Credit Default Risk Prediction

Dan Wei

2024-03-25

```r
#install.packages("tidyverse")
library(tidyverse)
```

```
## Warning: 程辑包'tidyverse'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'ggplot2'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'tibble'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'tidyr'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'readr'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'purrr'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'dplyr'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'stringr'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'forcats'是用R版本4.2.3 来建造的
```

```
## Warning: 程辑包'lubridate'是用R版本4.2.3 来建造的
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.2     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.3     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```r
#install.packages("ggplot")
library(ggplot2)
#install.packages("MASS")
library(MASS)
```

```
## Warning: 程辑包'MASS'是用R版本4.2.3 来建造的
```

```
##
## 载入程辑包: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

Applying Logistic Regression to credit default prediction

```
# install.packages("ISLR2")
library(ISLR2)
```

```
## Warning: 程辑包'ISLR2'是用R版本4.2.3 来建造的
```

```
##
## 载入程辑包: 'ISLR2'
```

```
## The following object is masked from 'package:MASS':
##
##     Boston
```

```
# load dataset
df_default <- Default
```

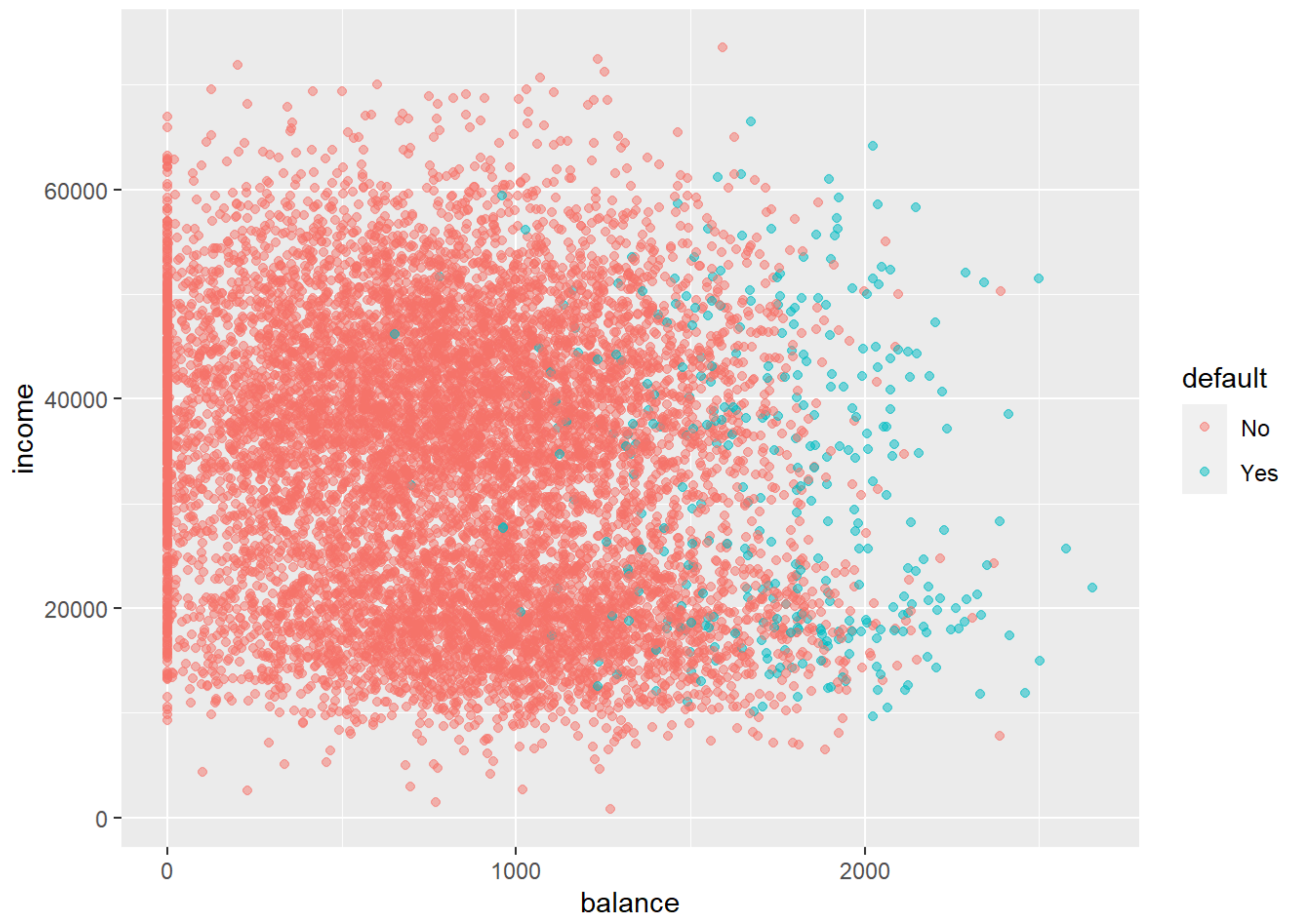
```
# ignore the student column
df_default$student <- NULL

