Homework 3

Group-1 Members [Prashanna Raj Pandit, Asha Shah, Nazma Vali Shaik, Hema Sai Paruchuri]

04/20/2025

Question 1: Student Performance Data Analysis

```
library(caret)
library(ggplot2)
library(readxl)
library(dplyr)

# Load the student performance data
student_data <- read.csv("Student_Performance.csv")
student_data$Extracurricular.Activities <- as.factor(student_data$
                                                     Extracurricular.Activities)
#summary(student_data)
```

a) Formulating a Question

How do academic preparation factors (Hours Studied, Previous Scores, Extracurricular Activities, Sleep Hours, and Sample Question Papers Practiced) influence a student's overall academic Performance Index?

- **Predictor Variables**: Hours Studied, Previous Scores, Extracurricular Activities, Sleep Hours, Sample Question Papers Practiced

- **Target Variable**: Performance Index

b) Split your data into a training (70%) and test set (30% test).

```
# Split data - using the correct column names with dots
set.seed(123)
trainIndex <- sample(1:nrow(student_data), 0.7 * nrow(student_data))
train <- student_data[trainIndex, ]
test <- student_data[-trainIndex, ]

# Check dimensions
dim(train)

## [1] 7000    6

dim(test)

## [1] 3000    6
```

c) Training a Multiple Linear Regression Model with Cross-Validation

Train a multiple linear regression model using the training set and a 10-fold cross validation. Use the train function from the caret package, specifying method = "lm" for linear regression. How

would you interpret any 2 of the regression coefficients in the context of the student performance data? Provide the fitted equation.

```
# Define training control for cross-validation
ctrl <- trainControl(method = "cv", number = 10, verboseIter = FALSE)

# Train the linear regression model - using the correct column names with dots
model <- train(
  Performance.Index ~ Hours.Studied + Previous.Scores + Extracurricular.Activities +
    Sleep.Hours + Sample.Question.Papers.Practiced,
  data = train,
  method = "lm",
  trControl = ctrl
)

# Display model summary
summary(model$finalModel)

##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.5864 -1.3743 -0.0113  1.3400  8.3193
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -34.047166   0.153166 -222.29   <2e-16 ***
## Hours.Studied                    2.850817   0.009516  299.57   <2e-16 ***
## Previous.Scores                  1.018021   0.001417  718.59   <2e-16 ***
## Extracurricular.ActivitiesYes    0.571091   0.049192   11.61   <2e-16 ***
## Sleep.Hours                      0.488381   0.014501   33.68   <2e-16 ***
## Sample.Question.Papers.Practiced 0.190325   0.008575   22.20   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.056 on 6994 degrees of freedom
## Multiple R-squared:  0.9885, Adjusted R-squared:  0.9885
## F-statistic: 1.205e+05 on 5 and 6994 DF,  p-value: < 2.2e-16
```

Interpretation of Model Results

Based on the model summary, we can observe the following:

1.**Hours Studied** (Coefficient: β1): For each additional hour spent studying, performance index increases by 2.85 points, holding other factors constant.

Practical meaning: A student who studies 5 more hours than another would be expected to score ≈14.25 points higher.

2.**Previous Scores** (Coefficient: β2):Each 1-point increase in previous test scores is associated with a 1.02-point increase in current performance.

Key insight: Past performance is nearly a 1:1 predictor of current performance in this model.

Regression Equation:

Performance_Index = -34.047166 + 2.850817 * Hours_Studied + 1.018021 * Previous_Scores + 0.571091 * Extracurricular_ActivitiesYes + 0.488381 * Sleep_Hours + 0.190325 * Sample_Question_Papers_Practiced

d) Model Performance on Test Set

Evaluate the performance of your regression model on the test set. Use metrics such as R-squared and Root Mean Squared Error (RMSE) to assess how well the model predicts the target variable. Comment on this.

```
# Make predictions on test set
prediction <- predict(model, test)
errors <- prediction - test$Performance.Index
rmse <- sqrt(mean(errors^2))
r_squared <- cor(prediction, test$Performance.Index)^2

cat("RMSE:", rmse, "\nR-squared:", r_squared)

## RMSE: 1.995368
## R-squared: 0.9892739
```

Comments on performace metrics

1. **RMSE (Root Mean Squared Error):** The model's predictions are, on average, ~2 (1.99) points away from the true Performance Index values. Since the Performance Index ranges from 10 to 100, an RMSE of 2 is extremely low, indicating very precise predictions.

