# STAT462 Assignment 2

David Ewing (82171165)  Xia Yu (62380486)

2025-04-14

## Contents



output: pdf_document: latex_engine: xelatex keep_tex: true

mainfont: Arial fontsize: 10pt

```
# Load heart and colour datasets from zip

# Set base directory
# Assign named paths based on filename match
# load the dataframes via pretty_read_csv
# preview structure
unzip_dir <- "../data/unzipped"
zip_path  <- "../data/data_assignment_2.zip"
csv_files <- unzip(zip_path, list = TRUE)$Name  # Extract filenames from the zip
```

```r
target_paths <- file.path(unzip_dir, csv_files)

idx_color   <- grep("color", csv_files)
idx_heart   <- grep("heart", csv_files)
path_colour <- target_paths[idx_color]
path_heart  <- target_paths[idx_heart]

heart_list  <- pretty_read_csv(path_heart, col_names = TRUE)
colour_list <- pretty_read_csv(path_colour, col_names = TRUE)

heart_df = heart_list$df
heart_ft = heart_list$ft

colour_df = colour_list$df
colour_ft = colour_list$ft


foo <- pretty_split_df(heart_df)

render_flextables <- function(ft_list) {
  for (ft in foo) {
  invisible(print(knitr::knit_print(ft)))
}
}

render_flextables(foo)
```

### 0.0.1 Split the dataset into a training set (80%) and a test set (20%).

```r
# Split the dataset into training (80%) and test (20%) sets
set.seed(82171165)  # For reproducibility

# Ensure the outcome variable is a factor
heart_df$DEATH <- as.factor(heart_df$DEATH)

# Generate train/test split
n <- nrow(heart_df)
train_indices <- sample(seq_len(n), size = 0.8 * n)
train_data <- heart_df[train_indices, ]
test_data  <- heart_df[-train_indices, ]
```

### 0.0.2 Visualise the relationship between DEATH, GLUCOSE and SYSBP.

```
### steps:

#    -   English:
#    -   English: Visualise the relationship between DEATH, GLUCOSE and SYSBP.
#    -   English: Form an initial hypothesis of what to look for when doing the classification
#    -   English: On the training set, fit a (multiple) logistic regression model.
#    -   English: Compute the misclassification rates on the test set.
#    -   English: Compute the confusion matrix on the test set.
#    -   English: Visualise your fitted classification models, e.g., by plotting the decision
#    -   English: Make a comment or observation regarding goodness of fit.
#    -   English: Modify your logistic regression to classify as "risky" if the risk is higher
 #   -   English: Repeat the tasks of question c) (misclassification rates, confusion matrix,
#    -   English: Compare the performance of logistic regression and discriminant analysis on
#    -   English: Identify strong risk factors from this dataset and communicate your results.
```

### 0.0.3 Form an initial hypothesis of what to look for when doing the classification.

```
# TODO: implement
```

### 0.0.4 On the training set, fit a (multiple) logistic regression model.

```
# Fit logistic regression model and assign to model_logistic
model_logistic <- glm(DEATH ~ GLUCOSE + SYSBP + AGE, data = train_data, family = "binomial")
summary(model_logistic)
```

```
##
## Call:
## glm(formula = DEATH ~ GLUCOSE + SYSBP + AGE, family = "binomial",
##     data = train_data)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.650012   0.208564 -31.885  < 2e-16 ***
## GLUCOSE      0.005814   0.001023   5.684 1.32e-08 ***
## SYSBP        0.015876   0.001187  13.379  < 2e-16 ***
## AGE          0.055944   0.002995  18.680  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9959.3  on 8145  degrees of freedom
## Residual deviance: 8999.3  on 8142  degrees of freedom
```

```
##   (1155 observations deleted due to missingness)
## AIC: 9007.3
##
## Number of Fisher Scoring iterations: 4
```

### 0.0.5 Compute the misclassification rates on the test set.

```
# Misclassification Rate
pred_probs <- predict(model_logistic, newdata = test_data, type = "response")
pred_class <- ifelse(pred_probs > 0.5, 1, 0)
misclassification_rate <- mean(pred_class != test_data$DEATH)
misclassification_rate
```

```
## [1] NA
```

