Module 3 HW

Alanna Hazlett

2024-06-05

Module 3

In this module, we learned classification models. Question (1) tests your understanding of the differences between LDA and QDA. In question (2), you will build several classification models for the NASA Asteroid dataset and estimate their predictive performance using a holdout/test set. Question (3) prepares you for the final project by researching approaches to handling class imbalance in classification problems.

Use *Tidyverse* and *Tidymodels* packages for the assignments.

You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-3.Rmd) and use it as a basis for your solution.

```
library(tidyverse)
library(tidymodels)
library(tidyverse)
library(tidymodels)
library(discrim) # for LDA and QDA
library(ggcorrplot)  # for correlation plot
library(GGally)  # scatterplot matrix
library(patchwork)  # for combining plots
library(probably)  # for threshold_perf
```

1. Differences between LDA and QDA.

a. If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?

- We would expect that QDA would perform better on the training set, as it has higher flexibility, which would result in smaller residuals. However the flexibility of QDA would then make it likely to overfit the data and would be worse for the test set, so the LDA would be better for the test set.

b. If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

- We would expect QDA to perform better on both the training and the test set as it's quadratic relationship allows it to fit non-linear data better. It is more flexible and will be able to capture other patterns in the data that LDA will not be able to, since LDA requires a linear relationship.

c. In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

- We would expect the test accuracy to improve for QDA as n increases, because the when n increases the model is not as easy influenced by a single observation. It will be less likely to overfit for a large n rather than for a small n. LDA is not as easily influenced by a single

observation, as it has a higher bias and lower variance than QDA, so at a certain point adding more observations to the sample will not impact LDA significantly anymore.

d. True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

- False, because the problem will not be absolutely perfectly linear, and so the QDA will pick up trends in the data from the training set and try to apply those trends to the test set (it will be overfitted) and will result in a worse test error rate than LDA would.

2. NASA: Asteroid classification

The dataset `nasa.csv` contains information about asteroids and if they are considered to be hazardous or not.

a. Load the data from https://gedeck.github.io/DS-6030/datasets/nasa.csv and preprocess the data. You can reuse the preprocessing pipeline we developed in class.

```
remove_columns <- c('Name', 'Est Dia in M(min)',
    'Semi Major Axis', 'Jupiter Tisserand Invariant',
    'Epoch Osculation', 'Mean Motion', 'Aphelion Dist',
    'Equinox', 'Orbiting Body', 'Orbit Determination Date',
    'Close Approach Date', 'Epoch Date Close Approach',
    'Miss Dist.(Astronomical)', 'Miles per hour')
asteroids <- read_csv("https://gedeck.github.io/DS-6030/datasets/nasa.csv",
                      show_col_types = FALSE) %>%
    select(-all_of(remove_columns)) %>%
    select(-contains("Relative Velocity")) %>%
    select(-contains("Est Dia in KM")) %>%
    select(-contains("Est Dia in Feet")) %>%
    select(-contains("Est Dia in Miles")) %>%
    select(-contains("Miss Dist.(lunar)")) %>%
    select(-contains("Miss Dist.(kilometers)")) %>%
    select(-contains("Miss Dist.(miles)")) %>%
    distinct() %>%
    mutate(Hazardous = base::as.factor(Hazardous))
dim(asteroids)
```

```
## [1] 3692   15
```

b. Split the dataset into a training and test set. Use 80% of the data for training and 20% for testing. Use stratified sampling to ensure that the training and test set have the same proportion of hazardous asteroids.

```
set.seed(12345)
asteroids_split <- initial_split(asteroids, prop=0.8, strata=Hazardous)
asteroids_train <- training(asteroids_split)
asteroids_holdout <- testing(asteroids_split)
```

