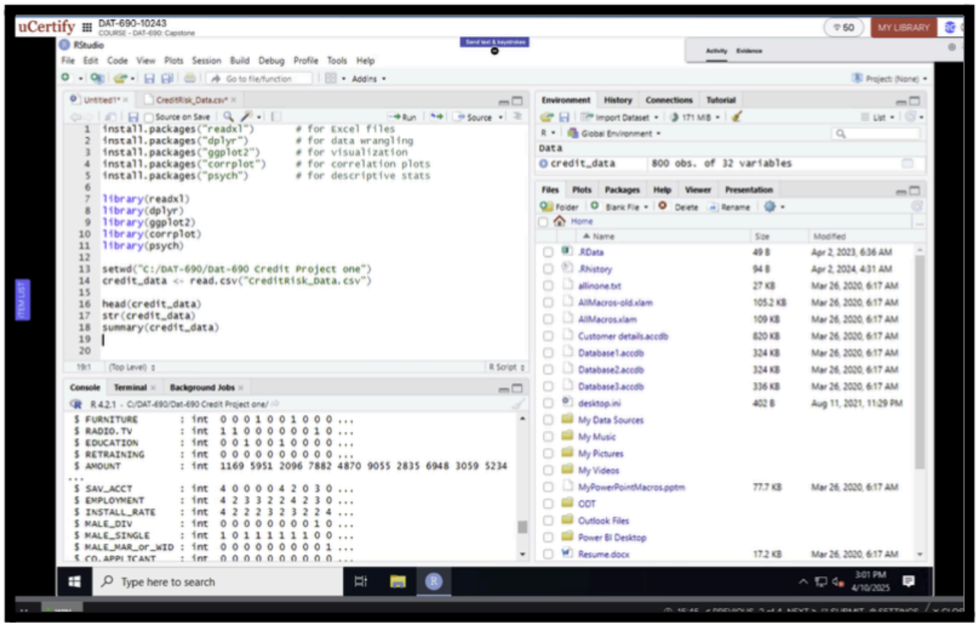
8-1 Programming Revision: Strengthening The Code

```
# Section 1: Load Libraries

# Install necessary packages if not already installed
packages <- c("readr", "dplyr", "ggplot2", "corrplot", "psych", "caret", "pROC")
installed <- packages %in% rownames(installed.packages())
if (any(!installed)) install.packages(packages[!installed])

# Load required libraries
library(readr)
library(dplyr)
library(ggplot2)
library(corrplot)
library(psych)
library(caret)
library(pROC)

# Section 2: Load and Inspect Data

setwd("C:/DAT-690/DAT-690 Credit Project one")
credit_data <- read.csv("CreditRisk_Data.csv")

# View initial structure and summary
head(credit_data)
str(credit_data)
summary(credit_data)
```

Figure 1:The functions head(), str(), and summary() functions



Section 3: Data Cleaning

Check and remove duplicates
sum(duplicated(credit_data)) # Should be 0
credit_data <- credit_data[!duplicated(credit_data),]

# Check for missing values
colSums(is.na(credit_data))

Impute
credit_data$AMOUNT[is.na(credit_data$AMOUNT)] <- median(credit_data$AMOUNT, na.rm = TRUE)

Section 4: Data Exploration

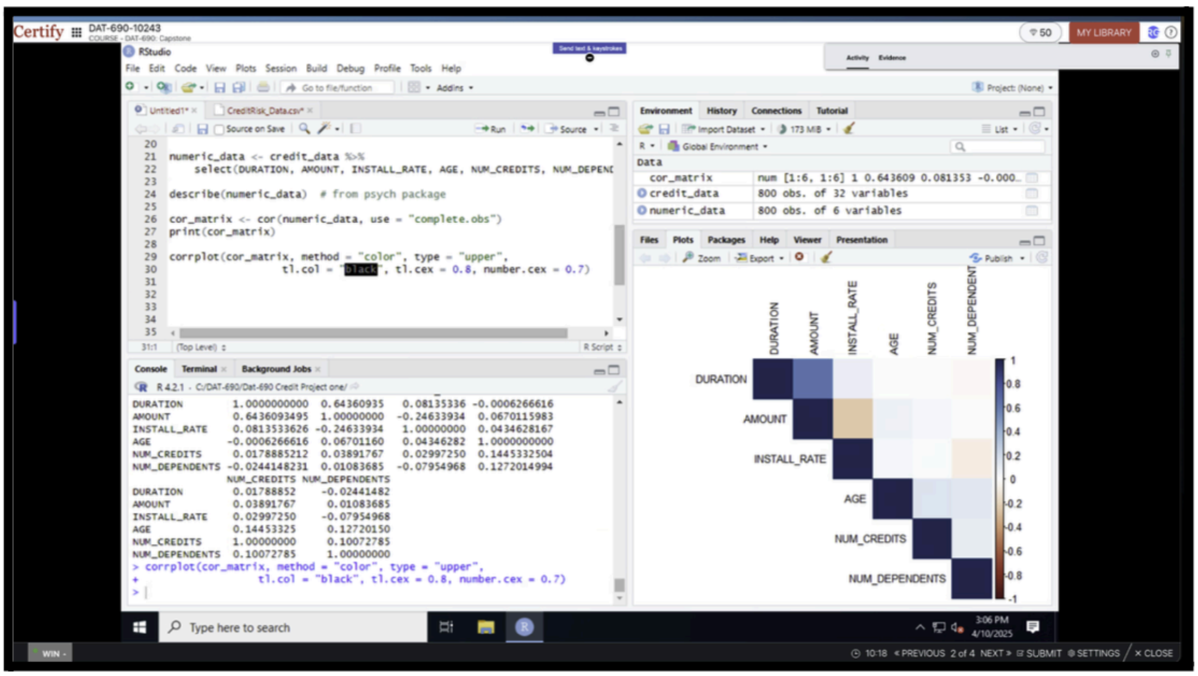
# Summary stats for selected features
summary(credit_data$AMOUNT)
summary(credit_data$AGE)

