Team #14

November 29, 2021

Video Streaming's Impact on the Box Office

Tony Allard
Namratha Acharya
Yueh-Ting Wu
Peter Yeu-Shyang Yeh

Abstract

In this paper, we analyze box office data from the past 20 years with regards to the potential impact video streaming has had on box office performance. A variety of data preprocessing were performed that included variables such as the timeline of video streaming services so we could effectively analyze the effect video streaming had on the box office. Logistic regression, K-Nearest Neighbors, Naive Bayes, decision tree, and neural network algorithms were used to analyze the data. After that, we close the paper with results and recommendations based on the outcomes of the algorithms. Our 1st recommendation is to not shut down movie theaters because of the impact of video streaming on profits. Our 2nd recommendation is movie theaters should consider hosting events on non-opening weekends because of the cyclical nature of earnings. Our final recommendation is that movie theaters should collaborate with streaming services to advertise their trailers for movies to the subscribers to boost movie theater attendance and profits.

Statement of the Problem

Over the past 20 years, streaming services such as Netflix and Hulu have impacted the way movies are consumed by the public. Rather than going into a movie theater, the public can stream and watch movies at home with a monthly subscription fee. In addition, original content is now produced and released exclusively on streaming services which could negatively impact the box office. The box office data is going to be used to determine the impact video streaming services have had on box office earnings, and develop suggestions as to how theaters and movie makers should proceed.

Dataset

We utilized two datasets for this analysis. The first dataset was acquired from Kaggle and contained data regarding the top weekend box office. Features included are listed in Table 1. The initial dataset had 3223 records from 1960 to 2021. The weekend gross was heavily right-skewed as can be seen in Figure 4.

The second dataset was self-generated and contained all of the Sundays for holiday weekends from the year 2000 to 2020. The reason why these years were selected will be discussed in the data preprocessing section. In total, there are 210 records in this dataset using the following holidays: New Year's, Valentine's Day, Mother's Day, Memorial Day, Father's Day, Fourth of July, Labor Day, Halloween, Veteran's Day, Thanksgiving, and Christmas.

Hypotheses:

1. The introduction of streaming services will lead to a loss in revenue for movie theaters. This is under the belief that accessibility to a library of movies at low prices will disincentivize potential customers from going to the theater.
2. The summer season is expected to have a direct relationship with box office gross. The remaining seasons are expected to have no relationship with box office gross. This is due to the idea that summer is often considered the blockbuster season.

Data Queries

1. Does the introduction of streaming services negatively impact movie box office earnings?
    - According to data preprocessing, there was no indication that streaming services negatively impacted the box office.
2. Are box office earnings higher over holiday weekends?
    - According to the generated linear model, holiday weekends have significantly higher earnings compared to non-holiday weekends.
3. Does season of the year impact or influence the box office earnings?
    - According to the generated linear model, seasons do have an impact on box office earnings. With the fall as a baseline, summer and spring do significantly increase box office earnings. Winter does not significantly change box office earnings.

Data Preprocessing

A list of generated features is stated in Table 2 in the appendix. A month feature was generated using the dates feature. Four binary season categories were also generated. If a record had the months 3, 4, or 5, SpringSeason was coded as 1. If a record had the months 6, 7, or 8, SummerSeason was coded as 1. If a record had the months 9, 10, or 11, FallSeason was coded as 1. All remaining records were coded with WinterSeason as 1.

Several binary features were generated regarding streaming services. From 2007 onwards, records were coded as 1 in Streaming to indicate that streaming services such as Netflix were available. From 2012 and onwards, records were coded as 1 in OriginalStreaming to indicate that streaming services were introducing their own original shows such as Netflix Originals. From 2017 and onwards, records were coded as 1 in InternationalStreaming to indicate that streaming services were now introducing content from foreign countries on their platforms.

In the interest of having comparable data, records from before 2000 were filtered out of the dataset because the year 2000 is often compared to the time when the entertainment industry and availability of the internet grew at unprecedented rates. The records after the year 2019 were also filtered out. These points were clearly outliers as seen by Figure 6. The low earnings were most likely due to the Covid-19 pandemic and lockdowns. After filtering the records, the dataset contained 1139 records.

A continuous variable called MilleniumWeek was also generated. The feature indicates the number of weeks into the millennium when the record occurred.

A binary feature called HolidayWeekend was generated and coded whether or not a record occurred during a holiday weekend. This was generated by comparing the date from box office data with the dates generated for the holiday weekend data. If the dates matched, the feature was coded as 1 to indicate that the records were during a holiday weekend.

A Box Cox transformation was applied to the gross to correct the skewness of the data. The linear model used for determining the necessary transformation used gross fitted to the millenium week number, winter season, spring season, summer season, holiday weekend, streaming, original streaming, and international streaming. Millennium week was used over the weekend and year features because correlation indicated that gross was more correlated to millennium week. The fall season was used as the baseline season. The suggested Box Cox transformation was -0.2 as seen in Figure 12 in the Appendix, and the NormalizedGross feature was generated. The results of the transformation can be seen in the histogram displayed in Figure 5.

Fitting the same model to the normalized gross, the scatter plot in Figure 7 was generated. A new binary Success feature was generated. If the records were on or above the fitted line, the record was coded as 0 or failures. Records below the fitted line were coded as 1 or success.

Algorithms

Logistic Regression

For logistic regression, seasons, streaming services, and holiday weekends were used as inputs to predict if the weekend box office would be a success or not. The model had a 58.2% accuracy. It was observed that the inclusion of holiday weekends improved the prediction accuracy. Streaming service features (Streaming,OriginalStreaming,InternationalStreaming) did not impact the accuracy to a great extent. The results can be seen on Table 18 in the Appendix.

K-Nearest Neighbors

The purpose of this model is to see how accurately season could predict a weekend box office's success. For the K-Nearest Neighbors model, Fourseason was set as the dependent variable. Gross, HolidayWeekend, Streaming, OriginalStreaming, InternationalStreaming, and Pandemic were the independent variables. To make a prediction is based on using those independent variables to predict which seasons the release of the film. The initial k was set as the square root of the number of records in creating the model. The predictive accuracy using this value of k was 31.03%. The confusion matrix is shown in Table 17 in the Appendix.
Using a for loop, a table of k values and their predictive accuracy was generated as seen in Table 20 in the Appendix. The highest predictive accuracy was 31.8%. While the predictive accuracy was low, this could be due to the fact that Fourseason was split into 4 categories.

Naïve Bayes

For the Naive Bayes model, we wished to see if features such as which season a movie was released in, whether or not the record was for a holiday weekend, and what streaming services were available would be good predictors as to whether or not a weekend was a success or not. Success was fitted using all of the binary variables. This model had a predictive accuracy of approximately 52%, however had the issue that many of the predictor variables were dependent on each other. In order to correct for the dependence, several other models were created using fewer variables. The confusion matrices are listed in Tables 3-14. The predictive accuracies can be seen in Table 19 in the Appendix. Because all of the predictive accuracies are approximately 50%, we can determine that seasons, holiday weekends, and streaming services are not good predictors for weekend box office success when using Naive Bayes. As a result of the ambiguity of the results, a conclusive answer regarding the effect of the predictor variables on the gross cannot be stated when using Naive Bayes methods.

Decision Tree

Our group created a decision tree model for the box office dataset. The intended purpose of this model is to find out which variables predict whether a film is successful or not in the box office. The variable that showed itself the most on this model was the MilleniumWeek variable. However, the predictive accuracy of this model was only 46.74%. This low predictive accuracy means that the decision tree model does not do a good job at predicting box office success based on the following variables: MilleniumWeek, WinterSeason, SpringSeason, SummerSeason, FallSeason, HolidayWeekend, Streaming,

OriginalStreaming and InternationalStreaming. The visualization of the decision tree is shown with Figure 10 in the Appendix.

Neural Network

Finally, our group created a neural network based on the box office dataset. The intention of the neural network was to have another model to predict box office success based on the presence of video streaming as an alternative platform for watching movies. The neural network model can be seen in Figure 11 in the Appendix. The predictive accuracy for the neural network model was 54.79%. This low predictive accuracy means that this neural network model is not good at predicting box office success based on the following variables: WinterSeason, SpringSeason, SummerSeason, FallSeason, HolidayWeekend, Streaming, OriginalStreaming and InternationalStreaming.

Best Model

For the Logistic Regression, K-Nearest Neighbors, Naïve Bayes, Decision Tree, and the Neural Network model we got predictive accuracies of 58.2%, 31.8%, 52%, 46.74%, and 54.79% respectively. All of the models were within close range in terms of predictive accuracy. However, with the given inputs we see that the Logistic Regression model is the most accurate model with 58.2% accuracy.

Recommendations

1. Don’t shut down movie theaters because of the impact of video streaming on profits.
2. Because of the cyclical nature of earnings, movie theaters should consider hosting events on non-opening weekends.
3. Movie theaters should collaborate with streaming services to advertise their trailers for movies to the subscribers to boost movie theater attendance.

Ethics

One ethical concern is that any data gained by the streaming services to recommend movie trailers should be kept confidential and not used for anything else. Streaming services users deserve to own their data and should not have to worry about other sources outside of streaming services using their data. Movie theaters should be concerned about how streaming services handle this data since the theaters are giving the streaming services the opportunity to collect it.

Team Members & Contributions

- Tony-27%
    - Abstract
    - Statement of the Problem
    - Decision Tree
    - Neural Network
    - Slides Formatting
    - Report Formatting
    - Ethics
- Peter-27%
    - Data Preprocessing (Code, Paper, Slides)
    - Naive Bayes (Code, Paper, Slides)
    - Appendix
- Alice-23%
    - K-Nearest Neighbors Model
    - Best model
- Namratha-23%
    - Hypothesis
    - Logistic Regression Model
    - Recommendations