# Split the dataset into a training set and a test set
set.seed(1)
train_ind <- sample(1:nrow(df_default), size = nrow(df_default)*0.8)

df_train <- df_default[train_ind, ]
df_test <- df_default[-train_ind, ]

nrow(df_train)
```

```
## [1] 8000
```

```
nrow(df_test)
```

```
## [1] 2000
```

```
# Visualise the relationship between balance,income and default
ggplot(data=df_default) +
  geom_point(aes(x=balance, y=income, color=default), alpha = 0.5)
```



Hypothesis: default may be related to high balance, and income level may not have a direct impact on default risk.

```
logreg1 <- glm(default ~ balance+income, data = df_train, family = binomial)
summary(logreg1)
```

```
##
## Call:
## glm(formula = default ~ balance + income, family = binomial,
##     data = df_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4758  -0.1413  -0.0563  -0.0210   3.4620
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.168e+01  4.893e-01 -23.879  < 2e-16 ***
## balance      5.613e-03  2.531e-04  22.176  < 2e-16 ***
## income       2.547e-05  5.631e-06   4.523  6.1e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2313.6  on 7999  degrees of freedom
## Residual deviance: 1239.2  on 7997  degrees of freedom
## AIC: 1245.2
##
## Number of Fisher Scoring iterations: 8
```

```
# Use the model to make predictions
df_test$predicted_class <- if_else(predict(logreg1, newdata = df_test, type = "response") >
0.5, "Yes", "No")

# Calculate the number of differences between the actual value and the predicted value
misclassified <- sum(df_test$predicted_class != df_test$default)