c. Build a classification model with *tidymodels* using four different methods: Null model, Logistic regression, LDA, and QDA. Use the training set to fit the models and the test set to evaluate the performance of the models.

```
formula <- Hazardous ~ .

reference_model <- null_model(mode="classification") %>%
        set_engine("parsnip") %>%
        fit(formula, asteroids_train)
```

```
logreg_model <- logistic_reg(mode="classification") %>%
        set_engine("glm") %>%
        fit(formula, asteroids_train)

lda_model <- discrim_linear(mode="classification") %>%
        set_engine("MASS") %>%
        fit(formula, asteroids_train)

qda_model <- discrim_quad(mode="classification") %>%
        set_engine("MASS") %>%
        fit(formula, asteroids_train)
```

```
all_metrics <- bind_rows(
    calculate_metrics(reference_model, asteroids_train, asteroids_holdout, "reference"),
    calculate_metrics(logreg_model, asteroids_train, asteroids_holdout, "logreg"),
    calculate_metrics(lda_model, asteroids_train, asteroids_holdout, "LDA"),
    calculate_metrics(qda_model, asteroids_train, asteroids_holdout, "QDA"),
)
metrics_table(all_metrics,"All Metrics")
```

Table 1: All Metrics

model	dataset	accuracy	kap	roc_auc
reference	train	0.844	0.000	0.500
reference	test	0.843	0.000	0.500
logreg	train	0.957	0.836	0.988
logreg	test	0.969	0.881	0.995
LDA	train	0.919	0.683	0.965
LDA	test	0.916	0.681	0.970
QDA	train	0.958	0.848	0.987
QDA	test	0.972	0.897	0.986

```
metrics_wide <- all_metrics %>%
    pivot_wider(names_from=.metric, values_from=.estimate)
g1 <- ggplot(metrics_wide, aes(y=model, x=roc_auc, color=dataset)) +
    geom_point()
g2 <- ggplot(metrics_wide, aes(y=model, x=accuracy, color=dataset)) +
    geom_point()
g3 <- ggplot(metrics_wide, aes(y=model, x=kap, color=dataset)) +
    geom_point()
g1 + g2 + g3
```

In figure 1 we can see that the reference model does the worst in all metrics, as we would expect. LDA is is second worst as the metrics are lower for LDA than they are for QDA and logistic regression. QDA and logistic regression perform pretty similarly, with QDA possibly beating out logistic regression. Logistic regression has a higher AUC, but QDA has a higher accuracy and kappa value. For accuracy and kappa we see that QDA and logistic regression have better values for the test set, while LDA has better values for the training set. For the AUC the reference model and QDA have almost exactly the same values for their respective training and test sets, and logistic and LDA have better values for the test set over the training set.

