Homework 1 resubmit

Yuning Li

2024-02-03

```
library('class')

library('dplyr')
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

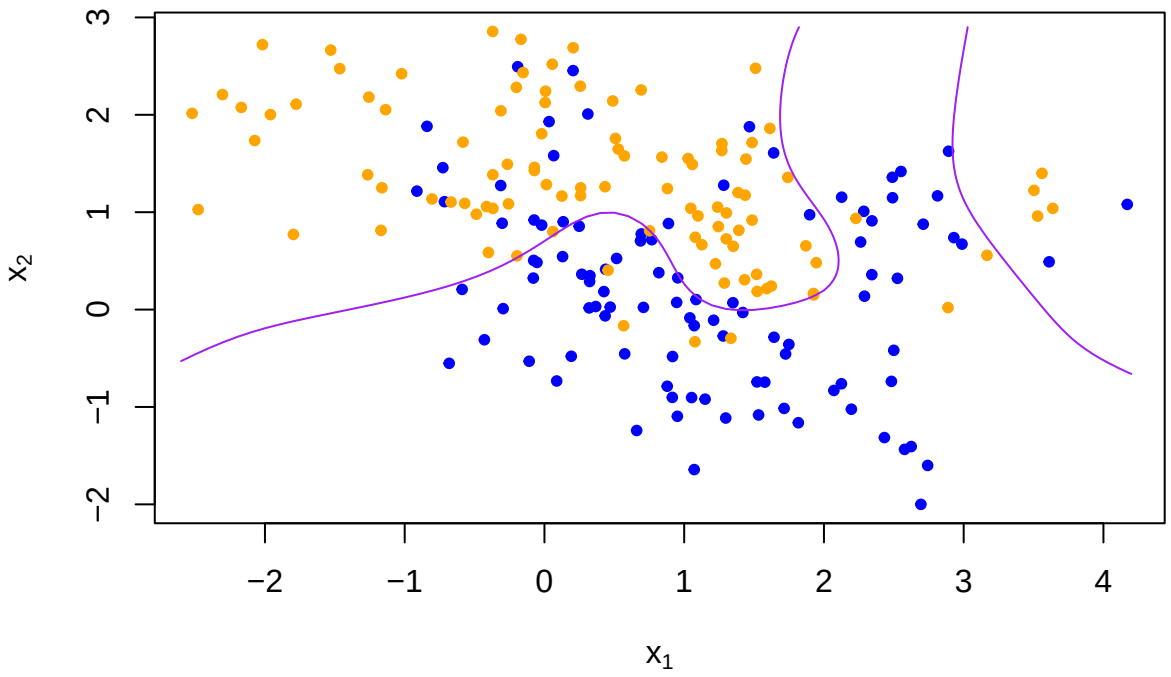
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
load(url('https://biostat.app.vumc.org/wiki/pub/Main/CourseDSI5640/ESL.mixture.rda'))
dat <- ESL.mixture
str(dat)
```

```
## List of 8
##  $ x       : num [1:200, 1:2] 2.5261 0.367 0.7682 0.6934 -0.0198 ...
##  $ y       : num [1:200] 0 0 0 0 0 0 0 0 0 0 ...
##  $ xnew    : 'matrix' num [1:6831, 1:2] -2.6 -2.5 -2.4 -2.3 -2.2 -2.1 -2 -1.9 -1.8 -1.7 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:6831] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:2] "x1" "x2"
##  $ prob    : num [1:6831] 3.55e-05 3.05e-05 2.63e-05 2.27e-05 1.96e-05 ...
##   ..- attr(*, ".Names")= chr [1:6831] "1" "2" "3" "4" ...
##  $ marginal: num [1:6831] 6.65e-15 2.31e-14 7.62e-14 2.39e-13 7.15e-13 ...
##   ..- attr(*, ".Names")= chr [1:6831] "1" "2" "3" "4" ...
##  $ px1     : num [1:69] -2.6 -2.5 -2.4 -2.3 -2.2 -2.1 -2 -1.9 -1.8 -1.7 ...
##  $ px2     : num [1:99] -2 -1.95 -1.9 -1.85 -1.8 -1.75 -1.7 -1.65 -1.6 -1.55 ...
##  $ means   : num [1:20, 1:2] -0.2534 0.2667 2.0965 -0.0613 2.7035 ...
```

```
plot_mix_data <- function(dat, datboot=NULL) {
  if(!is.null(datboot)) {
    dat$x <- datboot$x
    dat$y <- datboot$y
  }
  plot(dat$x[,1], dat$x[,2],
       col=ifelse(dat$y==0, 'blue', 'orange'),
       pch=20,
       xlab=expression(x[1]),
       ylab=expression(x[2]))
  ## draw Bayes (True) classification boundary
```