# Calculate misclassification rate
misclassification_rate <- misclassified / nrow(df_test)
misclassification_rate
```

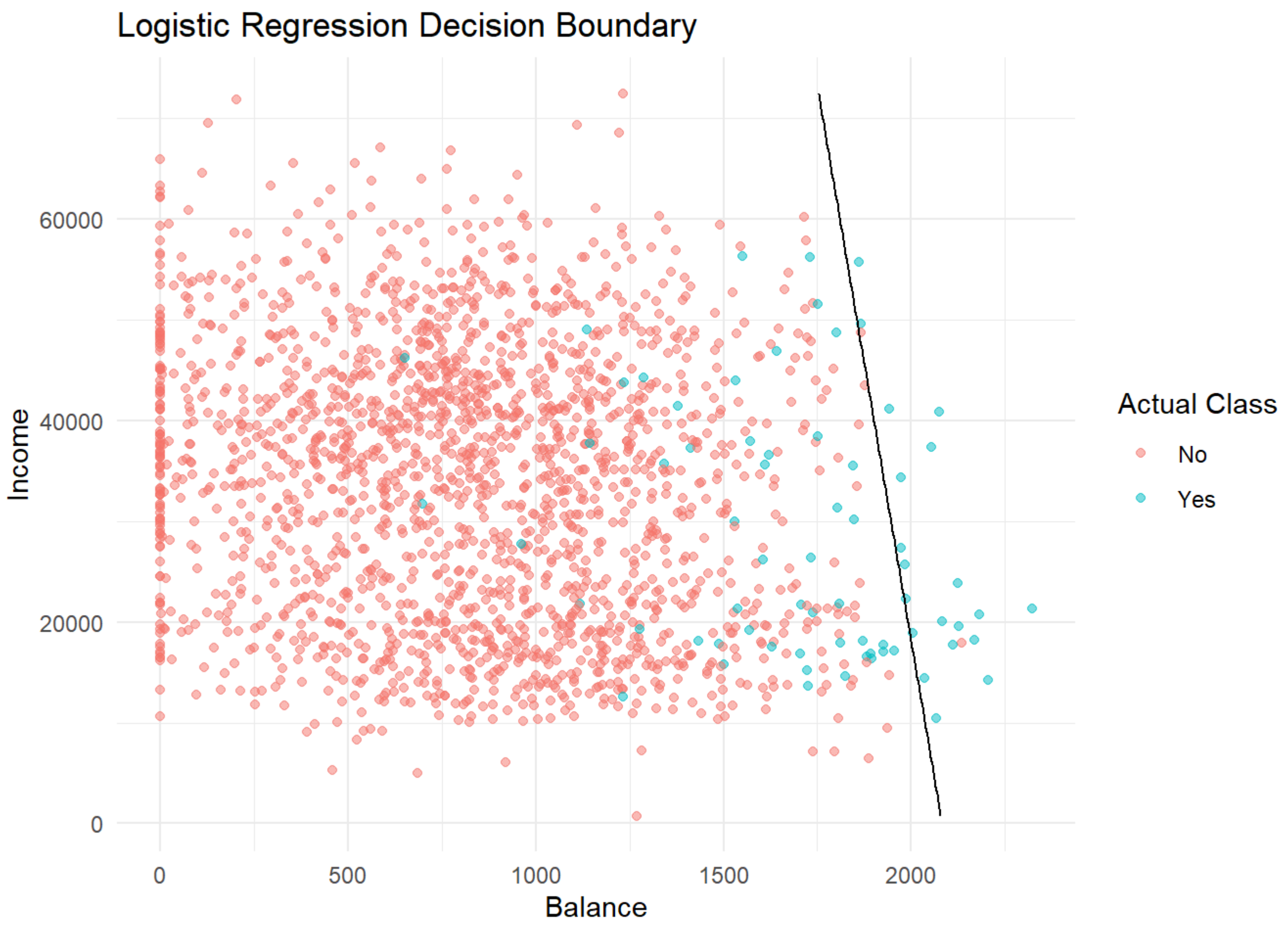
```
## [1] 0.026
```

The misclassification_rate is 2.6%.

```
# True Positive count
TP <- sum(df_test$default == "Yes" & df_test$predicted_class == "Yes")
# True Negative count
TN <- sum(df_test$default == "No" & df_test$predicted_class == "No")
# False Negative count
FN <- sum(df_test$default == "Yes" & df_test$predicted_class == "No")
# False Positive count
FP <- sum(df_test$default == "No" & df_test$predicted_class == "Yes")

confusion_matrix = matrix(c(TP,FN,FP,TN), 2, 2)
rownames(confusion_matrix) <- c("Actual Yes", "Actual No")
colnames(confusion_matrix) <- c("Pred Yes", "Pred No")
print(confusion_matrix)
```

```
##              Pred Yes Pred No
## Actual Yes         20       2
## Actual No          50    1928
```

```
# Create a prediction grid on a plane
plot_grid <- expand.grid(income = seq(min(df_test$income), max(df_test$income), length.out =
100),
                         balance = seq(min(df_test$balance), max(df_test$balance), length.out
= 100))

plot_grid$predicted_probability <- predict(logreg1, newdata = plot_grid, type = "response")

