```r
install.packages("tidyverse")
install.packages("ggplot2")
library(dplyr)
library(caret)
library(ggplot2)
library(tidyverse)

library(readr)
NetflixOriginals <- read_csv("NetflixOriginals.csv")
View(NetflixOriginals)

head(NetflixOriginals)
tail(NetflixOriginals)
dim(NetflixOriginals)
any(is.na(NetflixOriginals))

str(NetflixOriginals)
colnames(NetflixOriginals)
sapply(NetflixOriginals,class)

summary(NetflixOriginals)

is.na(NetflixOriginals)
sapply(NetflixOriginals, function(x) length(unique(x)))
unique(NetflixOriginals$Genre)
summary(is.na(NetflixOriginals))
colSums(is.na(NetflixOriginals))
min(NetflixOriginals$`IMDB Score`)
max(NetflixOriginals$`IMDB Score`)
sd(NetflixOriginals$`IMDB Score`)
# Now you can use the %>% operator
NetflixOriginals <- NetflixOriginals %>%
  mutate(Released = mdy(Premiere))

NetflixOriginals <- NetflixOriginals %>%
  mutate(Year = year(Released)) %>%
  mutate(Month = month(Released, label=TRUE)) %>%
  mutate(Date = day(Released)) %>%
  mutate(Day = wday(Released, label=TRUE, abbr=FALSE))


# Load required libraries
library(dplyr)
library(ggplot2)

# Create a bar chart for the number of movies per year
n_year <- NetflixOriginals %>% group_by(Year) %>% summarise(total=n())
n_year_graph <- ggplot(data=n_year, aes(x=Year, y=total)) +
  geom_bar(stat="identity", fill="yellow") +
  labs(title="Number of Netflix Movies Released Each Year",
       x="Year", y="Number of Movies") +
  theme_minimal()

n_year_graph

# Create a histogram for movie duration distribution


# Highest Rated Movies

n <- NetflixOriginals %>% arrange(desc(`IMDB Score`)) %>% head(5)

n_graph <- ggplot(data=n)+
  geom_col(mapping=aes(
    x=reorder(`Title`,`IMDB Score`),
    y=`IMDB Score`,
    fill=ifelse(
      `IMDB Score`==max(`IMDB Score`),
      "red","black")))+
  labs(title="Highest Rated Movies")+
  theme_minimal()+
  scale_fill_manual(values = c("#2d2d2d","#E50914"))+
  coord_flip()+
  theme(
    legend.position="none",
    plot.title = element_text(
      family="Bebas Neue",
      size=25,
      color="#E50914"),
    axis.title.x=element_blank(),
    axis.title.y=element_blank(),
    panel.grid.major.x=element_blank()
  )

n_graph
#Runtime vs IMDB-Score

n_graph <- ggplot(data=NetflixOriginals,aes(x = `IMDB Score`, y = Runtime))+
  geom_point()+
  geom_smooth(method = "lm", color="#E50914")+
  labs(title="Runtime vs IMDB Rating")+
  theme_minimal()+
  scale_fill_manual(values=c("#2d2d2d","#E50914"))+
  theme(
    legend.position = "none",
    plot.title=element_text(
      family="Bebas Neue",
      size=25,
      color="#E50914"),
    axis.title.x=element_blank(),
    axis.title.y=element_blank(),
    panel.grid.major.x=element_blank()
  )

n_graph
library(ggplot2)

#Most popular Genres
n <- NetflixOriginals%>% group_by(Genre) %>%
  summarise(Movies=n()) %>%
  arrange(desc(Movies)) %>%
  head(5)

n_graph <-
  ggplot(data=n)+
  geom_col(mapping = aes(
    x=reorder(Genre, -Movies),
    y=Movies,
    fill=ifelse(Movies == max(Movies),"red","black")))+
  labs(title="Most Popular Genres")+
  theme_minimal()+
  scale_fill_manual(values = c("#2d2d2d","#E50914"))+
  theme(
    legend.position="none",
    plot.title = element_text(
      family="Bebas Neue",
      size=25,
      color="#E50914"),
    axis.title.x=element_blank(),
    axis.title.y=element_blank(),
    panel.grid.major.x=element_blank(),
    panel.grid.minor = element_blank(),
    text = element_text(size=20)
  )

n_graph
```

```
# Load the necessary libraries
library(dplyr)
library(tidyr)

# Load your Netflix dataset (replace 'NetflixOriginals' with your actual dataset)
# Assuming your dataset includes columns like 'Year', 'Duration', 'Genre', and 'IMDb'

# Data cleaning and preprocessing
NetflixData <- NetflixOriginals %>%
  select(Year, Genre, `IMDB Score`) %>%
  na.omit()  # Remove rows with missing values
# Perform one-hot encoding for Genre
NetflixData <- NetflixData %>%
  separate_rows(Genre, sep=", ") %>%
  pivot_wider(names_from = Genre, values_from = Genre, values_fn = length, values_fill =
0)
# Load the necessary library for splitting
library(caret)
install.packages("caret")

# Set a seed for reproducibility
set.seed(123)

# Split the data into training (70%) and testing (30%) sets
splitIndex <- createDataPartition(NetflixData$`IMDB Score`, p = 0.7, list = FALSE)
trainData <- NetflixData[splitIndex, ]
testData <- NetflixData[-splitIndex, ]

# Load the necessary library for modeling
library(lmtest)
install.packages("lmtest")


# Build a linear regression model
model <- lm(`IMDB Score` ~ ., data = trainData)

# Make predictions on the test dataset
predictions <- predict(model, newdata = testData)

# Calculate RMSE
rmse <- sqrt(mean((testData$`IMDB Score` - predictions)^2))
cat("Root Mean Squared Error (RMSE):", rmse, "\n")

# Calculate MAE
mae <- mean(abs(testData$`IMDB Score` - predictions))
cat("Mean Absolute Error (MAE):", mae, "\n")

model <- lm(data=NetflixOriginals, formula = Runtime ~ `IMDB Score`)

summary(model)

res <- cor.test(NetflixOriginals$Runtime, NetflixOriginals$`IMDB
Score`,method="pearson")

res

#P-Value
res$p.value

#Correlation Coefficient
res$estimate
```