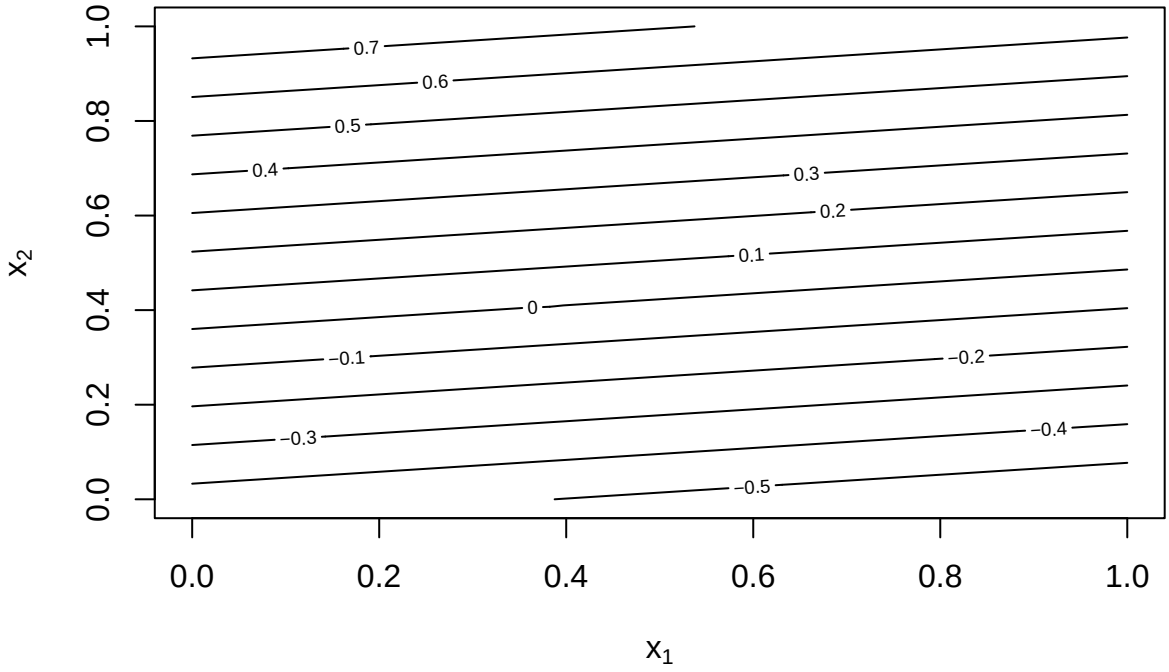
```
  prob <- matrix(dat$prob, length(dat$px1), length(dat$px2))
  cont <- contourLines(dat$px1, dat$px2, prob, levels=0.5)
  rslt <- sapply(cont, lines, col='purple')
}

plot_mix_data(dat)
```



