The prompt is given to ChatGPT:

import pandas as pd import numpy as np cancer_data_df = pd.read_csv('breast-cancer-wisconsin.csv') cancer_data_df # delete the rows with missing value. cancer_data_df = cancer_data_df.replace("?", np.NaN) # # replace "?" with NaN cancer_data_df = cancer_data_df.dropna() cancer_data_df.isnull().sum() # Use 30% test 70% training data. from sklearn.model_selection import train_test_split X = cancer_data_df.drop(['Sample', 'Class'], axis=1) y = cancer_data_df['Class'] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # Use the Naïve Bayes methodology to develop a classification model for the Diagnosis. from sklearn.naive_bayes import GaussianNB gnb = GaussianNB() gnb.fit(X_train, y_train) y_pred = gnb.predict(X_test) # Evaluate the model using the confusion matrix and the accuracy score. # printing classfication report, confusion matrix, accuracy, and Train set & Test set scores from sklearn.metrics import confusion_matrix, accuracy_score, classification_report print("Classification Report: \n") print(classification_report(y_test, y_pred), end="\n") print('Training set score: {:.4f}'.format(gnb.score(X_train, y_train))) print('\nTest set score: {:.4f}'.format(gnb.score(X_test, y_test))) print(f'\nConfusion Matrix:\n{confusion_matrix(y_test, y_pred)}') print(f'\nAccuracy: {accuracy_score(y_test, y_pred)}')
write the above code in R

ChatGPT response:

Here's the equivalent code in R to perform classification on the breast cancer dataset using Gaussian Naive Bayes and evaluate the model:

```
# Load required libraries
library(e1071)
library(caret)

# Read the dataset
cancer_data_df <- read.csv('breast-cancer-wisconsin.csv')

# Replace "?" with NA
cancer_data_df[cancer_data_df == "?"] <- NA

# Remove rows with missing values
cancer_data_df <- na.omit(cancer_data_df)

# Split the data into predictors (X) and the target variable (y)
X <- cancer_data_df[, -c(1, 11)]
y <- cancer_data_df$Class

# Split the data into training and test sets (70% training, 30% test)
set.seed(1)  # For reproducibility
split <- createDataPartition(y, p = 0.7, list = FALSE)
```

```
X_train <- X[split, ]
X_test <- X[-split, ]
y_train <- y[split]
y_test <- y[-split]

# Train a Gaussian Naive Bayes model
gnb <- naiveBayes(y_train ~ ., data = data.frame(X_train, y_train))

# Make predictions on the test set
y_pred <- predict(gnb, newdata = data.frame(X_test))

# Evaluate the model using confusion matrix and accuracy
conf_matrix <- table(Actual = y_test, Predicted = y_pred$class)
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)

# Print classification report, confusion matrix, accuracy, and train/test scores
print("Classification Report:")
print(conf_matrix)
print(paste("Accuracy:", accuracy))
print(paste("Training set score:", mean(predict(gnb, data.frame(X_train))$class == y_train)))
print(paste("Test set score:", mean(predict(gnb, newdata = data.frame(X_test))$class == y_test))
```

In this code, we first load the necessary libraries, read the dataset, handle missing values, split the data into training and test sets, train a Gaussian Naive Bayes model, make predictions, and evaluate the model's accuracy and performance.