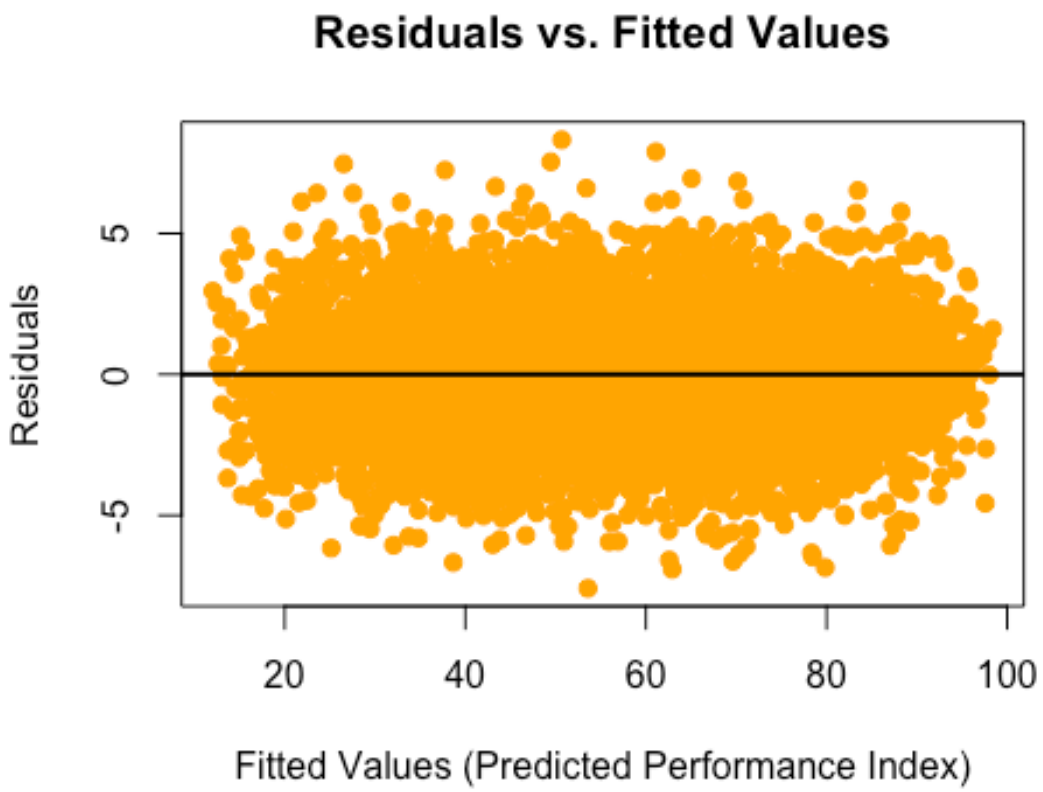
2. **R-squared:** The R^2 value of 0.9893 indicates the model explains 98.93% of variance in Performance Index, demonstrating exceptionally strong predictive performance. All predictors collectively account for nearly all variation in the target variable.

