Predictive Modeling

Chapter 13: Nonlinear Classification Models

STA 6543

The University of Texas at San Antonio

Overview

- Part I: General Strategies
- Part II: Regression Models
  - Chapter 6: Linear Regression and Its Cousins
  - Chapter 7: Nonlinear Regression Models
  - Chapter 8: Regression Trees and Rule-Based Models
- Part III: Classification Models
  - Chapter 11: Measuring Performance in Classification Models
  - Chapter 12: Discriminant Analysis and Other Linear Classification Models
  - Chapter 13: Nonlinear Classification Models
  - Chapter 14: Classification Trees and Rule-Based Models

Nonlinear Classification Models

- In Chapter 12, we discussed the models that were intrinsically linear — the structure of the model would produce *linear* class boundaries unless nonlinear functions of the predictors were manually specified.
- In this chapter, we deal with some intrinsically nonlinear models
  - Nonlinear discriminant analysis LDA $\frac{\mu_i + \mu_j}{2} > x \swarrow \sigma_i^2 = \sigma_j^2 = \sigma^2$
    - quadratic discriminant analysis (QDA)
    - regularized discriminant analysis (RDA)
    - mixture discriminant analysis (MDA).
  - Naïve Bayes
  - K-nearest neighbors
  - Neural networks
  - Flexible discriminant analysis
  - Support vector machines
- R demonstrations for the stock market data

Nonlinear discriminant analysis

Chapter 13: Nonlinear Classification Models

A general form of discriminant analysis

Bayes' theorem

- Discriminant analysis is based on

prior

density function of data in class k

$$\Pr(Y = k | X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

  - When we assume $f_k(x)$ to be Gaussian distributions with the *same* covariance matrix $\Sigma_k = \Sigma$ in each class, it leads to Linear Discriminant Analysis (LDA)
  - We assume $f_k(x)$ to be Gaussian distributions with the different covariance matrix Σ_k in each class, it leads to Quadratic Discriminant Analysis (QDA)
  - With $f_k(x) = \prod_{j=1}^{p} f_{jk}(x_j)$ (i.e., conditional independence model) in each class, it leads to naïve Bayes (NB). For Gaussian, this means Σ is diagonal.
  - Nonparametric models can also be used $f_k(x)$.