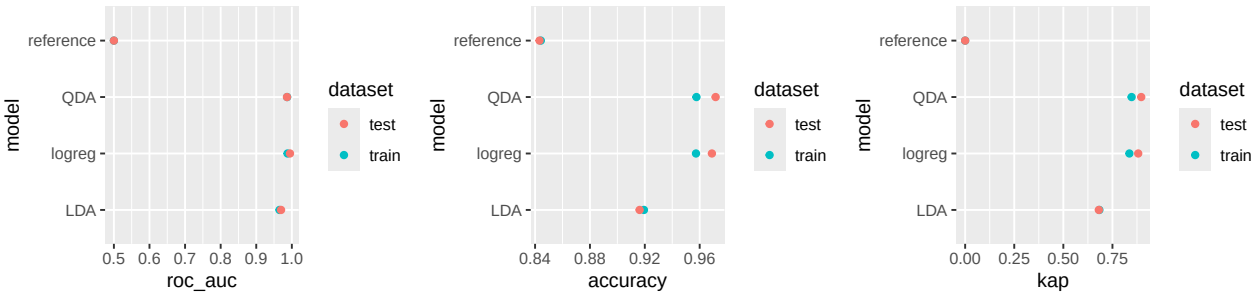

Figure 1: Classification Metrics as Graphs

(i.) For each model, determine and plot the ROC curves for both the training and test set. What do you observe?

```
get_ROC_plot <- function(model, data, model_name) {
    model %>%
        augment(data) %>%
        roc_curve(truth=Hazardous, .pred_TRUE, event_level="second") %>%
    autoplot() +
        labs(title=model_name)
}
g1 <- get_ROC_plot(reference_model, asteroids_train, "Null model")
g2 <- get_ROC_plot(logreg_model, asteroids_train, "Logistic regression")
g3 <- get_ROC_plot(lda_model, asteroids_train, "LDA")
g4 <- get_ROC_plot(qda_model, asteroids_train, "QDA")
g5 <- get_ROC_plot(reference_model, asteroids_holdout, "Null model")
g6 <- get_ROC_plot(logreg_model, asteroids_holdout, "Logistic regression")
g7 <- get_ROC_plot(lda_model, asteroids_holdout, "LDA")
g8 <- get_ROC_plot(qda_model, asteroids_holdout, "QDA")

(g1 + g5) / (g2 + g6) / (g3 + g7) / (g4 + g8) + plot_annotation('Training
```

In figure 2, the Null model is the same for both the training and the test data sets, as we would expect. Generally we see in the training set that the ROC curve is smoother and in the test set it is more staggered. This is simply due to the nature of the model being made based on the training data and so it will fit the training data better than it will the test data. We also see that generally across all models that the AUC is a bit less for logistic, LDA, and QDA on the test set for the same reason. This difference between the training sets and the test sets is not significant, in such a way that might lead us to believe that the models are overfitted. These models do not appear to be overfitted.

(ii.) Create a single plot that compares the ROC curves of the four models for the test set. Which model separates the two classes best?

```
bind_rows(roc_curve(augment(lda_model,asteroids_holdout),
                    truth=Hazardous,
                    .pred_TRUE,
                    event_level="second") %>%
            mutate(model="LDA"),
          roc_curve(augment(qda_model,asteroids_holdout),
                    truth=Hazardous,
```

