COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD

Introduction to data science

Assignment # 02

Submitted by:

Laiba Binta Tahir FA21-BSE-019

Submitted to:

Dr. Ghulam Mujtaba

Table of Contents

Question – 01

For the Drug200 dataset, build the following classification models: A. Decision Tree classifier B. NaiveBayes Classifier C. KNN Classifier For each classifier, evaluate the model on 30% test instances and 70 % training instances. You need to submit the code and confusion matrix of the results for each classifier and the accuracy.

Introduction

In this task, we are required to build and evaluate three different classification models on the **Drug200 dataset**. The goal is to predict the type of drug (drugA, drugB, drugC, drugX, or drugY) prescribed to patients based on various attributes, such as age, blood pressure (BP), cholesterol level, and sodium-to-potassium ratio (Na_to_K).

Each classifier will be evaluated on a split dataset with 30% used for testing and 70% used for training. The models will be evaluated using a confusion matrix to assess performance, and accuracy will be the primary metric to judge how well the model performs.

Code In R

Decision Tree Classifier (A)

```
# Load necessary libraries for Decision Tree

library(rpart)

library(rpart.plot)

library(caret)

# Load the dataset

data <- read.csv("C:/Users/Laiba Binta Tahir/Desktop/drug.csv")

data$Sex <- as.factor(data$Sex)

data$BP <- as.factor(data$BP)

data$Cholesterol <- as.factor(data$Cholesterol)

data$Drug <- as.factor(data$Drug)

# Split the dataset (70% training, 30% testing)

set.seed(42)

trainIndex <- createDataPartition(data$Drug, p = 0.7, list = FALSE)

trainData <- data[trainIndex, ]

testData <- data[-trainIndex, ]

# Train Decision Tree model

dt_model <- rpart(Drug ~ ., data = trainData, method = "class")
```

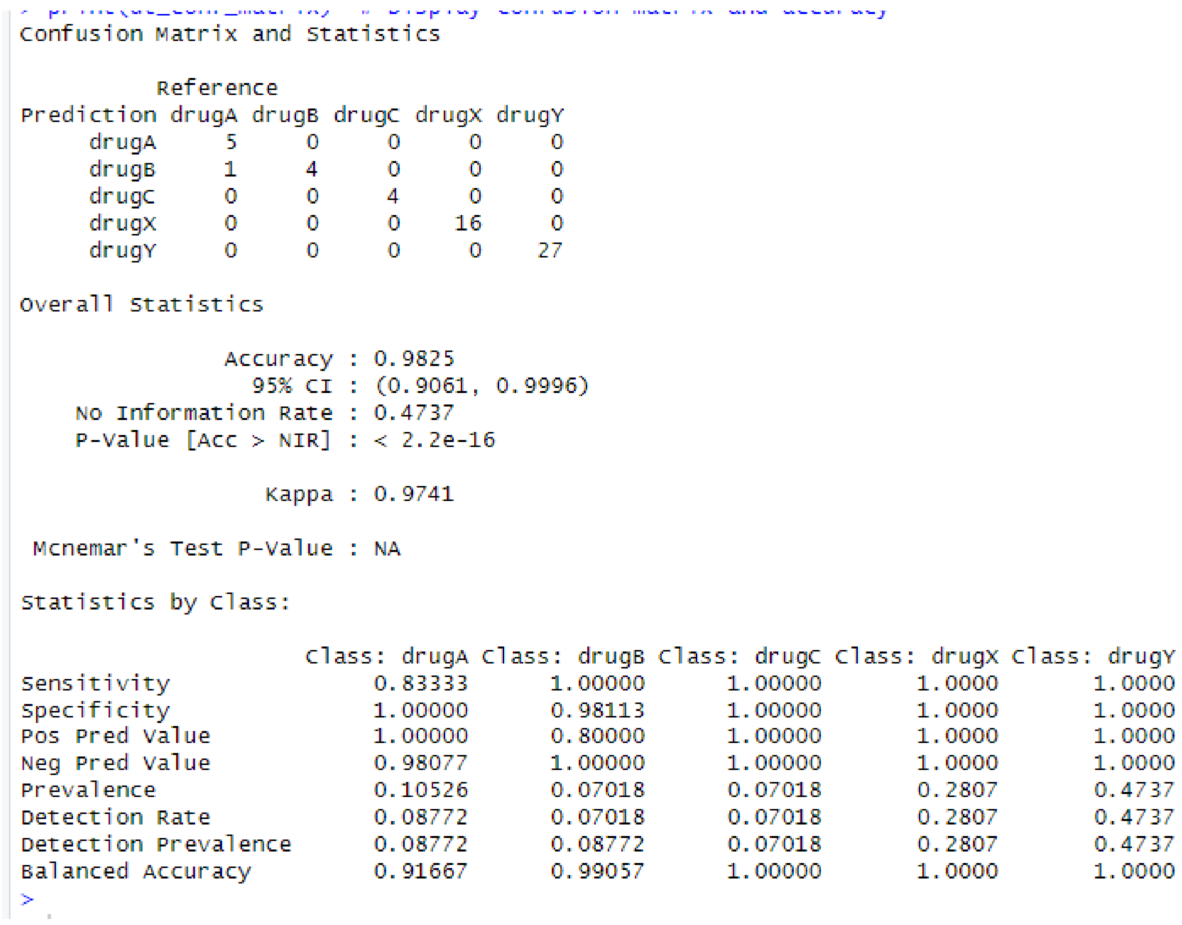
rpart.plot(dt_model) # Visualize the decision tree