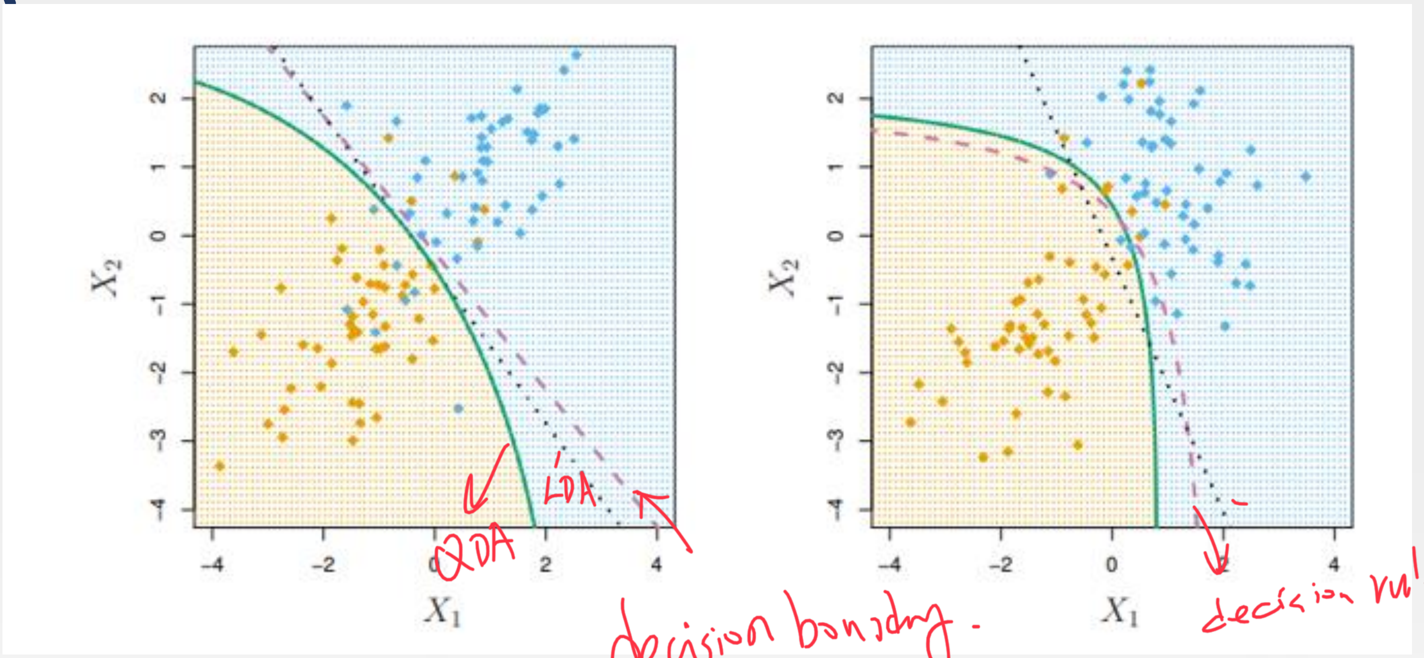
strong

is often violated. However this violation does not impact NB.

LDA, QDA, and RDA

- Recall that LDA needs the assumption that the predictors in each class shared a *same* covariance structure $\Sigma_k = \Sigma$ and that the class boundaries were *linear* functions of the predictors.
- QDA could relax these assumptions by making the decision boundaries become quadratically *curvilinear* in the predictor space.
- The increased discriminant function complexity may improve model performance for many problems.
- However, QDA brings a new restriction that the number of predictors p must be less than the number of cases within each class, so that each Σ_k is invertible. may not be used in high-dimensional case.

LDA vs. QDA



decision boundary.

decision rule.

- If the decision boundary (purple dashed) is linear, it is more accurately approximated by LDA (black dotted) than by QDA (green solid).
- If the decision boundary (purple dashed) is nonlinear, it is more accurately approximated by QDA (green solid) than by LDA (black dotted).

LDA, QDA, and RDA

Single multivariate Normal

- LDA and QDA minimize the total probability of misclassification assuming that the data can truly be separated by hyperplanes or quadratic surfaces.
- Reality may be, however, that the data are best separated by structures somewhere between linear and quadratic class boundaries.
- RDA, proposed by Friedman (1989), is one way to bridge the separating surfaces between LDA and QDA such that