# Plot test set data points and decision boundaries
ggplot() +
  geom_point(data = df_test, aes(x = balance, y = income, color = default), alpha = 0.5) +
  geom_contour(data = plot_grid, aes(x = balance, y = income, z = predicted_probability),
               breaks = 0.5, color = "black") +
  labs(title = "Logistic Regression Decision Boundary",
       x = "Balance",
       y = "Income",
       color = "Actual Class") +
  theme_minimal()
```



The decision boundary of the classification regression model looks like a line from the upper left to the lower right on the balance-income plane. This curve divides the data points into two categories: those predicted to default and those predicted not to default. While most of the non-default data points appear to be classified correctly, there are some default points that are classified as non-default, indicating that there may be some prediction error.

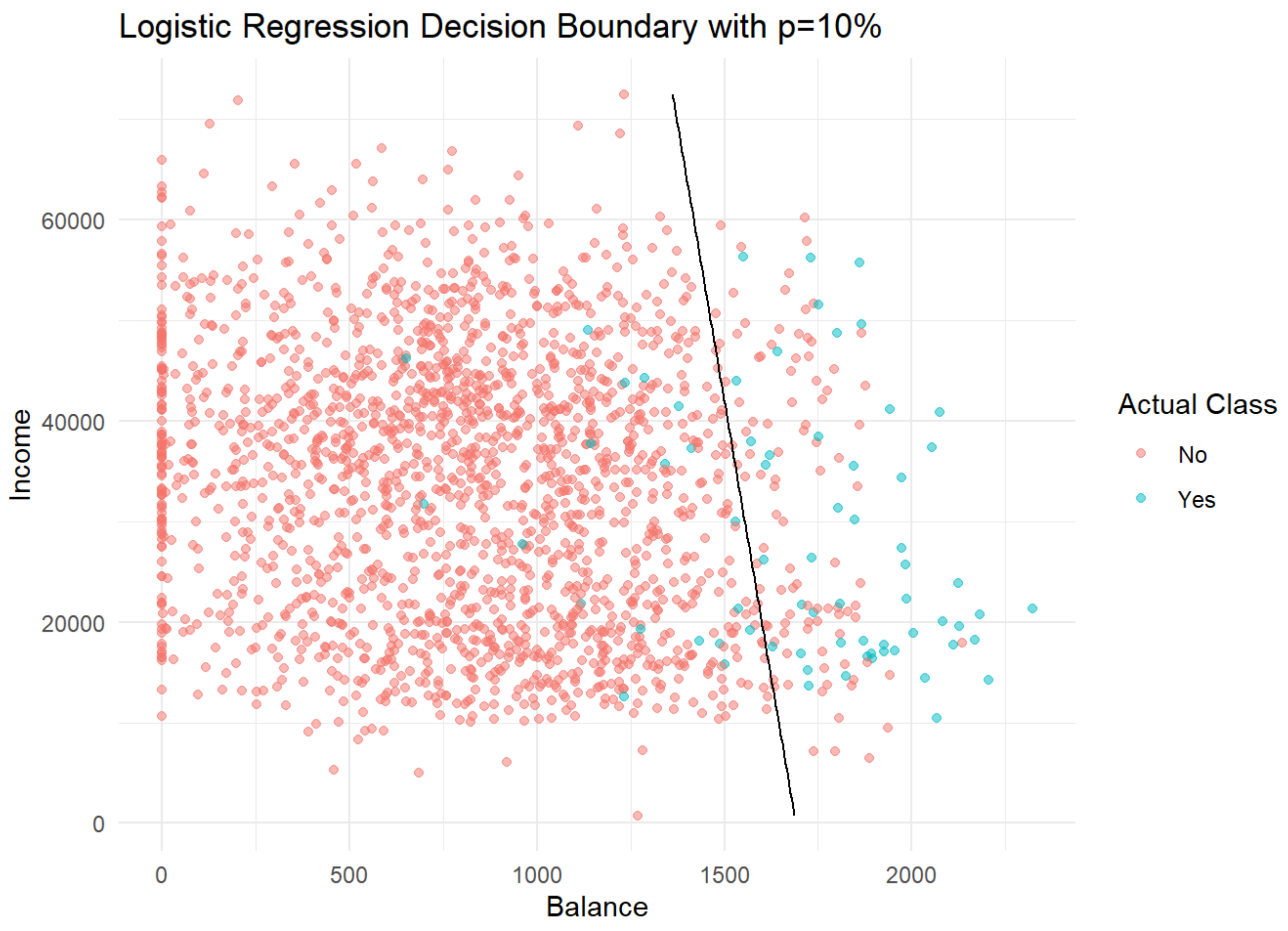
```
# Setting the decision boundary at probability p=10%

# Use the model to make predictions
df_test$predicted_class <- if_else(predict(logreg1, newdata = df_test, type = "response") >
0.1, "Yes", "No")