Appendix-R Code

```
# Preliminary Setup ------------------------------------------------------
# Sets the working directory

# K-Nearest Neighbors
# Works well without MASS package
# Installs and loads tidyverse library
# install.packages("tidyverse")
library(tidyverse)

# Installs and loads tidyverse library
# install.packages("lubridate")
library(lubridate)


# Installs and loads corrplot library
# install.packages("corrplot")
library(corrplot)

# Installs and loads e1071 library
# install.packages("e1071")
library(e1071)

# KNN R Code
# Install tidyverse packages
# install.packages("tidyverse")

# Load tidyverse libraries
library(class)
library(olsrr)

# Setting the working directory
setwd("/Users/robertallard/Desktop/MIS 545/Final Project")

# Read weekendsv2.csv into tibble called BoxOffice
# l for logical
# n for numeric
# i for integer
# c for character
# f for factor
# D for date
# T for datetime

BoxOffice <- read_csv(file = "weekendsv2.csv",
                      col_types = "ffnnn",
                      col_names = TRUE)

# Display the tibble
print(BoxOffice)

# Diplay the structure of the tibble
str(BoxOffice)

# Summary the tibble
summary(BoxOffice)

# Converts data from character into data
BoxOffice$Date <- as.Date(BoxOffice$Date, format="%m/%d/%Y")

# Explanatory Data Analysis ----------------------------------------------
# Recodes Date into Months
BoxOffice <- BoxOffice %>%
  mutate(Month = months(BoxOffice$Date))

# Recode Months into Season
BoxOffice <- BoxOffice %>%
  mutate(Season = 4)
for (i in 1:length(BoxOffice$Month)) {
  if (BoxOffice$Month[i] == "January" || BoxOffice$Month[i] =="February"
```

```
      || BoxOffice$Month[i] == "December") {
    BoxOffice$Season[i] <- 4
  } else if (BoxOffice$Month[i] == "November" || BoxOffice$Month[i] =="October"
             || BoxOffice$Month[i] == "September"){
    BoxOffice$Season[i] <-3
  } else if (BoxOffice$Month[i] == "July" || BoxOffice$Month[i] =="June"
             || BoxOffice$Month[i] == "August"){
    BoxOffice$Season[i] <- 2
  } else {
    BoxOffice$Season[i] <- 1
  }
}

# Dummy codes Seasons into 4 variables
BoxOffice <- BoxOffice %>%
  mutate(WinterSeason = 0)
BoxOffice <- BoxOffice %>%
  mutate(FallSeason = 0)
BoxOffice <- BoxOffice %>%
  mutate(SummerSeason = 0)
BoxOffice <- BoxOffice %>%
  mutate(SpringSeason = 0)
for (i in 1:length(BoxOffice$Month)) {
  if (BoxOffice$Month[i] == "January" || BoxOffice$Month[i] =="February"
      || BoxOffice$Month[i] == "December") {
    BoxOffice$WinterSeason[i] <- 1
  } else if (BoxOffice$Month[i] == "November" || BoxOffice$Month[i] =="October"
             || BoxOffice$Month[i] == "September"){
    BoxOffice$FallSeason[i] <- 1
  } else if (BoxOffice$Month[i] == "July" || BoxOffice$Month[i] =="June"
             || BoxOffice$Month[i] == "August"){
    BoxOffice$SummerSeason[i] <- 1
  } else {
    BoxOffice$SpringSeason[i] <- 1
  }
}
str(BoxOffice)

# Factorizes Winter Season
BoxOffice$WinterSeason <- as.factor(BoxOffice$WinterSeason)

# Factorizes Fall Season
BoxOffice$FallSeason <- as.factor(BoxOffice$FallSeason)

# Factorizes Summer Season
BoxOffice$SummerSeason <- as.factor(BoxOffice$SummerSeason)

# Factorizes Spring Season
BoxOffice$SpringSeason <- as.factor(BoxOffice$SpringSeason)

# Season factorization check
summary(BoxOffice)

# Select data from 2000 to 2021
BoxOffice2K <- filter(.data = BoxOffice,
                      Year > 1999)

# Week numeric check
summary(BoxOffice2K)

# HolidayWeekend Data for 2000-2020
holidayWeekend <- read_csv(file = "HolidayWeek.csv",
                           col_names = TRUE,
                           col_types = "f")

# Converts data from character into data
holidayWeekend$HolidayWeekend <- as.Date(holidayWeekend$HolidayWeekend,
                                         format="%m/%d/%Y")

# Summary indicates a NA output
summary(holidayWeekend)
```

```
# NA filled with the appropriate date
holidayWeekend$HolidayWeekend[82] <- as.Date("5/11/2008",
                                             format="%m/%d/%Y")

# Summary indicates a NA output
summary(holidayWeekend)

# Create For loop to generate a logical holiday weekend column
BoxOffice2K <- BoxOffice2K %>%
  mutate(HolidayWeekend = 0)
for (i in 1 : length(BoxOffice2K$Date)) {
  for (j in 1 : length(holidayWeekend$HolidayWeekend)) {
    if(BoxOffice2K$Date[i] == holidayWeekend$HolidayWeekend[j]) {
      BoxOffice2K$HolidayWeekend[i] <- 1
    }
  }
}

# Turns the HolidayWeekend into a logical feature
BoxOffice2K$HolidayWeekend <- as.factor(BoxOffice2K$HolidayWeekend)

# j does not match up with the number of holiday weekends
summary(BoxOffice2K$HolidayWeekend)

# Data Subsetting By Year ------------------------------------------------
# Streaming Availablility
BoxOffice2K <- BoxOffice2K %>%
  mutate(Streaming = ifelse(Year > 2007, 1,0))
BoxOffice2K$Streaming <- as.factor(BoxOffice2K$Streaming)

# Original Streaming Availablility
BoxOffice2K <- BoxOffice2K %>%
  mutate(OriginalStreaming  = ifelse(Year > 2012, 1, 0))
BoxOffice2K$OriginalStreaming <- as.factor(BoxOffice2K$OriginalStreaming)

# International Streaming Availablility
BoxOffice2K <- BoxOffice2K %>%
  mutate(InternationalStreaming = ifelse(Year > 2017, 1, 0))
BoxOffice2K$InternationalStreaming <-
  as.factor(BoxOffice2K$InternationalStreaming)

# Pandemic
BoxOffice2K <- BoxOffice2K %>%
  mutate(Pandemic = ifelse(Year > 2019, 1, 0))
BoxOffice2K$Pandemic <- as.factor(BoxOffice2K$Pandemic)

# Streaming factor check
summary(BoxOffice2K)

# Filter out the outlier data
BoxOfficePrePandemic <- filter(.data = BoxOffice2K, Pandemic ==0)

# Histogram
hist(BoxOfficePrePandemic$Gross)

# Create a new feature called Fourseason
BoxOfficePrePandemic <- BoxOfficePrePandemic %>%
  mutate(Fourseason = 0)
for (i in 1:length(BoxOfficePrePandemic$Month)) {
  if(BoxOfficePrePandemic$Season[i] == 4){
    BoxOfficePrePandemic$Fourseason[i] <- "Winter"
  } else if(BoxOfficePrePandemic$Season[i] ==3) {
    BoxOfficePrePandemic$Fourseason[i] <- "Fall"
  } else if (BoxOfficePrePandemic$Season[i] ==2) {
    BoxOfficePrePandemic$Fourseason[i] <- "Summer"
  } else{
    BoxOfficePrePandemic$Fourseason[i] <- "Spring"
  }

}
```

```
# K-nearest model
BoxOfficePrePandemicKnn <- BoxOfficePrePandemic %>% select(-Film)
BoxOfficePrePandemicKnnLabel <- BoxOfficePrePandemic %>% select(Fourseason)
BoxOfficePrePandemicKnn <- BoxOfficePrePandemic %>%
  select(Gross,
         HolidayWeekend,
         Streaming,
         OriginalStreaming,
         InternationalStreaming,
         Pandemic)

# Set.seed() function
set.seed(370)
# Split into BoxOfficeTraining and BoxOfficeTesting
sampleSet <- sample(nrow(BoxOfficePrePandemicKnn),
                    round(nrow(BoxOfficePrePandemicKnn) * 0.75),
                    replace = FALSE)
# Creating training set
BoxOfficeTraining <- BoxOfficePrePandemicKnn[sampleSet, ]
BoxOfficeTraingLabel <- BoxOfficePrePandemicKnnLabel[sampleSet, ]

# Creating the testing set
BoxOfficeTesting <- BoxOfficePrePandemicKnn[-sampleSet, ]
BoxOfficeTestingLabel <- BoxOfficePrePandemicKnnLabel[-sampleSet, ]

# Generate the model
BoxOfficePrediction <- knn(train = BoxOfficeTraining,
                           test = BoxOfficeTesting,
                           cl = BoxOfficeTraingLabel$Fourseason,
                           k = 59)

print(BoxOfficePrediction)
summary(BoxOfficePrediction)
# Confusion matrix
BoxOfficeConfusionMatrix <- table(BoxOfficeTestingLabel$Fourseason,
                                  BoxOfficePrediction)
print(BoxOfficeConfusionMatrix)

BoxOfficePredictiveAccuracy <- sum(diag(BoxOfficeConfusionMatrix)) /
  nrow(BoxOfficeTesting)
print(BoxOfficePredictiveAccuracy)

kValuematrix <- matrix(data = NA,
                       ncol = 2,
                       nrow = 0)
colnames(kValuematrix) <- c("k value", "Predictive accuracy")

for (kValue in 1:nrow(BoxOfficeTraining)) {
  if(kValue %% 2 != 0) {
    BoxOfficePrediction <- knn(train = BoxOfficeTraining,
                               test = BoxOfficeTesting,
                               cl = BoxOfficeTraingLabel$Fourseason,
                               k = 59)
    BoxOfficeConfusionMatrix <- table(BoxOfficeTestingLabel$Fourseason,
                                      BoxOfficePrediction)

    BoxOfficePredictiveAccuracy <- sum(diag(BoxOfficeConfusionMatrix)) /
      nrow(BoxOfficeTesting)

    kValuematrix <- rbind(kValuematrix, c(kValue, BoxOfficePredictiveAccuracy))

  }

}
print(kValuematrix)


# EDA and Naive Bayes R Code
```

```
# Peter Yeu-Shyang Yeh
# MIS 545 Section 02
# GroupProject.R
# Installs and loads MASS library
# install.packages("MASS")
library(MASS)

# Read in CSV ------------------------------------------------------------
# Reads the csv into R with date column
boxOffice <- read.csv(file = "weekendsv2.csv",
                      colClasses = "character",
                      na.strings="?")

# Converts data from character into data
boxOffice$Date <- as.Date(boxOffice$Date, format="%m/%d/%Y")

# Converts data from character into numeric values
boxOffice$Year <- as.numeric(boxOffice$Year)
boxOffice$Weekend <- as.numeric(boxOffice$Weekend)
boxOffice$Gross <- as.numeric(boxOffice$Gross)

# Converts to a tibble
as_tibble(boxOffice)

# Displays the tibble
print(boxOffice)
str(boxOffice)
summary(boxOffice)

# Explanatory Data Analysis ----------------------------------------------
# Recodes Date into Months
boxOffice <- boxOffice %>%
  mutate(Month = month(ymd(boxOffice$Date)))

boxOffice$Month <- as.numeric(boxOffice$Month)

# Recodes Months into Season
boxOffice <- boxOffice %>%
  mutate(WinterSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SpringSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SummerSeason = 0)

boxOffice <- boxOffice %>%
  mutate(FallSeason = 0)

for (i in 1:length(boxOffice$Month)) {
  if(boxOffice$Month[i] > 11) {
    boxOffice$WinterSeason[i] <- 1
  } else if(boxOffice$Month[i] > 8) {
    boxOffice$FallSeason[i] <- 1
  } else if(boxOffice$Month[i] > 5) {
    boxOffice$SummerSeason[i] <- 1
  } else if(boxOffice$Month[i] > 2) {
    boxOffice$SpringSeason[i] <- 1
  } else {
    boxOffice$WinterSeason[i] <- 1
  }
}

# Factorizes Winter Season
boxOffice$WinterSeason <- as.factor(boxOffice$WinterSeason)

# Factorizes Spring Season
boxOffice$SpringSeason <- as.factor(boxOffice$SpringSeason)

# Factorizes Season
boxOffice$SummerSeason <- as.factor(boxOffice$SummerSeason)
```

```
# Factorizes Season
boxOffice$FallSeason <- as.factor(boxOffice$FallSeason)

# Season factorization check
summary(boxOffice)

# Selects movies from 2000-2021
boxOffice2K <- filter(.data = boxOffice, Year > 1999)

# Coding in column of continuous week number since start of 2000
MilleniumWeek <- c(1:length(boxOffice2K$Gross))
boxOffice2K <- cbind(boxOffice2K, MilleniumWeek)

# Week numeric check
summary(boxOffice2K)

# HolidayWeekend Data for 2000-2020
holidayWeekend <- read.csv(file = "HolidayWeek.csv",
                           colClasses = "character",
                           na.strings="?")

# Converts data from character into data
holidayWeekend$HolidayWeekend <- as.Date(holidayWeekend$HolidayWeekend,
                                         format="%m/%d/%Y")
# Summary indicates a NA output
summary(holidayWeekend)

# NA filled with the appropriate date
holidayWeekend$HolidayWeekend[82] <- as.Date("5/11/2008",
                                             format="%m/%d/%Y")

# Summary indicates a NA output
summary(holidayWeekend)

# Generate holiday weekend column with 0 default
boxOffice2K <- boxOffice2K %>%
  mutate(HolidayWeekend = 0)

# Generate a nested for loop code holiday weekend with 1 for matching dates
for (i in 1:length(boxOffice2K$Date)) {
  for (j in 1:length(holidayWeekend$HolidayWeekend)) {
    if(boxOffice2K$Date[i] == holidayWeekend$HolidayWeekend[j]) {
      boxOffice2K$HolidayWeekend[i] <- 1
    }
  }
}

# Turns the HolidayWeekend into a logical feature
boxOffice2K$HolidayWeekend <- as.factor(boxOffice2K$HolidayWeekend)

# j does not match up with the number of holiday weekends
summary(boxOffice2K$HolidayWeekend)

# Data Subsetting By Year ------------------------------------------------
# Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(Streaming = ifelse(Year > 2007, 1, 0))
boxOffice2K$Streaming <- as.factor(boxOffice2K$Streaming)

# Original Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(OriginalStreaming = ifelse(Year > 2012, 1, 0))
boxOffice2K$OriginalStreaming <- as.factor(boxOffice2K$OriginalStreaming)

# International Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(InternationalStreaming = ifelse(Year > 2017, 1, 0))
boxOffice2K$InternationalStreaming <- as.factor(
  boxOffice2K$InternationalStreaming)
```

```
# Pandemic
boxOffice2K <- boxOffice2K %>%
  mutate(Pandemic= ifelse(Year > 2019, 1, 0))
boxOffice2K$Pandemic <- as.factor(boxOffice2K$Pandemic)

# Streaming factor check
summary(boxOffice2K)

# Visualizations ------------------------------------------------------------
# Plot, noticeable dip at year 2020.
plot(boxOffice2K$Gross ~ boxOffice2K$Date)

# Filter out the outlier data
boxOfficePrePandemic <- filter(.data = boxOffice2K, Pandemic == 0)

# Plot data of interest
plot(boxOfficePrePandemic$Gross ~ boxOfficePrePandemic$MilleniumWeek)

# Histogram
hist(boxOfficePrePandemic$Gross,
     xlab = "Gross")

# Correlation check, create dataset of only numeric columns
boxOfficePrePandemicCorrelation <- c(boxOfficePrePandemic$Gross,
                                     boxOfficePrePandemic$Year,