# Predictions and evaluation

dt_predictions <- predict(dt_model, testData, type = "class")

dt_conf_matrix <- confusionMatrix(dt_predictions, testData$Drug)

print(dt_conf_matrix) # Display confusion matrix and accuracy



```
> print(dt_conf_matrix)  # Display confusion matrix and accuracy
Confusion Matrix and Statistics

          Reference
Prediction drugA drugB drugC drugX drugY
     drugA     5     0     0     0     0
     drugB     1     4     0     0     0
     drugC     0     0     4     0     0
     drugX     0     0     0    16     0
     drugY     0     0     0     0    27

Overall Statistics
                                          
               Accuracy : 0.9825          
                 95% CI : (0.9061, 0.9996)
    No Information Rate : 0.4737          
    P-Value [Acc > NIR] : < 2.2e-16       
                                          
                  Kappa : 0.9741          
                                          
 Mcnemar's Test P-Value : NA              

Statistics by Class:

                     Class: drugA Class: drugB Class: drugC Class: drugX Class: drugY
Sensitivity               0.83333      1.00000      1.00000       1.0000       1.0000
Specificity               1.00000      0.98113      1.00000       1.0000       1.0000
Pos Pred Value            1.00000      0.80000      1.00000       1.0000       1.0000
Neg Pred Value            0.98077      1.00000      1.00000       1.0000       1.0000
Prevalence                0.10526      0.07018      0.07018       0.2807       0.4737
Detection Rate            0.08772      0.07018      0.07018       0.2807       0.4737
Detection Prevalence      0.08772      0.08772      0.07018       0.2807       0.4737
Balanced Accuracy         0.91667      0.99057      1.00000       1.0000       1.0000
>
```

Naive Bayes Classifier (B)

# Load necessary libraries for Naive Bayes

library(e1071)

