Untitled

Nikhil Prema Chandra Rao

2024-09-28

1. Introduction

The goal of this analysis is to develop a logistic regression model to detect genuine and forged banknotes based on wavelet-transformed features extracted from images of the banknotes. The dataset (`A6DATA.csv`) contains 1,372 observations of 5 variables. The features include variance, skewness, kurtosis, and entropy of the wavelet-transformed images, and the target variable is the class of the specimen (genuine or forged).

2. Data Loading and Initial Exploration

We load the dataset and examine its structure to understand the features and their types.

```r
# Load necessary libraries
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(ggplot2)

# Step 1: Data Loading and Initial Exploration
# Read the CSV data
data <- read.csv("A6DATA.csv")

# View the structure of the data
str(data)
```

```
## 'data.frame':    1372 obs. of  5 variables:
##  $ V1: num  3.622 4.546 3.866 3.457 0.329 ...
##  $ V2: num  8.67 8.17 -2.64 9.52 -4.46 ...
##  $ V3: num  -2.81 -2.46 1.92 -4.01 4.57 ...
##  $ V4: num  -0.447 -1.462 0.106 -3.594 -0.989 ...
##  $ V5: int  0 0 0 0 0 0 0 0 0 0 ...
```

output reveals that the dataset contains 1,372 observations with 5 variables, where V1 to V4 are numeric features, and V5 is an integer representing the class (0 for forged and 1 for genuine).

```r
# Check summary statistics of the data
summary(data)
```

