**1. Categorizing Iris Flowers**

r

Copy code

# Load necessary library

library(datasets)

library(ggplot2)

# Load iris dataset

data(iris)

# Scale the data for clustering

iris\_scaled <- scale(iris[, -5])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(iris\_scaled, centers = 3)

# Plot the clusters

ggplot(iris, aes(Petal.Length, Petal.Width, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Iris Flower Clusters", color = "Cluster")

**2. Classifying Cars based**

r

Copy code

# Load necessary library

library(datasets)

library(ggplot2)

# Load mtcars dataset

data(mtcars)

# Scale the data for clustering

mtcars\_scaled <- scale(mtcars[, c("mpg", "hp", "wt")])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(mtcars\_scaled, centers = 3)

# Plot the clusters

ggplot(mtcars, aes(mpg, hp, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Car Clusters based on Performance", color = "Cluster")

**3. Segmenting Wines based**

r

Copy code

# Load necessary library

library(tidyverse)

library(cluster)

# Load wine dataset

data(wine)

# Scale the data for clustering

wine\_scaled <- scale(wine)

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(wine\_scaled, centers = 3)

# Plot the clusters

ggplot(wine, aes(Alcohol, Phenols, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Wine Segmentation based on Chemical Composition", color = "Cluster")

**4. Classifying Breast Cancer**

r

Copy code

# Load necessary library

library(datasets)

library(ggplot2)

# Load Breast Cancer dataset

data(BreastCancer)

# Scale the data for clustering

bc\_scaled <- scale(BreastCancer[, -1])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(bc\_scaled, centers = 2)

# Plot the clusters

ggplot(BreastCancer, aes(Radius, Texture, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Breast Cancer Tumor Classification", color = "Cluster")

**5. Segmenting Patients at**

r

Copy code

# Load necessary library

library(ggplot2)

library(datasets)

# Load Pima Indians Diabetes dataset

data(pima)

# Scale the data for clustering

pima\_scaled <- scale(pima[, -9])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(pima\_scaled, centers = 2)

# Plot the clusters

ggplot(pima, aes(Glucose, BMI, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Diabetes Risk Patient Segmentation", color = "Cluster")

**6. Predicting Customer Purchase**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load retail dataset

data(retail)

# Preprocess data

retail\_processed <- retail %>%

select(browsing\_behavior, demographic\_info, previous\_purchase)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(retail\_processed$purchase, p = 0.8, list = FALSE)

trainData <- retail\_processed[trainIndex,]

testData <- retail\_processed[-trainIndex,]

# Build a Logistic Regression model

model <- glm(purchase ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$purchase, predictions)

plot(roc\_curve)

**7. Predicting Customer Churn**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Telco Customer Churn dataset

data(telco)

# Preprocess data

telco\_processed <- telco %>%

select(usage\_patterns, contract\_length, service\_calls, payment\_history)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(telco\_processed$churn, p = 0.8, list = FALSE)

trainData <- telco\_processed[trainIndex,]

testData <- telco\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(churn ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$churn, predictions)

plot(roc\_curve)

**8. Detecting Fraudulent Transactions**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load German Credit dataset

data(credit)

# Preprocess data

credit\_processed <- credit %>%

select(transaction\_amount, time, location, transaction\_patterns)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(credit\_processed$fraud, p = 0.8, list = FALSE)

trainData <- credit\_processed[trainIndex,]

testData <- credit\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(fraud ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$fraud, predictions)

plot(roc\_curve)

**9. Predicting Rain Tomorrow**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Australian Weather dataset

data(weather)

# Preprocess data

weather\_processed <- weather %>%

select(temperature, humidity, wind\_speed, pressure)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(weather\_processed$rain, p = 0.8, list = FALSE)

trainData <- weather\_processed[trainIndex,]

testData <- weather\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(rain ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$rain, predictions)

plot(roc\_curve)

**10. Predicting House Prices**

r

Copy code

# Load necessary libraries

library(Metrics)

library(tidyverse)

