Data Classification

1.Manually Add and Explore the Dataset:-

```
# Manually create a meaningful dataset
data <- data.frame(
  age = c(25, 45, 35, 50, 23, 40, 60, 28, 38, 55),
  blood_pressure = c(120, 140, 130, 150, 115, 135, 145, 125, 130, 160),
  cholesterol = c(200, 240, 220, 250, 180, 230, 260, 190, 210, 270),
  bmi = c(22.0, 30.5, 26.8, 32.0, 21.5, 28.0, 31.0, 23.5, 27.0, 33.5),
  risk_category = as.factor(c('low', 'high', 'medium', 'high', 'low', 'medium', 'high', 'low', 'medium', 'high'))
)

# Display the manually created dataset
print("Manually Created Dataset:")
print(data)
```

2. Preprocess the Data:-

```
# Check for missing values
print("Missing values count:")
print(sum(is.na(data)))

# Normalize the numerical features
normalize <- function(x) {
```

```
  return ((x - min(x)) / (max(x) - min(x)))
}

data[, 1:4] <- as.data.frame(lapply(data[, 1:4], normalize))

# Display the first few rows of the normalized dataset
print("Normalized Dataset:")
print(data)
```

3. Split the Data into Training and Testing Sets:-

```
# Set the seed for reproducibility
set.seed(42)

# Split the data into training (70%) and testing (30%) sets manually
sample_size <- floor(0.7 * nrow(data))
train_indices <- sample(seq_len(nrow(data)), size = sample_size)

training_set <- data[train_indices, ]
testing_set <- data[-train_indices, ]

# Display the dimensions of the training and testing sets
print("Dimensions of Training Set:")
print(dim(training_set))
print("Dimensions of Testing Set:")
print(dim(testing_set))
```

4.Build a Classification Model:-

```
# Simple classification using K-Nearest Neighbors (KNN) algorithm

# Define the distance function
euclidean_distance <- function(x1, x2) {
  return (sqrt(sum((x1 - x2) ^ 2)))
}

# Define the KNN function
knn <- function(train, test, k) {
  predictions <- c()

  for (i in 1:nrow(test)) {
    distances <- apply(train[, -5], 1, euclidean_distance, x2 = test[i, -5])
    nearest <- order(distances)[1:k]
    nearest_classes <- train[nearest, 5]
    predictions <- c(predictions, names(sort(table(nearest_classes), decreasing = TRUE))[1])
  }

  return (predictions)
}

# Predict the risk category on the testing set using KNN with k = 3
k <- 3
predictions <- knn(training_set, testing_set, k)
```

```
# Display predictions
print("Predictions:")
print(predictions)
```

5. Evaluate the Model

```
# Create a confusion matrix
confusion_matrix <- table(testing_set$risk_category, predictions)

# Print the confusion matrix
print("Confusion Matrix:")
print(confusion_matrix)

# Calculate the accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print(paste("Accuracy:", round(accuracy, 2)))
```