# Predictive modeling of COVID-19 using LDA and QDA: Handout

## When to Use LDA or QDA

LDA and QDA can be used to solve classification problems where the outcome variable is categorical and the relationship between the outcome variable and the predictor variables is thought to be linear (for LDA) or quadratic (for QDA). Our analysis used LDA to predict the class (survived or died) of a patient's COVID-19 mortality.

## Data Requirements, Assumptions, Solutions

**Data Requirements and Assumptions**: A categorical response variable and constant variance among response variable's classes are the most essential requirements of LDA. Additionally, independence among variables and normality of continuous predictor variables is assumed.

**Assessing Assumptions:** Box M's test can be used to determine if there is constant variance among a variable's classes. Normality can be tested with QQ-plots or Shapiro-Tests. Independence can often be inferred from the test design, or tested with R's plot() command.

## Important Information: LDA and QDA

**Step 1**: Ensure that the data is clean, that there are no missing values, that the variable types are correct, and that there is no multicollinearity, heteroscedasticity, or near zero variance before performing variable selection or model fitting. Here are some of the most important examples from our project:

```
# Create Binary Response Variable (method is unique to our project use case)
data$DIED <- ifelse(data$DATE_DIED != '9999-99-99', 1, 0)

# Change Variables: Example – Outcome variable to factor
data$DIED <- as.factor(data$DIED)

# Homoscedasticity check
leveneTest(AGE ~ DIED, data = data)

# Calculate VIF (from car package) in order to check for multicollinearity
multi_col_model <- lm(DIED ~ ., data = data)
vif(multi_col_model)

# Calculate the near-zero-variance variables (from caret)
near_zero_var <- nearZeroVar(data, saveMetrics = TRUE)
near_zero_var <- rownames(near_zero_var[near_zero_var$nzv == TRUE,])
near_zero_var
```

**Step 2**: Variable selection: fit a full model, pass it into stepAIC, then determine whether or not to use the variables produced by the step model. Choose the best model.

```
# LDA variable selection
full_model <- glm(DIED ~ ., data = data, family = binomial, trace = TRUE)
step_model <- stepAIC(full_model, direction = "both")
summary(step_model)
```

**Step 3**: Create a train test split. In this example, 70% training 30% testing was allocated.

```
# Train test split
tts <- rep(0:1,c(round(n*.3), n-round(n*.3)))

# Get the TTS Split
tts.split <- sample(tts, n)
training_data <- data[tts.split==1, ]
testing_data <- data[tts.split==0, ]
```

**Step 4**: Fit the LDA or QDA models on the training data (where outcome variable = DIED), then test the models on the testing data.

```
# Fit models on training data
lda_model <- lda(DIED ~ ., data = training_data)
qda_model <- qda(DIED ~ ., data = training_data)

# Make predictions on testing data
lda_model_predictions <- predict(lda_model, testing_data)$class
qda_model_predictions <- predict(qda_model, testing_data)$class
```

```
# Confusion Matrix: LDA
lda_conf <- table(predicted_deaths = lda_model_predictions, actual_deaths =
testing_data$DIED)

# Confusion Matrix: QDA
qda_conf <- table(predicted_deaths = qda_model_predictions, actual_deaths =
testing_data$DIED)

# Now, compute accuracy, recall, and precision; afterwards, choose the best model based on
these statistics
```

Our LDA Model produced 77% accuracy and 77% recall, which were both greater than the QDA model. Thus, we chose the LDA model.

**Step 5**: Produce ROC Curve, then get the AUC value to understand how the model performs against random chance

```
# Get predictions variable
lda_model_predictions <- predict(lda_model, testing_data)

# ROC Curve Step 1: Get probabilities for positive class
lda_probabilities <- lda_model_predictions$posterior[,2]

# ROC Curve Step 2: Create ROC curve
lda_roc <- roc(testing_data$DIED, lda_probabilities)

# Get AUC value by passing ROC
AUC <- auc(lda_roc)
```

0.5 - AUC = How much better or worse the model performs than random chance. Our model achieved 0.8 AUC, meaning that it performed 30% better than random chance.

## Understanding Cutoff

**Cutoff**: The cutoff value is the value which the LDA or QDA function uses to separate classes. In our example, the cutoff value was 0.5. Therefore, if the LDA function produced a value >= 0.5 for any observation, that patient would be predicted as the class of DIED=1.
**Effect of Increased Cutoff On Recall:** Fewer deaths predicted → increased number of false negatives → reduced recall
**Effect of Increased Cutoff On Precision:** Fewer deaths predicted → reduced number of false positives → increased precision

## Understanding The Model

**The Model Produced** by LDA and QDA includes a list of coefficients. The sum of all variables multiplied by their coefficients produces the result of LDA. If the result is greater than the cutoff value, the class of the outcome variable is assigned to 1; otherwise, it is assigned to 0. Therefore, variables with a positive coefficient in LDA / QDA output positively contribute to the outcome variable's assignment to the class of 1 (ex: DIED=1), and variables with a negative coefficient negatively contribute to the outcome variable's assignment to the class of 1.

## Additional Resources / Reading List


https://minerva.it.manchester.ac.uk/~saralees/statbook4.pdf
https://hastie.su.domains/ElemStatLearn/printings/ESLII_print10.pdf
https://rdrr.io/cran/MASS/man/lda.html
https://rdrr.io/cran/MASS/man/qda.html
https://uc-r.github.io/discriminant_analysis
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10800012/
https://www.geeksforgeeks.org/linear-discriminant-analysis-in-r-programming/
https://www.frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2022.1033872/full
https://onlinelibrary.wiley.com/doi/full/10.1155/2021/4733167
https://www.nature.com/articles/s41598-023-38133-6
https://bmcinfectdis.biomedcentral.com/articles/10.1186/s12879-024-09298-w