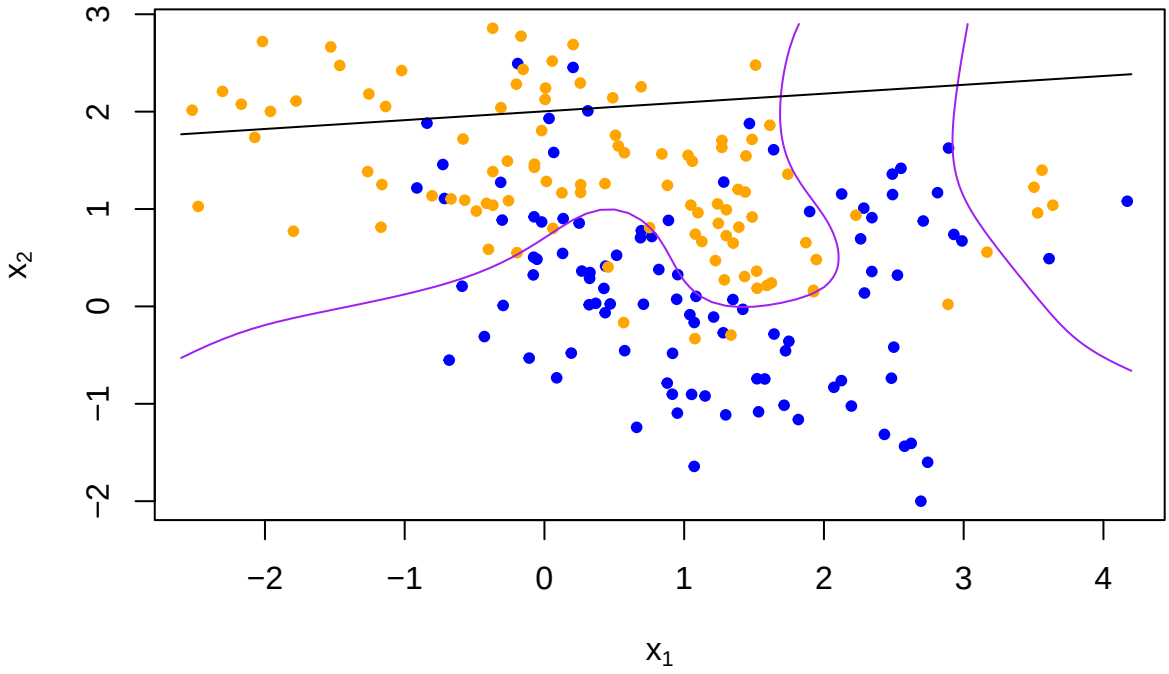

```
####################question 1: rewrite function by "lm" ####################

# Fit linear classifier using lm

fit_lc <- function(y, x) {
  data_df <- data.frame(y = y, x)
  model <- lm(y ~ ., data = data_df)
  return(coef(model)[-1])  # Exclude intercept term
}

# Make predictions from linear classifier
predict_lc <- function(x, beta) {
  cbind(1, x) %*% c(0, beta)  # Include intercept term
}

## fit model to mixture data and make predictions
lc_beta <- fit_lc(dat$y, dat$x)
lc_pred <- predict_lc(dat$xnew, lc_beta)

## reshape predictions as a matrix
lc_pred <- matrix(lc_pred, length(dat$px1), length(dat$px2))
```

```
## plot contour plot for linear classifier
contour(
  lc_pred,
  xlab = expression(x[1]),
```

```
  ylab = expression(x[2])
)
```




```
## find the contours in 2D space such that lc_pred == 0.5
lc_cont <- contourLines(dat$px1, dat$px2, lc_pred, levels = 0.5)

## plot data and decision surface for linear classifier
plot_mix_data(dat)
sapply(lc_cont, lines)
```