e) Residual Diagnostics

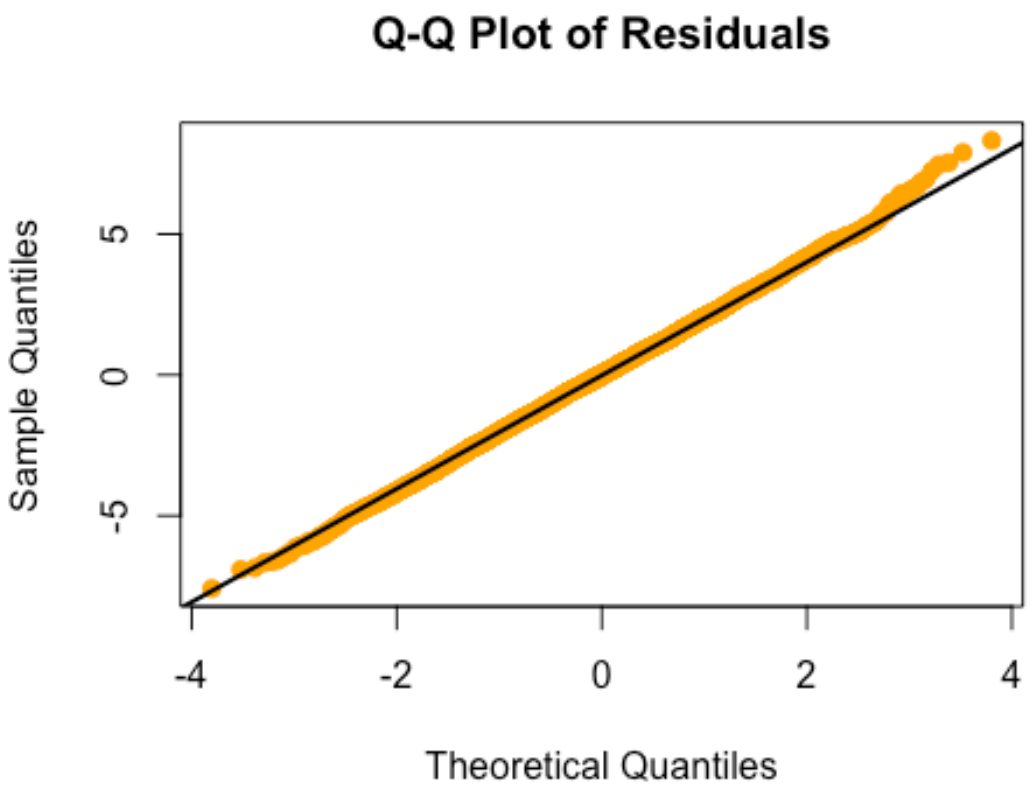
Investigate the residuals from your regression model in c) to check model if model assumptions are satisfied. What do these diagnostics tell you about the model's assumptions and its suitability for the data?

```
#Residuals vs. Fitted Values Plot ( checking linearity & homoscedasticity)
plot(fitted(model), residuals(model),
    xlab = "Fitted Values (Predicted Performance Index)",
    ylab = "Residuals",
    main = "Residuals vs. Fitted Values",
```

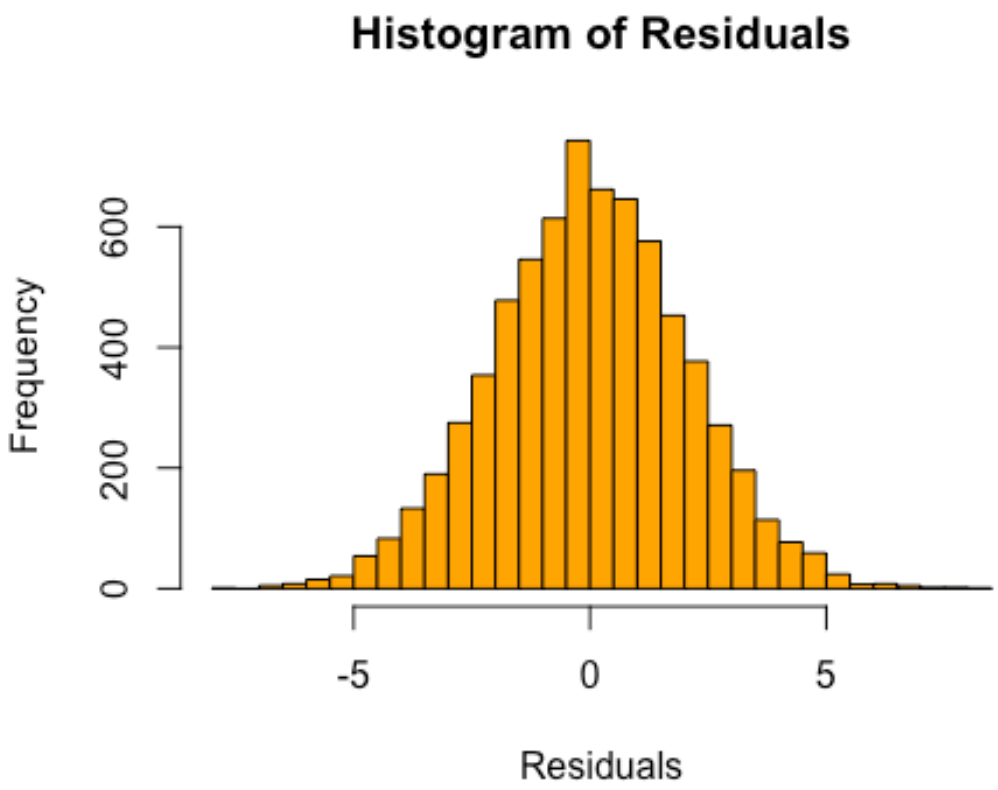
```
    col = "orange", pch = 19)
abline(h = 0, col = "black", lwd = 2)
```