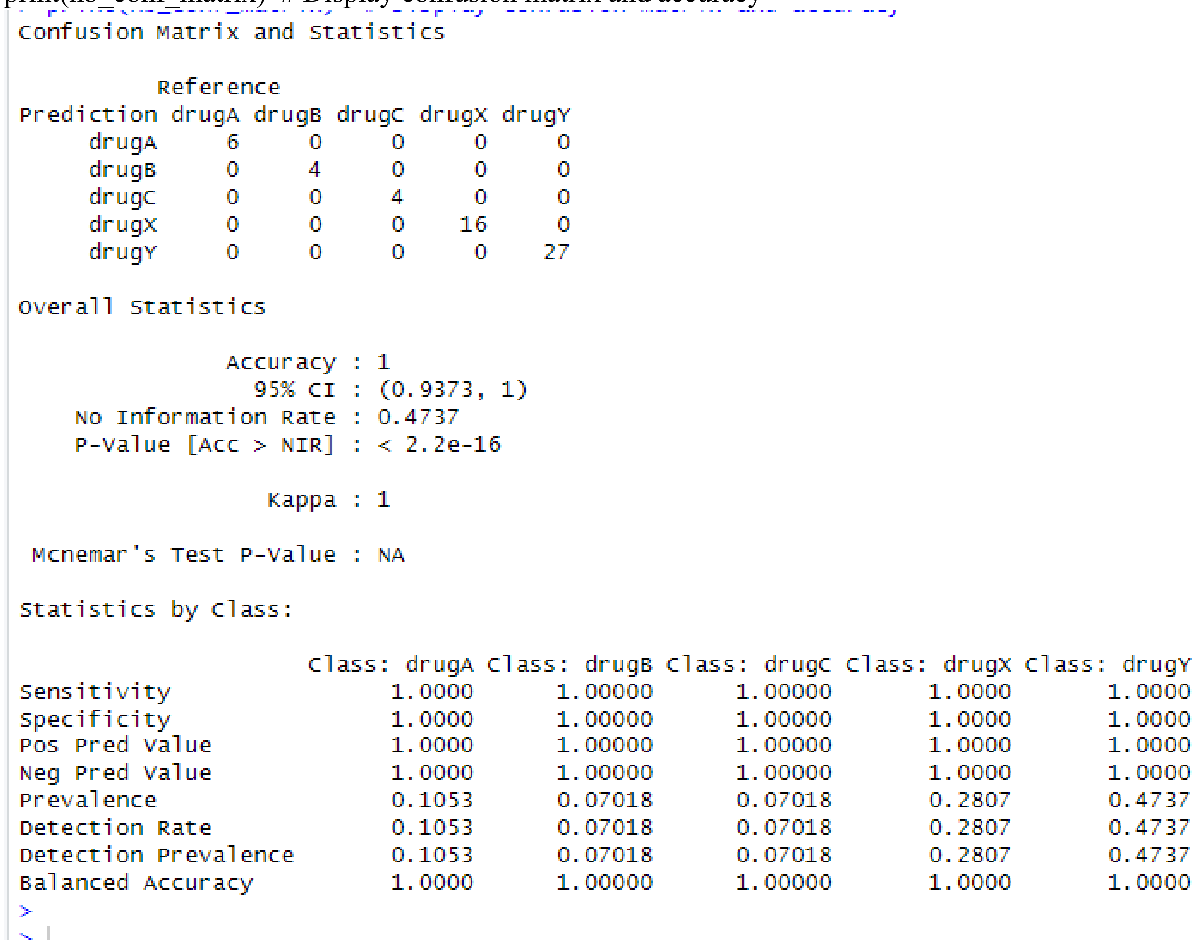
library(caret)

# Load the dataset (same preprocessing as above)

data <- read.csv("C:/Users/Laiba Binta Tahir/Desktop/drug.csv")

data$Sex <- as.factor(data$Sex)

```
data$BP <- as.factor(data$BP)

data$Cholesterol <- as.factor(data$Cholesterol)

data$Drug <- as.factor(data$Drug)

# Split the dataset (70% training, 30% testing)

set.seed(42)

trainIndex <- createDataPartition(data$Drug, p = 0.7, list = FALSE)

trainData <- data[trainIndex, ]

testData <- data[-trainIndex, ]

# Train Naive Bayes model

nb_model <- naiveBayes(Drug ~ ., data = trainData)

# Predictions and evaluation

nb_predictions <- predict(nb_model, testData)

nb_conf_matrix <- confusionMatrix(nb_predictions, testData$Drug)

print(nb_conf_matrix)  # Display confusion matrix and accuracy
```