                                     boxOfficePrePandemic$Weekend,
                                     boxOfficePrePandemic$MilleniumWeek)

# Gives dimensions to the matrix
dim(boxOfficePrePandemicCorrelation) <- c(1043,4)

# Gives labels to the matrix
colnames(boxOfficePrePandemicCorrelation) <- c("Gross",
                                               "Year",
                                               "Weekend",
                                               "MilleniumWeek")

# Visualizes the correlations
corrplot(cor(boxOfficePrePandemicCorrelation),
         method = "number",
         type = "lower")

# Correlation of Year and Gross
cor(boxOfficePrePandemicCorrelation[,1], boxOfficePrePandemicCorrelation[,2])

# Correlation of MilleniumWeek and Gross
cor(boxOfficePrePandemicCorrelation[,1], boxOfficePrePandemicCorrelation[,4])

# Gross is skewed. Need to generate box cox to determine transformation
boxOfficePrePandemicLinear <- lm(data = boxOfficePrePandemic, Gross ~
                                   MilleniumWeek +

                                   # FallSeason is the baseline
                                   WinterSeason +
                                   SpringSeason +
                                   SummerSeason +
                                   HolidayWeekend +

                                   # No Streaming is the baseline
                                   Streaming +
                                   OriginalStreaming +
                                   InternationalStreaming)

# Generates box cox transformations, -0.2 is the best transformation
grossBoxCox <- boxcox(boxOfficePrePandemicLinear,
                      lambda = seq(-.5,.3,0.1),
                      interp = F)

# Generates Normalized Gross using boxcox transformation
boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(NormalizedGross = Gross^(-0.2))
```

```
summary(boxOfficePrePandemic)

# Visualize normalized gross
hist(boxOfficePrePandemic$NormalizedGross)

# Correlation check, create dataset of only numeric columns
boxOfficePrePandemicCorrelation <- c(boxOfficePrePandemic$NormalizedGross,
                                     boxOfficePrePandemic$Year,
                                     boxOfficePrePandemic$Weekend,
                                     boxOfficePrePandemic$MilleniumWeek)

# Gives dimensions to the matrix
dim(boxOfficePrePandemicCorrelation) <- c(1043,4)

# Gives labels to the matrix
colnames(boxOfficePrePandemicCorrelation) <- c("NormalizedGross",
                                               "Year",
                                               "Weekend",
                                               "MilleniumWeek")

# Visualizes the correlations
corrplot(cor(boxOfficePrePandemicCorrelation),
         method = "number",
         type = "lower")

# Correlation of Year and NormalizedGross
cor(boxOfficePrePandemicCorrelation[,1], boxOfficePrePandemicCorrelation[,2])

# Correlation of MilleniumWeek and Gross
cor(boxOfficePrePandemicCorrelation[,1], boxOfficePrePandemicCorrelation[,4])

# Generate a scattterplot
plot(boxOfficePrePandemic$NormalizedGross ~ boxOfficePrePandemic$MilleniumWeek,
     xlab = "Millenium Week",
     ylab = "Normalized Gross")

# Generate linear model using normalized gross
boxOfficeNormalLM <- lm(data = boxOfficePrePandemic, NormalizedGross ~
                          MilleniumWeek +

                          # FallSeason is the baseline
                          WinterSeason +
                          SpringSeason +
                          SummerSeason +
                          HolidayWeekend +

                          # No streaming is the baseline
                          Streaming +
                          OriginalStreaming +
                          InternationalStreaming)

# Generates red line of linear model
abline(boxOfficeNormalLM,
       col = "red")

# Summary of linear model
summary(boxOfficeNormalLM)

# Generate a success criteria using the fitted values of the model
boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(SuccessCriteria = fitted(boxOfficeNormalLM))

# Generates a success binary column with default 0
boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(Success = 0)

# For loop to code for success and fail
for (i in 1:length(boxOfficePrePandemic$NormalizedGross)) {

  # Compares Normalized Gross with Success Criteria
  if(boxOfficePrePandemic$NormalizedGross[i] <
```

```
     boxOfficePrePandemic$SuccessCriteria[i]) {

    # Success is for Normalized Gross less than the criteria
    boxOfficePrePandemic$Success[i] <- 1
  } else{
    boxOfficePrePandemic$Success[i] <- 0
  }
}

# Factorizes the Success
boxOfficePrePandemic$Success <- as.factor(boxOfficePrePandemic$Success)

# Summary of the final data set
summary(boxOfficePrePandemic)

# Naive Bayes ------------------------------------------------------------
# Establishes seed for random sampling
# set.seed(154)

# Creates a 75:25 smaple split without replacement from sedanSize
sampleSet <- sample(nrow(boxOfficePrePandemic),
                    round(nrow(boxOfficePrePandemic) * 0.75),
                    replace = FALSE)

# Creates the Training set using 75% of the data
boxOfficeTraining <- boxOfficePrePandemic[sampleSet,]

# Creates the Testing set using 25% of the data
boxOfficeTesting <- boxOfficePrePandemic[-sampleSet,]

# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ WinterSeason +
                               SpringSeason +
                               SummerSeason +
                               FallSeason +
                               HolidayWeekend +
                               Streaming +
                               OriginalStreaming +
                               InternationalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 52
print(predictiveAccuracy)

# Correcting for Dependence ----------------------------------------------
# Streaming --------------------------------------------------------------
```

```
# Fall: 49.04%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ FallSeason +
                               HolidayWeekend +
                               Streaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Winter = 49.04%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ WinterSeason +
                               HolidayWeekend +
                               Streaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Spring = 49.04%
```

```
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ SpringSeason +
                               HolidayWeekend +
                               Streaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Summer = 50.57%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ SummerSeason +
                               HolidayWeekend +
                               Streaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.50
print(predictiveAccuracy)


# Original Streaming ----------------------------------------------------
```

```
# Fall: 49.04%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ FallSeason +
                               HolidayWeekend +
                               OriginalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Winter = 49.04%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ WinterSeason +
                               HolidayWeekend +
                               OriginalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Spring = 49.04%
```

```
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ SpringSeason +
                               HolidayWeekend +
                               OriginalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Summer = 50.57%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ SummerSeason +
                               HolidayWeekend +
                               OriginalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.50
print(predictiveAccuracy)


# International -----------------------------------------------------------
```

```
# Fall: 50.96%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ FallSeason +
                               HolidayWeekend +
                               InternationalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.51
print(predictiveAccuracy)

# Winter = 47.89%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ WinterSeason +
                               HolidayWeekend +
                               InternationalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.48
print(predictiveAccuracy)

# Spring = 49.04%
```

```
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ SpringSeason +
                               HolidayWeekend +
                               InternationalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.49
print(predictiveAccuracy)

# Summer = 50.57%
# Generates the Naive Bayes Model
boxOfficeNaive <- naiveBayes(formula = Success ~ SummerSeason +
                               HolidayWeekend +
                               InternationalStreaming,
                             data = boxOfficeTraining,
                             laplace = 1)

# Builds probabilities for each record in the testing data
boxOfficeProbability <- predict(boxOfficeNaive,
                                boxOfficeTesting,
                                type = "raw")

# Displays the probabilities
print(boxOfficeProbability)

# Predicts classes for each record in testing set
boxOfficePrediction <- predict(boxOfficeNaive,
                               boxOfficeTesting,
                               type = "class")

# Displays the predictions
print(boxOfficePrediction)

# Create the Confusion Matrix
confusionMatrix <- table(boxOfficeTesting$Success,
                         boxOfficePrediction)
print(confusionMatrix)

# Calculate the model prediction accuracy.
predictiveAccuracy <- sum(diag(confusionMatrix))/nrow(boxOfficeTesting)

# Displays the predictive accuracy 0.51
print(predictiveAccuracy)

# Logistic Regression R Code
# Preliminary Setup ------------------------------------------------------
```

```
# Installs and loads tidyverse library
# install.packages("tidyverse")
library(tidyverse)

# Installs and loads tidyverse library
# install.packages("lubridate")
library(lubridate)

# Installs and loads MASS library
# install.packages("MASS")
library(MASS)

# For group by
library(dplyr)

library(corrplot)
library(olsrr)
library(smotefamily)

# Read in CSV -------------------------------------------------------------
# Reads the csv into R with date column
boxOffice <- read.csv(file = "weekendsv2.csv",
                      colClasses = "character",
                      na.strings="?")

# Converts data from character into data
boxOffice$Date <- as.Date(boxOffice$Date, format="%m/%d/%Y")

# Converts data from character into numeric values
boxOffice$Year <- as.numeric(boxOffice$Year)
boxOffice$Weekend <- as.numeric(boxOffice$Weekend)
boxOffice$Gross <- as.numeric(boxOffice$Gross)

# Converts to a tibble
as_tibble(boxOffice)

# Displays the tibble
print(boxOffice)
str(boxOffice)
summary(boxOffice)

# Explanatory Data Analysis -----------------------------------------------
# Recodes Date into Months
boxOffice <- boxOffice %>%
  mutate(Month = month(ymd(boxOffice$Date)))

boxOffice$Month <- as.numeric(boxOffice$Month)

# Recodes Months into Season
boxOffice <- boxOffice %>%
  mutate(WinterSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SpringSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SummerSeason = 0)

boxOffice <- boxOffice %>%
  mutate(FallSeason = 0)

for (i in 1:length(boxOffice$Month)) {
  if(boxOffice$Month[i] > 11) {
    boxOffice$WinterSeason[i] <- 1
  } else if(boxOffice$Month[i] > 8) {
    boxOffice$FallSeason[i] <- 1
  } else if(boxOffice$Month[i] > 5) {
    boxOffice$SummerSeason[i] <- 1
  } else if(boxOffice$Month[i] > 2) {
    boxOffice$SpringSeason[i] <- 1
```

```
  } else {
    boxOffice$WinterSeason[i] <- 1
  }
}

# Factorizes Winter Season
boxOffice$WinterSeason <- as.factor(boxOffice$WinterSeason)

# Factorizes Spring Season
boxOffice$SpringSeason <- as.factor(boxOffice$SpringSeason)

# Factorizes Season
boxOffice$SummerSeason <- as.factor(boxOffice$SummerSeason)

# Factorizes Season
boxOffice$FallSeason <- as.factor(boxOffice$FallSeason)

# Season factorization check
summary(boxOffice)

# Selects movies from 2000-2021
boxOffice2K <- filter(.data = boxOffice, Year > 1999)

# Coding in column of continuous week number since start of 2000
MilleniumWeek <- c(1:length(boxOffice2K$Gross))

boxOffice2K <- cbind(boxOffice2K, MilleniumWeek)

# Week numeric check
summary(boxOffice2K)

# HolidayWeekend Data for 2000-2020
holidayWeekend <- read.csv(file = "HolidayWeek.csv",
                           colClasses = "character",
                           na.strings="?")

# Converts data from character into data
holidayWeekend$HolidayWeekend <- as.Date(holidayWeekend$HolidayWeekend,
                                         format="%m/%d/%Y")
# Summary indicates a NA output
summary(holidayWeekend)

# NA filled with the appropriate date
holidayWeekend$HolidayWeekend[82] <- as.Date("5/11/2008",
                                             format="%m/%d/%Y")

# Summary indicates a NA output
summary(holidayWeekend)

# Create For loop to generate a logical holiday weekend column
boxOffice2K <- boxOffice2K %>%
  mutate(HolidayWeekend = 0)

for (i in 1:length(boxOffice2K$Date)) {
  for (j in 1:length(holidayWeekend$HolidayWeekend)) {
    if(boxOffice2K$Date[i] == holidayWeekend$HolidayWeekend[j]) {