# Load Boston Housing dataset

data(boston)

# Preprocess data

boston\_processed <- boston %>%

select(size, rooms, age, neighborhood)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(boston\_processed$price, p = 0.8, list = FALSE)

trainData <- boston\_processed[trainIndex,]

testData <- boston\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(price ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

mse <- mse(testData$price, predictions)

print(mse)

**11. Predicting Employee Salary**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Salary Prediction dataset

data(salary)

# Preprocess data

salary\_processed <- salary %>%

select(experience, education\_level, job\_role)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(salary\_processed$salary, p = 0.8, list = FALSE)

trainData <- salary\_processed[trainIndex,]

testData <- salary\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(salary ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**12. Segmenting Customers using Hierarchical Clustering**

r

Copy code

# Load necessary libraries

library(tidyverse)

library(cluster)

# Load Mall Customer Segmentation dataset

data(mall)

# Scale the data for clustering

mall\_scaled <- scale(mall[, c("purchase\_frequency", "spending\_amount", "product\_types")])

# Apply Hierarchical Clustering

hclust\_result <- hclust(dist(mall\_scaled), method = "complete")

# Plot the dendrogram

plot(hclust\_result, labels = FALSE, main = "Customer Segmentation Dendrogram")

**13. Classifying Patients using Hierarchical Clustering**

r

Copy code

# Load necessary libraries

library(tidyverse)

library(cluster)

# Load Hospital Readmissions dataset

data(hospital)

# Scale the data for clustering

hospital\_scaled <- scale(hospital[, c("chronic\_diseases", "lifestyle\_factors", "age")])

# Apply Hierarchical Clustering

hclust\_result <- hclust(dist(hospital\_scaled), method = "complete")

# Plot the dendrogram

plot(hclust\_result, labels = FALSE, main = "Patient Classification Dendrogram")

**14. Identifying User Groups using Hierarchical Clustering**

r

Copy code

# Load necessary libraries

library(tidyverse)

library(cluster)

# Load Facebook Social Network dataset

data(facebook)

# Scale the data for clustering

facebook\_scaled <- scale(facebook[, c("activity\_level", "friends\_count", "posts\_per\_day")])

# Apply Hierarchical Clustering

hclust\_result <- hclust(dist(facebook\_scaled), method = "complete")

# Plot the dendrogram

plot(hclust\_result, labels = FALSE, main = "User Group Classification Dendrogram")

**15. Segmenting Retail Customers using K-Means Clustering**

r

Copy code

# Load necessary library

library(datasets)

library(ggplot2)

# Load retail dataset

data(retail)

# Scale the data for clustering

retail\_scaled <- scale(retail[, c("purchase\_amount", "frequency", "visit\_duration")])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(retail\_scaled, centers = 4)

# Plot the clusters

ggplot(retail, aes(purchase\_amount, frequency, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Retail Customer Segmentation", color = "Cluster")

**16. Segmenting Movie Ratings**

r

Copy code

# Load necessary library

library(datasets)

library(ggplot2)

# Load Movie Ratings dataset

data(movies)

# Scale the data for clustering

movies\_scaled <- scale(movies[, c("rating", "duration", "box\_office")])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(movies\_scaled, centers = 3)

# Plot the clusters

ggplot(movies, aes(rating, box\_office, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Movie Ratings Segmentation", color = "Cluster")

**17. Predicting Diabetes Using Logistic Regression**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Pima Indians Diabetes dataset

data(pima)

# Preprocess data

pima\_processed <- pima %>%

select(glucose, BMI, age, insulin)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(pima\_processed$diabetes, p = 0.8, list = FALSE)

trainData <- pima\_processed[trainIndex,]

testData <- pima\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(diabetes ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$diabetes, predictions)

plot(roc\_curve)

**18. Predicting Car Prices Using Linear Regression**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Car Prices dataset

data(cars)