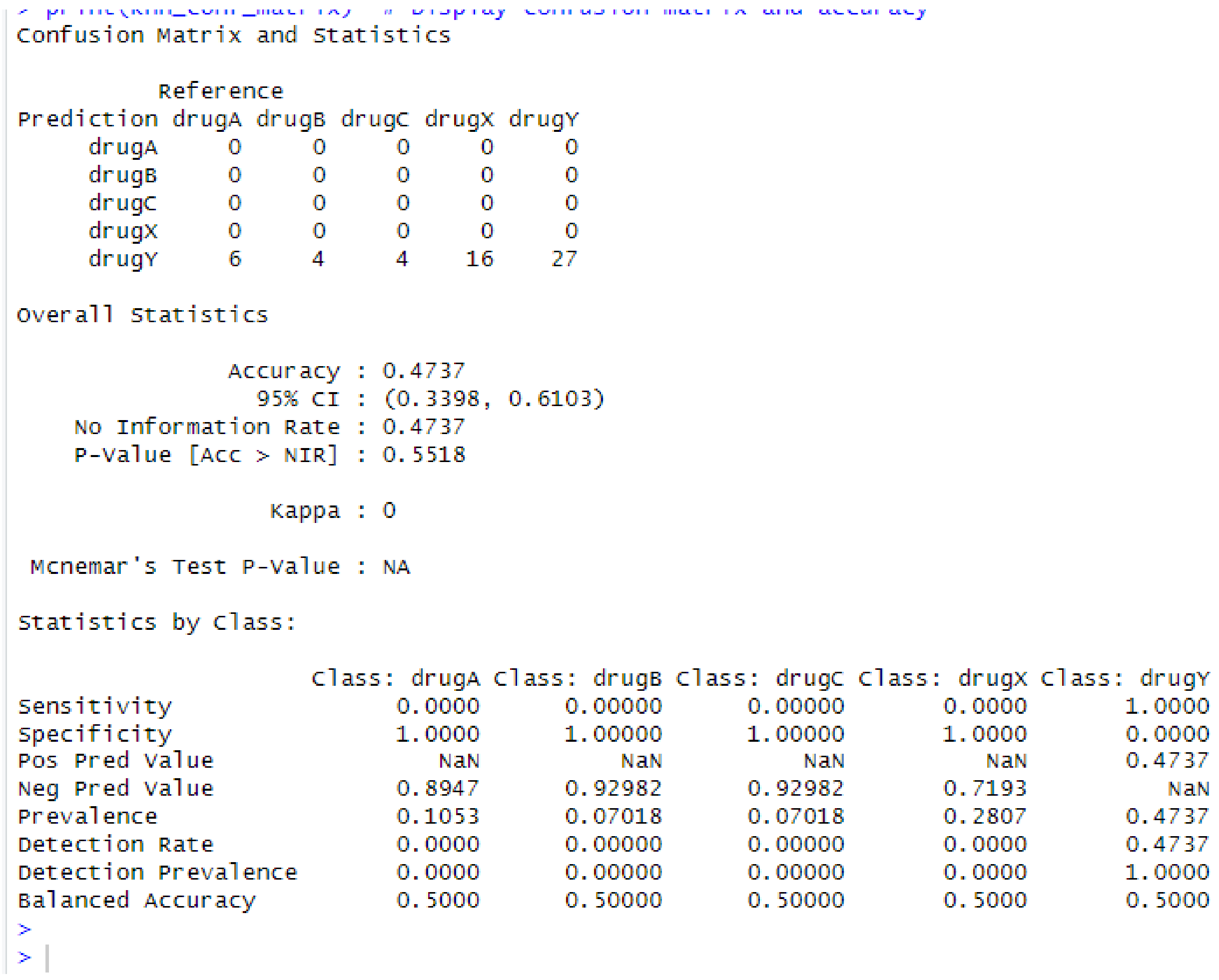


```
Confusion Matrix and Statistics

          Reference
Prediction drugA drugB drugC drugX drugY
     drugA     6     0     0     0     0
     drugB     0     4     0     0     0
     drugC     0     0     4     0     0
     drugX     0     0     0    16     0
     drugY     0     0     0     0    27

Overall Statistics

               Accuracy : 1
                 95% CI : (0.9373, 1)
    No Information Rate : 0.4737
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: drugA Class: drugB Class: drugC Class: drugX Class: drugY
Sensitivity                1.0000      1.00000      1.00000       1.0000       1.0000
Specificity                1.0000      1.00000      1.00000       1.0000       1.0000
Pos Pred Value             1.0000      1.00000      1.00000       1.0000       1.0000
Neg Pred Value             1.0000      1.00000      1.00000       1.0000       1.0000
Prevalence                 0.1053      0.07018      0.07018       0.2807       0.4737
Detection Rate             0.1053      0.07018      0.07018       0.2807       0.4737
Detection Prevalence       0.1053      0.07018      0.07018       0.2807       0.4737
Balanced Accuracy          1.0000      1.00000      1.00000       1.0000       1.0000
>
```