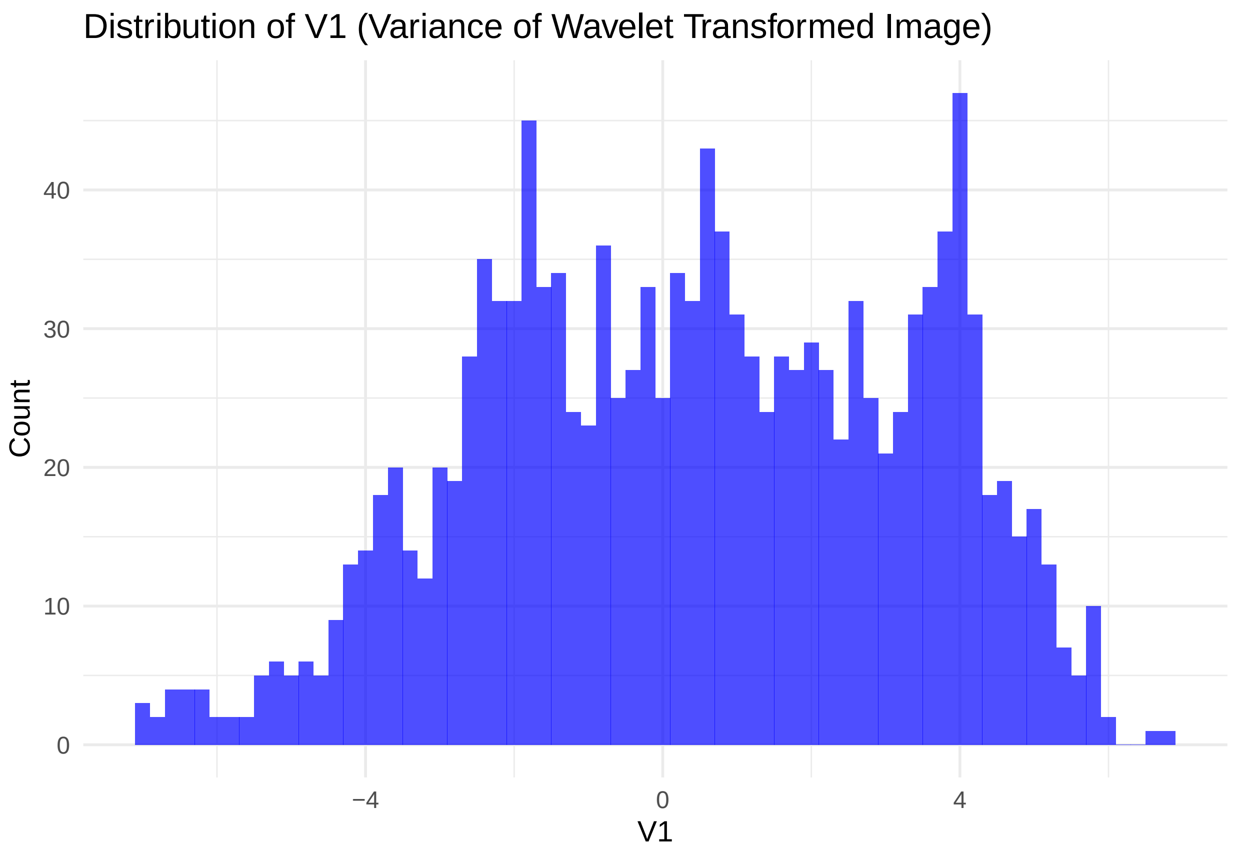
```
##        V1                V2                V3                V4
##  Min.   :-7.0421   Min.   :-13.773   Min.   :-5.2861   Min.   :-8.5482
##  1st Qu.:-1.7730   1st Qu.: -1.708   1st Qu.:-1.5750   1st Qu.:-2.4135
##  Median : 0.4962   Median :  2.320   Median : 0.6166   Median :-0.5867
##  Mean   : 0.4337   Mean   :  1.922   Mean   : 1.3976   Mean   :-1.1917
##  3rd Qu.: 2.8215   3rd Qu.:  6.815   3rd Qu.: 3.1793   3rd Qu.: 0.3948
##  Max.   : 6.8248   Max.   : 12.952   Max.   :17.9274   Max.   : 2.4495
##        V5
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4446
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

The output provides descriptive statistics for each variable, including minimum, maximum, mean, and quartiles, indicating the range and distribution of the data.