# Preprocess data

cars\_processed <- cars %>%

select(age, mileage, brand)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(cars\_processed$price, p = 0.8, list = FALSE)

trainData <- cars\_processed[trainIndex,]

testData <- cars\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(price ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**19. Predicting Credit Risk**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load German Credit dataset

data(credit)

# Preprocess data

credit\_processed <- credit %>%

select(transaction\_amount, time, location, transaction\_patterns)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(credit\_processed$credit\_risk, p = 0.8, list = FALSE)

trainData <- credit\_processed[trainIndex,]

testData <- credit\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(credit\_risk ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$credit\_risk, predictions)

plot(roc\_curve)

**20. Detecting Heart Disease**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Heart Disease dataset

data(heart)

# Preprocess data

heart\_processed <- heart %>%

select(age, cholesterol, blood\_pressure, family\_history)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(heart\_processed$heart\_disease, p = 0.8, list = FALSE)

trainData <- heart\_processed[trainIndex,]

testData <- heart\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(heart\_disease ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$heart\_disease, predictions)

plot(roc\_curve)

**21. Predicting House Sales Prices**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Boston Housing dataset

data(boston)

# Preprocess data

boston\_processed <- boston %>%

select(size, rooms, age, neighborhood)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(boston\_processed$price, p = 0.8, list = FALSE)

trainData <- boston\_processed[trainIndex,]

testData <- boston\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(price ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

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

print(mse)

**22. Predicting Loan Default**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Loan Default dataset

data(loan\_default)

# Preprocess data

loan\_default\_processed <- loan\_default %>%

select(income, loan\_amount, credit\_score, marital\_status)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(loan\_default\_processed$default, p = 0.8, list = FALSE)

trainData <- loan\_default\_processed[trainIndex,]

testData <- loan\_default\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(default ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$default, predictions)

plot(roc\_curve)

**23. Predicting Online Shopping Behavior**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Online Shopping dataset

data(online\_shopping)

# Preprocess data

shopping\_processed <- online\_shopping %>%

select(browser, time\_on\_site, product\_type, previous\_visits)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(shopping\_processed$purchase, p = 0.8, list = FALSE)

trainData <- shopping\_processed[trainIndex,]

testData <- shopping\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(purchase ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$purchase, predictions)

plot(roc\_curve)

**24. Predicting Product Demand**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Retail Sales dataset

data(retail\_sales)

# Preprocess data

sales\_processed <- retail\_sales %>%

select(product\_category, price, marketing\_spend, seasonality)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(sales\_processed$demand, p = 0.8, list = FALSE)

trainData <- sales\_processed[trainIndex,]

testData <- sales\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(demand ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**25. Predicting Employee Retention**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Employee Retention dataset

data(employee\_retention)

# Preprocess data

retention\_processed <- employee\_retention %>%

select(age, tenure, salary, performance\_score)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(retention\_processed$retention, p = 0.8, list = FALSE)

trainData <- retention\_processed[trainIndex,]

testData <- retention\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(retention ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$retention, predictions)

plot(roc\_curve)

**26. Customer Segmentation using K-Means**

r

Copy code

# Load necessary library

library(datasets)

library(ggplot2)

# Load Customer dataset

data(customers)

# Scale the data for clustering

customers\_scaled <- scale(customers[, c("spending\_score", "annual\_income")])

# Apply K-means clustering

set.seed(42)

kmeans\_result <- kmeans(customers\_scaled, centers = 5)

# Plot the clusters

ggplot(customers, aes(spending\_score, annual\_income, color = factor(kmeans\_result$cluster))) +

geom\_point() +

labs(title = "Customer Segmentation", color = "Cluster")

**27. Predicting Traffic Accidents**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Traffic Accidents dataset

data(traffic\_accidents)

# Preprocess data

traffic\_processed <- traffic\_accidents %>%

select(weather\_condition, road\_condition, time\_of\_day, accident\_severity)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(traffic\_processed$accident\_severity, p = 0.8, list = FALSE)

trainData <- traffic\_processed[trainIndex,]

testData <- traffic\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(accident\_severity ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$accident\_severity, predictions)

plot(roc\_curve)