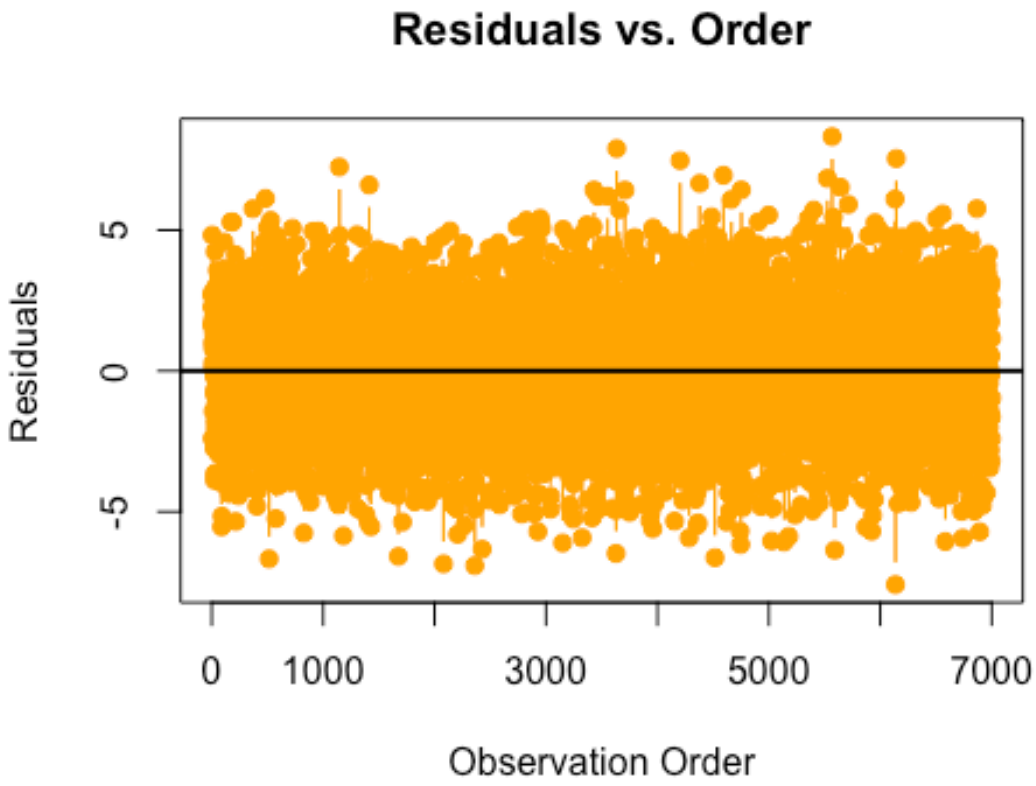

```
#(Normality Check)
qqnorm(residuals(model), col = "orange", pch = 19, main = "Q-Q Plot of Residuals")
qqline(residuals(model), col = "black", lwd = 2)
```