      boxOffice2K$HolidayWeekend[i] <- 1
    }
  }
}

# Turns the HolidayWeekend into a logical feature
boxOffice2K$HolidayWeekend <- as.factor(boxOffice2K$HolidayWeekend)

# j does not match up with the number of holiday weekends
summary(boxOffice2K$HolidayWeekend)

# Data Subsetting By Year ------------------------------------------------
# Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(Streaming = ifelse(Year > 2007, 1, 0))
```

```
boxOffice2K$Streaming <- as.factor(boxOffice2K$Streaming)

# Original Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(OriginalStreaming = ifelse(Year > 2012, 1, 0))
boxOffice2K$OriginalStreaming <- as.factor(boxOffice2K$OriginalStreaming)

# International Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(InternationalStreaming = ifelse(Year > 2017, 1, 0))
boxOffice2K$InternationalStreaming <- as.factor(
  boxOffice2K$InternationalStreaming)

# Pandemic
boxOffice2K <- boxOffice2K %>%
  mutate(Pandemic= ifelse(Year > 2019, 1, 0))
boxOffice2K$Pandemic <- as.factor(boxOffice2K$Pandemic)

# Streaming factor check
summary(boxOffice2K)

# Visualizations ------------------------------------------------------------
# Plot, noticeable dip at year 2020.
plot(boxOffice2K$Gross ~ boxOffice2K$Date)

# Filter out the outlier data
boxOfficePrePandemic <- filter(.data = boxOffice2K, Pandemic == 0)

# Plot data of interest
plot(boxOfficePrePandemic$Gross ~ boxOfficePrePandemic$MilleniumWeek)

# Histogram
hist(boxOfficePrePandemic$Gross)

# Correlation check, Use MilleniumWeek since most correlated to Gross
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$MilleniumWeek)
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$Year)
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$Weekend)

# Gross is skewed. Need to generate box cox to determine transformation
boxOfficePrePandemicLinear <- lm(data = boxOfficePrePandemic, Gross ~
                                   MilleniumWeek +
                                   WinterSeason +
                                   SpringSeason +
                                   SummerSeason +
                                   HolidayWeekend +
                                   Streaming +
                                   OriginalStreaming +
                                   InternationalStreaming)

grossBoxCox <- boxcox(boxOfficePrePandemicLinear,
                      lambda = seq(-.5,.3,0.1),
                      interp = F)

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(NormalizedGross = Gross^(-0.2))

summary(boxOfficePrePandemic)

hist(boxOfficePrePandemic$NormalizedGross)

cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$MilleniumWeek)
cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$Year)
cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$Weekend)

plot(boxOfficePrePandemic$NormalizedGross ~ boxOfficePrePandemic$MilleniumWeek)
boxOfficeNormalLM <- lm(data = boxOfficePrePandemic, NormalizedGross ~
                          MilleniumWeek +
                          WinterSeason +
                          SpringSeason +
                          SummerSeason +
```

```
                          HolidayWeekend +
                          Streaming +
                          OriginalStreaming +
                          InternationalStreaming)
abline(boxOfficeNormalLM,
       col = "red")
summary(boxOfficeNormalLM)

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(SuccessCriteria = fitted(boxOfficeNormalLM))

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(Success = 0)
for (i in 1:length(boxOfficePrePandemic$NormalizedGross)) {
  if(boxOfficePrePandemic$NormalizedGross[i] <
     boxOfficePrePandemic$SuccessCriteria[i]) {
    boxOfficePrePandemic$Success[i] <- 1
  } else{
    boxOfficePrePandemic$Success[i] <- 0
  }
}

boxOfficePrePandemic$Succes <- as.factor(boxOfficePrePandemic$Success)


summary(boxOfficePrePandemic)

-----------------------------------

#seed variable is set to ensure we get the same result every time we run a
#sampling process

# set.seed(203)

#create a vector of 75% randomly sampled rows from the original data
sample<- sample(nrow(boxOfficePrePandemic),
                      round(nrow(boxOfficePrePandemic) * 0.75),
                      replace = FALSE)

# Put the records (75%) sample into traning data set
boxOfficePrePandemicTraining <- boxOfficePrePandemic[sample, ]

# Put the records (25%) sample into testing data set
boxOfficePrePandemicTesting <- boxOfficePrePandemic[-sample, ]

# Check for class imbalance
summary(boxOfficePrePandemic$Succes)

# Generate the logistic regression model
boxOfficePrePandemicModel <- glm(boxOfficePrePandemic,
                                 family = binomial,
                                 formula = Succes ~ WinterSeason +
                                   SpringSeason +
                                   SummerSeason +
                                   FallSeason +
                                   HolidayWeekend +
                                   Streaming +
                                   OriginalStreaming +
                                   InternationalStreaming)

boxOfficePrePandemicModel <- glm(boxOfficePrePandemic,
                                 family = binomial,
                                 formula = Succes ~ WinterSeason +
                                   SpringSeason +
                                   SummerSeason +
                                   FallSeason +
                                   Year +
                                   HolidayWeekend +
                                   Streaming +
                                   OriginalStreaming +
                                   InternationalStreaming)
```

```
summary(boxOfficePrePandemicModel)


# Predicting testing data
boxOfficePrePandemicModelpredict<- predict(boxOfficePrePandemicModel,
                                           boxOfficePrePandemicTesting,
                                           type="response")
# Print predicted data
print(boxOfficePrePandemicModelpredict)

# Anything below or equal to 0.5 as 0, anything above 0.5 as 1.
boxOfficePrePandemicModelpredict<-
  ifelse(boxOfficePrePandemicModelpredict >= 0.5,1,0)

print(boxOfficePrePandemicModelpredict)

# Creating confusion matrix
boxOfficePrePandemicConfusionMatrix <- table(boxOfficePrePandemicTesting$Succes,
                                             boxOfficePrePandemicModelpredict)

print(boxOfficePrePandemicConfusionMatrix)


# False positive rate
boxOfficePrePandemicConfusionMatrix[1,2] /
  (boxOfficePrePandemicConfusionMatrix[1,2]+
     boxOfficePrePandemicConfusionMatrix[1,1] )

# False negative
boxOfficePrePandemicConfusionMatrix[2,1] /
  (boxOfficePrePandemicConfusionMatrix[2,1]+
     boxOfficePrePandemicConfusionMatrix[2,2] )

# Accuracy
sum(diag(boxOfficePrePandemicConfusionMatrix)) /
  nrow(boxOfficePrePandemicTesting)

# Decision Tree R Code
# Preliminary Setup ------------------------------------------------------
# Sets the working directory


# Installs and loads tidyverse library
# install.packages("tidyverse")
library(tidyverse)

# Installs and loads tidyverse library
# install.packages("lubridate")
library(lubridate)

# Installs and loads MASS library
# install.packages("MASS")
library(MASS)

# install.packages("rpart.plot")
library(rpart)
library(rpart.plot)

# Read in CSV ------------------------------------------------------------
# Reads the csv into R with date column
boxOffice <- read.csv(file = "weekendsv2.csv",
                      colClasses = "character",
                      na.strings="?")

# Converts data from character into data
boxOffice$Date <- as.Date(boxOffice$Date, format="%m/%d/%Y")

# Converts data from character into numeric values
boxOffice$Year <- as.numeric(boxOffice$Year)
```

```
boxOffice$Weekend <- as.numeric(boxOffice$Weekend)
boxOffice$Gross <- as.numeric(boxOffice$Gross)

# Converts to a tibble
as_tibble(boxOffice)

# Displays the tibble
print(boxOffice)
str(boxOffice)
summary(boxOffice)

# Exploratory Data Analysis ------------------------------------------------
# Recodes Date into Months
boxOffice <- boxOffice %>%
  mutate(Month = month(ymd(boxOffice$Date)))

boxOffice$Month <- as.numeric(boxOffice$Month)

# Recodes Months into Season
boxOffice <- boxOffice %>%
  mutate(WinterSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SpringSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SummerSeason = 0)

boxOffice <- boxOffice %>%
  mutate(FallSeason = 0)

for (i in 1:length(boxOffice$Month)) {
  if(boxOffice$Month[i] > 11) {
    boxOffice$WinterSeason[i] <- 1
  } else if(boxOffice$Month[i] > 8) {
    boxOffice$FallSeason[i] <- 1
  } else if(boxOffice$Month[i] > 5) {
    boxOffice$SummerSeason[i] <- 1
  } else if(boxOffice$Month[i] > 2) {
    boxOffice$SpringSeason[i] <- 1
  } else {
    boxOffice$WinterSeason[i] <- 1
  }
}

# Factorizes Winter Season
boxOffice$WinterSeason <- as.factor(boxOffice$WinterSeason)

# Factorizes Spring Season
boxOffice$SpringSeason <- as.factor(boxOffice$SpringSeason)

# Factorizes Season
boxOffice$SummerSeason <- as.factor(boxOffice$SummerSeason)

# Factorizes Season
boxOffice$FallSeason <- as.factor(boxOffice$FallSeason)

# Season factorization check
summary(boxOffice)

# Selects movies from 2000-2021
boxOffice2K <- filter(.data = boxOffice, Year > 1999)

# Coding in column of continuous week number since start of 2000
MilleniumWeek <- c(1:length(boxOffice2K$Gross))

boxOffice2K <- cbind(boxOffice2K, MilleniumWeek)

# Week numeric check
summary(boxOffice2K)
```

```
# HolidayWeekend Data for 2000-2020
holidayWeekend <- read.csv(file = "HolidayWeek.csv",
                           colClasses = "character",
                           na.strings="?")

# Converts data from character into data
holidayWeekend$HolidayWeekend <- as.Date(holidayWeekend$HolidayWeekend,
                                         format="%m/%d/%Y")
# Summary indicates a NA output
summary(holidayWeekend)

# NA filled with the appropriate date
holidayWeekend$HolidayWeekend[82] <- as.Date("5/11/2008",
                                             format="%m/%d/%Y")

# Summary indicates a NA output
summary(holidayWeekend)

# Create For loop to generate a logical holiday weekend column
boxOffice2K <- boxOffice2K %>%
  mutate(HolidayWeekend = 0)

for (i in 1:length(boxOffice2K$Date)) {
  for (j in 1:length(holidayWeekend$HolidayWeekend)) {
    if(boxOffice2K$Date[i] == holidayWeekend$HolidayWeekend[j]) {
      boxOffice2K$HolidayWeekend[i] <- 1
    }
  }
}

# Turns the HolidayWeekend into a logical feature
boxOffice2K$HolidayWeekend <- as.factor(boxOffice2K$HolidayWeekend)

# j does not match up with the number of holiday weekends
summary(boxOffice2K$HolidayWeekend)

# Data Subsetting By Year -------------------------------------------------
# Streaming Availability
boxOffice2K <- boxOffice2K %>%
  mutate(Streaming = ifelse(Year > 2007, 1, 0))
boxOffice2K$Streaming <- as.factor(boxOffice2K$Streaming)

# Original Streaming Availability
boxOffice2K <- boxOffice2K %>%
  mutate(OriginalStreaming = ifelse(Year > 2012, 1, 0))
boxOffice2K$OriginalStreaming <- as.factor(boxOffice2K$OriginalStreaming)

# International Streaming Availability
boxOffice2K <- boxOffice2K %>%
  mutate(InternationalStreaming = ifelse(Year > 2017, 1, 0))
boxOffice2K$InternationalStreaming <- as.factor(
  boxOffice2K$InternationalStreaming)

# Pandemic
boxOffice2K <- boxOffice2K %>%
  mutate(Pandemic= ifelse(Year > 2019, 1, 0))
boxOffice2K$Pandemic <- as.factor(boxOffice2K$Pandemic)

# Streaming factor check
summary(boxOffice2K)

# Visualizations ----------------------------------------------------------
# Plot, noticeable dip at year 2020.
plot(boxOffice2K$Gross ~ boxOffice2K$Date)

# Filter out the outlier data
boxOfficePrePandemic <- filter(.data = boxOffice2K, Pandemic == 0)

# Plot data of interest
plot(boxOfficePrePandemic$Gross ~ boxOfficePrePandemic$MilleniumWeek)
```

```
# Histogram
hist(boxOfficePrePandemic$Gross)

# Correlation check, Use MilleniumWeek since most correlated to Gross
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$MilleniumWeek)
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$Year)
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$Weekend)

# Gross is skewed. Need to generate box cox to determine transformation
boxOfficePrePandemicLinear <- lm(data = boxOfficePrePandemic, Gross ~
                                   MilleniumWeek +
                                   WinterSeason +
                                   SpringSeason +
                                   SummerSeason +
                                   HolidayWeekend +
                                   Streaming +
                                   OriginalStreaming +
                                   InternationalStreaming)

grossBoxCox <- boxcox(boxOfficePrePandemicLinear,
                      lambda = seq(-.5,.3,0.1),
                      interp = F)

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(NormalizedGross = Gross^(-0.2))

summary(boxOfficePrePandemic)

hist(boxOfficePrePandemic$NormalizedGross)

cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$MilleniumWeek)
cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$Year)
cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$Weekend)

plot(boxOfficePrePandemic$NormalizedGross ~ boxOfficePrePandemic$MilleniumWeek)
boxOfficeNormalLM <- lm(data = boxOfficePrePandemic, NormalizedGross ~
                          MilleniumWeek +