# Use psych package for detailed stats
numeric_data <- credit_data %>%
  select(DURATION, AMOUNT, INSTALL_RATE, AGE, NUM_CREDITS, NUM_DEPENDENTS)

describe(numeric_data)

# Correlation matrix and heatmap
cor_matrix <- cor(numeric_data, use = "complete.obs")
corrplot(cor_matrix, method = "color", type = "upper", tl.col = "black", tl.cex = 0.8, number.cex = 0.7)

Figure 3: Heatmap

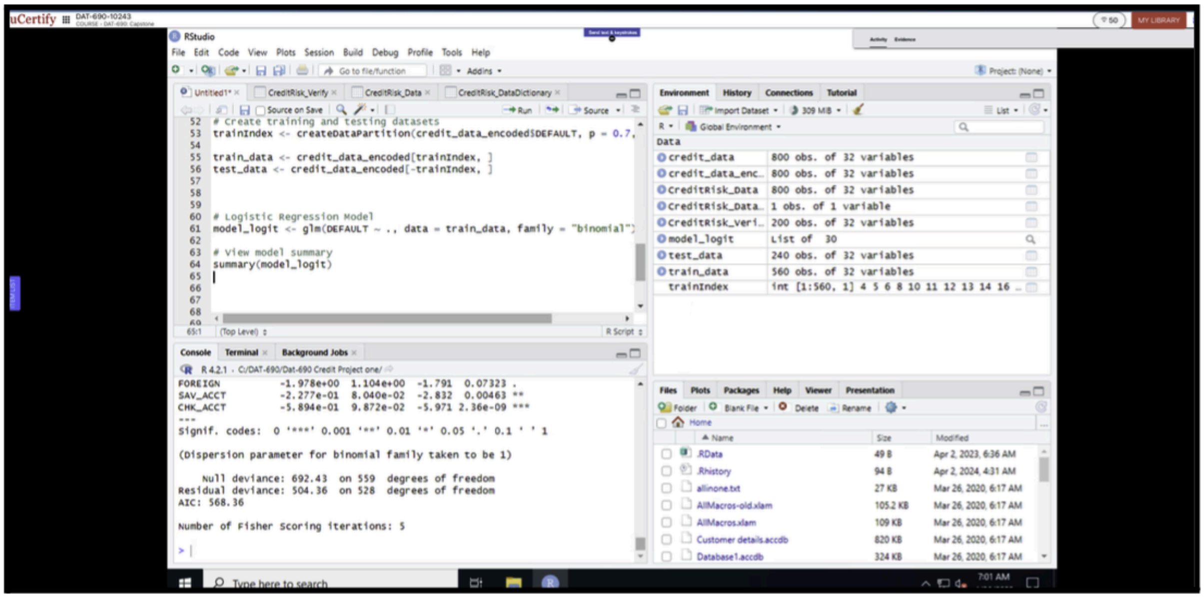


Section 5: Feature Engineering

Convert relevant categorical columns to factors
credit_data$SAV_ACCT <- as.factor(credit_data$SAV_ACCT)
credit_data$CHK_ACCT <- as.factor(credit_data$CHK_ACCT)
credit_data$FOREIGN <- as.factor(credit_data$FOREIGN)

```
# Scale numeric columns
credit_data$AGE <- scale(credit_data$AGE)
credit_data$AMOUNT <- scale(credit_data$AMOUNT)


# Section 6: Data Splitting

# Note: Model will be trained on 70% of the data and tested on 30% to support
validation and avoid overfitting. This helps estimate model generalizability
and serves as a baseline for future performance comparison.

set.seed(123)
trainIndex <- createDataPartition(credit_data$DEFAULT, p = 0.7, list = FALSE)
train_data <- credit_data[trainIndex, ]
test_data <- credit_data[-trainIndex, ]


# Section 7: Logistic Regression Modeling

# Assumption check: Logistic regression assumes a linear relationship between
the log-odds of the outcome and predictor variables. Variables like AGE and
AMOUNT are standardized, and categorical variables are encoded as factors.

# This model is designed to answer the business question:
 Can we predict loan default using applicant features?
The output (predicted default) aligns with this goal.

# Future-proofing: To accommodate future retraining, this section can be reused
with updated data sources. Parameters and predictors can be easily modified.


model_logit <- glm(DEFAULT ~ SAV_ACCT + CHK_ACCT + AGE + AMOUNT +
FOREIGN,
                data = train_data,
                family = "binomial")

summary(model_logit)
```

Figure 1: Model summary coefficients and p-values.