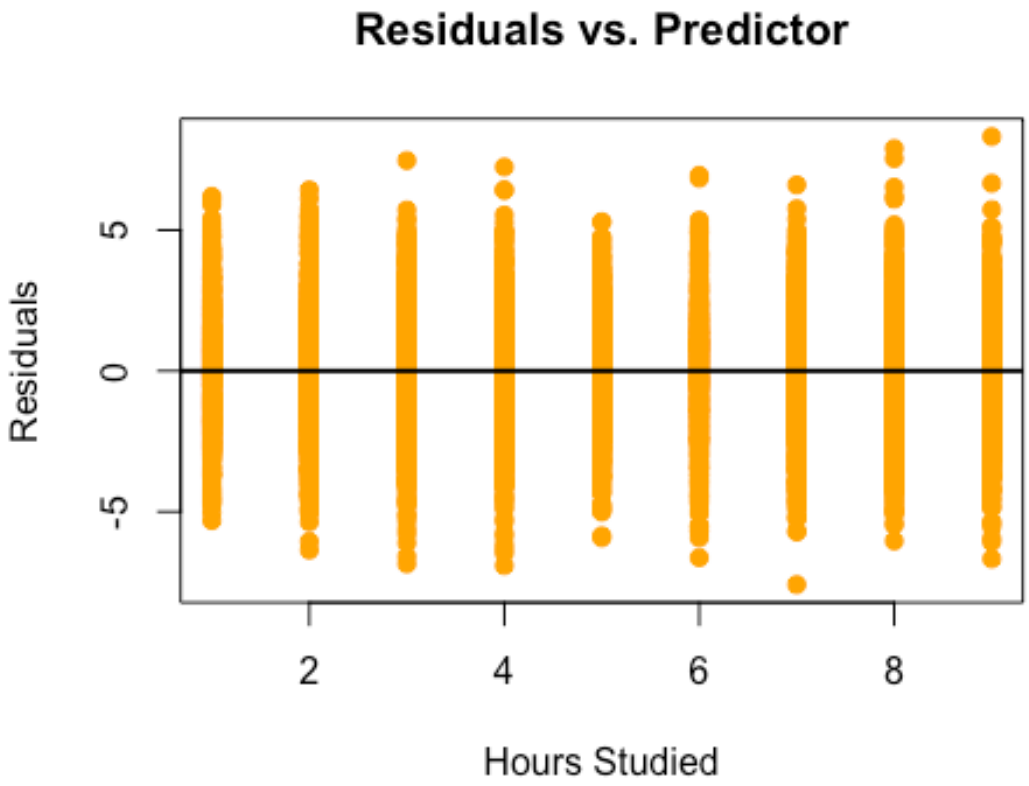

```
hist(residuals(model), main = "Histogram of Residuals",
     xlab = "Residuals", col = "orange", breaks = 30)
```




```r
# Independence check
plot(residuals(model), type = "b", pch = 19, col = "orange",
     xlab = "Observation Order", ylab = "Residuals",
     main = "Residuals vs. Order")
abline(h = 0, col = "black", lwd = 2)
```




```r
#1. Plot of residuals against X (Hours Studied)
plot(train$Hours.Studied, residuals(model$finalModel),
     xlab = "Hours Studied", ylab = "Residuals",
     main = "Residuals vs. Predictor",
     col = "orange", pch = 19)
abline(h = 0, col = "black", lwd = 2)
```



Residual Diagnostics Interpretation

1. **Residuals vs. Fitted Values Plot**
   - Residuals are randomly scattered around 0, indicating **linearity** and **constant variance (homoscedasticity)** assumptions are satisfied.
2. **Q-Q Plot of Residuals**
   - Residuals mostly follow the diagonal line, confirming they are **approximately normally distributed** (minor deviations at the tails).
3. **Histogram of Residuals**
   - The symmetric, bell-shaped histogram confirms the **normality** of residuals.
4. **Residuals vs. Observation Order**
   - Random scatter without trends or clustering suggests **independence of residuals**.
5. **Residuals vs. Predictor (Hours Studied)**
   - Residuals are randomly distributed around 0, confirming **linearity** and **constant variance** for this predictor.

Conclusion
The regression model satisfies all key assumptions (linearity, normality, homoscedasticity, and independence). Minor deviations in the Q-Q plot tails are negligible, and the model is suitable for the data.

f) Proposed Interventions to Improve Student Performance

Based on the insights gained from your analysis, propose interventions that could potentially improve student performance. Are there any violations of assumptions? Suggest remedies for these violations.