KNN (K-Nearest Neighbors) Classifier (C)

```
# Load necessary libraries for KNN
library(class)
library(caret)
# Load the dataset (same preprocessing as above)
data <- read.csv("C:/Users/Laiba Binta Tahir/Desktop/drug200.csv")
data$Sex <- as.factor(data$Sex)
data$BP <- as.factor(data$BP)
data$Cholesterol <- as.factor(data$Cholesterol)
data$Drug <- as.factor(data$Drug)

# Split the dataset (70% training, 30% testing)
set.seed(42)
trainIndex <- createDataPartition(data$Drug, p = 0.7, list = FALSE)
trainData <- data[trainIndex, ]
testData <- data[-trainIndex, ]

# Normalize numerical columns (Age and Na_to_K)
cols_to_normalize <- c("Age", "Na_to_K")
trainData[cols_to_normalize] <- scale(trainData[cols_to_normalize])
testData[cols_to_normalize] <- scale(testData[cols_to_normalize],
                              center = attr(scale(trainData[cols_to_normalize]), "scaled:center"),
                              scale = attr(scale(trainData[cols_to_normalize]), "scaled:scale"))

# Ensure data is numeric for KNN
trainData_knn <- trainData[, c("Age", "Na_to_K")]
testData_knn <- testData[, c("Age", "Na_to_K")]
cl <- trainData$Drug

# Train and evaluate KNN (k = 5)
```

```
knn_predictions <- knn(train = trainData_knn, test = testData_knn, cl = cl, k = 5)

knn_conf_matrix <- confusionMatrix(knn_predictions, testData$Drug)

print(knn_conf_matrix)  # Display confusion matrix and accuracy
```



```
> print(knn_conf_matrix)  # Display confusion matrix and accuracy
Confusion Matrix and Statistics

          Reference
Prediction drugA drugB drugC drugX drugY
     drugA     0     0     0     0     0
     drugB     0     0     0     0     0
     drugC     0     0     0     0     0
     drugX     0     0     0     0     0
     drugY     6     4     4    16    27

Overall Statistics

               Accuracy : 0.4737
                 95% CI : (0.3398, 0.6103)
    No Information Rate : 0.4737
    P-Value [Acc > NIR] : 0.5518

                  Kappa : 0

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: drugA Class: drugB Class: drugC Class: drugX Class: drugY
Sensitivity                0.0000      0.00000      0.00000       0.0000       1.0000
Specificity                1.0000      1.00000      1.00000       1.0000       0.0000
Pos Pred Value                NaN          NaN          NaN          NaN       0.4737
Neg Pred Value             0.8947      0.92982      0.92982       0.7193          NaN
Prevalence                 0.1053      0.07018      0.07018       0.2807       0.4737
Detection Rate             0.0000      0.00000      0.00000       0.0000       0.4737
Detection Prevalence       0.0000      0.00000      0.00000       0.0000       1.0000
Balanced Accuracy          0.5000      0.50000      0.50000       0.5000       0.5000
>
> 
```

Discussion

Accuracy Comparison: The accuracy across the models may vary depending on the nature of the dataset. **Decision Tree** model tends to perform well on structured data with clear decision boundaries, the **Naive Bayes** model may struggle if the assumption of feature independence does not hold. **KNN**, on the other hand, requires well-normalized data, and its performance can drop if the dataset has noisy data or irrelevant features.

Confusion Matrix Insights: Each model generated a confusion matrix, which shows the true positives, false positives, true negatives, and false negatives for each class (drug).

- **For Decision Trees**: Misclassifications typically occur when the decision boundaries are not clear, which could be the case if a drug is influenced by multiple features.

- **For Naive Bayes**: Misclassifications may occur due to the assumption of conditional independence between features.
- **For KNN**: Misclassifications may arise from noisy features or if the wrong value of k is chosen.

