6. Explore conditional probabilities, independence, and expected outliers. Examine distribution shapes, skewness, correlations, and compare different outlier detection methods to assess consistency and insights.

✅ **Clean & Compact R Code**

```
# Load libraries

library(dplyr)

library(ggplot2)

library(e1071)

library(car)


# Load dataset

data <- mtcars

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

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


# 1. Conditional Probability: P(am = 1 | gear = 4)

cond_prob <- mean(data$am[data$gear == 4])

cat("P(am = 1 | gear = 4):", round(cond_prob, 3), "\n")


# 2. Independence: Chi-squared test between am and gear

chi_p <- chisq.test(table(data$am, data$gear))$p.value

cat("Chi-squared p-value (am vs gear):", round(chi_p, 4),

   ifelse(chi_p < 0.05, "→ Dependent\n", "→ Independent\n"))


# 3. Skewness

cat("Skewness of mpg:", round(skewness(data$mpg), 3), "\n")

cat("Skewness of hp:", round(skewness(data$hp), 3), "\n")


# 4. Correlation matrix

cat("\nCorrelation matrix:\n")

print(cor(select_if(data, is.numeric)))
```

```
# 5. Distribution of mpg
ggplot(data, aes(mpg)) +
  geom_histogram(binwidth = 2, fill = "skyblue", color = "black") +
  ggtitle("MPG Distribution") + theme_minimal()

# 6. Outlier Detection
## (a) IQR Method
Q1 <- quantile(data$mpg, 0.25)
Q3 <- quantile(data$mpg, 0.75)
IQR_val <- Q3 - Q1
out_iqr <- data$mpg[data$mpg < Q1 - 1.5 * IQR_val | data$mpg > Q3 + 1.5 * IQR_val]
cat("\nIQR Outliers (mpg):", out_iqr, "\n")

## (b) Z-score Method
z_scores <- scale(data$mpg)
out_z <- data$mpg[abs(z_scores) > 2]
cat("Z-score Outliers (mpg):", out_z, "\n")

## (c) Bonferroni Method
model <- lm(mpg ~ ., data = data)
cat("Bonferroni Outlier Test:\n")
print(outlierTest(model))
```

💡 **Notes:**

- scale() standardizes for Z-score.
- outlierTest() from the car package flags outliers based on studentized residuals.
- You can change mpg to other variables if needed.
- This version is **short, readable, and informative**, great for labs or reports.

7. Using Pearson and Spearman methods. Identify key influencing variables and visualize insights using heatmaps, scatter plots, box plots, bar plots, and clustering dendrograms

```
# Load necessary libraries
library(ggplot2)
library(corrplot)
library(heatmaply)
library(cluster)
library(RColorBrewer)

# Load example dataset
data(mtcars)

# 1. Pearson and Spearman Correlation
pearson_corr <- cor(mtcars, method = "pearson")
spearman_corr <- cor(mtcars, method = "spearman")

# 2. Pearson Correlation Heatmap
corrplot(pearson_corr, method = "color", type = "upper",
         col = brewer.pal(n = 9, name = "Blues"),
         title = "Pearson Correlation Matrix")

# 3. Spearman Correlation Heatmap
corrplot(spearman_corr, method = "color", type = "upper",
         col = brewer.pal(n = 9, name = "RdYlBu"),
         title = "Spearman Correlation Matrix")

# 4. Scatter Plot (mpg vs wt)
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point(aes(color = mpg)) +
  labs(title = "Scatter Plot of mpg vs wt") +
  theme_minimal()

# 5. Box Plot (mpg)
ggplot(mtcars, aes(x = factor(1), y = mpg)) +
  geom_boxplot() +
  labs(title = "Box Plot of mpg") +
  theme_minimal()

# 6. Bar Plot (cyl)
ggplot(mtcars, aes(x = factor(cyl))) +
  geom_bar(aes(fill = factor(cyl))) +
  labs(title = "Bar Plot of Cars with Different Cylinder Counts") +
  theme_minimal()

# 7. Clustering Dendrogram
dist_matrix <- dist(mtcars)
hc <- hclust(dist_matrix)
plot(hc, main = "Dendrogram of mtcars")

# 8. Clustered Heatmap
heatmaply::heatmaply(mtcars, main = "Clustered Heatmap of mtcars")
```