                          WinterSeason +
                          SpringSeason +
                          SummerSeason +
                          HolidayWeekend +
                          Streaming +
                          OriginalStreaming +
                          InternationalStreaming)
abline(boxOfficeNormalLM,
       col = "red")
summary(boxOfficeNormalLM)

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(SuccessCriteria = fitted(boxOfficeNormalLM))

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(Success = 0)
for (i in 1:length(boxOfficePrePandemic$NormalizedGross)) {
  if(boxOfficePrePandemic$NormalizedGross[i] <
     boxOfficePrePandemic$SuccessCriteria[i]) {
    boxOfficePrePandemic$Success[i] <- 0
  } else{
    boxOfficePrePandemic$Success[i] <- 1
  }
}

boxOfficePrePandemic$Success <- as.factor(boxOfficePrePandemic$Success)

summary(boxOfficePrePandemic)

# Setting random seed
# set.seed(370)

# Randomly splitting the data into training(75) and testing(25)
sampleSet <- sample(nrow(boxOfficePrePandemic),
```

```
                    round(nrow(boxOfficePrePandemic) * .75),
                    replace = FALSE)

# Putting 75% into training
boxOfficePrePandemicTraining <- boxOfficePrePandemic[sampleSet, ]

# Putting 25% into testing
boxOfficePrePandemicTesting <- boxOfficePrePandemic[-sampleSet, ]

# Creating the Decision Tree Model
boxOfficePrePandemicDecisionTreeModel <- rpart(formula = Success ~
                                                 MilleniumWeek +
                                                 WinterSeason +
                                                 SpringSeason +
                                                 SummerSeason +
                                                 HolidayWeekend +
                                                 Streaming +
                                                 OriginalStreaming +
                                                 InternationalStreaming,
                                               method = "class",
                                               cp = .01,
                                               data = boxOfficePrePandemicTraining)

# Displaying the decision tree visualization
rpart.plot(boxOfficePrePandemicDecisionTreeModel)

# Calculating the prediction
boxOfficePrePandemicPrediction <- predict(boxOfficePrePandemicDecisionTreeModel,
                                          boxOfficePrePandemicTesting,
                                          type = "class")

# Displaying the prediction in the console
print(boxOfficePrePandemicPrediction)

# Creating a confusion matrix
boxOfficePrePandemicConfusionMatrix <- table(
  boxOfficePrePandemicTesting$Success,
  boxOfficePrePandemicPrediction)

# Displaying the confusion matrix
print(boxOfficePrePandemicConfusionMatrix)

# Calculating the predictive accuracy
predictiveAccuracy <- sum(diag(boxOfficePrePandemicConfusionMatrix)) /
  nrow(boxOfficePrePandemicTesting)

# Displaying predictive accuracy in the console
print(predictiveAccuracy)

# Neural Network R Code
# Preliminary Setup ------------------------------------------------------
# Sets the working directory


# Installs and loads tidyverse library
# install.packages("tidyverse")
library(tidyverse)

# Installs and loads tidyverse library
# install.packages("lubridate")
library(lubridate)

# Installs and loads MASS library
# install.packages("MASS")
library(MASS)

# install.packages("neuralnet")
library(neuralnet)
# Read in CSV ------------------------------------------------------------
# Reads the csv into R with date column
boxOffice <- read.csv(file = "weekendsv2.csv",
```

```
                      colClasses = "character",
                      na.strings="?")

# Converts data from character into data
boxOffice$Date <- as.Date(boxOffice$Date, format="%m/%d/%Y")

# Converts data from character into numeric values
boxOffice$Year <- as.numeric(boxOffice$Year)
boxOffice$Weekend <- as.numeric(boxOffice$Weekend)
boxOffice$Gross <- as.numeric(boxOffice$Gross)

# Converts to a tibble
as_tibble(boxOffice)

# Displays the tibble
print(boxOffice)
str(boxOffice)
summary(boxOffice)

# Explanatory Data Analysis ------------------------------------------------
# Recodes Date into Months
boxOffice <- boxOffice %>%
  mutate(Month = month(ymd(boxOffice$Date)))

boxOffice$Month <- as.numeric(boxOffice$Month)

# Recodes Months into Season
boxOffice <- boxOffice %>%
  mutate(WinterSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SpringSeason = 0)

boxOffice <- boxOffice %>%
  mutate(SummerSeason = 0)

boxOffice <- boxOffice %>%
  mutate(FallSeason = 0)

for (i in 1:length(boxOffice$Month)) {
  if(boxOffice$Month[i] > 11) {
    boxOffice$WinterSeason[i] <- 1
  } else if(boxOffice$Month[i] > 8) {
    boxOffice$FallSeason[i] <- 1
  } else if(boxOffice$Month[i] > 5) {
    boxOffice$SummerSeason[i] <- 1
  } else if(boxOffice$Month[i] > 2) {
    boxOffice$SpringSeason[i] <- 1
  } else {
    boxOffice$WinterSeason[i] <- 1
  }
}

# Factorizes Winter Season
boxOffice$WinterSeason <- as.factor(boxOffice$WinterSeason)

# Factorizes Spring Season
boxOffice$SpringSeason <- as.factor(boxOffice$SpringSeason)

# Factorizes Season
boxOffice$SummerSeason <- as.factor(boxOffice$SummerSeason)

# Factorizes Season
boxOffice$FallSeason <- as.factor(boxOffice$FallSeason)

# Season factorization check
summary(boxOffice)

# Selects movies from 2000-2021
boxOffice2K <- filter(.data = boxOffice, Year > 1999)
```

```
# Coding in column of continuous week number since start of 2000
MilleniumWeek <- c(1:length(boxOffice2K$Gross))

boxOffice2K <- cbind(boxOffice2K, MilleniumWeek)

# Week numeric check
summary(boxOffice2K)

# HolidayWeekend Data for 2000-2020
holidayWeekend <- read.csv(file = "HolidayWeek.csv",
                           colClasses = "character",
                           na.strings="?")

# Converts data from character into data
holidayWeekend$HolidayWeekend <- as.Date(holidayWeekend$HolidayWeekend,
                                         format="%m/%d/%Y")
# Summary indicates a NA output
summary(holidayWeekend)

# NA filled with the appropriate date
holidayWeekend$HolidayWeekend[82] <- as.Date("5/11/2008",
                                             format="%m/%d/%Y")

# Summary indicates a NA output
summary(holidayWeekend)

# Create For loop to generate a logical holiday weekend column
boxOffice2K <- boxOffice2K %>%
  mutate(HolidayWeekend = 0)

for (i in 1:length(boxOffice2K$Date)) {
  for (j in 1:length(holidayWeekend$HolidayWeekend)) {
    if(boxOffice2K$Date[i] == holidayWeekend$HolidayWeekend[j]) {
      boxOffice2K$HolidayWeekend[i] <- 1
    }
  }
}

# Turns the HolidayWeekend into a logical feature
boxOffice2K$HolidayWeekend <- as.factor(boxOffice2K$HolidayWeekend)

# j does not match up with the number of holiday weekends
summary(boxOffice2K$HolidayWeekend)

# Data Subsetting By Year ---------------------------------------------------
# Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(Streaming = ifelse(Year > 2007, 1, 0))
boxOffice2K$Streaming <- as.factor(boxOffice2K$Streaming)

# Original Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(OriginalStreaming = ifelse(Year > 2012, 1, 0))
boxOffice2K$OriginalStreaming <- as.factor(boxOffice2K$OriginalStreaming)

# International Streaming Availablility
boxOffice2K <- boxOffice2K %>%
  mutate(InternationalStreaming = ifelse(Year > 2017, 1, 0))
boxOffice2K$InternationalStreaming <- as.factor(
  boxOffice2K$InternationalStreaming)

# Pandemic
boxOffice2K <- boxOffice2K %>%
  mutate(Pandemic= ifelse(Year > 2019, 1, 0))
boxOffice2K$Pandemic <- as.factor(boxOffice2K$Pandemic)

# Streaming factor check
summary(boxOffice2K)

# Visualizations ------------------------------------------------------------
# Plot, noticeable dip at year 2020.
```

```
plot(boxOffice2K$Gross ~ boxOffice2K$Date)

# Filter out the outlier data
boxOfficePrePandemic <- filter(.data = boxOffice2K, Pandemic == 0)

# Plot data of interest
plot(boxOfficePrePandemic$Gross ~ boxOfficePrePandemic$MilleniumWeek)

# Histogram
hist(boxOfficePrePandemic$Gross)

# Correlation check, Use MilleniumWeek since most correlated to Gross
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$MilleniumWeek)
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$Year)
cor(boxOfficePrePandemic$Gross, boxOfficePrePandemic$Weekend)

# Gross is skewed. Need to generate box cox to determine transformation
boxOfficePrePandemicLinear <- lm(data = boxOfficePrePandemic, Gross ~
                                   MilleniumWeek +
                                   WinterSeason +
                                   SpringSeason +
                                   SummerSeason +
                                   HolidayWeekend +
                                   Streaming +
                                   OriginalStreaming +
                                   InternationalStreaming)

grossBoxCox <- boxcox(boxOfficePrePandemicLinear,
                      lambda = seq(-.5,.3,0.1),
                      interp = F)

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(NormalizedGross = Gross^(-0.2))

summary(boxOfficePrePandemic)

hist(boxOfficePrePandemic$NormalizedGross)

cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$MilleniumWeek)
cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$Year)
cor(boxOfficePrePandemic$NormalizedGross, boxOfficePrePandemic$Weekend)

plot(boxOfficePrePandemic$NormalizedGross ~ boxOfficePrePandemic$MilleniumWeek)
boxOfficeNormalLM <- lm(data = boxOfficePrePandemic, NormalizedGross ~
                          MilleniumWeek +
                          WinterSeason +
                          SpringSeason +
                          SummerSeason +
                          HolidayWeekend +
                          Streaming +
                          OriginalStreaming +
                          InternationalStreaming)
abline(boxOfficeNormalLM,
       col = "red")
summary(boxOfficeNormalLM)

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(SuccessCriteria = fitted(boxOfficeNormalLM))

boxOfficePrePandemic <- boxOfficePrePandemic %>%
  mutate(Success = 0)
for (i in 1:length(boxOfficePrePandemic$NormalizedGross)) {
  if(boxOfficePrePandemic$NormalizedGross[i] <
     boxOfficePrePandemic$SuccessCriteria[i]) {
    boxOfficePrePandemic$Success[i] <- 0
  } else{
    boxOfficePrePandemic$Success[i] <- 1
  }
}

# Turning Success into a logical
```

```
boxOfficePrePandemic$Success <- as.logical(boxOfficePrePandemic$Success)
# Turning WinterSeason into numeric
boxOfficePrePandemic$WinterSeason <- as.numeric(
  boxOfficePrePandemic$WinterSeason) - 1
# Turning SpringSeason into numeric
boxOfficePrePandemic$SpringSeason <- as.numeric(
  boxOfficePrePandemic$SpringSeason) - 1
# Turning SummerSeason into numeric
boxOfficePrePandemic$SummerSeason <- as.numeric(
  boxOfficePrePandemic$SummerSeason) - 1
# Turning FallSeason into numeric
boxOfficePrePandemic$FallSeason <- as.numeric(
  boxOfficePrePandemic$FallSeason) - 1
# Turning HolidayWeekend into numeric
boxOfficePrePandemic$HolidayWeekend <- as.numeric(
  boxOfficePrePandemic$HolidayWeekend) -1
# Turning Streaming into numeric
boxOfficePrePandemic$Streaming <- as.numeric(
  boxOfficePrePandemic$Streaming) - 1
# Turning OriginalStreaming into numeric
boxOfficePrePandemic$OriginalStreaming <- as.numeric(
  boxOfficePrePandemic$OriginalStreaming) - 1
# Turning InternationalStreaming into numeric
boxOfficePrePandemic$InternationalStreaming <- as.numeric(
  boxOfficePrePandemic$InternationalStreaming) -1

# Summary of the boxOfficePrePandemic
summary(boxOfficePrePandemic)

# Setting Random Seed
# set.seed(370)

# Randomly splitting the data into training(75) and testing(25)
sampleSet <- sample(nrow(boxOfficePrePandemic),
                    round(nrow(boxOfficePrePandemic) * .75),
                    replace = FALSE)

# Putting 75% into training
boxOfficePrePandemicTraining <- boxOfficePrePandemic[sampleSet, ]

# Putting 25% into testing
boxOfficePrePandemicTesting <- boxOfficePrePandemic[-sampleSet, ]

# Generating the neural network
boxOfficePrePandemicNeuralNet <- neuralnet(
  formula = Success ~ WinterSeason +
    SpringSeason +
    SummerSeason +
    FallSeason +
    HolidayWeekend +
    Streaming +
    OriginalStreaming +
    InternationalStreaming,
  data = boxOfficePrePandemicTraining,
  hidden = 9,
  act.fct = "logistic",
  linear.output = FALSE)

# Displaying Neural Network numeric results
print(boxOfficePrePandemicNeuralNet$result.matrix)

# Visualizing the neural network
plot(boxOfficePrePandemicNeuralNet)

# Generating the probabilities
boxOfficePrePandemicProbability <- compute(boxOfficePrePandemicNeuralNet,
                                           boxOfficePrePandemicTesting)

# Displaying the probabilities
print(boxOfficePrePandemicProbability$net.result)
```

```
# Converting the probabilities to 0 & 1
boxOfficePrePandemicPrediction <-
  ifelse(boxOfficePrePandemicProbability$net.result > 0.5, 1, 0)

# displaying fishingCharterPrediction
print(boxOfficePrePandemicPrediction)

# Creating the confusion matrix
boxOfficePrePandemicConfusionMatrix <- table(
  boxOfficePrePandemicTesting$Success,boxOfficePrePandemicPrediction)

# Printing the confusion matrix
print(boxOfficePrePandemicConfusionMatrix)

# Calculating predictive accuracy
predictiveAccuracy <- sum(diag(boxOfficePrePandemicConfusionMatrix)) /
  nrow(boxOfficePrePandemicTesting)

# Printing the predictive accuracy
print(predictiveAccuracy)
```