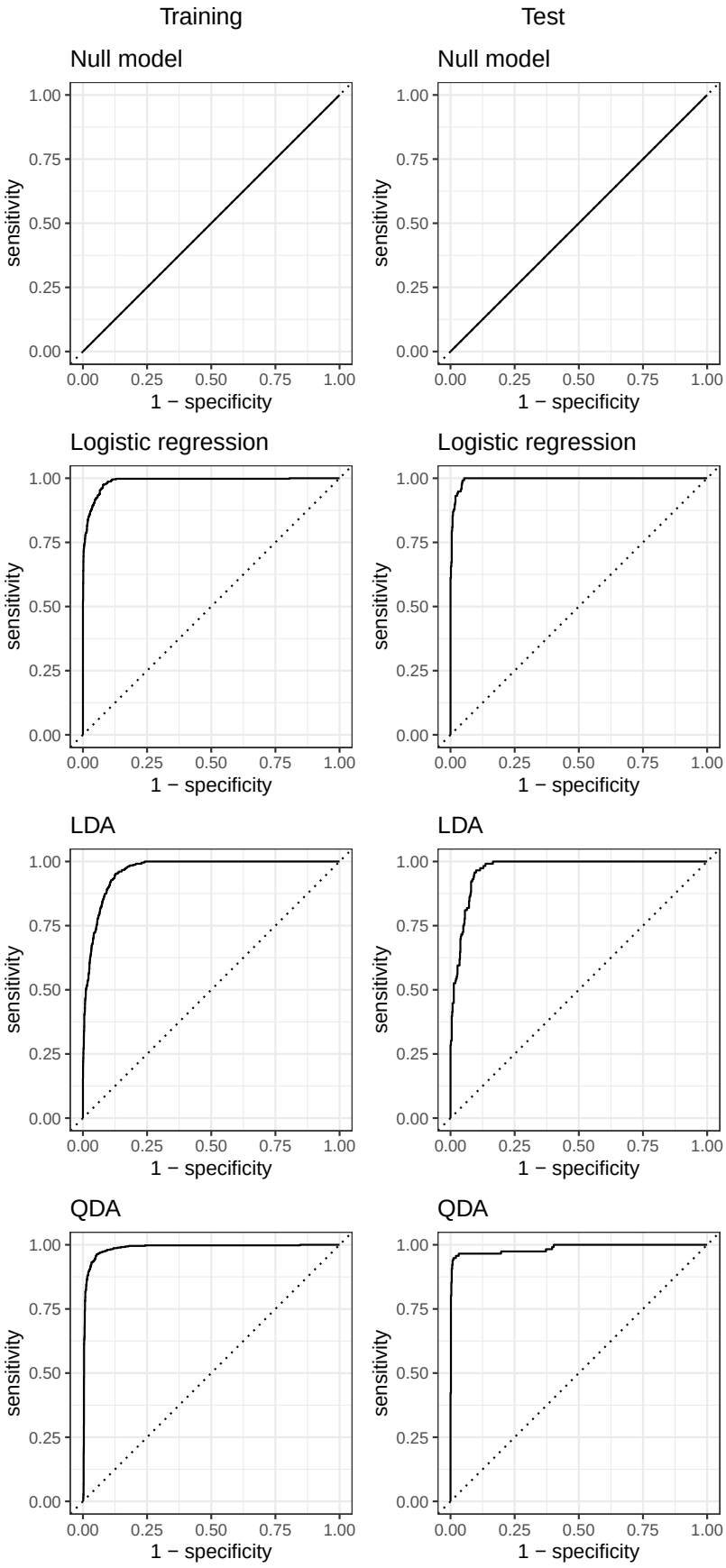

Figure 2: ROC curves by models for training and test sets

```
                    .pred_TRUE,
                    event_level="second") %>%
            mutate(model="QDA"),
          roc_curve(augment(logreg_model, asteroids_holdout),
                    truth=Hazardous,
                    .pred_TRUE,
                    event_level="second") %>%
            mutate(model="Logistic"),
          roc_curve(augment(reference_model,asteroids_holdout),
                    truth=Hazardous,
                    .pred_TRUE,
                    event_level="second") %>%
            mutate(model="Reference")) %>%
#autoplot(aes(color=model))
ggplot(aes(x=1-specificity,y=sensitivity,color=model))+
geom_line()+
labs(title = "ROC Curves across Models")
```