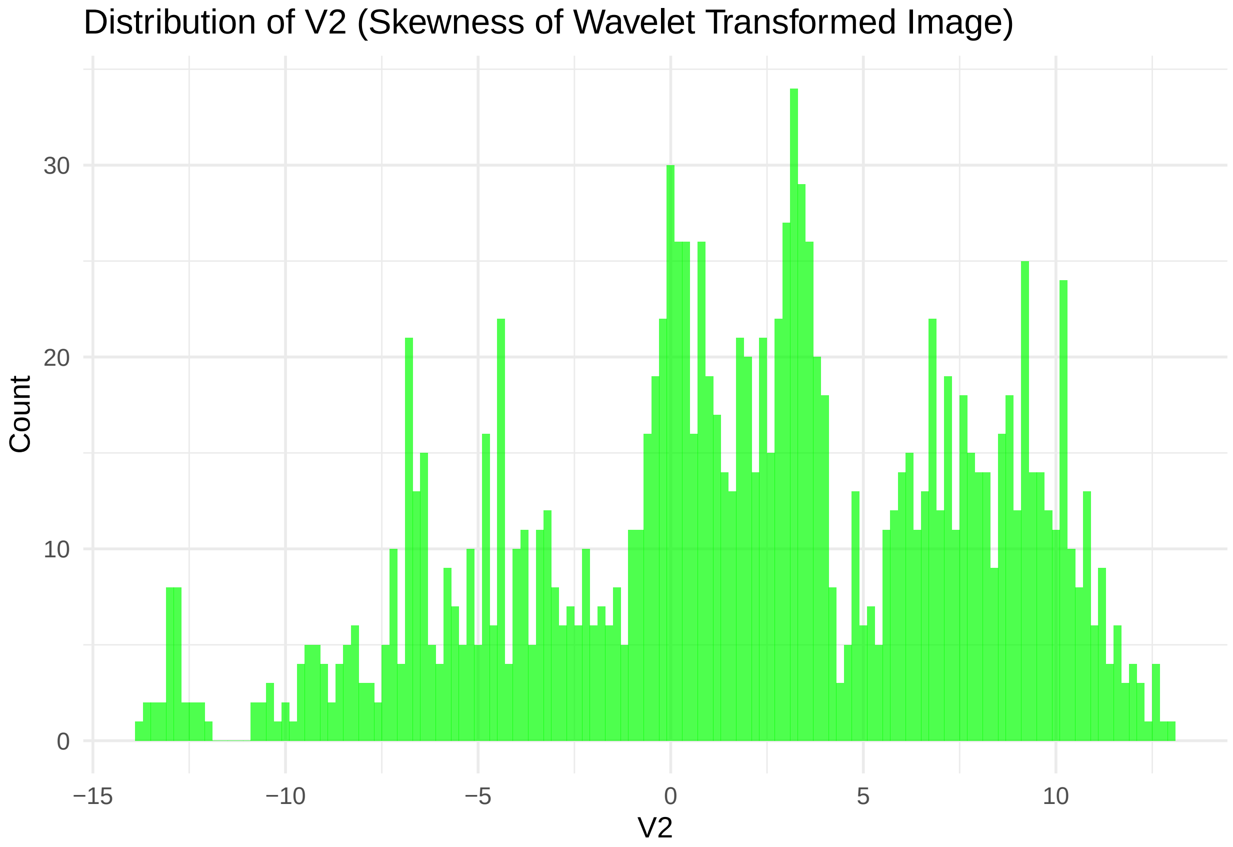
3. Visualizing the Distribution of Variables

We visualize the distribution of key variables to understand their behavior.

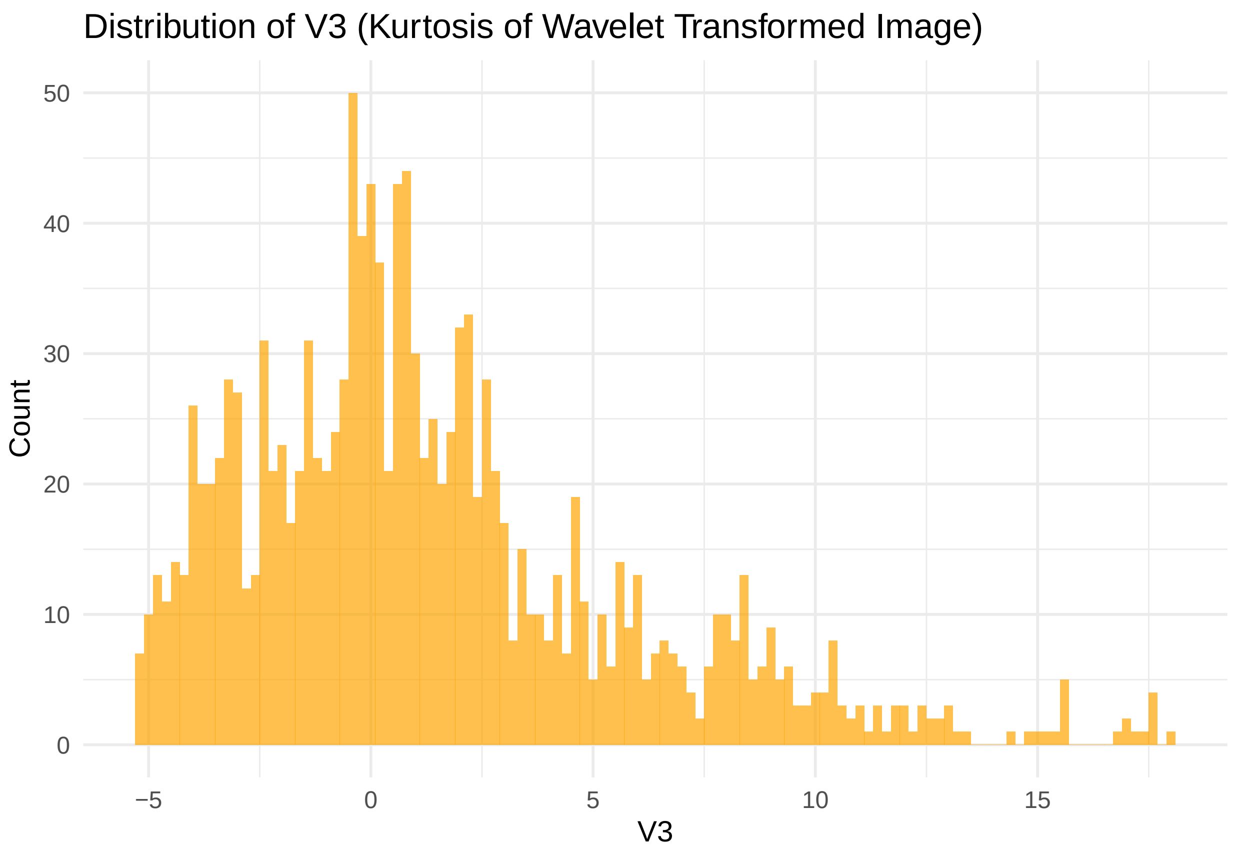
```r
# Visualizing the distribution of the variables
ggplot(data, aes(x = V1)) +
  geom_histogram(binwidth = 0.2, fill = "blue", alpha = 0.7) +
  labs(title = "Distribution of V1 (Variance of Wavelet Transformed Image)",
       x = "V1", y = "Count") +
  theme_minimal()
```



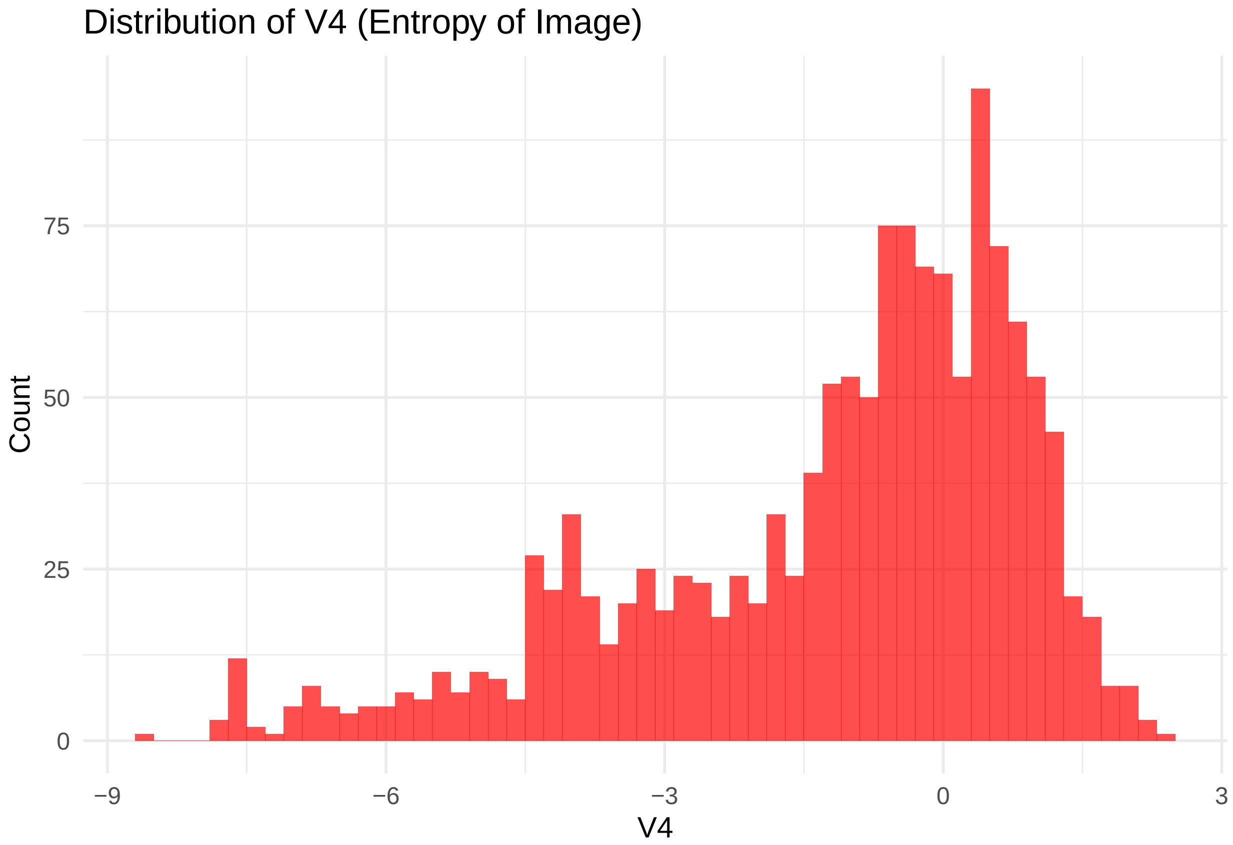
```r
ggplot(data, aes(x = V2)) +
  geom_histogram(binwidth = 0.2, fill = "green", alpha = 0.7) +
  labs(title = "Distribution of V2 (Skewness of Wavelet Transformed Image)",
       x = "V2", y = "Count") +
  theme_minimal()
```