$\lambda = 0 \quad \tilde{\Sigma}_l = \Sigma \quad (LDA)$

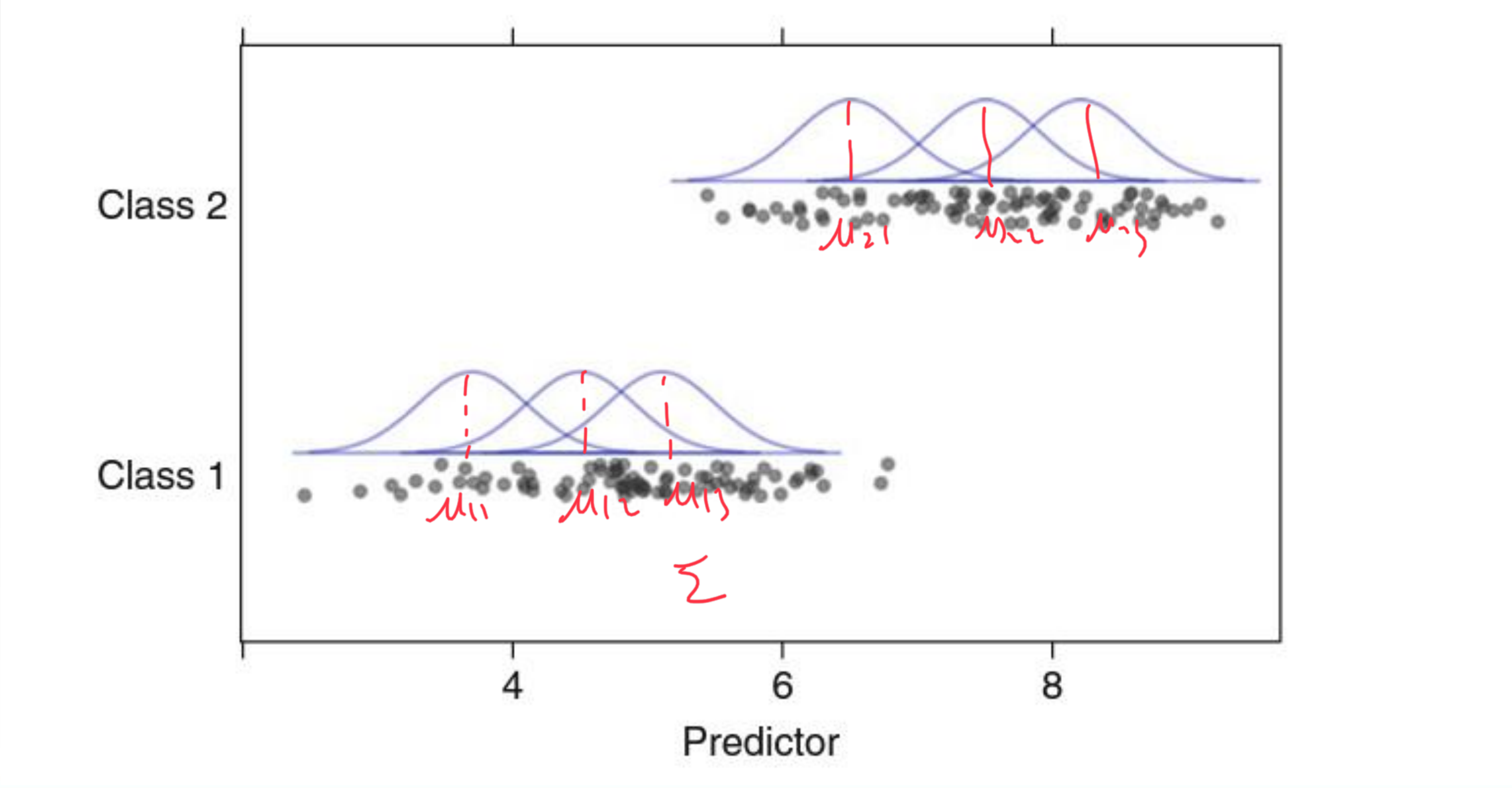
$$\tilde{\Sigma}_l = \lambda \Sigma_l + (1 - \lambda)\Sigma,$$

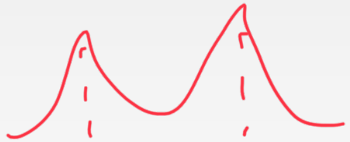
$\lambda = 1 \quad \tilde{\Sigma}_l = \Sigma_l \quad (QDA)$

where Σ_l is the covariance matrix of the l-th class and Σ is the pooled covariance matrix across all classes.

Mixture discriminant analysis (MDA)

- MDA is as an extension of LDA by allowing each class to be represented by *multiple* multivariate normal distributions.





Mixture discriminant analysis (MDA)

- These distributions can have different means but, like LDA, the covariance structures are assumed to be the same.
- We would specify how many different distributions should be used and the MDA model would determine their optimal locations in the predictor space.
- The number of distributions per class is the tuning parameter for the model (they need not be equal per class).
- Also, similar to LDA, using ridge- and lasso-like penalties to MDA would integrate feature selection into the MDA model.