Appendix-Visualizations

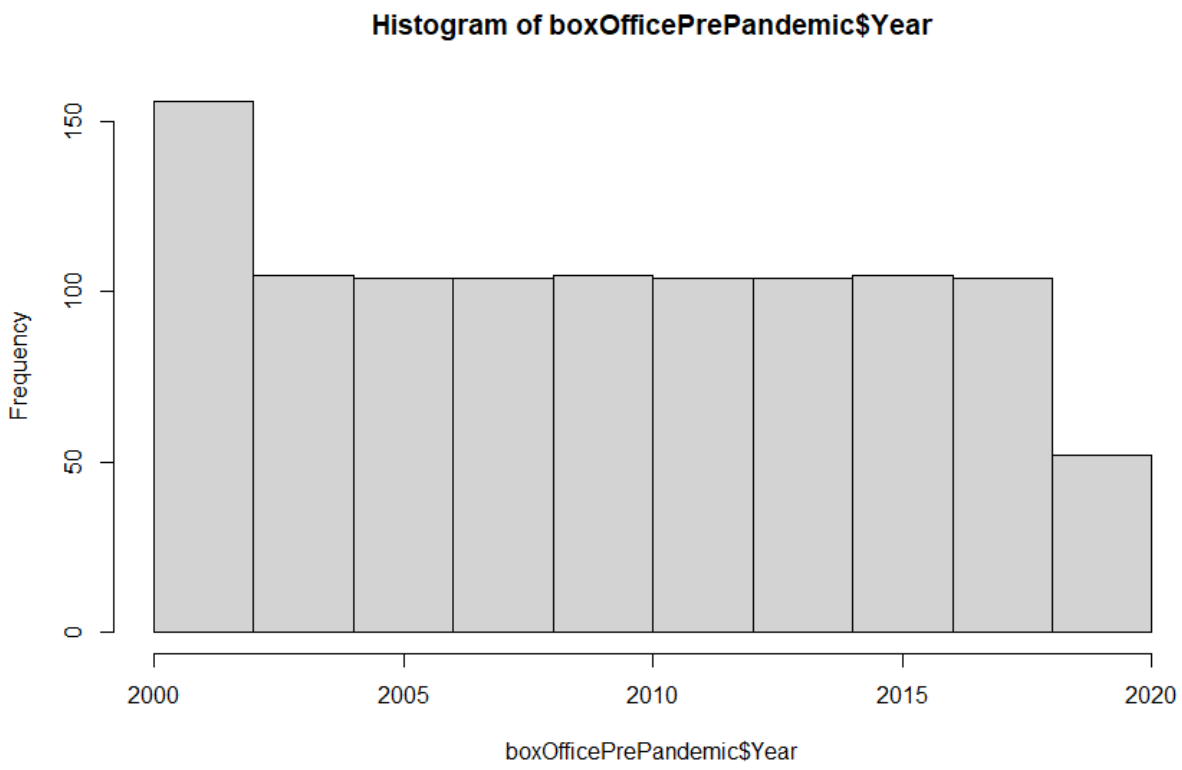

Figure 1: Histogram of the frequency of box office weekends by Year

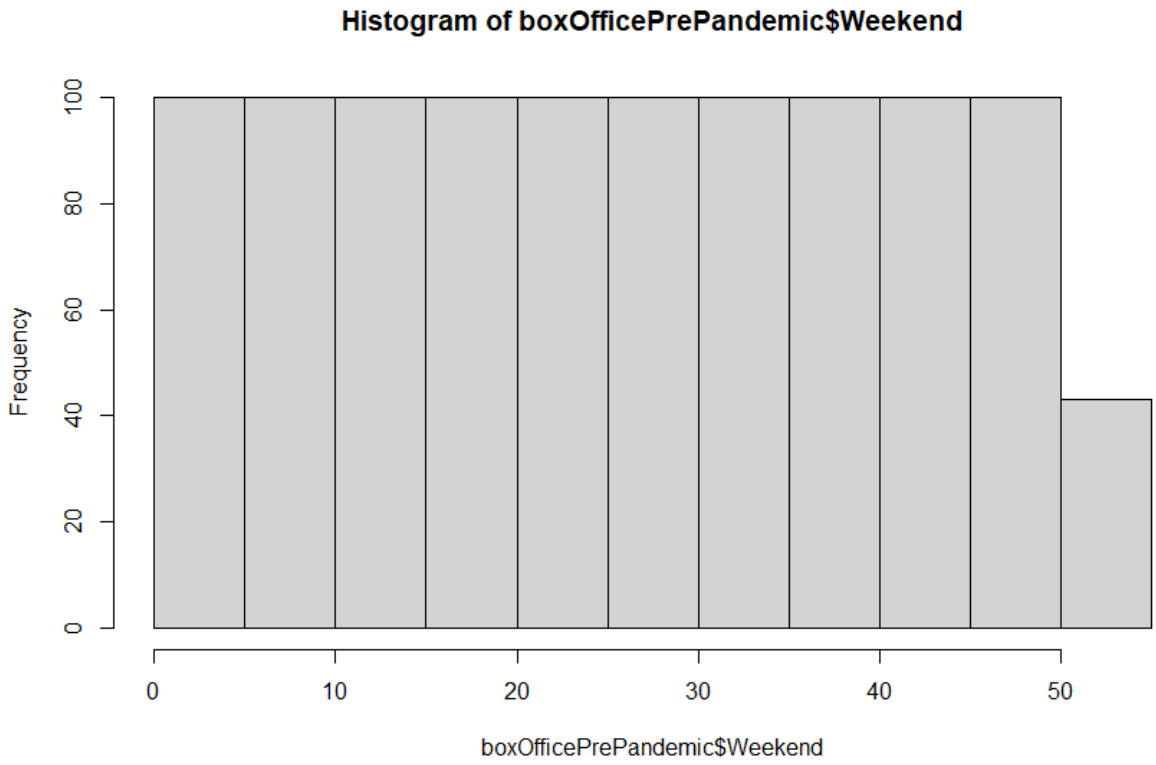

Figure 2: Histogram of the frequency of box office weekends by Weekend

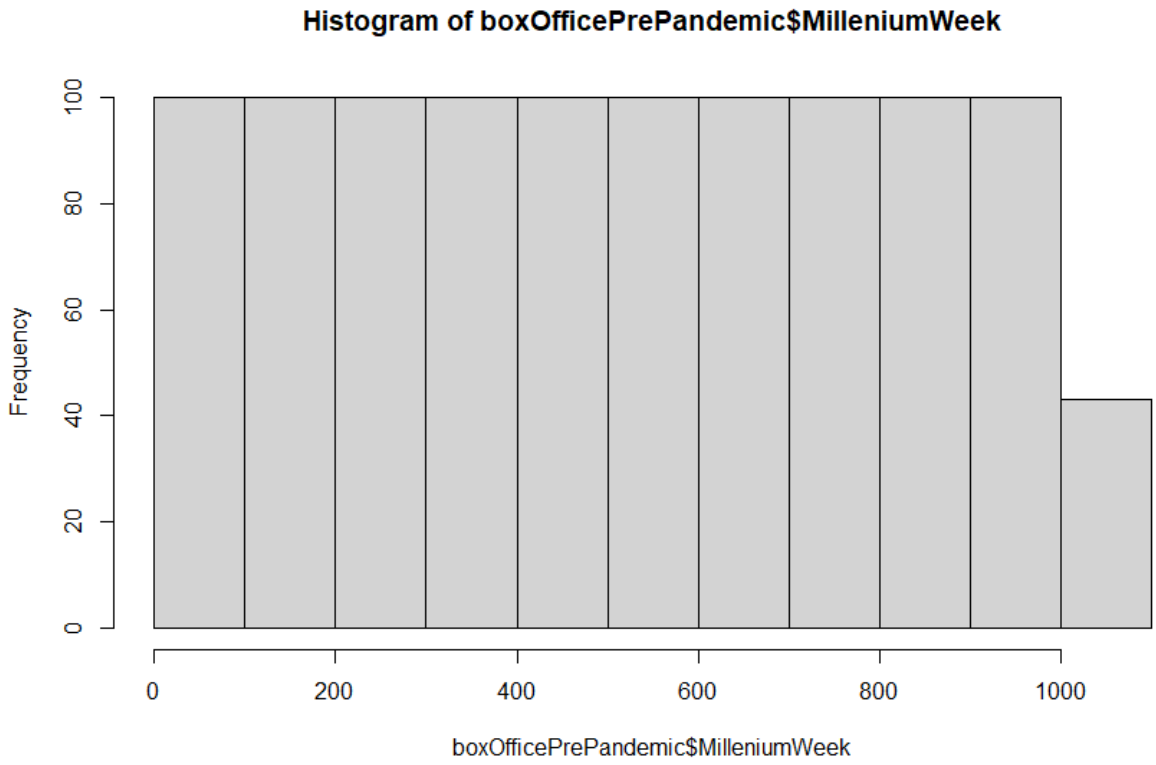

Figure 3: Histogram of MilleniumWeek

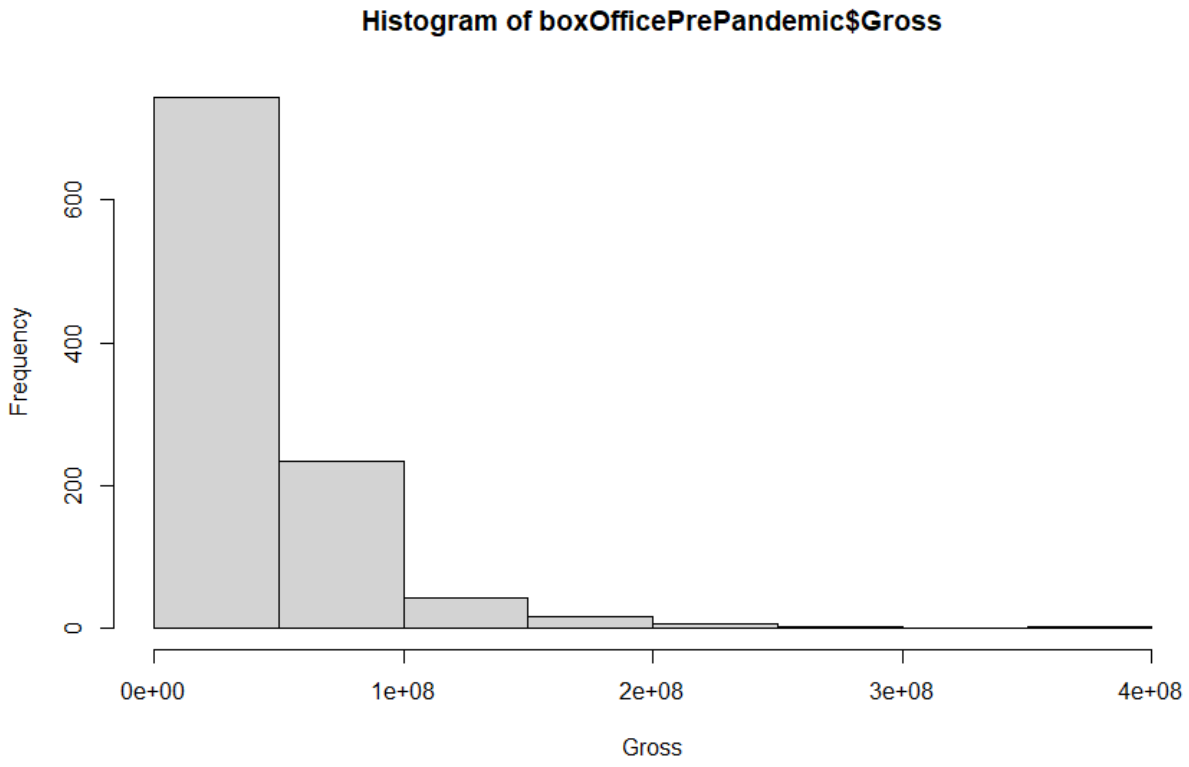

Figure 4: Histogram of Gross between the years 2000 and 2019