```r
ggplot(data, aes(x = V3)) +
  geom_histogram(binwidth = 0.2, fill = "orange", alpha = 0.7) +
  labs(title = "Distribution of V3 (Kurtosis of Wavelet Transformed Image)",
       x = "V3", y = "Count") +
  theme_minimal()
```



```r
ggplot(data, aes(x = V4)) +
  geom_histogram(binwidth = 0.2, fill = "red", alpha = 0.7) +
  labs(title = "Distribution of V4 (Entropy of Image)",
       x = "V4", y = "Count") +
  theme_minimal()
```



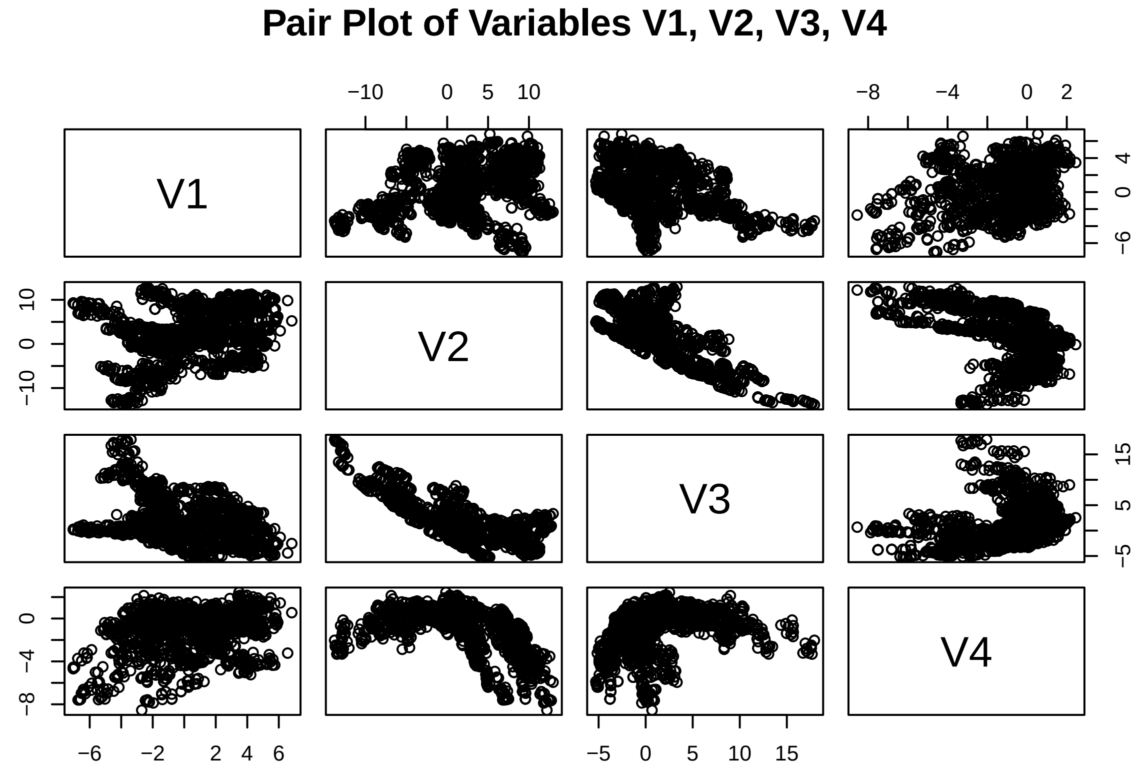
These histograms represent the distributions of three features derived from wavelet-transformed images:

1. **V1**: Distribution of the variance, showing a symmetrical spread around 0, indicating balanced variance.
2. **V2**: Distribution of skewness, where values are more spread out with a noticeable rightward skew, reflecting asymmetry in the data.
3. **V3**: Distribution of kurtosis, which is heavily right-skewed, indicating the presence of outliers or heavy tails in the data.
4. **V3**: Distribution of V4 Entropy of Image, show how frequently certain entropy levels occur.

4. Checking Relationships Between Variables

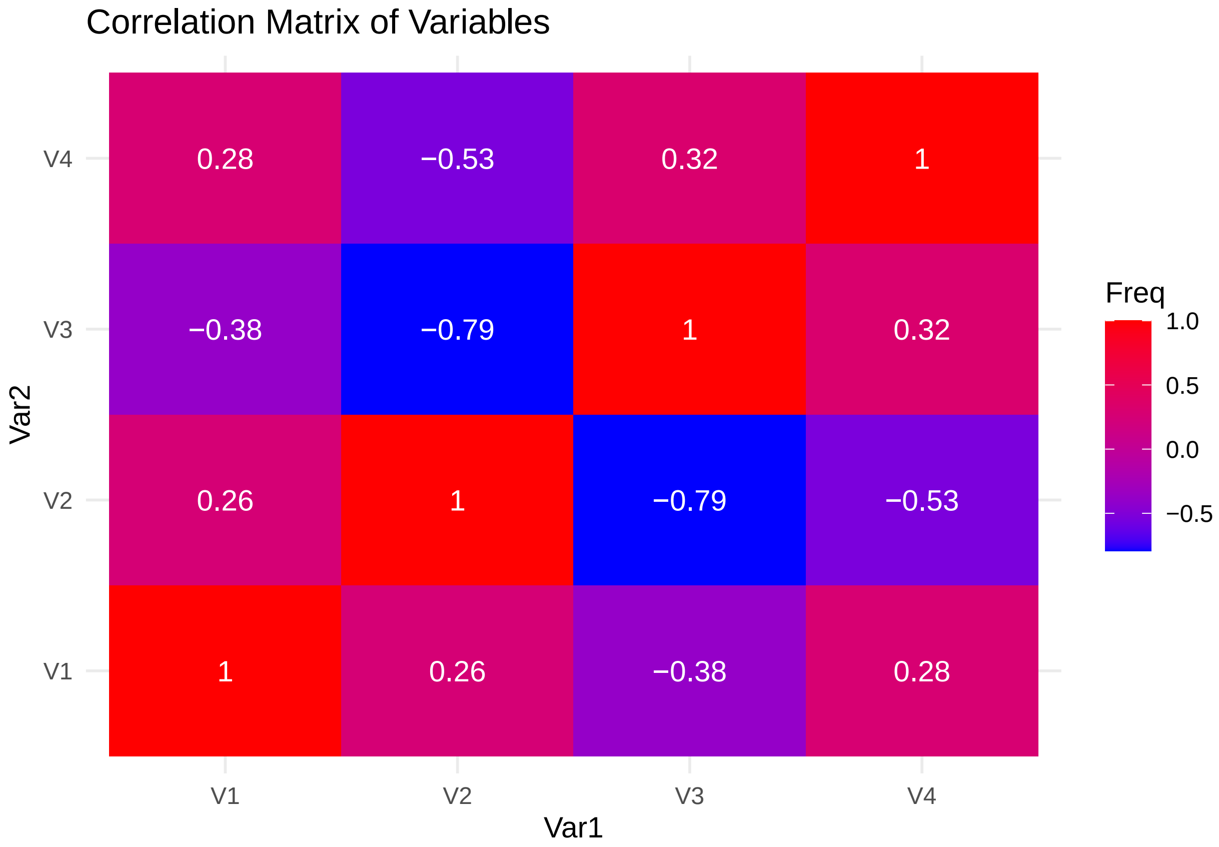
To examine relationships between the variables, we use pair plots and correlation heatmaps.