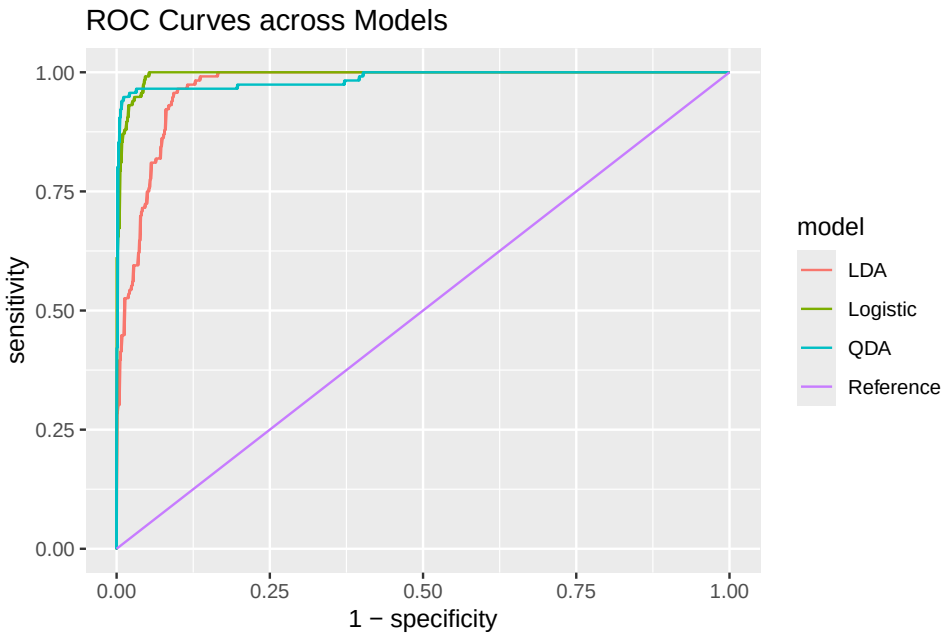

Figure 3: ROC curves of the four classification models determined using the holdout set.

For figure 3: Obviously the null model can be ruled out as it classifies at random. The model that is easiest to rule out is LDA, as it has a lower sensitivity and a lower specificity (higher 1 - specificity). The QDA and the logistic regression are a little more difficult to distinguish from the plot alone, but the addition of the calculated AUC from part c table 1 allows us to confirm that the logistic regression just beats out the QDA, with an AUC of 0.995. QDA has an AUC of 0.986. There is still room for debate about which model does the best at predicting the classes. The ROC curve and AUC are just a couple of measures of predictive performance. From part c table 1 we can see that QDA actually has higher values for accuracy and for kappa, where higher values indicate better predictive capability. It is possible that since QDA has two metrics that show better performance than the logistic regression that it could be the better option. Given what I have learned later in this assignment in part iii, that the data is imbalanced, which would

mean accuracy would be skewed, I would say that the logistic model would be better to represent this data.

(iii.) For each model, determine the threshold that maximizes the J-index using the training set. Why is the J-index a better metric than accuracy in this case? Create plots that show the dependence of the J-index from the threshold.

```
performance_reference<-reference_model %>%
                    augment(asteroids_train) %>%
                    probably::threshold_perf(Hazardous, .pred_TRUE,
                    thresholds=seq(0.05, 0.95, 0.01), event_level="second",
                    metrics=metric_set(j_index))
reference_max_j_index <- performance_reference %>%
    #filter(.metric == "j_index") %>%
    filter(.estimate == max(.estimate))

performance_logreg<-logreg_model %>%
                    augment(asteroids_train) %>%
                    probably::threshold_perf(Hazardous, .pred_TRUE,
                    thresholds=seq(0.05, 0.95, 0.01), event_level="second",
                    metrics=metric_set(j_index))
logreg_max_j_index <- performance_logreg %>%
    #filter(.metric == "j_index") %>%
    filter(.estimate == max(.estimate))

performance_lda<-lda_model %>%
                    augment(asteroids_train) %>%
                    probably::threshold_perf(Hazardous, .pred_TRUE,
                    thresholds=seq(0.05, 0.95, 0.01), event_level="second",
                    metrics=metric_set(j_index))
lda_max_j_index <- performance_lda %>%
    #filter(.metric == "j_index") %>%
    filter(.estimate == max(.estimate))

performance_qda<-qda_model %>%
                    augment(asteroids_train) %>%
                    probably::threshold_perf(Hazardous, .pred_TRUE,
                    thresholds=seq(0.05, 0.95, 0.01), event_level="second",
                    metrics=metric_set(j_index))
qda_max_j_index <- performance_qda %>%
    #filter(.metric == "j_index") %>%
    filter(.estimate == max(.estimate))

Names<-c("Logistic Regression","LDA","QDA")
metrics_table(bind_cols(Names, bind_rows(logreg_max_j_index,lda_max_j_index,qda_max_j_index)),
              "Thresholds")
```

```
## New names:
## * `` -> `...1`
```

Table 2: Thresholds

...1	.threshold	j_index
Logistic Regression	0.12	0.896
LDA	0.22	0.823

QDA	0.31	0.907

In the table 2 we see that the thresholds that maximize the J-index is different for each model. For logisitic regression the threshold that results in the maximum j_index (0.8962) is 0.12. The threshold that maximizes the j_index (0.8234) for LDA is 0.22. The threshold that maximizes the j_index (0.9075) for QDA is 0.31.