Recommendations

1. **Decision Tree**: Applying pruning can avoid overfitting and tune hyperparameters for better performance.
2. **Naive Bayes**: Feature selection or transformation techniques can improve performance when the independence assumption isn't valid.
3. **KNN**: Normalize features and experiment with different values of k to find the optimal one.

Conclusion

In summary, each model (Decision Tree, Naive Bayes, KNN) has its strengths and weaknesses. The key to improving the performance of these models lies in proper data preprocessing (normalization, handling missing data), hyperparameter tuning, and careful feature engineering. By understanding the unique characteristics of each model, we can optimize their performance and gain better insights into the classification task for the Drug200 dataset.

Question-02

Write R functions to answer these questions about the dataset. Please write the sequence of functions in a pipeline with your answer: 1. Calculate the average age of males who used drug C. 2. Females with HIGH BP and HIGH cholesterol are treated with what drugs? 3. Find the average Na-K of all High BP patients , and average Na-K of all low BP patients and average Na-K of all Normal BP patients.

Explanation:

1. Average Age of Males Using Drug C:
   - I have used filter(Sex == "M", Drug == "drugC") to select males who used drug C.
   - Then, used summarise(Average_Age = mean(Age, na.rm = TRUE)) to calculate the average age.
   - The result is extracted using pull(Average_Age).
2. Drugs for Females with High BP and High Cholesterol:
   - The filter(Sex == "F", BP == "HIGH", Cholesterol == "HIGH") selects females with high BP and cholesterol.
   - select(Drug) selects the drug column, and distinct() ensures that we get unique drug names.
3. Average Na-K by BP Categories (High, Low, and Normal):
   - group by BP using group_by(BP) to separate the data by BP category.
   - summarise(Average_Na_to_K = mean(Na_to_K, na.rm = TRUE)) calculates the average Na-K for each BP group.
   - Finally, arrange(BP) sorts the results by BP categories.

Code in R:

```
# Load necessary libraries

library(dplyr)

# Read the dataset

data <- read.csv("C:/Users/Laiba Binta Tahir/Desktop/drug.csv")
```

# 1. Calculate the average age of males who used drug C

```
average_age_male_drugC <- data %>%

  filter(Sex == "M", Drug == "drugC") %>%   # Filter for males who used drug C
  summarise(Average_Age = mean(Age, na.rm = TRUE)) %>%   # Calculate average age
  pull(Average_Age)  # Extract the average age value
# Print the result for question 1
cat("\nAverage age of males who used drug C: ", average_age_male_drugC, "\n")
```