8. explain how Principal Component Regression (PCR) can be used to address multicollinearity issues while improving predictive accuracy.

```
# Install and load required package
if (!require(pls)) install.packages("pls", dependencies = TRUE)
library(pls)

# Load dataset
data(mtcars)

# Prepare data
```

```
X <- scale(mtcars[, -1])  # All predictors except mpg
Y <- mtcars$mpg           # Response variable

# Perform PCR with Cross-Validation
pcr_model <- pcr(Y ~ X, validation = "CV")

# Summary of the PCR model
summary(pcr_model)

# Plot RMSEP to find optimal number of components
validationplot(pcr_model, val.type = "RMSEP", main = "PCR Cross-Validation")

# Predict mpg using 3 components
predicted <- predict(pcr_model, ncomp = 3)

# Compare predicted vs actual values
plot(Y, predicted, col = "blue", pch = 16,
     xlab = "Actual MPG", ylab = "Predicted MPG",
     main = "PCR: Actual vs Predicted")
abline(a = 0, b = 1, col = "red")
```

9. Fit linear and quadratic regression, compare their R-squared and RMSE values, and interpret results. Identify the best-fitting model and predict sales

```
# Load libraries
library(ggplot2)
library(Metrics)  # For RMSE calculation

# Load dataset
data(mtcars)

# Fit linear regression model
linear_model <- lm(mpg ~ wt, data = mtcars)
```

```r
# Fit quadratic regression model
quadratic_model <- lm(mpg ~ wt + I(wt^2), data = mtcars)

# Summary outputs
summary(linear_model)
summary(quadratic_model)

# Predictions
linear_pred <- predict(linear_model, newdata = mtcars)
quadratic_pred <- predict(quadratic_model, newdata = mtcars)

# R-squared values
linear_r2 <- summary(linear_model)$r.squared
quadratic_r2 <- summary(quadratic_model)$r.squared

# RMSE values
linear_rmse <- rmse(mtcars$mpg, linear_pred)
quadratic_rmse <- rmse(mtcars$mpg, quadratic_pred)

# Print comparison
cat("Linear Model R-squared:", linear_r2, "\n")
cat("Quadratic Model R-squared:", quadratic_r2, "\n")
cat("Linear Model RMSE:", linear_rmse, "\n")
cat("Quadratic Model RMSE:", quadratic_rmse, "\n")

# Model Comparison
if (linear_r2 > quadratic_r2) {
  cat("Linear model provides a better fit based on R-squared.\n")
} else {
  cat("Quadratic model provides a better fit based on R-squared.\n")
}

# Predict mpg when wt = 3
new_data <- data.frame(wt = 3)
linear_prediction <- predict(linear_model, newdata = new_data)
quadratic_prediction <- predict(quadratic_model, newdata = new_data)

cat("\nPrediction from Linear Model (wt = 3):", linear_prediction, "\n")
cat("Prediction from Quadratic Model (wt = 3):", quadratic_prediction, "\n")
```

10. Compute the Pearson and Spearman correlation coefficients for a given dataset and analyze the relationships between variables. Explain how correlation matrices assist in feature selection and why highly correlated features might lead to redundancy in predictive models.

```r
# Load library
library(corrplot)

# Load dataset
data(mtcars)

# Compute Pearson and Spearman correlations
pearson_corr <- cor(mtcars, method = "pearson")
spearman_corr <- cor(mtcars, method = "spearman")

# Pearson heatmap
corrplot(pearson_corr, method = "circle", type = "upper",
         title = "Pearson Correlation Matrix", mar = c(0, 0, 1, 0))

# Spearman heatmap
corrplot(spearman_corr, method = "circle", type = "upper",
         title = "Spearman Correlation Matrix", mar = c(0, 0, 1, 0))
```

```
# Print correlation matrices
cat("Pearson Correlation Matrix:\n")
print(pearson_corr)

cat("\nSpearman Correlation Matrix:\n")
print(spearman_corr)
```