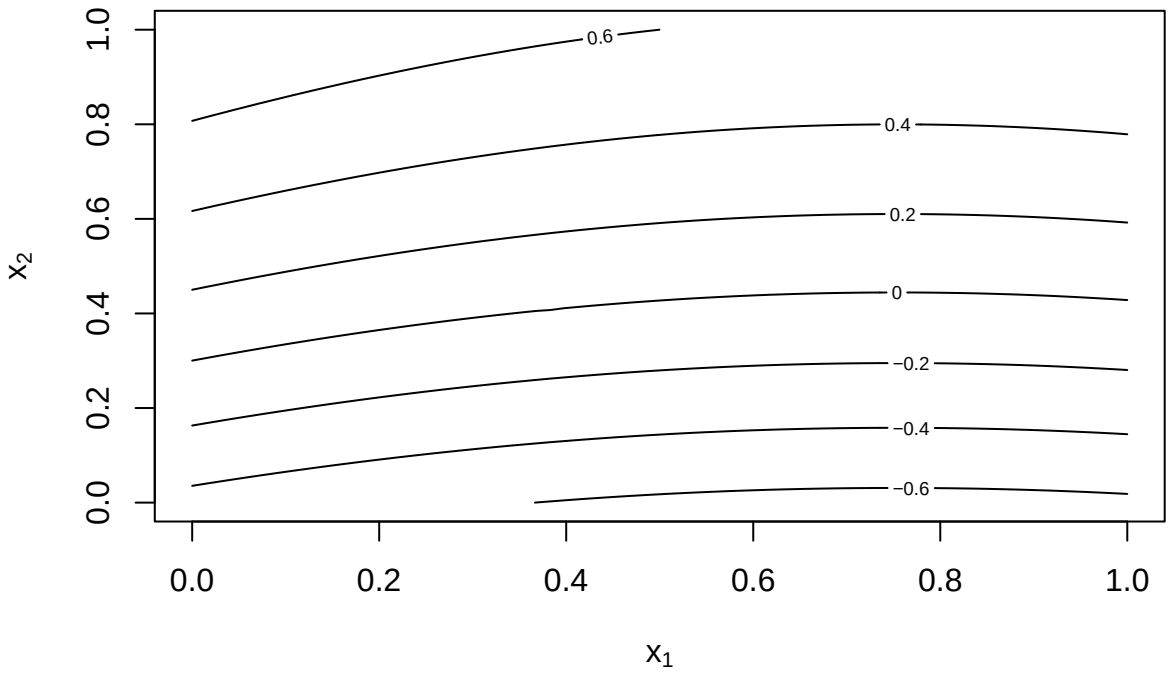



```
## [[1]]
```

```
## NULL
```

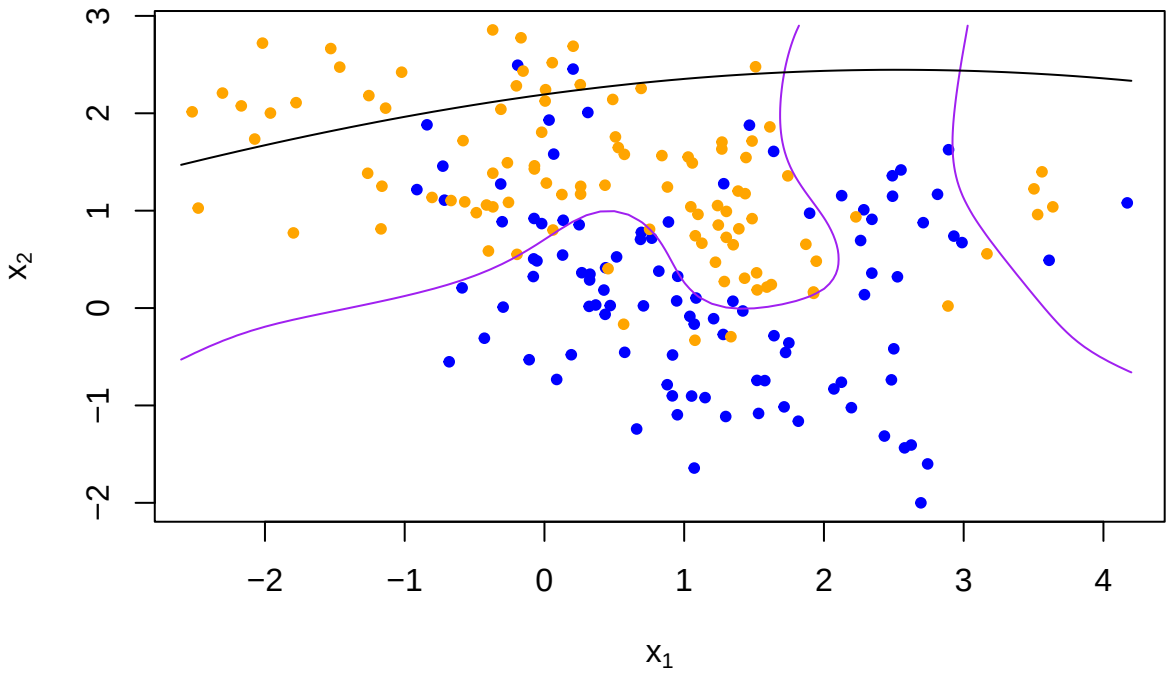
```
################question 2: add squared terms###############
## Make the linear classifier more flexible by adding squared terms
fit_flexible_lc <- function(y, x) {
  data_df <- data.frame(y = y, x, x1_squared = x[, 1]^2, x2_squared = x[, 2]^2)
  model <- lm(y ~ . + x1_squared + x2_squared, data = data_df)
  return(coef(model)[-1])  # Exclude intercept term
}

## Re-write predict_lc for the flexible model
predict_flexible_lc <- function(x, beta) {
  cbind(1, x, x1_squared = x[, 1]^2, x2_squared = x[, 2]^2) %*% c(0, beta)  # Include intercept term
}

## fit model to mixture data and make predictions for the flexible model
lc_beta_flexible <- fit_flexible_lc(dat$y, dat$x)
lc_pred_flexible <- predict_flexible_lc(dat$xnew, lc_beta_flexible)

## reshape predictions as a matrix for the flexible model

lc_pred_flexible <- matrix(lc_pred_flexible, length(dat$px1), length(dat$px2))
```

```
## plot contour plot for the flexible model
contour(
  lc_pred_flexible,
  xlab = expression(x[1]),
  ylab = expression(x[2])
)
```




```
## find the contours in 2D space such that lc_pred_flexible == 0.5
lc_cont_flexible <- contourLines(dat$px1, dat$px2, lc_pred_flexible, levels = 0.5)

## plot data and decision surface for the flexible model
```

```
plot_mix_data(dat)
sapply(lc_cont_flexible, lines)
```




```
## [[1]]
## NULL
```

##################question 3############### #Adding squared terms for x1 and x2 in the linear model increases the model's flexibility, allowing it to better capture non-linear relationships in the data. This increased flexibility may lead to reduced bias but could also result in higher variance, potentially making the model more sensitive to noise in the training data.