Based on the insights gained from the analysis, the following interventions are proposed to enhance student performance:

1. **Increase Study Hours**:
    - Since "Hours Studied" has the strongest positive impact on performance, structured study schedules, peer study groups, and study-time tracking tools should be introduced to encourage students to dedicate more time to studying.
2. **Target Low-Performing Students**:
    - "Previous Scores" are a strong predictor of performance. Students with low prior scores should be identified and offered remedial programs, tutoring, or personalized learning plans.
3. **Promote Extracurricular Activities**:
    - Participation in extracurricular activities has a positive effect on performance. Schools should encourage students to engage in such activities to foster holistic development and improve academic outcomes.
4. **Encourage Healthy Sleep Habits**:
    - Proper sleep is crucial for cognitive functioning and performance. Awareness campaigns and student counseling can help students adopt healthier sleep patterns.
5. **Increase Practice with Sample Papers**:
    - Practicing more sample question papers helps familiarize students with exam formats and reduces test anxiety. Schools should provide additional resources and time for mock exams.

Addressing Minor Assumption Violations

While the model assumptions are largely satisfied, there are minor deviations in the Q-Q plot at the tails, indicating potential outliers or slight non-normality. Remedies include:

- **Handling Outliers**:
    - Investigate the data points contributing to the deviations. If necessary, apply robust regression techniques or data transformations (e.g., log or square root) to minimize their impact.
- **Improving Model Complexity**:
    - If additional non-linear relationships are suspected, consider adding polynomial or interaction terms to capture more complex patterns in the data.

Through these targeted interventions and minor refinements, student performance can be further improved, while maintaining a robust and reliable regression model.

Question 2 – Loan Approval Prediction

```
library(caret)
library(pROC)
library(readxl)
```

a) Load and split data

Split the data set into a training set and a test set (80% training, 30% test).

```
# Read the loan data
loan_data <- read.csv("loan_data.csv")

# Check for missing values and handle them
#summary(loan_data)
colSums(is.na(loan_data))
```

```
##          Loan_ID           Gender          Married       Dependents
##                0                0                0                0
##        Education    Self_Employed   ApplicantIncome CoapplicantIncome
##                0                0                0                0
##       LoanAmount  Loan_Amount_Term    Credit_History     Property_Area
##                0               11               30                0
##      Loan_Status
##                0
```

```
# Simple imputation for missing values (as needed)
# Example for categorical variables:
loan_data$Gender[is.na(loan_data$Gender)] <- "Male"  # Replace with most common value
loan_data$Dependents[is.na(loan_data$Dependents)] <- "0"
loan_data$Self_Employed[is.na(loan_data$Self_Employed)] <- "No"

# Example for numeric variables:
loan_data$Loan_Amount_Term[is.na(loan_data$Loan_Amount_Term)] <-
  median(loan_data$Loan_Amount_Term, na.rm = TRUE)
loan_data$Credit_History[is.na(loan_data$Credit_History)] <-
  median(loan  data$Credit  History, na.rm = TRUE)

# Convert categorical variables to factors
loan_data$Gender <- as.factor(loan_data$Gender)
loan_data$Married <- as.factor(loan_data$Married)
loan_data$Dependents <- as.factor(loan_data$Dependents)
loan_data$Education <- as.factor(loan_data$Education)
loan_data$Self_Employed <- as.factor(loan_data$Self_Employed)
loan_data$Property_Area <- as.factor(loan_data$Property_Area)
loan_data$Loan_Status <- as.factor(loan_data$Loan_Status)

# Split the data - note we're using 80% for training as specified in your code
set.seed(123)
trainIndex <- createDataPartition(loan_data$Loan_Status, p = 0.8, list = FALSE)
train <- loan_data[trainIndex, ]
```

```
test <- loan_data[-trainIndex, ]

# Check dimensions
dim(train)

## [1] 305  13

dim(test)

## [1] 76 13
```

Data Preprocessing Summary

Before model training, we addressed several preprocessing concerns:

1. **Missing Values**: We found missing values in the following variables and imputed them:
    - Gender: Imputed with "Male" (the most frequent category)
    - Dependents: Imputed with "0" (the most frequent category)
    - Self_Employed: Imputed with "No" (the most frequent category)
    - Loan_Amount_Term: Imputed with the median value
    - Credit_History: Imputed with the median value
2. **Data Types**: All categorical variables were converted to factors for proper model handling.
3. **Train-Test Split**: We divided the data into 80% training and 20% testing sets to evaluate model performance on unseen data.

b) Train logistic regression with caret

Train the logistic regression model using the training set with a 10-fold cross-validation to optimize model parameters (Use the caret package).

```
# Define training control
ctrl <- trainControl(
  method = "cv",
  number = 10,
  classProbs = TRUE,
  summaryFunction = twoClassSummary,
  verboseIter = FALSE
)

# Train the logistic regression model
log_model <- train(
  Loan_Status ~ Gender + Married + Dependents + Education + Self_Employed +
    ApplicantIncome + CoapplicantIncome + LoanAmount + Loan_Amount_Term +
    Credit_History + Property_Area,
  data = train,
  method = "glm",
  family = "binomial",
  trControl = ctrl,
```

```
  metric = "ROC"
)

# Display model summary
summary(log_model$finalModel)

##
## Call:
## NULL
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -5.830e+00  2.436e+00  -2.393  0.01672 *
## GenderFemale            2.351e+00  1.449e+00   1.623  0.10464
## GenderMale              1.981e+00  1.433e+00   1.383  0.16679
## MarriedYes              3.800e-01  4.140e-01   0.918  0.35867
## Dependents0             9.600e-01  1.224e+00   0.784  0.43301
## Dependents1             2.181e-01  1.278e+00   0.171  0.86454
## Dependents2             1.478e+00  1.329e+00   1.112  0.26610
## `Dependents3+`          7.157e-01  1.356e+00   0.528  0.59755
## `EducationNot Graduate` -1.051e+00  3.742e-01  -2.808  0.00498 **
## Self_EmployedNo         -4.098e-02  6.595e-01  -0.062  0.95045
## Self_EmployedYes        -2.748e-02  8.163e-01  -0.034  0.97315
## ApplicantIncome         -6.688e-05  1.358e-04  -0.492  0.62247
## CoapplicantIncome       -5.715e-05  5.486e-05  -1.042  0.29749
## LoanAmount              9.231e-03  6.317e-03   1.461  0.14395
## Loan_Amount_Term        -2.924e-03  2.799e-03  -1.045  0.29624
## Credit_History          4.630e+00  6.751e-01   6.858 6.98e-12 ***
## Property_AreaSemiurban   1.135e+00  4.416e-01   2.570  0.01017 *
## Property_AreaUrban       1.180e-01  4.214e-01   0.280  0.77954
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 366.50  on 304  degrees of freedom
## Residual deviance: 233.61  on 287  degrees of freedom
## AIC: 269.61
##
## Number of Fisher Scoring iterations: 5
```

c) Model Evaluation

Evaluate the model on the test set using appropriate metrics ; Accuracy, Sensitivity, Specificity, and AUC (Area Under the ROC Curve). Analyze the confusion matrix to understand the model's performance in predicting an individual's loan status. Interpret each of these metrics in terms of the data and the model.

```
# Make predictions
pred_class <- predict(log_model, test)
pred_prob <- predict(log_model, test, type = "prob")
```

```r
# Confusion matrix
conf_matrix <- confusionMatrix(pred_class, test$Loan_Status, positive = "Y")
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction  N  Y
##          N 11  0
##          Y 11 54
##
##               Accuracy : 0.8553
##                 95% CI : (0.7558, 0.9255)
##    No Information Rate : 0.7105
##    P-Value [Acc > NIR] : 0.002508
##
##                  Kappa : 0.587
##
##  Mcnemar's Test P-Value : 0.002569
##
##             Sensitivity : 1.0000
##             Specificity : 0.5000
##          Pos Pred Value : 0.8308
##          Neg Pred Value : 1.0000
##              Prevalence : 0.7105
##          Detection Rate : 0.7105
##    Detection Prevalence : 0.8553
##       Balanced Accuracy : 0.7500
##
##        'Positive' Class : Y
##
```

```r
# ROC and AUC
roc_obj <- roc(response = test$Loan_Status, predictor = pred_prob[, "Y"])
```