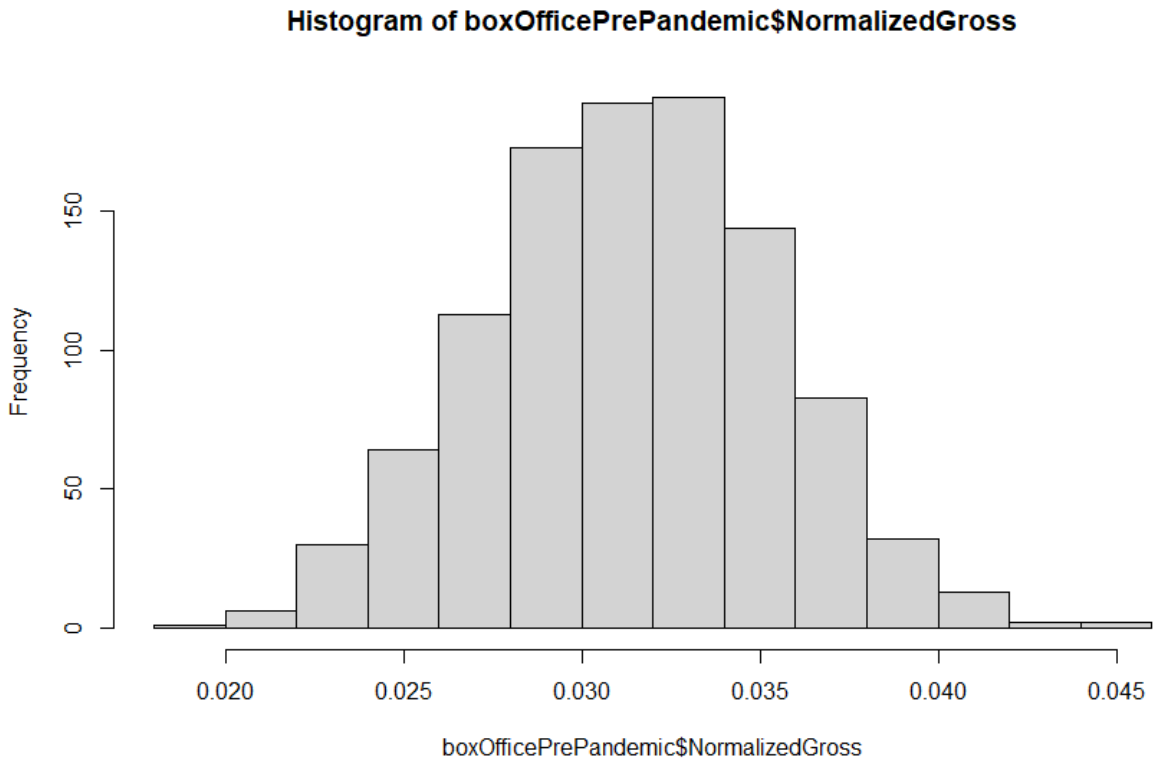

Figure 5: Histogram of NormalizedGross between the years 2000 and 2019

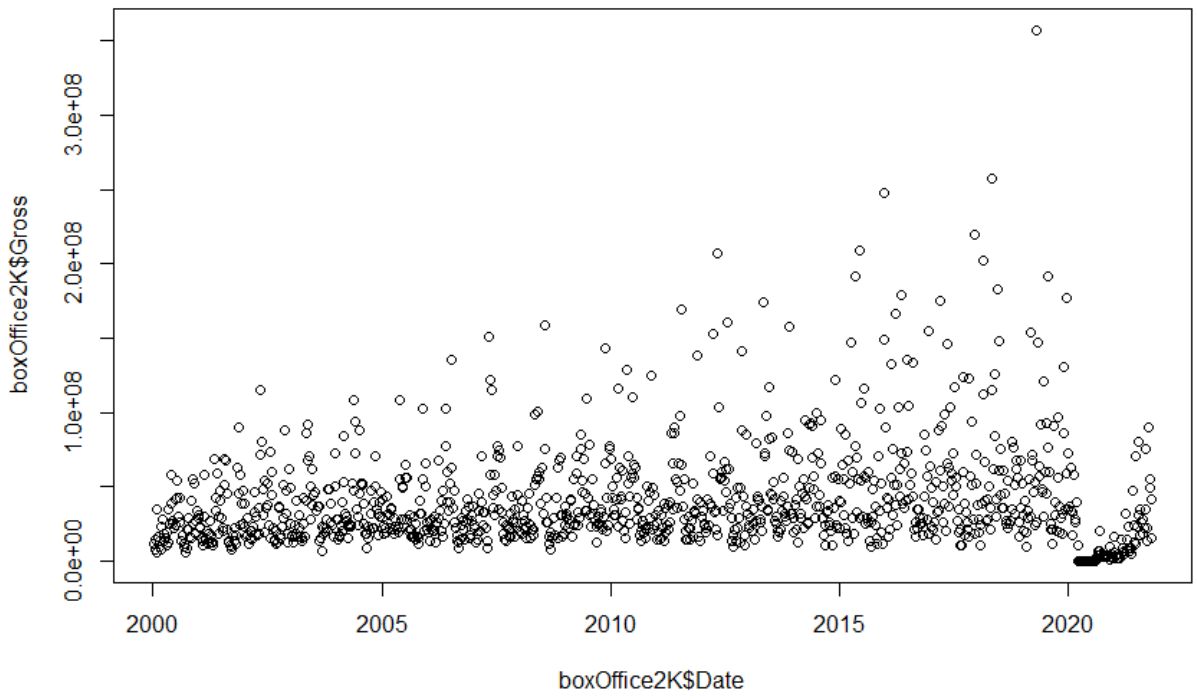

Figure 6: Scatterplot of gross against date. Year 2020 and afterwards appear to be outliers

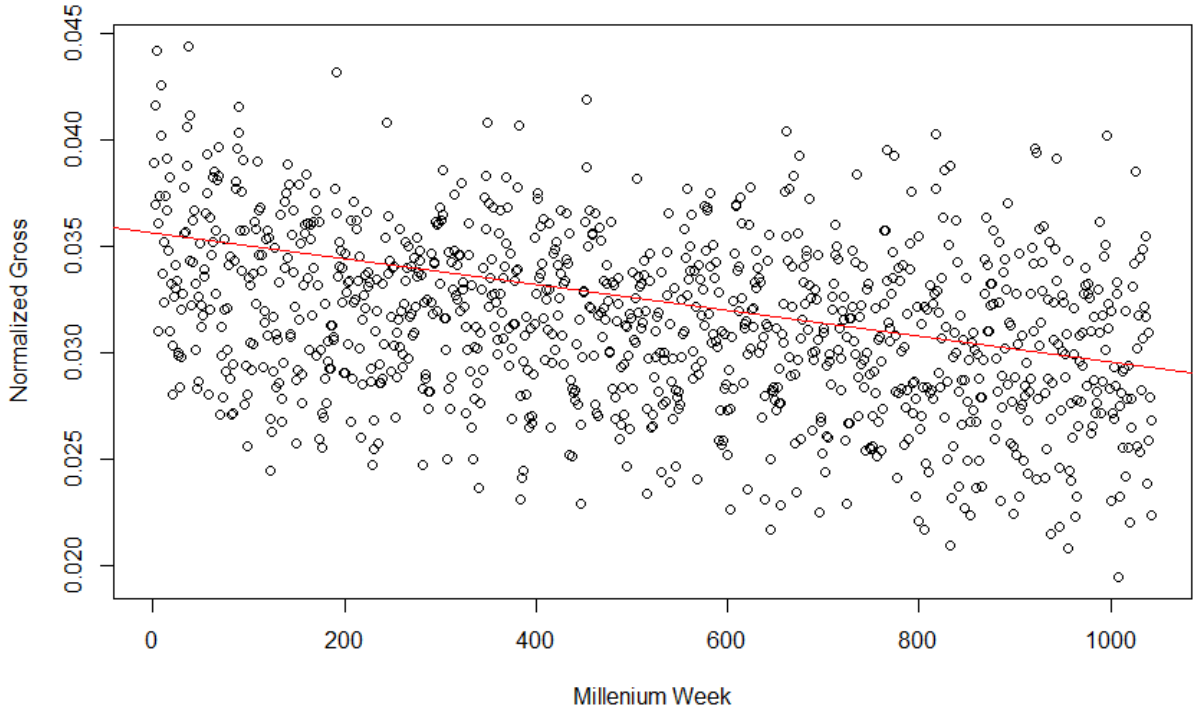

Figure 7: Scatterplot of the normalized gross against millennium week with fitted regression line.