11. Visualize correlations, split data, and train a Logistic Regression model. Evaluate using accuracy, precision, recall, and F1-score. Apply PCA for dimensionality reduction and assess if performance improves for better

```
library(ggplot2); library(caret); library(corrplot); library(pls)

data(mtcars)
mtcars$mpg_binary <- ifelse(mtcars$mpg > 20, 1, 0)

# Correlation plot
corrplot(cor(mtcars[, -c(1, 11)]), method = "circle")

# Train-test split
set.seed(42)
split <- createDataPartition(mtcars$mpg_binary, p = 0.7, list = FALSE)
train <- mtcars[split, ]; test <- mtcars[-split, ]

# Logistic Regression
model <- glm(mpg_binary ~ ., data = train, family = "binomial")
pred <- ifelse(predict(model, test, type = "response") > 0.5, 1, 0)

# Evaluation
conf <- confusionMatrix(as.factor(pred), as.factor(test$mpg_binary))
cat("Accuracy:", mean(pred == test$mpg_binary), "\n")
cat("Precision:", conf$byClass["Pos Pred Value"], "\n")
cat("Recall:", conf$byClass["Sensitivity"], "\n")
cat("F1 Score:", conf$byClass["F1"], "\n")

# PCA
X <- scale(mtcars[, -c(1, 11)])
pca <- prcomp(X)
train_pca <- cbind(pca$x[split, 1:2], mpg_binary = train$mpg_binary)
test_pca <- cbind(pca$x[-split, 1:2], mpg_binary = test$mpg_binary)

# PCA model
model_pca <- glm(mpg_binary ~ ., data = train_pca, family = "binomial")
pred_pca <- ifelse(predict(model_pca, test_pca, type = "response") > 0.5, 1, 0)
conf_pca <- confusionMatrix(as.factor(pred_pca), as.factor(test_pca$mpg_binary))

cat("\nWith PCA - Accuracy:", mean(pred_pca == test_pca$mpg_binary), "\n")
cat("F1 Score with PCA:", conf_pca$byClass["F1"], "\n")
```

12 . Formulate the problem as a multiple linear regression equation and express it in matrix form. Using the normal equation, calculate the regression coefficients to fit the model. Discuss how multicollinearity among predictor variables affects the reliability of the regression model and explain how Principal Component Regression (PCR) can be used to address multicollinearity issues while improving predictive accuracy.

```
library(pls)

data(mtcars)
X <- as.matrix(mtcars[, -1])
Y <- mtcars$mpg
X <- cbind(1, X)

# Normal Equation
beta <- solve(t(X) %*% X) %*% t(X) %*% Y
cat("Regression Coefficients:\n"); print(beta)
```

```
# PCR
pcr_model <- pcr(mpg ~ ., data = mtcars, validation = "CV")
summary(pcr_model)
pred <- predict(pcr_model, ncomp = 5)
cat("RMSE:", sqrt(mean((Y - pred)^2)), "\n")
```

13.Explore conditional probabilities, independence, and expected outliers. Examine distribution shapes, skewness, correlations

```
library(ggplot2); library(corrplot); library(e1071)
data(mtcars)

# Conditional Probability
pAandB <- sum(mtcars$mpg > 20 & mtcars$hp > 100) / nrow(mtcars)
pB <- sum(mtcars$hp > 100) / nrow(mtcars)
cat("P(High MPG | HP > 100):", pAandB / pB, "\n")

# Independence (Correlation)
cat("Correlation (wt vs hp):", cor(mtcars$wt, mtcars$hp), "\n")

# Outliers using Z-score
z <- scale(mtcars$mpg)
cat("Outliers:", which(abs(z) > 3), "\n")

# Skewness
cat("Skewness of MPG:", skewness(mtcars$mpg), "\n")

# Histogram
ggplot(mtcars, aes(mpg)) + geom_histogram(binwidth = 2, fill = "skyblue", color = "black")

# Correlation Heatmap
corrplot(cor(mtcars), method = "circle")
```

14. calculate the regression coefficients to fit the model. Discuss how multicollinearity among predictor variables affects the reliability

```
data(mtcars)


X <- as.matrix(mtcars[, -1])

Y <- mtcars$mpg

X <- cbind(1, X)


# Normal Equation

beta <- solve(t(X) %*% X) %*% t(X) %*% Y

cat("Regression Coefficients:\n"); print(beta)
```

15. Show the relationship between cyl (Cylinders) and hp (Horsepower) using a box/scatter plot.

```
library(ggplot2)

data(mtcars)

mtcars$cyl <- as.factor(mtcars$cyl)


# Box Plot

ggplot(mtcars, aes(x = cyl, y = hp)) +

  geom_boxplot(fill = "skyblue") +

  labs(title = "Box Plot: Cylinders vs Horsepower")


# Scatter Plot

ggplot(mtcars, aes(x = cyl, y = hp)) +

  geom_jitter(width = 0.1, color = "blue") +

  labs(title = "Scatter Plot: Cylinders vs Horsepower")
```