Section 8: Prediction and Evaluation

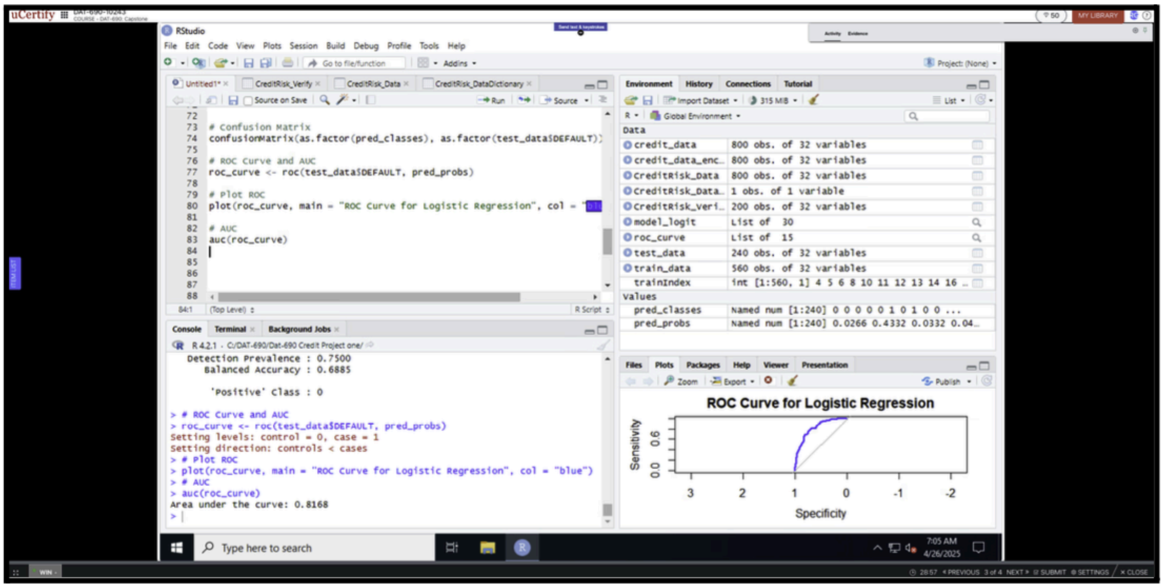
# Generate predicted probabilities
pred_probs <- predict(model_logit, newdata = test_data, type = "response")
pred_classes <- ifelse(pred_probs > 0.5, 1, 0)

# Confusion Matrix
conf_matrix <- table(Predicted = pred_classes, Actual = test_data$DEFAULT)
print(conf_matrix)

# ROC Curve and AUC
roc_curve <- roc(test_data$DEFAULT, pred_probs)
plot(roc_curve, main = "ROC Curve for Logistic Regression", col = "blue")
auc(roc_curve)

# Detailed metrics
confusionMatrix(as.factor(pred_classes), as.factor(test_data$DEFAULT))

Figure 3: ROC curve plot and AUC score result



The ROC and confusion matrix evaluate model performance.
Current baseline: AUC of 0.8168, balanced accuracy of ~0.689.
Future predictions can be compared to this benchmark.

Application to new data: The model will later be applied to creditRisk_Verify.csv and validated to assess generalization to unseen applicants.

Maintenance Plan:The model will undergo regular validation and retraining as new applicant data becomes available. Future versions may use threshold tuning or resampling (e.g., SMOTE) to improve performance on imbalanced classes.

Communication Plan: Model results and metrics (AUC, confusion matrix, etc.) will be shared through stakeholder-facing dashboards and reporting tools. Plots and summaries will accompany each update.

Section 9: Final Evaluation Using Verification Dataset

Apply the trained model to the external verification dataset. This step simulates applying the model to unseen, real-world applicant data. It helps assess generalization and robustness outside the training/testing pipeline.

pred_probs <- predict(model_logit, newdata = creditRisk_verify, type = "response")
pred_classes <- ifelse(pred_probs > 0.5, 1, 0)

# Check distribution of actual defaults in the verification dataset
table(creditRisk_verify$DEFAULT)

# Create confusion matrix and performance metrics on the verification data
confusionMatrix(as.factor(pred_classes), as.factor(creditRisk_verify$DEFAULT))

Figure: Evaluation results on verification data