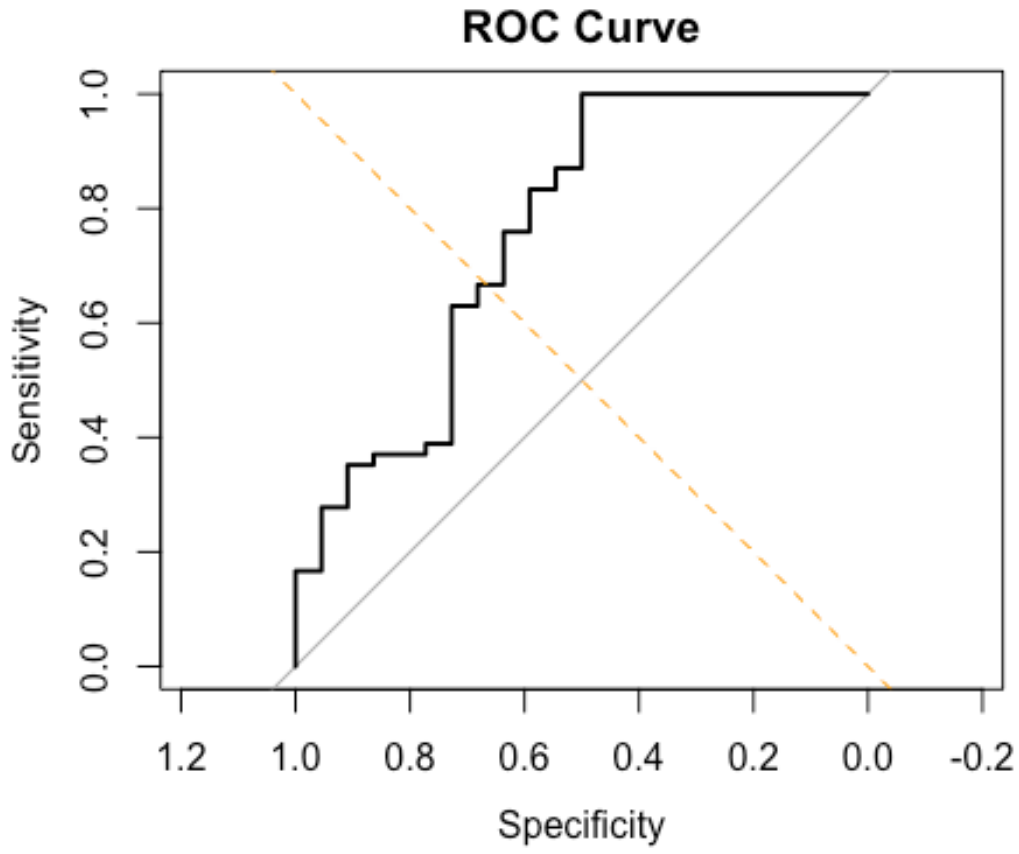
```
## Setting levels: control = N, case = Y
```

```
## Setting direction: controls < cases
```

```r
auc_value <- auc(roc_obj)
cat("AUC:", auc_value, "\n")
```

```
## AUC: 0.7584175
```

```r
# Plot ROC curve
plot(roc_obj, main = "ROC Curve")
abline(a = 0, b = 1, lty = 2, col = "orange")
```



###

Confusion Matrix Metrics Accuracy: 85.53%
- The model correctly classified the majority of cases.
Sensitivity (Recall for Approvals): 100%
- All actual approvals were correctly identified.
Specificity (Recall for Denials): 50%
- The model struggles to correctly identify loan denials.
AUC (Area Under the Curve): 75.84%
- Good ability to distinguish between approved and denied loans.

ROC Curve

- The ROC curve (Receiver Operating Characteristic) plots sensitivity against 1-specificity at various decision thresholds.
- The curve being above the diagonal line indicates the model performs better than random guessing.