# Calculate the number of differences between the actual value and the predicted value
misclassified <- sum(df_test$predicted_class != df_test$default)

# Calculate misclassification rate
misclassification_rate <- misclassified / nrow(df_test)
misclassification_rate
```

```
## [1] 0.059
```

```
# True Positive count
TP <- sum(df_test$default == "Yes" & df_test$predicted_class == "Yes")
# True Negative count
TN <- sum(df_test$default == "No" & df_test$predicted_class == "No")
# False Negative count
FN <- sum(df_test$default == "Yes" & df_test$predicted_class == "No")
# False Positive count
FP <- sum(df_test$default == "No" & df_test$predicted_class == "Yes")

confusion_matrix = matrix(c(TP,FN,FP,TN), 2, 2)
rownames(confusion_matrix) <- c("Actual Yes", "Actual No")
colnames(confusion_matrix) <- c("Pred Yes", "Pred No")
print(confusion_matrix)
```

```
##            Pred Yes Pred No
## Actual Yes       51      99
## Actual No        19    1831
```

```
# Create a prediction grid on a plane
plot_grid <- expand.grid(income = seq(min(df_test$income), max(df_test$income), length.out =
100),
                         balance = seq(min(df_test$balance), max(df_test$balance), length.out
= 100))

plot_grid$predicted_probability <- predict(logreg1, newdata = plot_grid, type = "response")

# Plot test set data points and decision boundaries
ggplot() +
  geom_point(data = df_test, aes(x = balance, y = income, color = default), alpha = 0.5) +
  geom_contour(data = plot_grid, aes(x = balance, y = income, z = predicted_probability),
               breaks = 0.1, color = "black") +
  labs(title = "Logistic Regression Decision Boundary with p=10%",
       x = "Balance",
       y = "Income",
       color = "Actual Class") +
  theme_minimal()
```



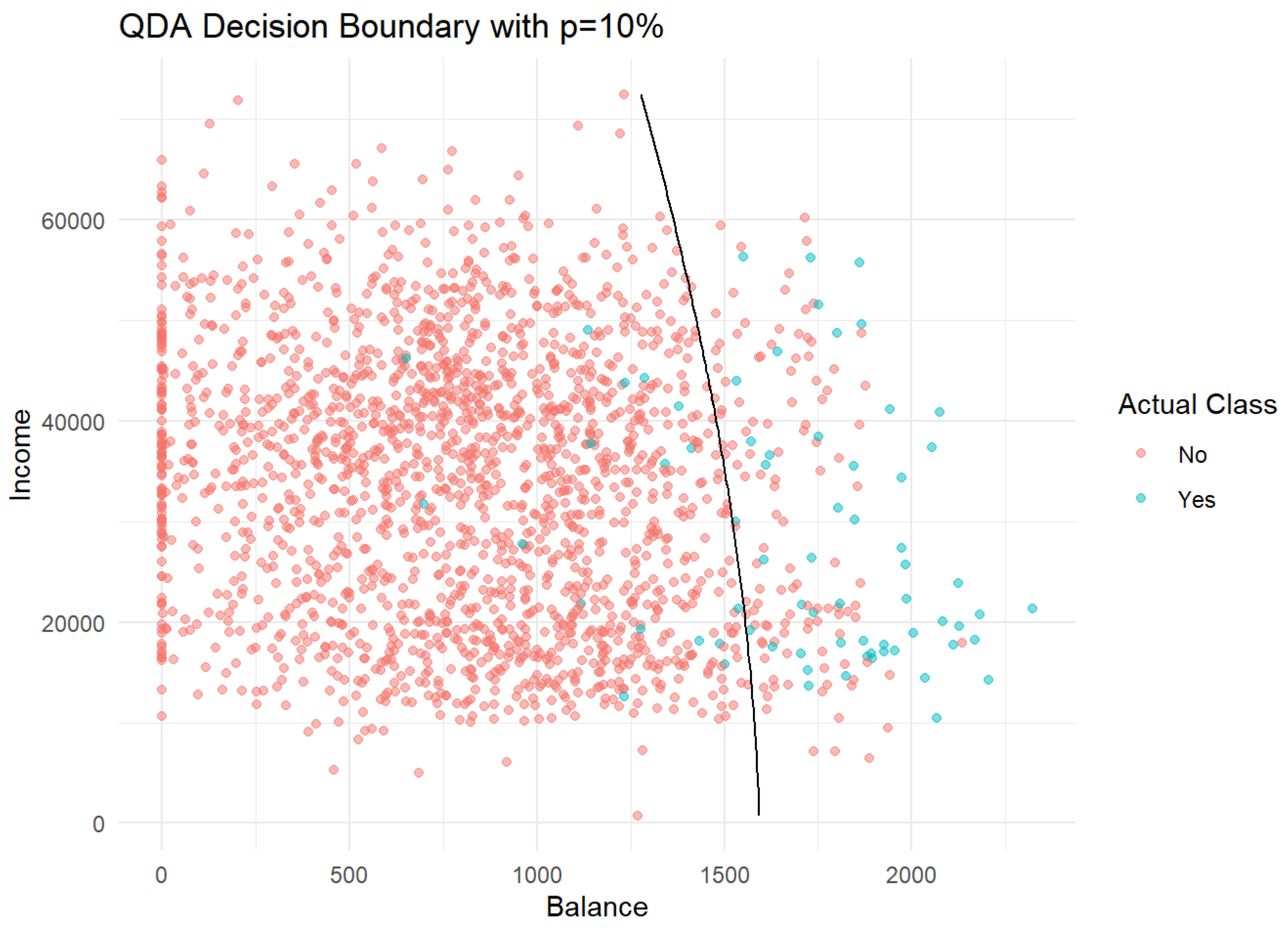
```r
# fit a (multiple) logistic regression model using QDA method.
qda_model <- qda(default ~ income + balance, data = df_train)

