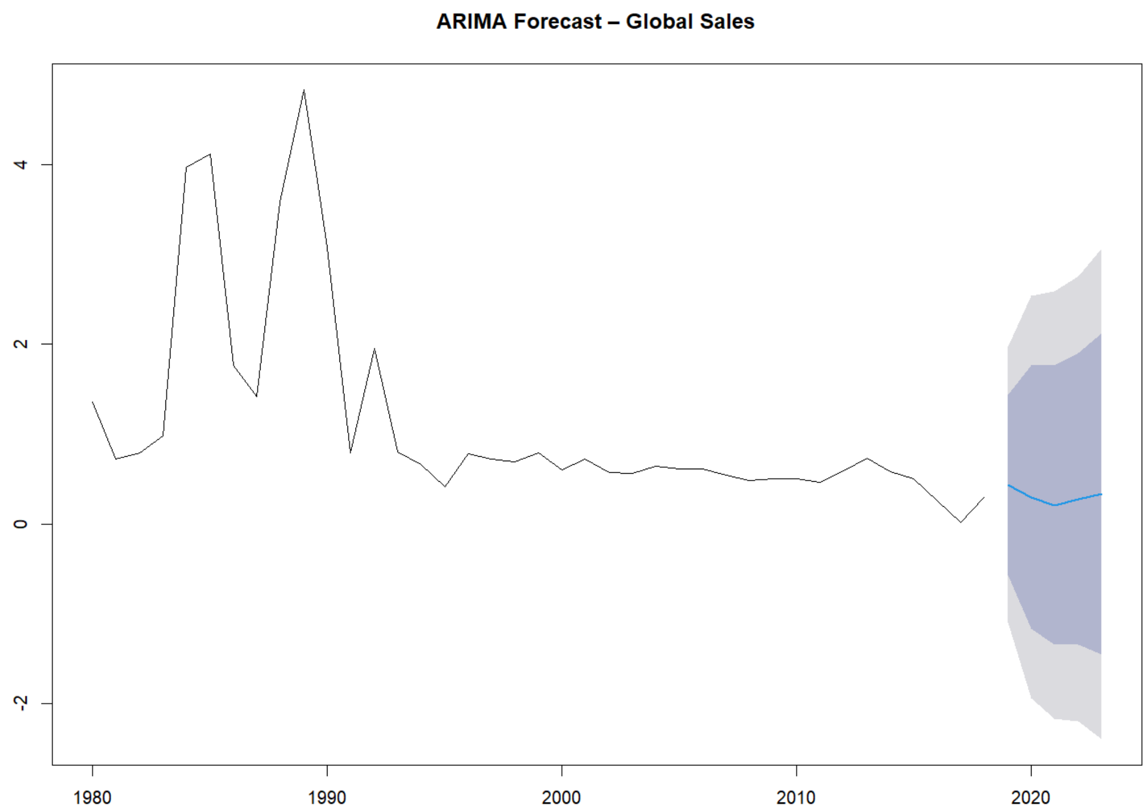
```r
# ---------------------------------------------------|
# Stationarity Checks: ADF, ACF, PACF - Time Series |
# ---------------------------------------------------|

avg_sales_yearly <- df_clean %>%
  group_by(Year) %>%
  summarise(avg_global_sales = mean(Global_Sales, na.rm = TRUE)) %>%
  filter(!is.na(Year) & Year >= 1980 & Year <= 2020)

avg_sales_ts <- ts(avg_sales_yearly$avg_global_sales, start = 1980, frequency = 1)
diff_sales_ts <- diff(avg_sales_ts)

adf.test(diff_sales_ts)
acf(diff_sales_ts, main = "ACF: Differenced Global Sales")
pacf(diff_sales_ts, main = "PACF: Differenced Global Sales")

# ------------------------------------|
# Framework 5: ARIMA Forecasting Model |
# ------------------------------------|

model_arima <- auto.arima(avg_sales_ts)
summary(model_arima)

forecast_arima <- forecast(model_arima, h = 5)
plot(forecast_arima, main = "ARIMA Forecast - Global Sales")
```

```
        Augmented Dickey-Fuller Test

data:  diff_sales_ts
Dickey-Fuller = -4.0759, Lag order = 3, p-value = 0.01742
alternative hypothesis: stationary
```






```
> summary(model_arima)
Series: avg_sales_ts
ARIMA(2,1,0)

Coefficients:
         ar1      ar2
      0.0622  -0.5474
s.e.  0.1337   0.1286

sigma^2 = 0.6161:  log likelihood = -44.05
AIC=94.1   AICc=94.8   BIC=99.01

Training set error measures:
                     ME      RMSE       MAE      MPE    MAPE      MASE       ACF1
Training set -0.0388768 0.7541407 0.4046993 -52.4915 75.1254 0.7891611 -0.1342948
>
```


ARIMA Forecast – Global Sales
4
2
0
-2
1980
1990
2000
2010
2020