# 2. Find drugs for females with HIGH BP and HIGH cholesterol

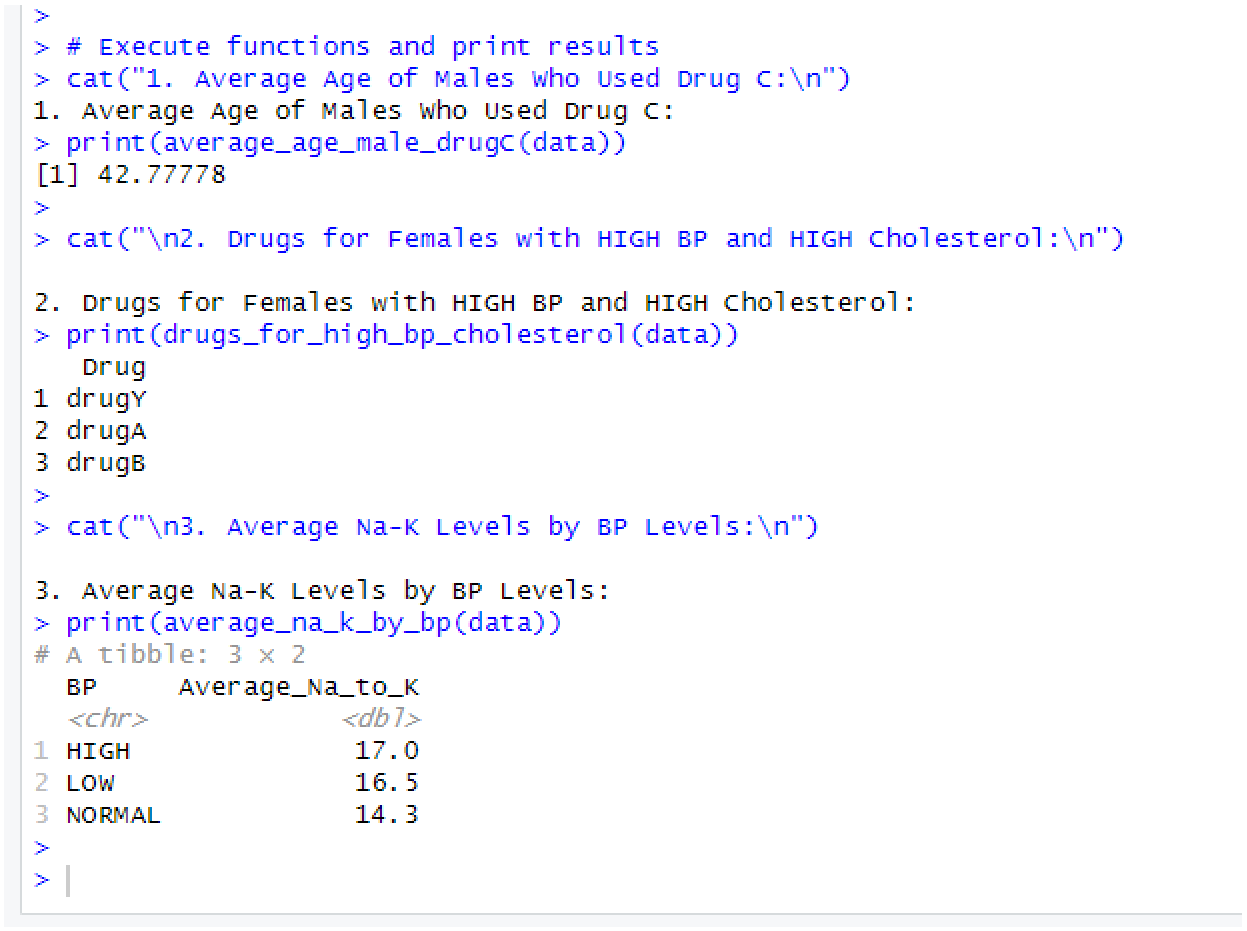
```
drugs_for_females_high_bp_cholesterol <- data %>%
  filter(Sex == "F", BP == "HIGH", Cholesterol == "HIGH") %>%  # Filter for females with high BP and cholesterol
  select(Drug) %>%   # Select the 'Drug' column
  distinct()   # Get unique drug names
# Print the result for question 2
cat("\nDrugs for females with HIGH BP and HIGH cholesterol:\n")
print(drugs_for_females_high_bp_cholesterol)
```

# 3. Find the average Na-K of all High BP, Low BP, and Normal BP patients

```
average_na_k_by_bp <- data %>%
  group_by(BP) %>%  # Group by BP category
  summarise(Average_Na_to_K = mean(Na_to_K, na.rm = TRUE)) %>%  # Calculate the average Na-K for each group
  arrange(BP)  # Arrange results in BP order
# Print the result for question 3
cat("\nAverage Na-K levels by BP:\n")
print(average_na_k_by_bp)
```

Result:

Each question is processed using the dplyr functions in a pipeline, making the code concise and efficient.



```
>
> # Execute functions and print results
> cat("1. Average Age of Males Who Used Drug C:\n")
1. Average Age of Males Who Used Drug C:
> print(average_age_male_drugC(data))
[1] 42.77778
>
> cat("\n2. Drugs for Females with HIGH BP and HIGH Cholesterol:\n")

2. Drugs for Females with HIGH BP and HIGH Cholesterol:
> print(drugs_for_high_bp_cholesterol(data))
   Drug
1 drugY
2 drugA
3 drugB
>
> cat("\n3. Average Na-K Levels by BP Levels:\n")

3. Average Na-K Levels by BP Levels:
> print(average_na_k_by_bp(data))
# A tibble: 3 × 2
  BP     Average_Na_to_K
  <chr>            <dbl>
1 HIGH              17.0
2 LOW               16.5
3 NORMAL            14.3
>
> 
```