```r
# Visualizing the relationship between variables
pairs(data[,1:4], main = "Pair Plot of Variables V1, V2, V3, V4")
```



The image is a pair plot showing scatter plot between all the combinations of variables V1, V2, V3, and V4. It helps to visualize the relationships, trends, and possible correlations among these variables, where the diagonal element are the variables.

```r
# Correlation heatmap to check multicollinearity
cor_matrix <- round(cor(data[,1:4]), 2)
ggplot(data = as.data.frame(as.table(cor_matrix)), aes(Var1, Var2, fill = Freq)) +
  geom_tile() +
  geom_text(aes(label = Freq), color = "white") +
  scale_fill_gradient(low = "blue", high = "red") +
  labs(title = "Correlation Matrix of Variables") +
  theme_minimal()
```



This is a heatmap showing the correlation coefficients between the variables V1, V2, V3, and V4. Positive correlations are shown in red, while negative correlations appear in blue, helps to identify the strength and direction of relationships.

5. Data Partitioning

We split the data into training and testing sets (60% and 40%, respectively) to ensure reproducibility and validate the model.

```
# Step 2: Data Partitioning
# Set seed for reproducibility
set.seed(681)

# Split data into training (60%) and testing (40%)
trainIndex <- createDataPartition(data$V5, p = 0.6, list = FALSE)
trainData <- data[trainIndex, ]
testData <- data[-trainIndex, ]

# Check the number of genuine (1) and forged (0) specimens in training and testing sets
cat("Training Set Specimen Counts:\n")
```

```
## Training Set Specimen Counts:
```

```
print(table(trainData$V5))
```

```
## 
##   0   1 
## 456 368
```

```
cat("Testing Set Specimen Counts:\n")
```

```
## Testing Set Specimen Counts:
```

```
print(table(testData$V5))
```

```
## 
##   0   1 
## 306 242
```

The printed counts show that the training set consists of 456 forged and 368 genuine specimens, while the testing set has 306 forged and 242 genuine specimens.

6. Logistic Regression Model Development

We develop the logistic regression model, starting with all variables and using stepwise selection to optimize the model.

```
model <- glm(V5 ~ V1 + V2 + V3 + V4, data = trainData, family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(model)
```

```
## 
## Call:
## glm(formula = V5 ~ V1 + V2 + V3 + V4, family = binomial, data = trainData)
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)   7.5848     2.0453   3.708 0.000209 ***
## V1           -8.4394     2.3157  -3.644 0.000268 ***
## V2           -4.2266     1.1648  -3.628 0.000285 ***
## V3           -5.3989     1.4888  -3.626 0.000287 ***
## V4           -0.4134     0.4371  -0.946 0.344317    
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 1132.891  on 823  degrees of freedom
## Residual deviance:   29.278  on 819  degrees of freedom
## AIC: 39.278
## 
## Number of Fisher Scoring iterations: 12
```

Model Summary

The summary of the model results from the logistic regression analysis that links between the predictor variables (V1, V2, V3, V4) and the response variable (V5), which determine whether the specimen is authentic or forged.

1. Coefficients

The output contains the estimated coefficients for each predictor variable, together with the intercept:

- **(Intercept)**: The intercept value (7.5848) represents the log-odds of the outcome when all predictors are zero.
- **V1, V2, V3**: Each coefficient for these variables is negative, implying that as the values of V1, V2, and V3 rise, the log-odds of the outcome being genuine (V5 = 1) fall. This shows an inverse relationship: higher values in these features indicate a greater possibility that the specimen is forged (V5 = 0).
- **V4**: The coefficient for V4 (-0.4134) is not statistically significant (p-value = 0.344), demonstrating that this variable does not contribute meaningfully to determining if a specimen is genuine.

2. Statistical Significance

The z-value and corresponding p-value are used to determine the statistical significance of each coefficient:

- The p-values for V1 (0.000209), V2 (0.000285), and V3 (0.000287) are all below the customary threshold of 0.05, suggesting high statistical significance. This indicates that we can confidently state these variables significantly influence the likelihood of the outcome.
- V4's p-value of 0.344 indicates that it has no significant effect on the result, leading to its exclusion from the final model.

3. Model Fit Statistics

The summary also reports the null deviance and residual deviance, which are measures of the model's fit to the data:

- **Null Deviance (1132.891)**: This represents the fit of a model that only includes the intercept. A smaller deviance suggests a better fit.
- **Residual Deviance (29.278)**: This measures the model's fit with the predictors included. The decrease in the deviance from the null to residual indicates that the model with predictors has significantly better fit than the null model.

4. Akaike Information Criterion (AIC)

The AIC value (39.278) is useful for assessing the quality of various stats models; lower AIC values indicate a better fit when comparing models.

Conclusion

Overall, the `summary(model)` output shows that V1, V2, and V3 are important predictors of whether a banknote is genuine or forged, providing valuable insights for model interpretation and further analysis.

```r
# Perform stepwise selection to find the best model
final_model <- step(model)
```

```
## Start:  AIC=39.28
## V5 ~ V1 + V2 + V3 + V4

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##        Df Deviance    AIC
## - V4    1    30.18  38.18
## <none>       29.28  39.28
## - V2    1   367.55 375.55
## - V3    1   438.74 446.74
## - V1    1   719.34 727.34

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=38.18
## V5 ~ V1 + V2 + V3
##
##        Df Deviance    AIC
## <none>       30.18  38.18
## - V3    1   440.47 446.47
## - V2    1   480.29 486.29
## - V1    1   866.27 872.27
```

The final model selection process starts with an AIC of 39.28, including all predictors (V1, V2, V3, V4). Stepwise selection removes V4, resulting in a new model with an AIC of 38.18, indicating a better fit. The warnings about fitted probabilities being numerically 0 or 1 suggest that the model may have encountered perfect separation for some observations, which can impact stability and interpretation.

```r
# Display the final model
summary(final_model)
```

```
##
## Call:
## glm(formula = V5 ~ V1 + V2 + V3, family = binomial, data = trainData)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    7.503      1.987   3.776 0.000159 ***
## V1            -7.911      2.078  -3.806 0.000141 ***
## V2            -3.859      1.011  -3.818 0.000134 ***
## V3            -4.974      1.306  -3.810 0.000139 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1132.891  on 823  degrees of freedom
## Residual deviance:   30.177  on 820  degrees of freedom
## AIC: 38.177
##
## Number of Fisher Scoring iterations: 12
```

7. Model Evaluation

Evaluate the model on both training and testing data. We use confusion matrices and calculate misclassification errors.