### 0.0.6 Compute the confusion matrix on the test set.

```
# TODO: implement
```

### 0.0.7 Visualise your fitted classification models, e.g., by plotting the decision boundaries in the GLUCOSE-SYSBP-plane.

```
### steps:

#    -   English:
#    -   English: Visualise the relationship between DEATH, GLUCOSE and SYSBP.
#    -   English: Form an initial hypothesis of what to look for when doing the classification
#    -   English: On the training set, fit a (multiple) logistic regression model.
#    -   English: Compute the misclassification rates on the test set.
#    -   English: Compute the confusion matrix on the test set.
#    -   English: Visualise your fitted classification models, e.g., by plotting the decision
#    -   English: Make a comment or observation regarding goodness of fit.
#    -   English: Modify your logistic regression to classify as "risky" if the risk is higher
 #   -   English: Repeat the tasks of question c) (misclassification rates, confusion matrix,
#    -   English: Compare the performance of logistic regression and discriminant analysis on
#    -   English: Identify strong risk factors from this dataset and communicate your results.
```

### 0.0.8 Make a comment or observation regarding goodness of fit.

```
# TODO: implement
```

#### 0.0.9 Modify your logistic regression to classify as "risky" if the risk is higher than 10%.

```
# Modify Threshold to 10%
pred_risky <- ifelse(pred_probs > 0.1, 1, 0)
table(Predicted = pred_risky, Actual = test_data$DEATH)
```

```
##          Actual
## Predicted    0    1
##         0   69    6
##         1 1346  620
```

#### 0.0.10 Repeat the tasks of question c) (misclassification rates, confusion matrix, visualisation) with the modified threshold.

```
# Confusion Matrix and Misclassification Rate with Threshold 10%
conf_matrix_10 <- table(Predicted = pred_risky, Actual = test_data$DEATH)
misclassification_rate_10 <- mean(pred_risky != test_data$DEATH)
conf_matrix_10
```

```
##          Actual
## Predicted    0    1
##         0   69    6
##         1 1346  620
```

```
misclassification_rate_10
```

```
## [1] NA
```

#### 0.0.11 Compare the performance of logistic regression and discriminant analysis on this classification problem.

```
# Compare Logistic Regression with Discriminant Analysis

# Drop missing values to avoid warnings from LDA
train_data_clean <- na.omit(train_data[, c("GLUCOSE", "SYSBP", "AGE", "DEATH")])
test_data_clean  <- na.omit(test_data[,  c("GLUCOSE", "SYSBP", "AGE", "DEATH")])
```

```r
# Fit LDA model
library(MASS)
model_lda <- lda(DEATH ~ GLUCOSE + SYSBP + AGE, data = train_data_clean)

# Predict on test data
pred_lda <- predict(model_lda, newdata = test_data_clean)$class
misclassification_lda <- mean(pred_lda != test_data_clean$DEATH)

# Optional: Format output using pretty_split_df
ft <- pretty_split_df(test_data_clean)
ft
```

```
## $`test_data_clean (1)`
## a flextable object.
## col_keys: `GLUCOSE`, `SYSBP`, `AGE`, `DEATH`
## header has 1 row(s)
## body has 5 row(s)
## original dataset sample:
##    GLUCOSE SYSBP AGE DEATH
## 5       71 108.0  58     0
## 6       70 127.5  48     0
## 16      85 138.0  63     0
## 23      82 168.0  64     0
## 25      74 147.0  49     0
```

```r
misclassification_lda
```

```
## [1] 0.2748653
```

### 0.0.12 Identify strong risk factors from this dataset and communicate your results.

```r
# Identify Strong Risk Factors
summary(model_logistic)
```

```
##
## Call:
## glm(formula = DEATH ~ GLUCOSE + SYSBP + AGE, family = "binomial",
##     data = train_data)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.650012   0.208564 -31.885  < 2e-16 ***
## GLUCOSE      0.005814   0.001023   5.684 1.32e-08 ***
## SYSBP        0.015876   0.001187  13.379  < 2e-16 ***
```

```
## AGE          0.055944   0.002995  18.680  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9959.3  on 8145  degrees of freedom
## Residual deviance: 8999.3  on 8142  degrees of freedom
##   (1155 observations deleted due to missingness)
## AIC: 9007.3
##
## Number of Fisher Scoring iterations: 4
```

#{r child="../doc/heart-analysis.Rmd" , cache = FALSE} #

### 0.1 A(a) Train-Test Split

### 0.2 A(b) Exploratory Visualisation

### 0.3 A(c) Logistic Regression

- (i) Misclassification Rate
- (ii) Confusion Matrix
- (iii) Decision Boundary Visualisation