16. Visualize correlations, split data, and train a Logistic Regression model. Evaluate using accuracy, precision, recall, and F1-score.

```
library(ggplot2)

library(caret)

library(ggcorrplot)
library(pROC)

data(mtcars)

# Correlation Heatmap
cor_matrix <- cor(mtcars)
ggcorrplot(cor_matrix, hc.order = TRUE, type = "lower", lab = TRUE)

# Create binary target
mtcars$mpg_binary <- ifelse(mtcars$mpg > 20, 1, 0)

# Train-test split
set.seed(123)
trainIndex <- createDataPartition(mtcars$mpg_binary, p = 0.8, list = FALSE)
train_data <- mtcars[trainIndex, ]
test_data <- mtcars[-trainIndex, ]

# Logistic Regression
model <- glm(mpg_binary ~ ., data = train_data, family = "binomial")
probs <- predict(model, test_data, type = "response")
pred <- ifelse(probs > 0.5, 1, 0)

# Evaluation
cm <- confusionMatrix(factor(pred), factor(test_data$mpg_binary))
print(cm)

cat("Accuracy: ", cm$overall['Accuracy'], "\n")
cat("Precision: ", cm$byClass['Pos Pred Value'], "\n")
cat("Recall: ", cm$byClass['Sensitivity'], "\n")
cat("F1 Score: ", 2 * ((cm$byClass['Pos Pred Value'] * cm$byClass['Sensitivity']) /
                    (cm$byClass['Pos Pred Value'] + cm$byClass['Sensitivity'])), "\n")
```

17.Identify numerical/categorical variables and handle missing values by imputing with mean, median, or mode. Convert categorical data into factors

```
library(dplyr)


data(mtcars)


# Introduce missing value
mtcars[1, "mpg"] <- NA


# Impute numeric with mean
mtcars$mpg[is.na(mtcars$mpg)] <- mean(mtcars$mpg, na.rm = TRUE)


# Mode function
get_mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}


# Convert & Impute categorical with mode
mtcars$cyl <- as.factor(mtcars$cyl)
mtcars$gear <- as.factor(mtcars$gear)
mtcars$carb <- as.factor(mtcars$carb)


# Print structure
str(mtcars)
```

18 .Visualize mpg vs. wt (Weight) with color for cyl. 5.Create a bar plot showing car counts by gear (Transmission gears).

```
library(ggplot2)


data(mtcars)
```

```
# Scatter plot
ggplot(mtcars, aes(x = wt, y = mpg, color = as.factor(cyl))) +
  geom_point(size = 3) +
  labs(title = "mpg vs wt", color = "Cylinders") +
  theme_minimal()

# Bar plot
ggplot(mtcars, aes(x = as.factor(gear))) +
  geom_bar(fill = "skyblue", color = "black") +
  labs(title = "Car Counts by Gear", x = "Gears", y = "Count") +
  theme_minimal()
```

19. handle missing values by imputing with mean, median, or mode.

```
library(dplyr)

data(mtcars)

# Introduce missing values
mtcars[1, "mpg"] <- NA
mtcars[3, "cyl"] <- NA

# Impute numeric
mtcars$mpg[is.na(mtcars$mpg)] <- mean(mtcars$mpg, na.rm = TRUE)
mtcars$cyl[is.na(mtcars$cyl)] <- median(mtcars$cyl, na.rm = TRUE)

# Mode function
get_mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
```

```
# Impute categorical
mtcars$gear[is.na(mtcars$gear)] <- get_mode(mtcars$gear)

# Show final data
print(mtcars)
```

20. Apply PCA for dimensionality reduction and assess if performance improves for better

```
library(caret)
library(e1071)

data(mtcars)

# Standardize data
scaled_data <- scale(mtcars)

# PCA
pca <- prcomp(scaled_data, center = TRUE, scale. = TRUE)
summary(pca)
screeplot(pca, main = "Scree Plot", col = "blue")

# Keep first 2 components
pca_data <- as.data.frame(pca$x[, 1:2])
pca_data$mpg <- mtcars$mpg

# Train/test split
set.seed(123)
index <- createDataPartition(pca_data$mpg, p = 0.8, list = FALSE)
train <- pca_data[index, ]
test <- pca_data[-index, ]
```

```
# Model on PCA data
model <- glm(mpg ~ ., data = train, family = "gaussian")
pred <- predict(model, test)
rmse <- sqrt(mean((pred - test$mpg)^2))
print(paste("RMSE (PCA model):", rmse))
```