```
# Step 4: Model Evaluation

# Predict on training and testing data using the final model
trainPred <- predict(final_model, newdata = trainData, type = "response")
testPred <- predict(final_model, newdata = testData, type = "response")

# Convert probabilities to class labels (0 or 1)
trainPredClass <- ifelse(trainPred > 0.5, 1, 0)
testPredClass <- ifelse(testPred > 0.5, 1, 0)

# Confusion matrix for training data with labels
trainConfMat <- table(Predicted = trainPredClass, Actual = trainData$V5)
cat("Training Confusion Matrix:\n")
```

```
## Training Confusion Matrix:
```

```
print(trainConfMat)
```

```
##          Actual
## Predicted   0   1
##         0 451   4
##         1   5 364
```

```
# Confusion matrix for testing data with labels
testConfMat <- table(Predicted = testPredClass, Actual = testData$V5)
cat("Testing Confusion Matrix:\n")
```

```
## Testing Confusion Matrix:
```

```
print(testConfMat)
```

```
##          Actual
## Predicted   0   1
##         0 303   3
##         1   3 239
```

The training confusion matrix shows 451 true negatives and 364 true positives, while the testing confusion matrix indicates 303 true negatives and 239 true positives, reflecting the model's predictive performance.

```
# Misclassification error for training and testing sets
trainError <- mean(trainPredClass != trainData$V5)
testError <- mean(testPredClass != testData$V5)

# Print misclassification errors
cat("Training Error: ", trainError, "\n")
```

```
## Training Error:  0.01092233
```

```
cat("Testing Error: ", testError, "\n")
```

```
## Testing Error:  0.01094891
```

The calculated misclassification errors reveal a training error of approximately 1.09% and a testing error of about 1.09%, indicating high model accuracy.

8. Additional Model Performance Metrics

We calculate the accuracy, sensitivity, and specificity for both training and testing datasets.

```
trainAccuracy <- sum(diag(trainConfMat)) / sum(trainConfMat)
trainSensitivity <- trainConfMat[2, 2] / sum(trainConfMat[, 2])  # True Positives / (True Positives + F
trainSpecificity <- trainConfMat[1, 1] / sum(trainConfMat[, 1])  # True Negatives / (True Negatives + F

cat("Training Accuracy: ", trainAccuracy, "\n")
```

```
## Training Accuracy:  0.9890777
```

```
cat("Training Sensitivity: ", trainSensitivity, "\n")
```

```
## Training Sensitivity:  0.9891304
```

```
cat("Training Specificity: ", trainSpecificity, "\n")
```

```
## Training Specificity:  0.9890351
```

The calculated accuracy, sensitivity, and specificity values for both the training and testing sets show that the model effectively distinguishes between genuine and forged specimens with accuracy around 98.9%.

```
# Accuracy, Sensitivity, and Specificity for Testing Set
testAccuracy <- sum(diag(testConfMat)) / sum(testConfMat)
testSensitivity <- testConfMat[2, 2] / sum(testConfMat[, 2])
testSpecificity <- testConfMat[1, 1] / sum(testConfMat[, 1])

cat("Testing Accuracy: ", testAccuracy, "\n")
```

```
## Testing Accuracy:  0.9890511
```

```
cat("Testing Sensitivity: ", testSensitivity, "\n")
```

```
## Testing Sensitivity:  0.9876033
```

```
cat("Testing Specificity: ", testSpecificity, "\n")
```

```
## Testing Specificity:  0.9901961
```

The AUC values of 0.9997795 for the training set and 0.9997029 for the testing set, derived from the ROC curve outputs, indicate near-perfect classification performance by the model.

9. ROC Curves and AUC

We generate ROC curves and calculate AUC to assess the model's ability to distinguish between classes.

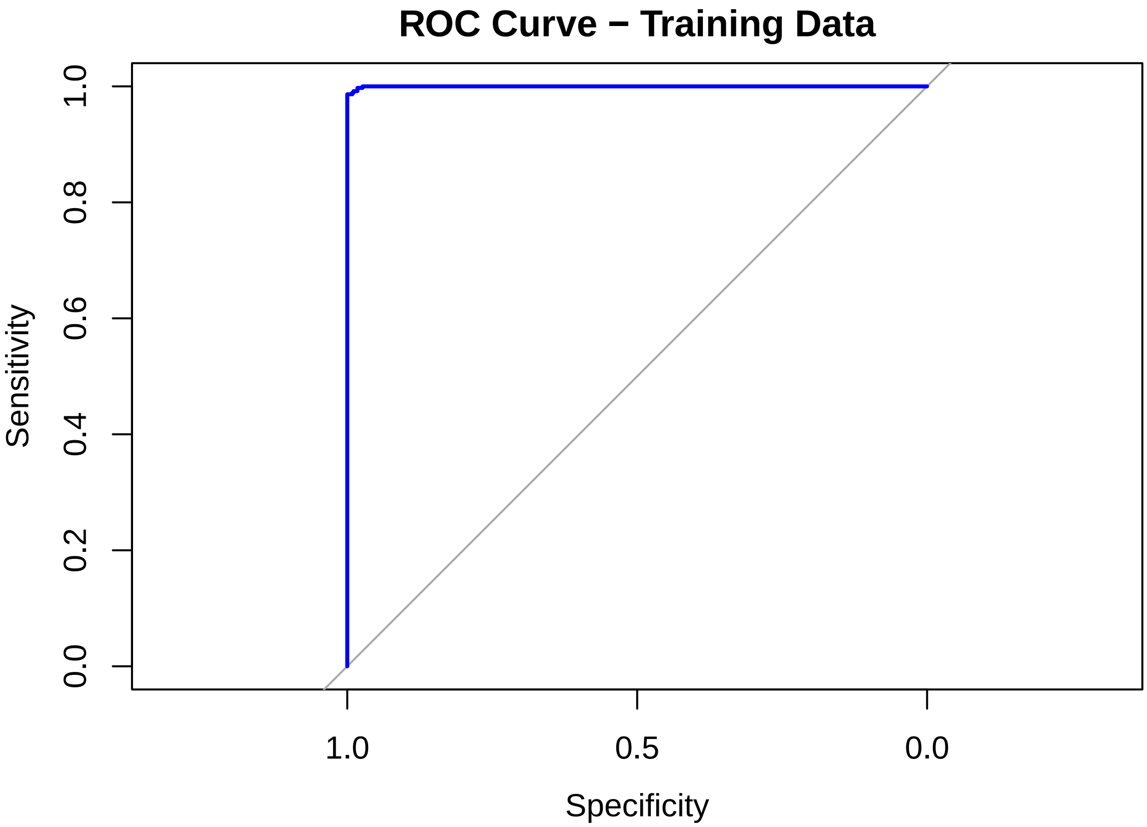
```
# Step 6: ROC Curve and AUC

# ROC Curve and AUC for Training Set
roc_train <- roc(trainData$V5, trainPred)
```

```
## Setting levels: control = 0, case = 1
```

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