Variable selection

The stock market data

```
# required packages
library(AppliedPredictiveModeling)
library(caret)
library(ISLR) #the stock market data
library(pROC) #roc
library(MASS) #lda
library(klaR) #rda
library(mda)  #mda
library(earth) #fda
library(e1071)
library(kernlab) #SVM

###Data splitting
train = which(Year<2005)
Smarket.train= Smarket[train,]; # observations before 2005 are served as test data.
Smarket.test= Smarket[-train,]; # observations from 2005 are served as test data.
```

Train control function

```
### Create a control function that will be used across models.
set.seed(100)
ctrl <- trainControl(method = "LGOCV",
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE,
                     savePredictions = TRUE)

# LGOCV: Repeated Train/Test Splits Estimated (25 reps, 75%)
```

QDA

```
set.seed(476)
QDATune <- train(x = as.matrix(Smarket.train[,1:8]),
y = Smarket.train$Direction,
method = "qda",
metric = "ROC",
trControl = ctrl)
QDATune

#library(pROC)
### Predict the test set based the logistic regression
Smarket.test$QDA <- predict(QDATune,Smarket.test, type = "prob")[,1]
#ROC for QDA model
QDAROC <- roc(Smarket.test$Direction, Smarket.test$QDA)
plot(QDAROC, col=1, lty=1, lwd=2)

#Confusion matrix of QDA model
confusionMatrix(data = predict(QDATune, Smarket.test), reference =
Smarket.test$Direction)
```

response from the test data

QDA output

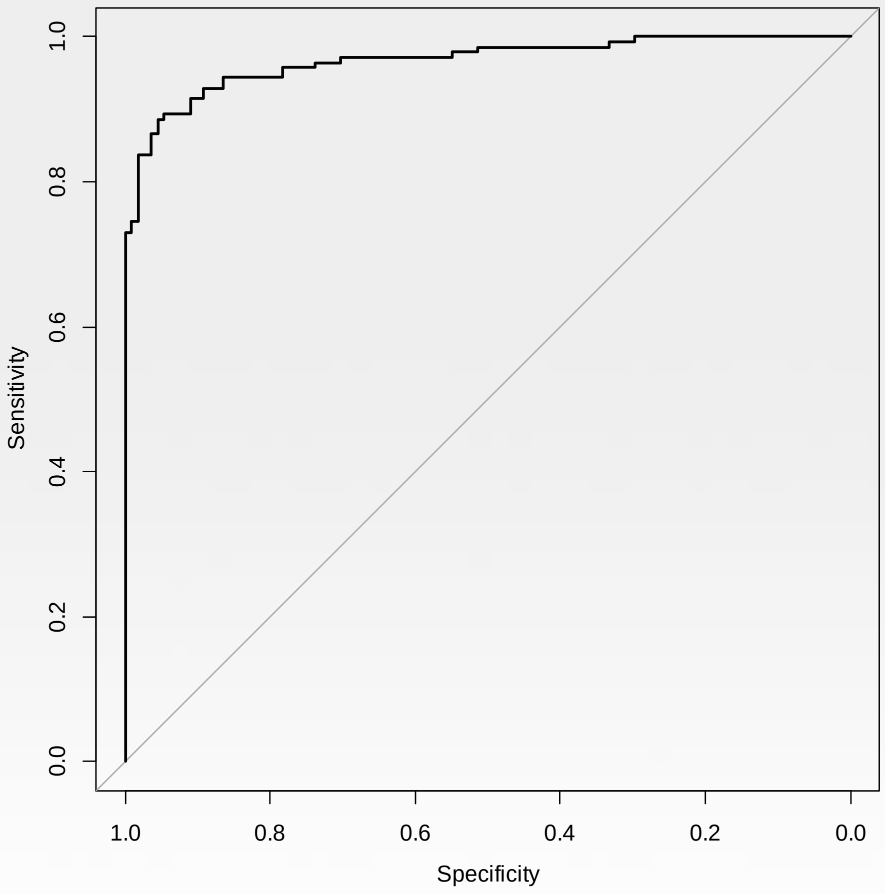
```
>  QDATune
Quadratic Discriminant Analysis

998 samples
  8 predictor
  2 classes: 'Down', 'Up'

No pre-processing
Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
Summary of sample sizes: 750, 750, 750, 750, 750, 750, ...
Resampling results:

  ROC        Sens       Spec
  0.9873849  0.9147541  0.9593651
```