### 0.4 A(d) Sensitivity-Prioritised Model

### 0.5 A(e) Comparison with Discriminant Analysis

### 0.6 A(f) Feature Importance and Risk Factors

# 1 Question B: Colour Classification (`colour_df_train.csv`)

‘

#### 1.0.1 Determine the number of distinct colour classes present in the dataset.

```
print(" Determine the number of distinct colour classes present in the dataset.")
```

```
## [1] " Determine the number of distinct colour classes present in the dataset."
```

```r
n_classes <- length(unique(colour_df$color))
n_classes
```

```
## [1] 5
```

```r
cat("distinct colour classes",n_classes,"\n")
```

```
## distinct colour classes 5
```

### 1.0.2 Fit a Quadratic Discriminant Analysis (QDA) algorithm to the classification problem.

```r
print(" Fit a Quadratic Discriminant Analysis (QDA) algorithm to the classification problem.")
```

```
## [1] " Fit a Quadratic Discriminant Analysis (QDA) algorithm to the classification problem."
```

```r
model_qda <- qda(color ~ r + b, data = colour_df)
model_qda
```

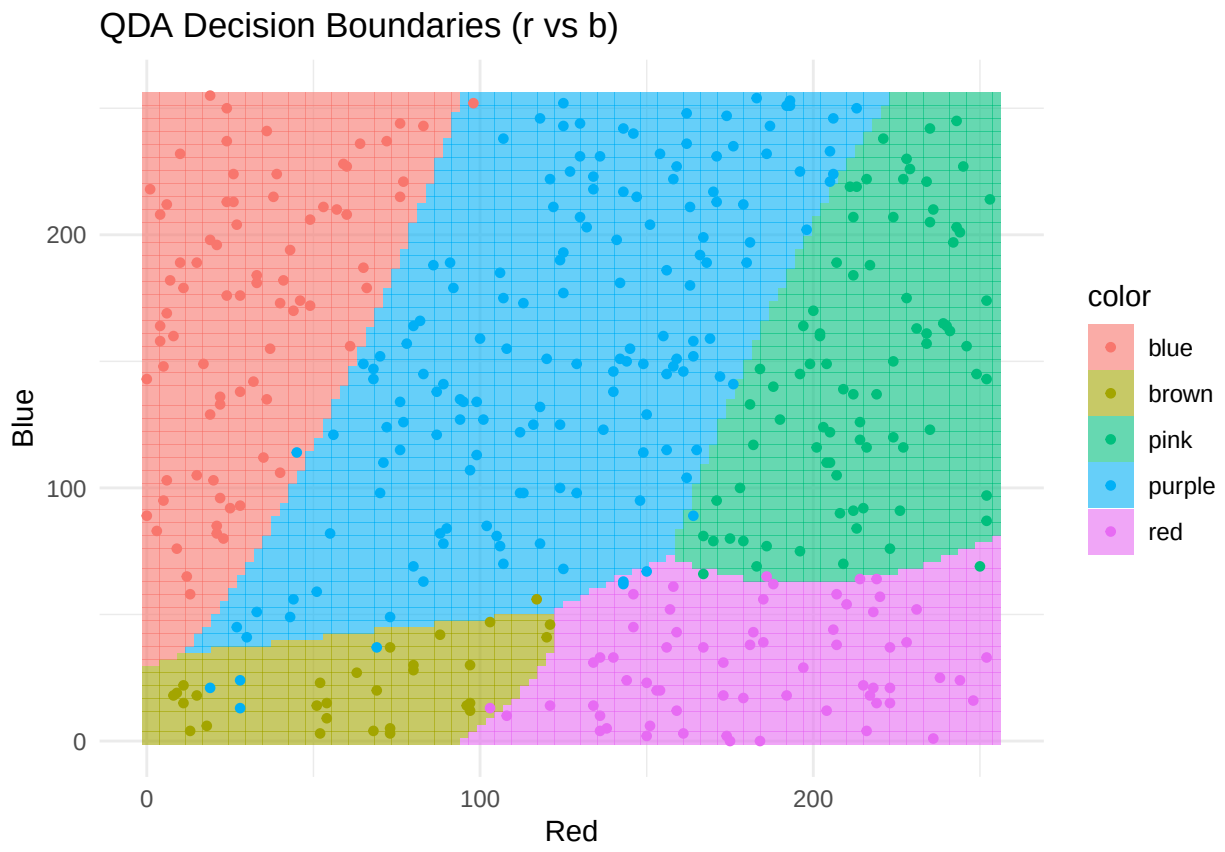
```
## Call:
## qda(color ~ r + b, data = colour_df)
##
## Prior probabilities of groups:
##   blue  brown   pink purple    red
## 0.1925 0.0725 0.2025 0.3750 0.1575
##
## Group means:
##                r         b
## blue    30.93506 170.16883
## brown   64.24138  21.48276
## pink   215.41975 146.43210
## purple 124.89333 155.15333
## red    182.33333  28.77778
```