```
plot(roc_train, main = "ROC Curve - Training Data", col = "blue")
```



```r
cat("AUC for Training Data: ", auc(roc_train), "\n")
```

```
## AUC for Training Data:  0.9997795
```

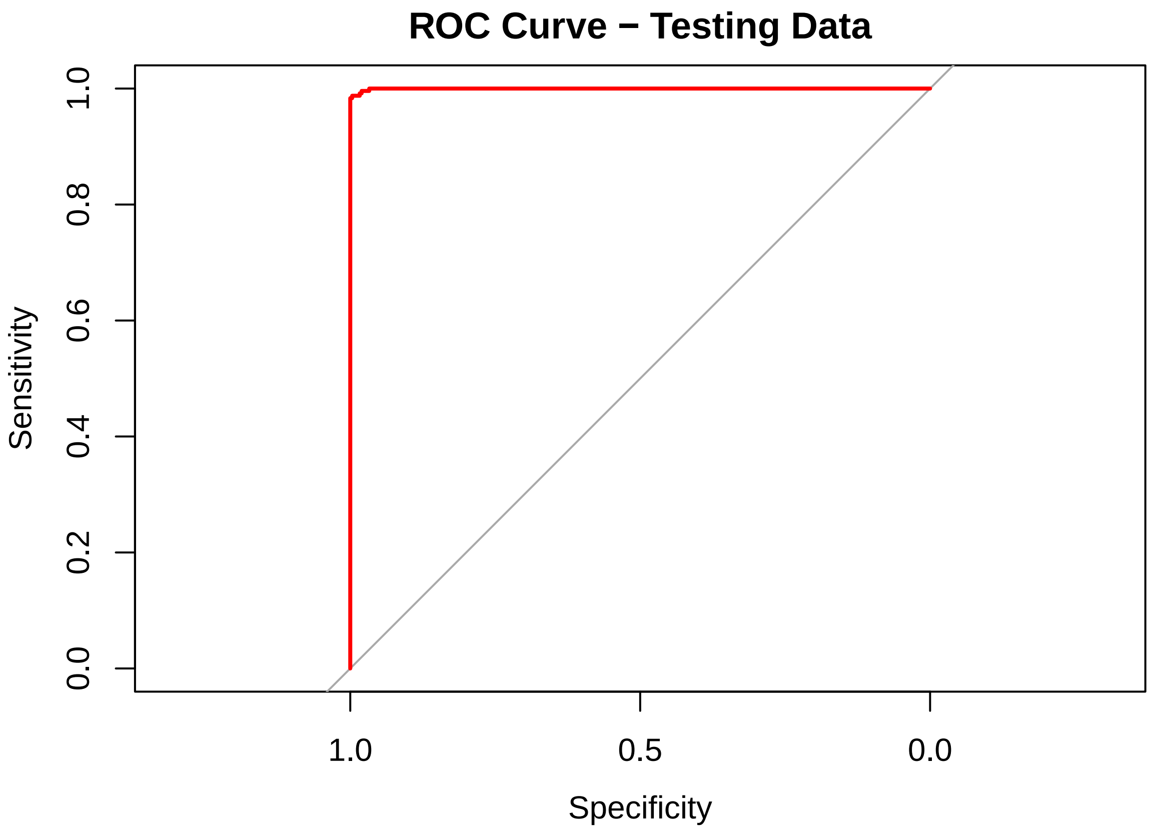
This is Receiver Operating Characteristic (ROC) curve evaluating the performance of the model based on the sensitivity (true positive rate) and specificity (false positive rate). The sharp curve indicates that the model has a high level of accuracy on the training data and the AUC for Training Data is 0.9997795

```r
# ROC Curve and AUC for Testing Set
roc_test <- roc(testData$V5, testPred)
```

```
## Setting levels: control = 0, case = 1
```

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

```r
plot(roc_test, main = "ROC Curve - Testing Data", col = "red")
```



```r
cat("AUC for Testing Data: ", auc(roc_test), "\n")
```

```
## AUC for Testing Data:  0.9997029
```

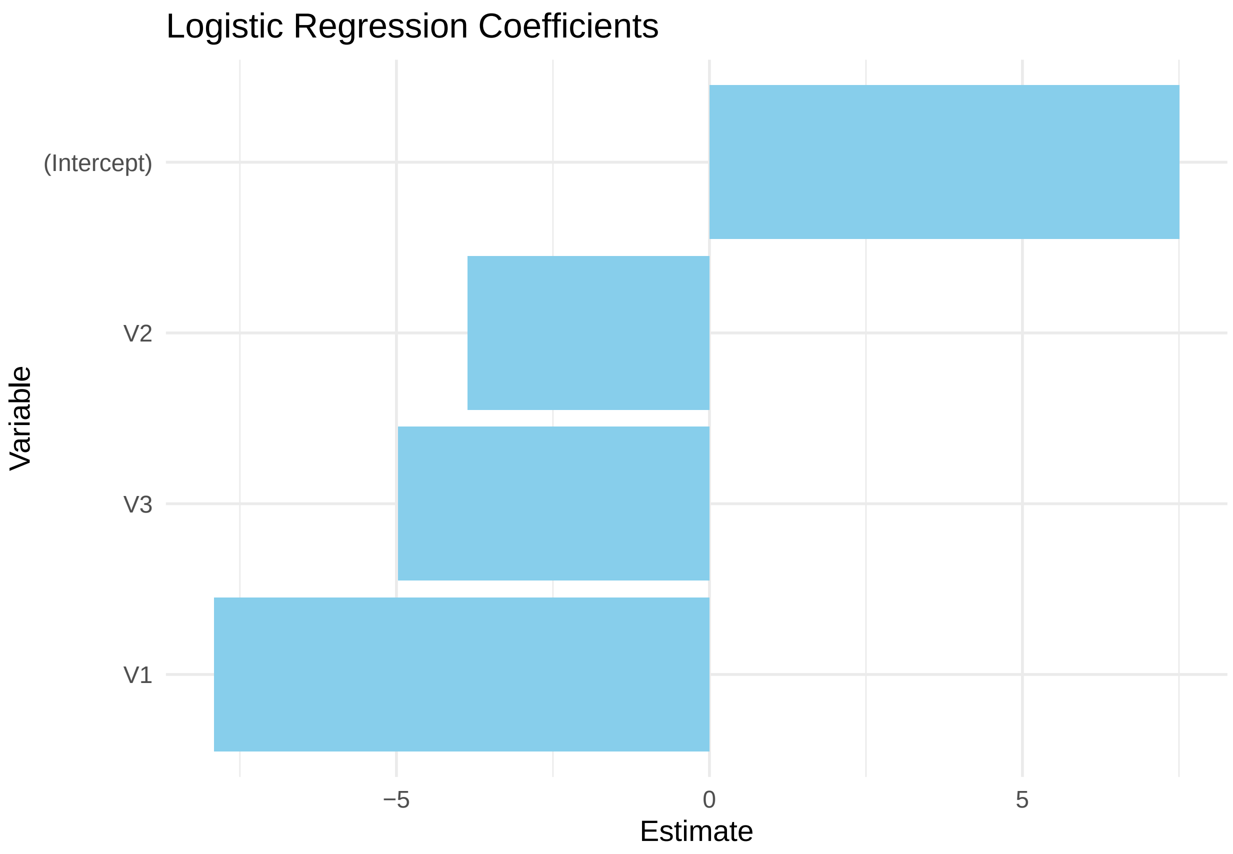
This ROC curve says the model's performance on the testing data, By plotting sensitivity (true positive rate) against specificity (1 - false positive rate). The curve indicates a strong classification ability, with sensitivity 1.0, suggesting that the model effectively distinguishes between genuine and counterfeit specimens, although steep rise suggests potential overfitting.

10. Visualizations

Lastly, we visualize the model coefficients and the distribution of predicted probabilities.

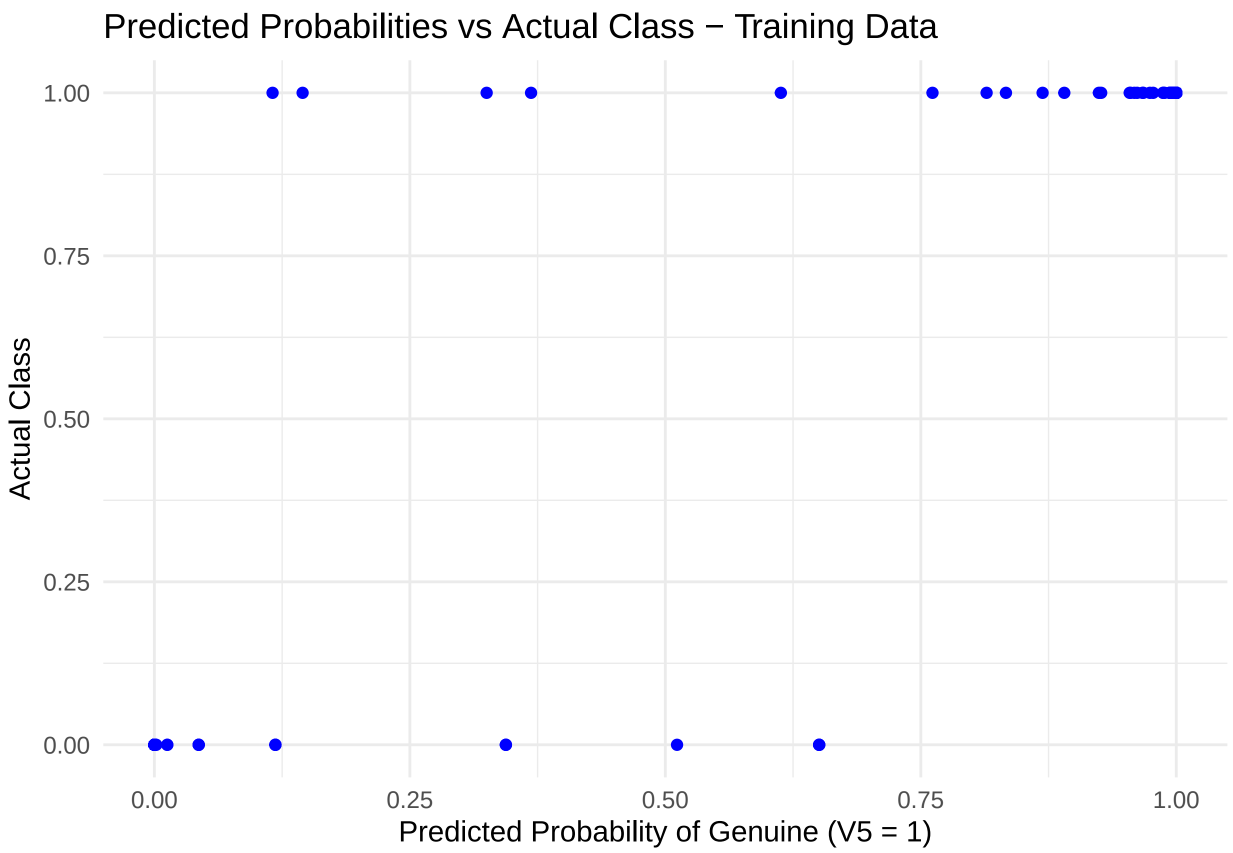
```r
# Step 7: Visualizations