ROC from QDA



Confusion matrix from QDA

LDA is superior than QDA

```
Confusion Matrix and Statistics

          Reference
Prediction Down  Up
      Down  101  12
      Up     10 129

               Accuracy : 0.9127
                 95% CI : (0.8708, 0.9445)
    No Information Rate : 0.5595
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8232

 Mcnemar's Test P-Value : 0.8312

            Sensitivity : 0.9099
            Specificity : 0.9149
         Pos Pred Value : 0.8938
         Neg Pred Value : 0.9281
             Prevalence : 0.4405
         Detection Rate : 0.4008
   Detection Prevalence : 0.4484
      Balanced Accuracy : 0.9124

       'Positive' Class : Down
```

22

← QDA

```
Confusion Matrix and Statistics

          Reference
Prediction Down  Up
      Down  100   2
      Up     11 140

               Accuracy : 0.9524
                 95% CI : (0.9183, 0.9752)
    No Information Rate : 0.5595
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9025

 Mcnemar's Test P-Value : 0.009375

            Sensitivity : 0.9009
            Specificity : 0.9929
         Pos Pred Value : 0.9901
         Neg Pred Value : 0.9272
             Prevalence : 0.4405
         Detection Rate : 0.3968
   Detection Prevalence : 0.4008
      Balanced Accuracy : 0.9469

       'Positive' Class : Down
```

13

← LDA

RDA

```
########################regularized discriminant analysis########################
 set.seed(476)
 RDATune <- train(x = as.matrix(Smarket.train[,1:8]),
 y = Smarket.train$Direction,
 method = "rda",
 preProc = c('center', 'scale'),
 metric = "ROC",
 trControl = ctrl)
 RDATune
```

RDA output

(λ tuning Paramter)

```
Regularized Discriminant Analysis

998 samples
  8 predictor
  2 classes: 'Down', 'Up'

Pre-processing: centered (8), scaled (8)
Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
Summary of sample sizes: 750, 750, 750, 750, 750, 750, ...
Resampling results across tuning parameters:

  gamma  lambda  ROC        Sens       Spec
  0.0    0.0     0.9873849  0.9147541  0.9593651
  0.0    0.5     0.9934765  0.9275410  0.9768254
  0.0    1.0     0.9962972  0.9367213  0.9857143
  0.5    0.0     0.9869945  0.9052459  0.9653968
  0.5    0.5     0.9921364  0.9134426  0.9736508
  0.5    1.0     0.9948426  0.9245902  0.9803175
  1.0    0.0     0.9889045  0.8947541  0.9622222
  1.0    0.5     0.9907494  0.9036066  0.9666667
  1.0    1.0     0.9918111  0.9091803  0.9695238

ROC was used to select the optimal model using the largest value.
The final values used for the model were gamma = 0 and lambda = 1.
```

MDA

```
######################################mixture discriminant analysis##########################
 set.seed(476)
 MDATune <- train(as.matrix(Smarket.train[,1:8]),
 y =  Smarket.train$Direction,
 method = "mda",
 tuneGrid = expand.grid(.subclasses = 3:10),
 metric = "ROC",
 trControl = ctrl)
 MDATune
```

the number of distribution in each class from 3 to 10

MDA output

```
Mixture Discriminant Analysis

998 samples
  8 predictor
  2 classes: 'Down', 'Up'

No pre-processing
Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
Summary of sample sizes: 750, 750, 750, 750, 750, 750, ...
Resampling results across tuning parameters:

  subclasses  ROC        Sens       Spec
   3          0.9901379  0.9186885  0.9669841
   4          0.9882201  0.9157377  0.9660317
   5          0.9865600  0.9068852  0.9673016
   6          0.9864507  0.9039344  0.9650794
   7          0.9843039  0.9013115  0.9644444
   8          0.9855295  0.9075410  0.9650794
   9          0.9827791  0.9052459  0.9612698
  10          0.9842857  0.9036066  0.9612698

ROC was used to select the optimal model using the largest value.
The final value used for the model was subclasses = 3.
```