### 1.0.3 Visualise the decision boundaries of the fitted QDA model in a suitable way.

```r
print("Visualise the decision boundaries of the fitted QDA model in a suitable way.")
```

```
## [1] "Visualise the decision boundaries of the fitted QDA model in a suitable way."
```

```
grid_points <- expand.grid(
  r = seq(0, 255, length.out = 100),
  b = seq(0, 255, length.out = 100)
)
grid_pred <- predict(model_qda, grid_points)$class
grid_points$color <- grid_pred
plot_qda_boundary <- ggplot(grid_points, aes(x = r, y = b, fill = color)) +
  geom_tile(alpha = 0.6) +
  geom_point(data = colour_df, aes(x = r, y = b, colour = color), size = 1.2, shape = 21) +
  theme_minimal() +
  labs(title = "QDA Decision Boundaries (r vs b)", x = "Red", y = "Blue")
plot_qda_boundary
```




### 1.0.4 Test the QDA algorithm on the input (200, 0, 200).

```
print(" Test the QDA algorithm on the input (200, 0, 200).")
```

```
## [1] " Test the QDA algorithm on the input (200, 0, 200)."
```

```r
new_colour <- data.frame(r = 200, b = 200)
qda_prediction <- predict(model_qda, new_colour)
qda_prediction
```

```
## $class
## [1] pink
## Levels: blue brown pink purple red
##
## $posterior
##          blue        brown      pink    purple          red
## 1 7.14596e-15 2.072491e-35 0.5538038 0.4461962 8.530348e-18
```

### 1.0.5 Determine what colour name is predicted by the algorithm for the input (200, 0, 200).

```r
print(" Determine what colour name is predicted by the algorithm for the input (200, 0, 200).")
```

```
## [1] " Determine what colour name is predicted by the algorithm for the input (200, 0, 200).
```

```r
predicted_class <- qda_prediction$class
predicted_class
```

```
## [1] pink
## Levels: blue brown pink purple red
```

### 1.0.6 (Optional) Implement k-Nearest Neighbours and compare its performance with the QDA algorithm.

```r
print(" Learn about k-Nearest Neighbors (kNN) classification, implement it, and compare perfor
```

```
## [1] " Learn about k-Nearest Neighbors (kNN) classification, implement it, and compare perfo
```

```r
library(class)
set.seed(82171165)
library(caret)
control <- trainControl(method = "cv", number = 10)
train_qda <- train(color ~ r + b, data = colour_df, method = "qda", trControl = control)
train_knn <- train(color ~ r + b, data = colour_df, method = "knn", tuneLength = 5, trControl =
qda_results <- train_qda$results
knn_results <- train_knn$results
qda_results
```

```
##   parameter  Accuracy     Kappa AccuracySD    KappaSD
## 1      none 0.9548655 0.9404838 0.02892084 0.03760597
```

```
knn_results
```

```
##    k  Accuracy     Kappa AccuracySD    KappaSD
## 1  5 0.9499171 0.9339224 0.04449267 0.05842528
## 2  7 0.9499812 0.9340583 0.03367963 0.04427702
## 3  9 0.9397311 0.9209044 0.05494836 0.07173767
## 4 11 0.9424203 0.9240809 0.04914093 0.06458621
## 5 13 0.9371701 0.9172428 0.05379689 0.07031433
```

```
#{r child="../doc/colour-analysis.Rmd", cache = FALSE} #
```

### 1.1 B(a) Number of Classes

### 1.2 B(b) QDA Classification and Visualisation

### 1.3 B(c) Classification of Colour (200, 0, 200)

### 1.4 B(d) kNN Comparison (Optional)

## 2 Discussion

This section synthesises findings from both classification tasks, comparing model performance and interpretability.

## 3 Conclusion

Summary of key results, limitations, and future recommendations.