The J-index is (sensitivity + specificity) - 1. This means we want the highest sensitivity and specificity, just like the ROC Curve and AUC. Of our 2,952 observations in the training dataset 2,502 of them are not Hazardous (84.76%) and the remaining 450 are Hazardous (15.24%). The classes are imbalanced in our training data set. This means that accuracy will be affected by the imbalance, as the model is more likely to classify as not Hazardous, since it is the majority class. This means that it will correctly classify all of the not Hazardous asteroids and because these make up a large portion of the dataset it will automatically lead to an accuracy of similarly large proportion. It will however miss the predicted Hazardous when they really are Hazardous. Because j_index is based on specificity and sensitivity it is not as easily affected by the imbalance of the data. The specificity will capture the proportion of not Hazardous correctly classified by the model and the sensitivity will capture the proportion of Hazardous correctly classified by the model. This combination provides a more robust understanding of the test results.

```
threshold_scan <- function(model, data, model_name) {
    model %>%
        augment(data) %>%
        probably::threshold_perf(Hazardous, .pred_TRUE,
            thresholds=seq(0.05, 0.95, 0.01), event_level="second",
            metrics=metric_set(j_index)) %>%
    ggplot(aes(x=.threshold, y=.estimate)) +
        geom_line() +
        labs(title=model_name, x="Threshold", y="Accuracy")
}
t0<-threshold_scan(reference_model,asteroids_train,"Reference Model")
t1 <- threshold_scan(logreg_model, asteroids_train, "Logistic regression")
t2 <- threshold_scan(lda_model, asteroids_train, "LDA")
t3 <- threshold_scan(qda_model, asteroids_train, "QDA")
t0 + t1 + t2 + t3
```

In figure 4 we see for all models, besides the null model, we see that as threshold increases our accuracy decreases.

(iv.) Determine the accuracy, sensitivity, specificity, and J-index for each model at the determined thresholds. Which model performs best? How does this compare to the result from the ROC curves?

```
augment_model<-function(model,data,thresh_level){
  model %>%
    augment(data) %>%
    mutate(pred=as.factor(ifelse(.pred_TRUE>= thresh_level,TRUE,FALSE)))
}

logreg2<-augment_model(logreg_model,asteroids_holdout,0.12)
lda2<-augment_model(lda_model,asteroids_holdout,0.22)
qda2<-augment_model(qda_model,asteroids_holdout,0.31)
```