**28. Predicting Disease Spread**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Disease Spread dataset

data(disease\_spread)

# Preprocess data

disease\_processed <- disease\_spread %>%

select(temperature, humidity, population\_density, spread\_rate)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(disease\_processed$spread\_rate, p = 0.8, list = FALSE)

trainData <- disease\_processed[trainIndex,]

testData <- disease\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(spread\_rate ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**29. Predicting Energy Consumption**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Energy Consumption dataset

data(energy\_consumption)

# Preprocess data

energy\_processed <- energy\_consumption %>%

select(temperature, humidity, time\_of\_day, energy\_usage)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(energy\_processed$energy\_usage, p = 0.8, list = FALSE)

trainData <- energy\_processed[trainIndex,]

testData <- energy\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(energy\_usage ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**30. Predicting Air Quality Index**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Air Quality dataset

data(air\_quality)

# Preprocess data

air\_quality\_processed <- air\_quality %>%

select(temperature, humidity, wind\_speed, pm2.5)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(air\_quality\_processed$pm2.5, p = 0.8, list = FALSE)

trainData <- air\_quality\_processed[trainIndex,]

testData <- air\_quality\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(pm2.5 ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

**31. Predicting Crop Yield**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Crop Yield dataset

data(crop\_yield)

# Preprocess data

crop\_processed <- crop\_yield %>%

select(temperature, rainfall, soil\_quality, crop\_type)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(crop\_processed$yield, p = 0.8, list = FALSE)

trainData <- crop\_processed[trainIndex,]

testData <- crop\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(yield ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**32. Customer Churn Prediction**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Customer Churn dataset

data(customer\_churn)

# Preprocess data

churn\_processed <- customer\_churn %>%

select(age, tenure, service\_plan, customer\_support\_calls)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(churn\_processed$churn, p = 0.8, list = FALSE)

trainData <- churn\_processed[trainIndex,]

testData <- churn\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(churn ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$churn, predictions)

plot(roc\_curve)

**33. Predicting Student Performance**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Student Performance dataset

data(student\_performance)

# Preprocess data

student\_processed <- student\_performance %>%

select(math\_score, reading\_score, writing\_score, family\_support)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(student\_processed$final\_score, p = 0.8, list = FALSE)

trainData <- student\_processed[trainIndex,]

testData <- student\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(final\_score ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

**34. Predicting Hospital Readmissions**

r

Copy code

# Load necessary libraries

library(caret)

library(dplyr)

# Load Hospital Readmission dataset

data(hospital\_readmissions)

# Preprocess data

readmission\_processed <- hospital\_readmissions %>%

select(age, comorbidities, treatment\_type, previous\_visits)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(readmission\_processed$readmitted, p = 0.8, list = FALSE)

trainData <- readmission\_processed[trainIndex,]

testData <- readmission\_processed[-trainIndex,]

# Build Logistic Regression model

model <- glm(readmitted ~ ., data = trainData, family = "binomial")

# Evaluate the model

predictions <- predict(model, newdata = testData, type = "response")

roc\_curve <- roc(testData$readmitted, predictions)

plot(roc\_curve)

**35. Predicting Movie Success**

r

Copy code

# Load necessary libraries

library(tidyverse)

# Load Movie dataset

data(movie\_data)

# Preprocess data

movie\_processed <- movie\_data %>%

select(budget, genre, director, star\_rating)

# Split the data into training and testing sets

set.seed(42)

trainIndex <- createDataPartition(movie\_processed$box\_office, p = 0.8, list = FALSE)

trainData <- movie\_processed[trainIndex,]

testData <- movie\_processed[-trainIndex,]

# Build Linear Regression model

model <- lm(box\_office ~ ., data = trainData)

# Evaluate the model

predictions <- predict(model, newdata = testData)

summary(predictions)

These are the final code snippets, covering a variety of predictive models across different types of data, including crop yield, student performance, and movie success.