# Load necessary libraries

library(dplyr) # for data manipulation

library(ggplot2) # for visualization

library(corrplot) # for correlation plot

library(randomForest) # for feature importance

# Step 3: Exploratory Data Analysis (EDA)

# Summary statistics

summary(articles\_data)

# Histogram of shares

ggplot(articles\_data, aes(x = shares)) +

geom\_histogram(binwidth = 1000, fill = "skyblue") +

labs(title = "Distribution of Shares", x = "Shares", y = "Frequency")

# Step 4: Statistical Analysis

# Correlation matrix

correlation\_matrix <- cor(articles\_data[, c('shares', 'category', 'length', 'multimedia')])

corrplot(correlation\_matrix, method = "color")

# Step 5: Modeling

# Assuming you have split the data into training and testing sets

# Build a random forest model

rf\_model <- randomForest(shares ~ category + length + multimedia, data = train\_data)

# Step 6: Feature Importance

# Plot feature importance

varImpPlot(rf\_model)

# Step 7: Tailoring Content Production

# Based on the feature importance, prioritize content production for features with higher importance scores

# Step 8: Validation and Iteration

# Evaluate model performance on the test set

predicted\_shares <- predict(rf\_model, newdata = test\_data)

accuracy <- mean(predicted\_shares == test\_data$shares)

print(paste("Accuracy:", accuracy))

# Monitor article performance metrics over time and iterate on content production strategies as needed

# Step 5: Modeling (Linear Regression)

# Fit a linear regression model

lm\_model <- lm(shares ~ category + length + multimedia, data = train\_data)

# Step 6: Interpret the Model

# Print the summary of the linear regression model

summary(lm\_model)

# Step 7: Assess Model Performance

# Assess model performance using metrics such as R-squared and adjusted R-squared

# Predict shares on the test set

predicted\_shares <- predict(lm\_model, newdata = test\_data)

# Calculate R-squared

rsquared <- cor(test\_data$shares, predicted\_shares)^2

print(paste("R-squared:", rsquared))

# Calculate adjusted R-squared

num\_predictors <- length(coefficients(lm\_model)) - 1 # excluding intercept

n <- nrow(test\_data)

adj\_rsquared <- 1 - (1 - rsquared) \* (n - 1) / (n - num\_predictors - 1)

print(paste("Adjusted R-squared:", adj\_rsquared))

# Step 8: Tailoring Content Production

# Interpret coefficients to understand the impact of predictors on the number of shares

# Prioritize content production based on coefficients and statistical significance

Linear Regression:

# Assuming you have a dataset named 'data' with columns 'category', 'length', 'multimedia', and 'shares'

# Load necessary library

library(caret)

# Split the data into training and testing sets

set.seed(123)

trainIndex <- createDataPartition(data$shares, p = .8, list = FALSE)

trainData <- data[trainIndex, ]

testData <- data[-trainIndex, ]

# Fit linear regression model

lm\_model <- lm(shares ~ category + length + multimedia, data = trainData)

# Print summary of the model

summary(lm\_model)

# Predict on test data

predictions <- predict(lm\_model, newdata = testData)

# Evaluate model performance

mse <- mean((testData$shares - predictions)^2)

rmse <- sqrt(mse)

cat("Root Mean Squared Error:", rmse)

Decision Tree:

# Load necessary library

library(rpart)

# Fit decision tree model

tree\_model <- rpart(shares ~ category + length + multimedia, data = trainData, method = "anova")

# Print summary of the model

summary(tree\_model)

# Plot the decision tree

plot(tree\_model)

text(tree\_model)

# Predict on test data

predictions <- predict(tree\_model, newdata = testData)

# Evaluate model performance

mse <- mean((testData$shares - predictions)^2)

rmse <- sqrt(mse)

cat("Root Mean Squared Error:", rmse)

KNN:

# Load necessary library

library(class)

# Fit KNN model

knn\_model <- knn(train = trainData[, c("category", "length", "multimedia")],

test = testData[, c("category", "length", "multimedia")],

cl = trainData$shares,

k = 5)

# Predict on test data

predictions <- as.numeric(knn\_model)

# Evaluate model performance

mse <- mean((testData$shares - predictions)^2)

rmse <- sqrt(mse)

cat("Root Mean Squared Error:", rmse)