df_test <- df_test %>%
  mutate(
    pred_qda = predict(qda_model, newdata = df_test)$class
  )

# Calculate the number of differences between the actual value and the predicted value
misclassified <- sum(df_test$predicted_class != df_test$default)

# Calculate misclassification rate
misclassification_rate <- misclassified / nrow(df_test)
misclassification_rate
```

```
## [1] 0.059
```

```
# True Positive count
TP <- sum(df_test$default == "Yes" & df_test$predicted_class == "Yes")
# True Negative count
TN <- sum(df_test$default == "No" & df_test$predicted_class == "No")
# False Negative count
FN <- sum(df_test$default == "Yes" & df_test$predicted_class == "No")
# False Positive count
FP <- sum(df_test$default == "No" & df_test$predicted_class == "Yes")

confusion_matrix = matrix(c(TP,FN,FP,TN), 2, 2)
rownames(confusion_matrix) <- c("Actual Yes", "Actual No")
colnames(confusion_matrix) <- c("Pred Yes", "Pred No")
print(confusion_matrix)
```

```
##            Pred Yes Pred No
## Actual Yes       51      99
## Actual No        19    1831
```

```
# Create a prediction grid on a plane
plot_grid <- expand.grid(income = seq(min(df_test$income), max(df_test$income), length.out =
100),
                         balance = seq(min(df_test$balance), max(df_test$balance), length.out
= 100))

# Calculate the posterior probability of each point on the grid
plot_grid$posterior_probs <- predict(qda_model, newdata = plot_grid)$posterior[, "Yes"]

# Plot test set data points and decision boundaries
ggplot(df_test, aes(x = balance, y = income)) +
  geom_point(aes(color = default), alpha = 0.5) +
  geom_contour(data = plot_grid, aes(z = posterior_probs, x = balance, y = income), breaks =
0.1, color = "black") +
  labs(title = "QDA Decision Boundary with p=10%",
       x = "Balance", y = "Income", color = "Actual Class") +
  theme_minimal()
```



The data involved in algorithm training may contain sensitive personal information. Ensuring the confidentiality of this information and preventing data leakage is critical to protecting personal privacy.

The decision-making process of the algorithm should be transparent and provide a clear basis for decision-making. When it comes to important decisions like loan or credit card applications, customers have the right to know how algorithms make decisions and based on what criteria.

When an algorithm makes an error, such as incorrectly denying a credit application, it should be clear who is responsible. There needs to be clear procedures to correct errors and provide remedies to affected individuals.