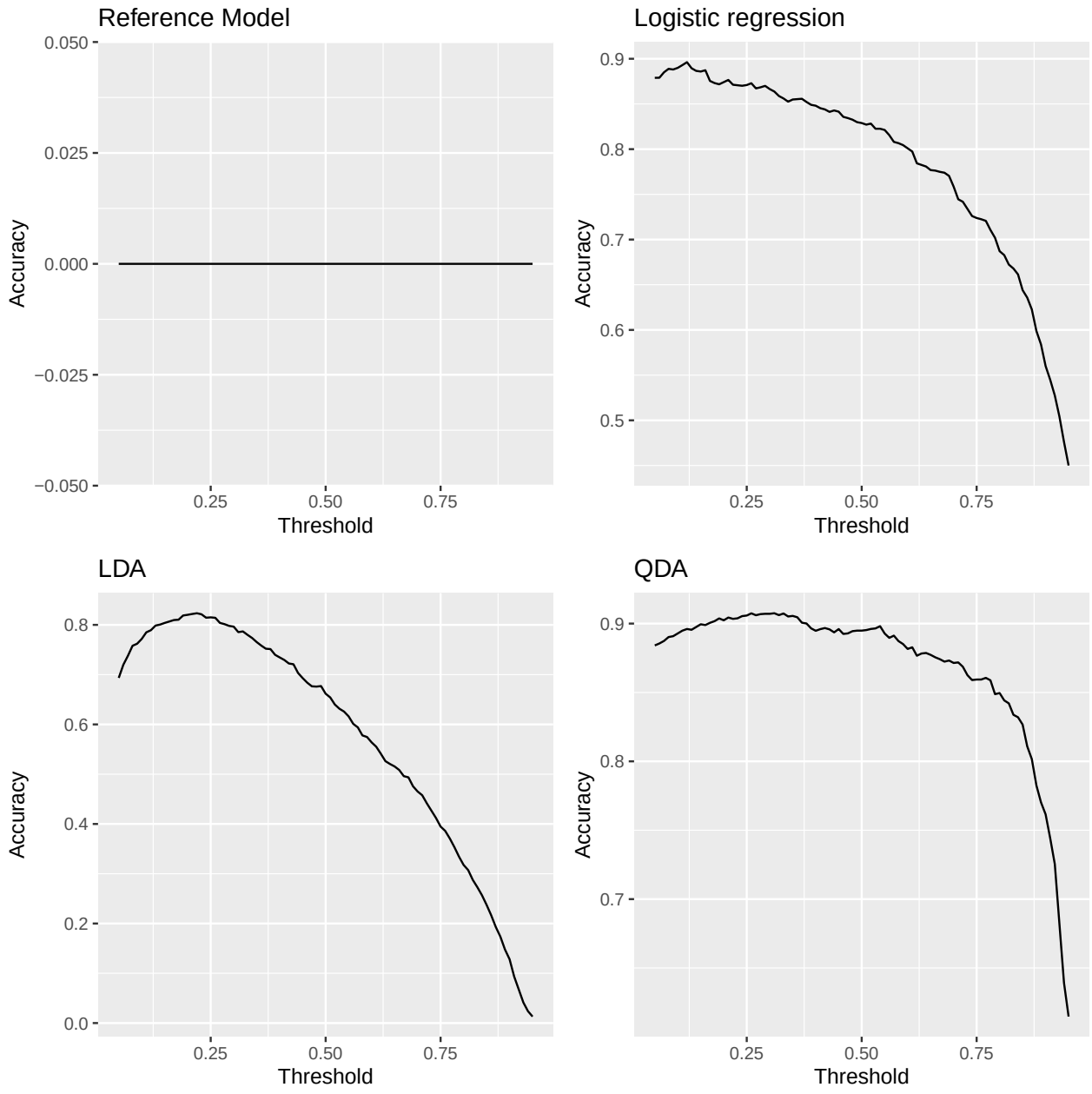

Figure 4: Accuracy as a function of threshold for the three classification models

```r
#This outputs a function
class_metrics<-metric_set(accuracy,sensitivity,specificity,j_index)
calculate_metrics2 <- function(model, train, test, model_name,thresh_level) {
    roc_auc(model %>% augment(train), Hazardous, .pred_TRUE, event_level="second")
    bind_rows(
        bind_cols(
            model=model_name,
            dataset="train",
            class_metrics(model %>% augment_model(train,thresh_level),
                          truth=Hazardous,
                          estimate=pred),
        ),
        bind_cols(
            model=model_name,
            dataset="train",
            roc_auc(model %>% augment_model(train,thresh_level),
                    Hazardous,
                    .pred_TRUE,
                    event_level="second"),
        ),
        bind_cols(
            model=model_name,
            dataset="test",
            class_metrics(model %>% augment_model(test,thresh_level),
                          truth=Hazardous,
                          estimate=pred),
        ),
        bind_cols(
            model=model_name,
            dataset="test",
            roc_auc(model %>% augment_model(test,thresh_level),
                    Hazardous,
                    .pred_TRUE,
                    event_level="second"),
        ),
    )
}
```

```r
#accuracy, sensitivity, specificity, and J-index
ASSJ<-bind_rows(calculate_metrics2(logreg_model,asteroids_train,asteroids_holdout,"Logistic",0.12),
          calculate_metrics2(lda_model,asteroids_train,asteroids_holdout,"LDA",0.22),
          calculate_metrics2(qda_model,asteroids_train,asteroids_holdout,"QDA",0.31))
metrics_table(ASSJ,"Threshold Metrics")
```