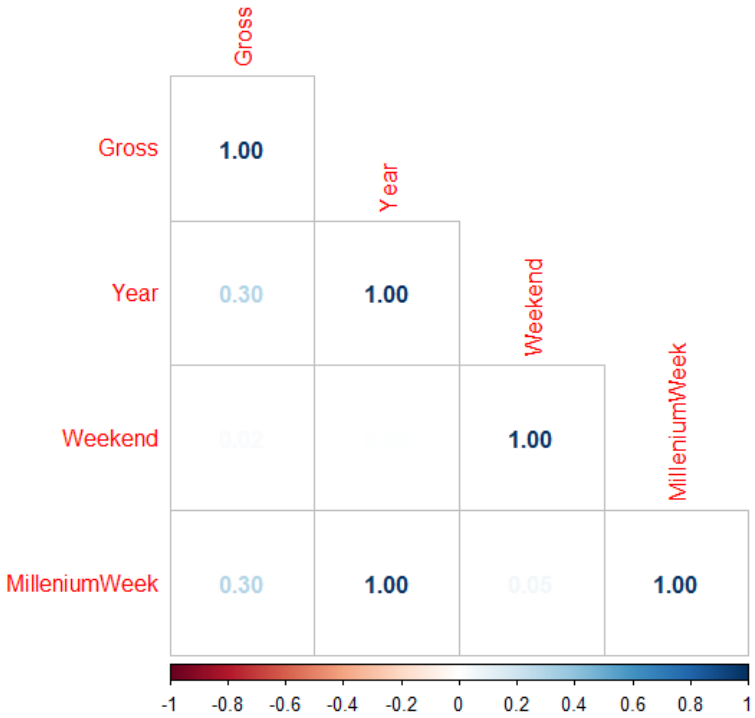

Figure 8: Correlation plot of continuous variables with Gross

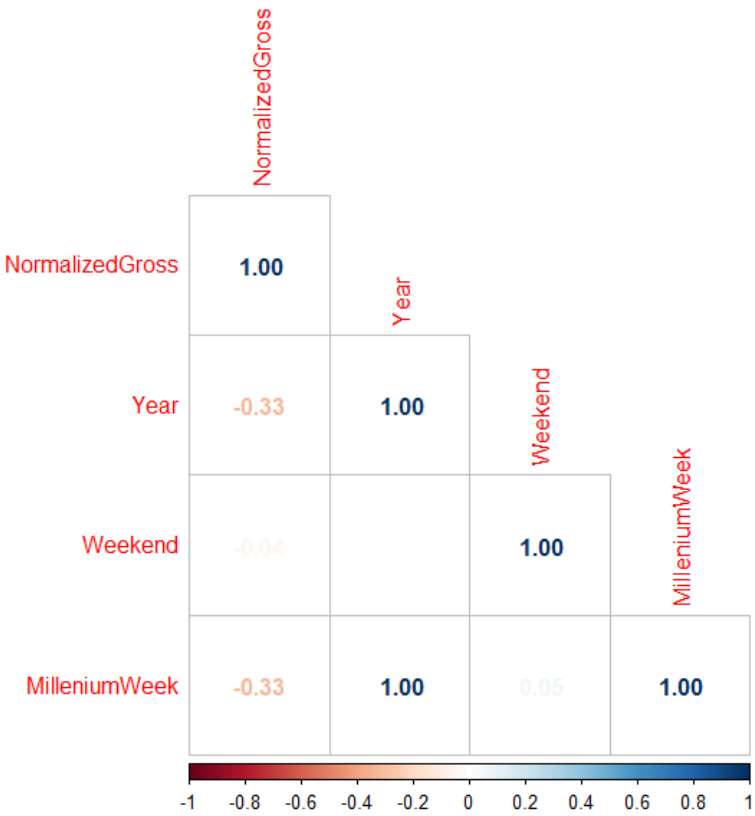

Figure 9: Correlation plot of continuous variables with NormalizedGross

Appendix-Model Results

- Model results screenshots from R for each of the 5 models

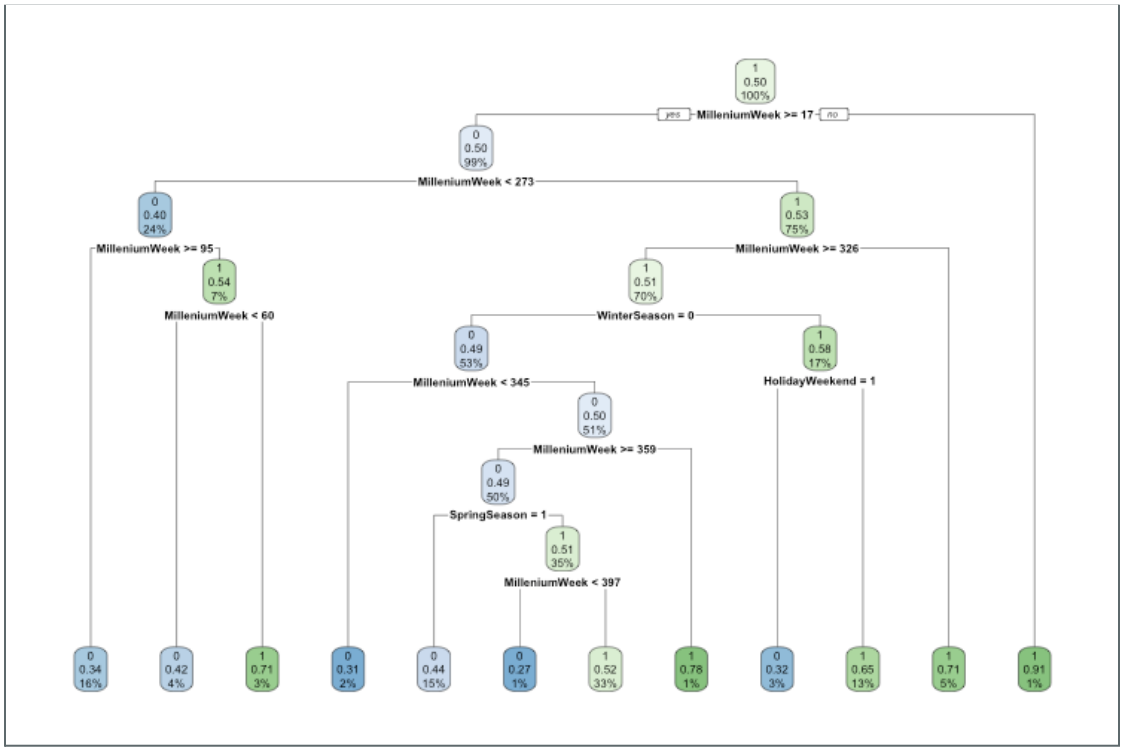

Figure 10: Decision Tree plot

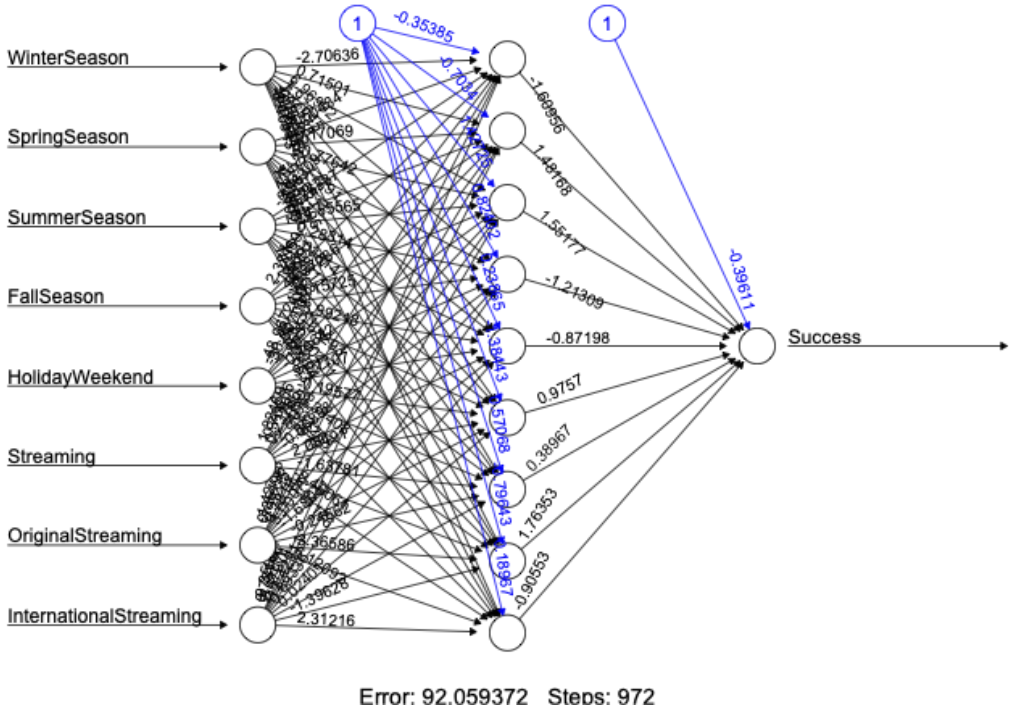

Figure 11: Neural Network plot

Variable	Type	Description
Film	Character	Name of the top grossing weekend box office
Date	Date	Sunday of the weekend box office
Year	Numeric	Year for the weekend box office (1960-2021)
Weekend	Numeric	Week number of the year (1:53)
Gross	Numeric	Box office earnings of the top film for the weekend

Table 1: Initial Box Office Features

Variable	Type	Description
Month	Numeric	Month of the weekend box office (1:12)
WinterSeason	Logical	1 = box office in the winter 0 = box office did not occur during the winter
SpringSeason	Logical	1 = box office in the spring 0 = box office did not occur during the spring
SummerSeason	Logical	1 = box office in the summer 0 = box office did not occur during the summer
FallSeason	Logical	1 = box office in the fall 0 = box office did not occur during the fall
MilleniumWeek	Numeric	Week number into the millenium
HolidayWeekend	Logical	1 = box office was during the holiday weekend 0 = box office did not occur during the holiday weekend
Streaming	Logical	1 = streaming services were available (2008) 0 = streaming services were not available
OriginalStreaming	Logical	1 = original streaming content was available (2013) 0 = original streaming content was not available
InternationalStreaming	Logical	1 = international streaming content was available (2018) 0 = international streaming content was not available
NormalizedGross	Numeric	Gross normalized with a Box Cox Transformation

Success	Logical	1 = box office generated sufficient earnings to be a success 0 = box office generated to little to be a success
Fourseason	character	Season of the weekend box office (Winter, Spring, Summer, and Fall)

Table 2: Generated Box Office Features

Naive Bayes Confusion Matrices

	0	1
0	72	48
1	76	65

Table 3: Naive Bayes Confusion Matrix using all variables

	0	1
0	104	16
1	117	24

Table 4: Naive Bayes Confusion Matrix using FallSeason, HolidayWeekend, Streaming

	0	1
0	104	16
1	117	24

Table 5: Naive Bayes Confusion Matrix using WinterSeason, HolidayWeekend, Streaming

	0	1
0	104	16
1	117	24

Table 6: Naive Bayes Confusion Matrix using SpringSeason, HolidayWeekend, Streaming

	0	1
0	78	42
1	87	54

Table 7: Naive Bayes Confusion Matrix using SummerSeason, HolidayWeekend, Streaming

	0	1
0	104	16
1	117	24

Table 8: Naive Bayes Confusion Matrix using FallSeason, HolidayWeekend, OriginalStreaming

	0	1
0	104	16
1	117	24

Table 9: Naive Bayes Confusion Matrix using WinterSeason, HolidayWeekend, OriginalStreaming

	0	1
0	104	16
1	117	24

Table 10: Naive Bayes Confusion Matrix using SpringSeason, HolidayWeekend, OriginalStreaming

	0	1
0	78	42
1	87	54

Table 11: Naive Bayes Confusion Matrix using SummerSeason, HolidayWeekend, OriginalStreaming

	0	1
0	103	17
1	111	30

Table 12: Naive Bayes Confusion Matrix using FallSeason, HolidayWeekend, InternationalStreaming

	0	1
0	92	28
1	108	33

Table 13: Naive Bayes Confusion Matrix using WinterSeason, HolidayWeekend, InternationalStreaming

	0	1

0	93	27
1	106	35

Table 14: Naive Bayes Confusion Matrix using SpringSeason, HolidayWeekend, InternationalStreaming

	0	1
0	78	42
1	87	54

Table 14: Naive Bayes Confusion Matrix using SummerSeason, HolidayWeekend, InternationalStreaming

	0	1
0	59	73
1	66	63

Table 15: Decision Tree Confusion Matrix

	0	1
0	63	69
1	49	80

Table 16: Neural Network Confusion Matrix

	Fall	Spring	Summer	Winter
Fall	17	3	17	27
Spring	7	16	16	27
Summer	8	13	24	18
Winter	15	11	18	24

Table 17: K-Nearest Neighbor Confusion Matrix

Appendix-Other Relevant Visuals

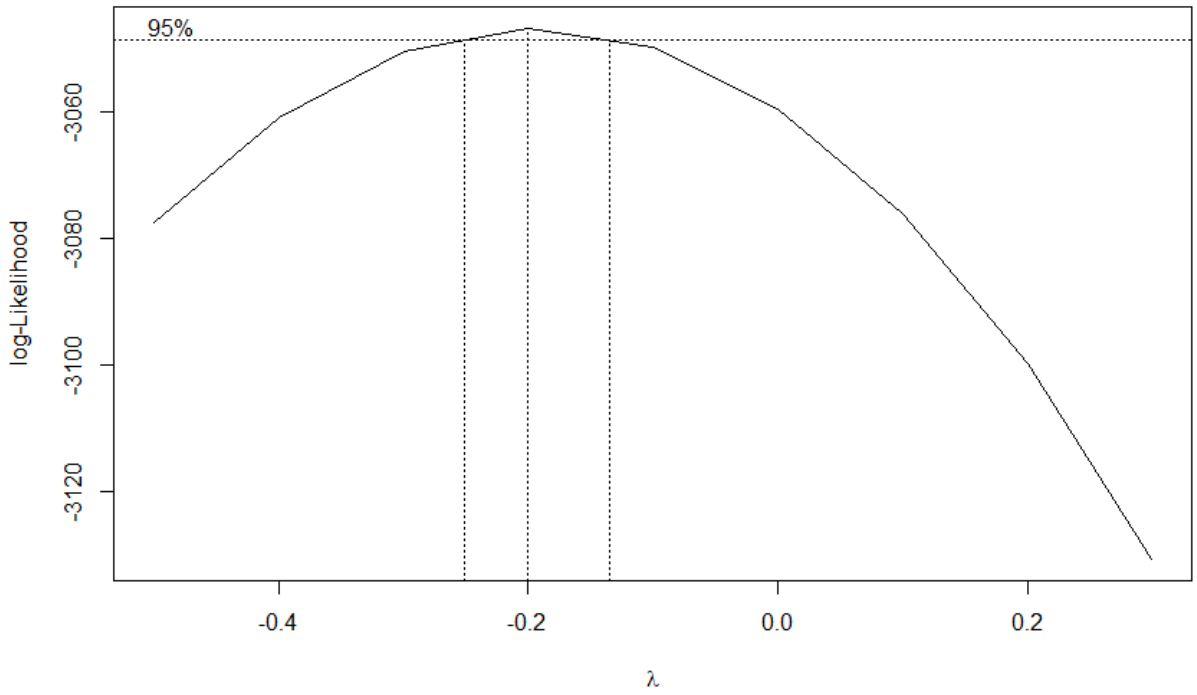

Figure 12: Box Cox transformation of Gross, -0.2 is suggested transformation for normalizing Gross

Variable Included	Predictive Accuracy
FallSeason, HolidayWeekend, Streaming	55.10%
WinterSeason, HolidayWeekend, Streaming	55.10%
SpringSeason, HolidayWeekend, Streaming	55.10%
SummerSeason, HolidayWeekend, Streaming	57.40%
FallSeason, HolidayWeekend, OriginalStreaming	55.10%
WinterSeason, HolidayWeekend, OriginalStreaming	53.60%
SpringSeason, HolidayWeekend, OriginalStreaming	55.10%
SummerSeason, HolidayWeekend, OriginalStreaming	57.40%
FallSeason, HolidayWeekend, InternationalStreaming	55.10%
WinterSeason, HolidayWeekend, InternationalStreaming	55.10%
SpringSeason, HolidayWeekend, InternationalStreaming	55.10%
SummerSeason, HolidayWeekend, InternationalStreaming	57.40%
WinterSeason,SpringSeason,SummerSeason,FallSeason,Year,HolidayWeekend,Streaming,OriginalStreaming,InternationalStreaming	55.55%
WinterSeason,SpringSeason,SummerSeason,FallSeason,HolidayWeekend,Streaming,OriginalStreaming,InternationalStreaming	58.20%
Year,HolidayWeekend,Streaming,OriginalStreaming, InternationalStreaming	55.93%
Year,Streaming,OriginalStreaming,InternationalStreaming	49.80%
WinterSeason, SpringSeason, SummerSeason, FallSeason, HolidayWeekend, Year	57.47%
SummerSeason, HolidayWeekend, Year	57.47%

Table 18: Logistic Regression Predictive Accuracy

Variables Included	Predictive Accuracy
FallSeason, HolidayWeekend, Streaming	49.04%
WinterSeason, HolidayWeekend, Streaming	49.04%

SpringSeason, HolidayWeekend, Streaming	49.04%
SummerSeason, HolidayWeekend, Streaming	50.57%
FallSeason, HolidayWeekend, OriginalStreaming	49.04%
WinterSeason, HolidayWeekend, OriginalStreaming	49.04%
SpringSeason, HolidayWeekend, OriginalStreaming	49.04%
SummerSeason, HolidayWeekend, OriginalStreaming	50.57%
FallSeason, HolidayWeekend, InternationalStreaming	50.96%
WinterSeason, HolidayWeekend, InternationalStreaming	47.89%
SpringSeason, HolidayWeekend, InternationalStreaming	49.04%
SummerSeason, HolidayWeekend, InternationalStreaming	50.57%

Table 19: Naive Bayes Predictive Accuracy

K-value	Predictive Accuracy
153	31.80%
171	31.80%
193	31.80%
219	31.80%
253	31.80%
323	31.80%
405	31.80%
467	31.80%
473	31.80%
621	31.80%
665	31.80%

Table 20: The best value of k for the K-Nearest Neighbor