# Coefficients Plot
coef_df <- data.frame(Variable = names(coef(final_model)),
                      Estimate = coef(final_model))
ggplot(coef_df, aes(x = reorder(Variable, Estimate), y = Estimate)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +
  theme_minimal() +
  labs(title = "Logistic Regression Coefficients", x = "Variable", y = "Estimate")
```

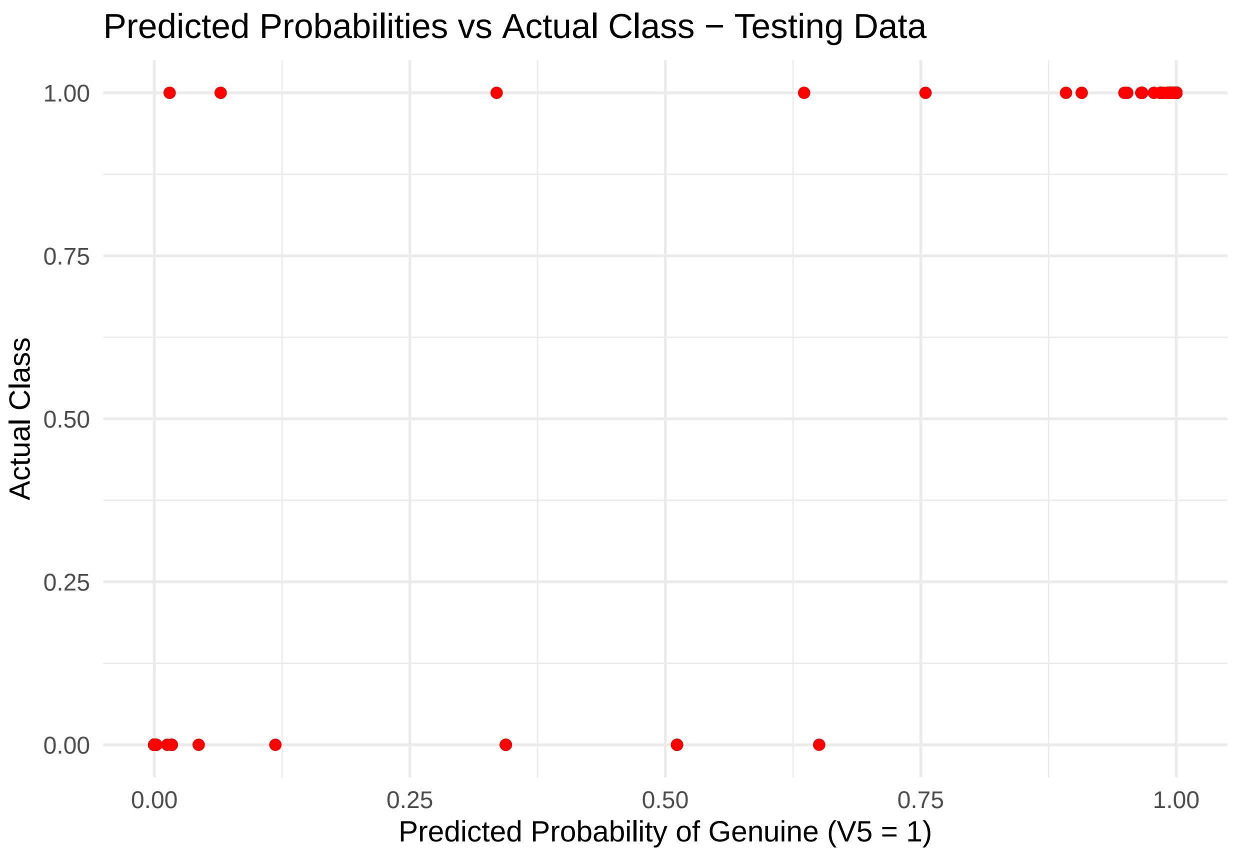


The bar plot says estimated coefficients from the logistic regression model for variables V1, V2, V3, and the intercept. Negative coefficients for V1, V2, and V3 suggest, increase in these variables are associated with lower log-odds of the outcome being genuine, while intercept the baseline log-odds when all predictor variable are zero.

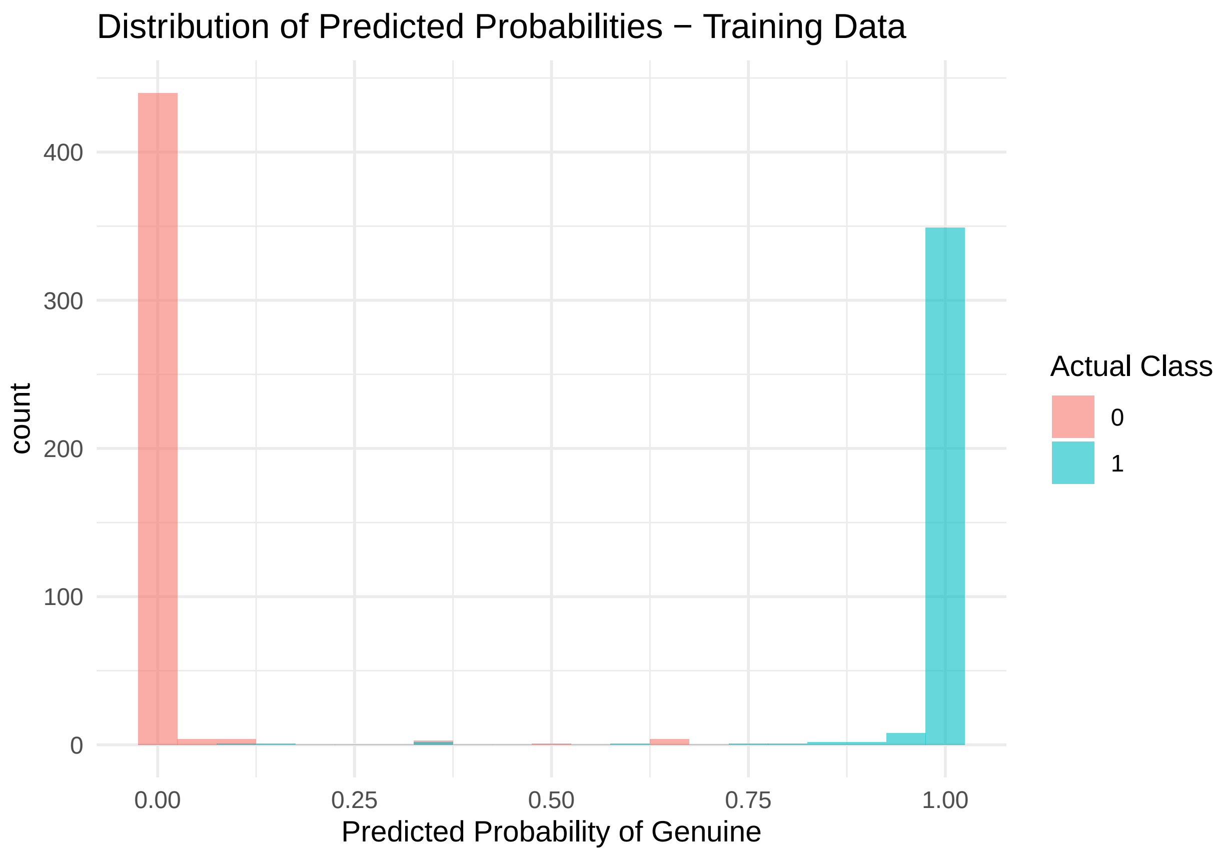
```r
# Predicted probabilities vs actual class for training data
ggplot(trainData, aes(x = trainPred, y = V5)) +
  geom_point(color = "blue") +
  labs(title = "Predicted Probabilities vs Actual Class - Training Data",
       x = "Predicted Probability of Genuine (V5 = 1)",
       y = "Actual Class") +
  theme_minimal()
```



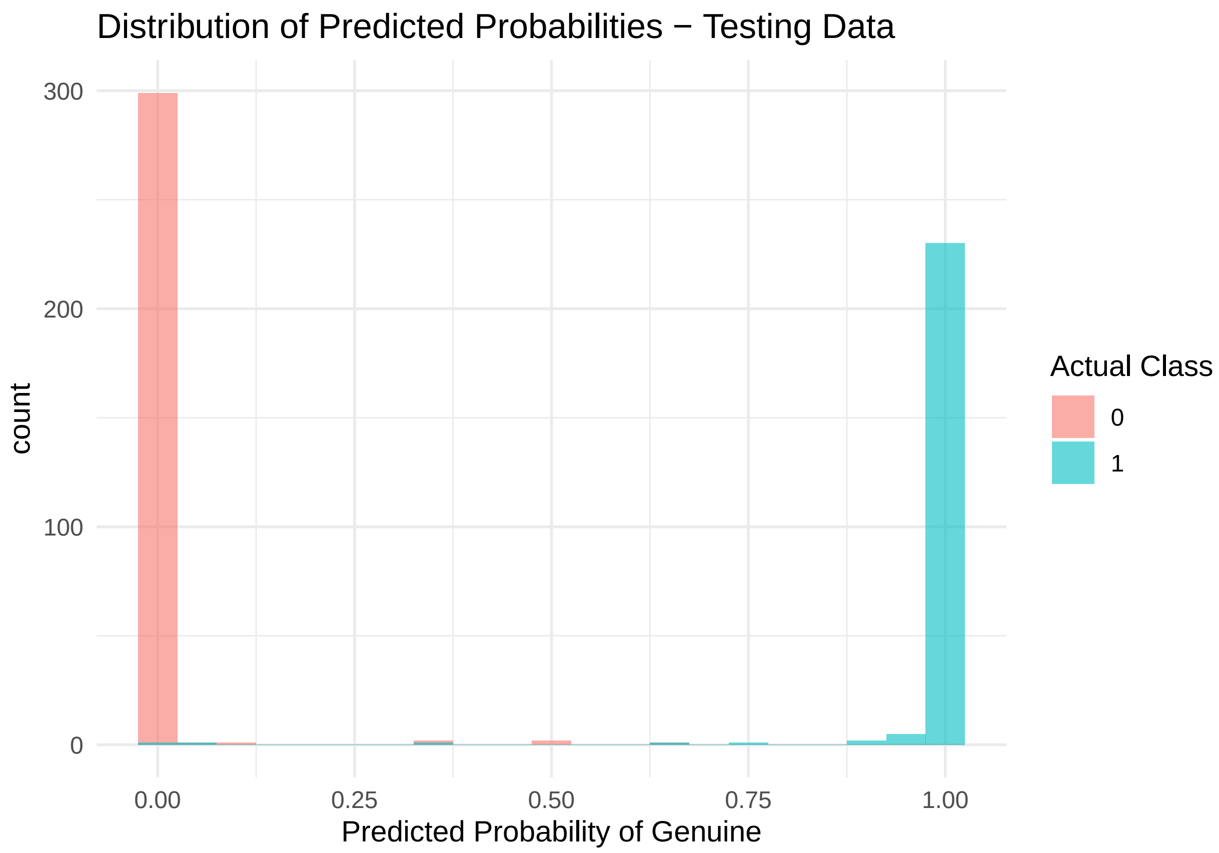
```r
# Predicted probabilities vs actual class for testing data
ggplot(testData, aes(x = testPred, y = V5)) +
  geom_point(color = "red") +
  labs(title = "Predicted Probabilities vs Actual Class - Testing Data",
       x = "Predicted Probability of Genuine (V5 = 1)",
       y = "Actual Class") +
  theme_minimal()
```



```r
# Distribution of predicted probabilities for genuine (1) and forged (0) specimens
ggplot(trainData, aes(x = trainPred, fill = as.factor(V5))) +
  geom_histogram(binwidth = 0.05, alpha = 0.6, position = "identity") +
  labs(title = "Distribution of Predicted Probabilities - Training Data",
       x = "Predicted Probability of Genuine",
       fill = "Actual Class") +
  theme_minimal()
```



```r
# The same for testing data
ggplot(testData, aes(x = testPred, fill = as.factor(V5))) +
  geom_histogram(binwidth = 0.05, alpha = 0.6, position = "identity") +
  labs(title = "Distribution of Predicted Probabilities - Testing Data",
       x = "Predicted Probability of Genuine",
       fill = "Actual Class") +
  theme_minimal()
```



11. Conclusion

The logistic regression model successfully identifies genuine and forged banknotes with a high degree of accuracy. Further exploration could involve testing other machine learning models to improve detection performance.