Table 3: Threshold Metrics

model	dataset	accuracy	sensitivity	specificity	j_index	roc_auc
Logistic	train	0.929	0.920	0.976	0.896	0.988
Logistic	test	0.945	0.934	1.000	0.934	0.995
LDA	train	0.884	0.871	0.952	0.823	0.965
LDA	test	0.897	0.883	0.974	0.857	0.970
QDA	train	0.949	0.947	0.961	0.907	0.987
QDA	test	0.964	0.963	0.966	0.929	0.986

In table 3 the model that performs the best on the test data is still somewhat open to debate between logistic and QDA, as logistic has better values for specificity, j index, and AUC, but QDA has better values for accuracy and sensitivity. Given what we know about how accuracy is affected by the imbalance of the data on training the model, I would say that logistic model does a better job in it's predicting capability as it has more metrics in it's favor, particularly j index that utilizes both sensitivity and specificity which each represent the proportions of correctly classified Hazardous and not Hazardous, respectively. These results are the same conclusions I made with the ROC curves, as the curves will show the balance of sensitivity and specificity. The threshold value isn't shown on the curves, but we know that the best threshold value will be when the sensitivity and specificity are the highest.

3. Handling class imbalance in classification problems

Write a short essay (about 1 page) on the topic of handling class imbalance in classification problems. Here are a few questions to guide your research:

- What is class imbalance, and why is it a problem in classification problems?
- What are common strategies to handle class imbalance?
- What are appropriate evaluation metrics for imbalanced classification problems?

Class imbalance is where the percent of the total number of observations varies for each class from the training data. The amount of imbalance can vary widely, from being slightly imbalanced to being severly imbalanced. This is applicable for both binary responses or multiclass responses. It is a problem for classification problems, because the statistical learning/machine learning models assume that the responses will be evenly distributed among the classes. This will lead to the model learning the characteristics of the majority class(es) and not learning the characteristics of the minority class as well. The model will not as accurately predict outcomes for new data/test data. This is a problem, because naturally what we often want to learn more about is in regards to the minority class.

There is no one correct strategy to handle class imbalance, but one of the first things that should be attempted is fitting different models, one should not assume that a certain model will be what will fit and predict for the data the best. If you have a lot of data in your training dataset you could remove some observations that indicate the majority class. This is called under-sampling. Another option is to add observations for the minority class, over-sampling. This can be done by by collecting more data or generating synthetic samples. One can do this by repeating some of the minority class observations already in the training dataset or utilizing algorithms that are available, like Synthetic Minority Over-Sampling Technique (SMOTE). There are also methods available that impose penalties on the model when it classifies incorrectly for the minority class during training. This makes the model focus more on the minority class.

There are many metrics available that would be better choices than others to help us understand our model when the training data is imbalanced.A simple and thorough metric is the confusion matrix, which summarizes the number of correct and incorrect classifications based on our test data. As we have seen in class kappa can be used, as it is a measure of classification accuracy, but is normalized by the imbalance of the classes in the data. ROC curves, which display both specificity and sensitivity are useful to help determine the threshold that should be utilized. A couple of other metrics that I stumbled upon were precision and recall, these measure the classifiers exactness and completeness, respectively. Another metric called the F score utilizes the weighted average of these two metrics.

You can use the following resource to get started:

References utilized, last date of access June 9th, 2024.

- https://machinelearningmastery.com/what-is-imbalanced-classification/
- https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/