# Lab 2

Daniel Tshiani

2025-05-25

```
library(ISLR)
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
```

## 1

```
load("../data/Auto-3.rda")
```

### b

```
library(dplyr)

Auto <- Auto %>%
  mutate(Economy = case_when(
    mpg <= 17 ~ "Heavy",
    mpg <= 22.75 ~ "OK",
    mpg <= 29 ~ "Eco",
    mpg > 29 ~ "Excellent"
  )) %>%
  mutate(Economy = as.factor(Economy)) %>%
  mutate(origin = as.factor(origin))
```

### c

```
n <-  nrow(Auto)
set.seed(1234)
training_data <-  sample(n, n/2)

train_set <- Auto[training_data, ]
test_set  <- Auto[-training_data, ]
```

### d

```r
library(class)

x_train <- train_set %>%
  select(mpg:origin)

x_test <- test_set %>%
  select(mpg:origin)

cl <- train_set$Economy

y_hat <- knn(train = x_train, test = x_test, cl = cl, k = 4)
```

```r
y <- test_set$Economy
y_test <- test_set$Economy

confusion_matrix <- table(y_test,y_hat)
print(confusion_matrix)
```

```
##            y_hat
## y_test      Eco Excellent Heavy OK
##   Eco        28         9     0  7
##   Excellent   9        37     0  4
##   Heavy       0         0    35  9
##   OK         13         1     4 40
```

```r
classification_rate <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print(paste("Classification rate:", round(classification_rate, 4)))
```

```
## [1] "Classification rate: 0.7143"
```

### e

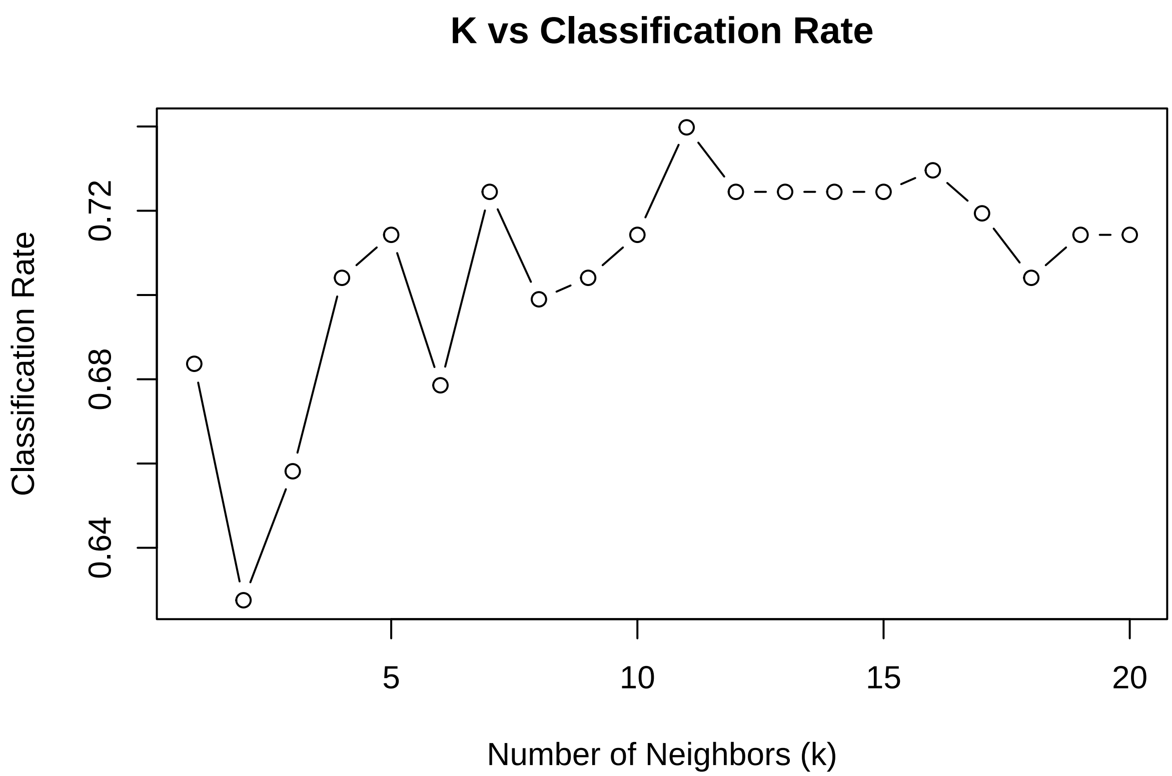
```r
accuracy_values <- numeric(20)

for (k in 1:20) {
  knn_pred <- knn(train = x_train, test = x_test, cl = cl, k = k)
  confusion_matrix <- table(Predicted = knn_pred, Actual = test_set$Economy)
  accuracy_values[k] <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
}

# Print accuracy for each k
print(accuracy_values)
```

```
##  [1] 0.6836735 0.6275510 0.6581633 0.7040816 0.7142857 0.6785714 0.7244898
##  [8] 0.6989796 0.7040816 0.7142857 0.7397959 0.7244898 0.7244898 0.7244898
## [15] 0.7244898 0.7295918 0.7193878 0.7040816 0.7142857 0.7142857
```

```r
# Optional: plot accuracy vs k
plot(1:20, accuracy_values, type = "b",
     xlab = "Number of Neighbors (k)",
     ylab = "Classification Rate",
     main = "K vs Classification Rate")
```




it looks like the classification is highest when k is 13.

# 2

## a

```
library(readr)
depression <- read.csv("../data/depression_data.csv")

depression <- na.omit(depression)
```

## b

```
depression <- depression%>%
  mutate(Diagnosis = as.factor(Diagnosis),
         Guardian_status = as.factor(Guardian_status),
         Gender = as.factor(Gender))
```

```
model <- glm(Diagnosis ~ Gender + Guardian_status + Cohesion_score + Depression_score,
             data = depression,
             family = binomial)
summary(model)
```

```
##
## Call:
## glm(formula = Diagnosis ~ Gender + Guardian_status + Cohesion_score +
##     Depression_score, family = binomial, data = depression)
##
## Coefficients:
```

```
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -2.13928    0.83048  -2.576   0.0100 **
## GenderMale        -0.31940    0.30568  -1.045   0.2961
## Guardian_status1 -0.70964    0.29379  -2.415   0.0157 *
## Cohesion_score    -0.01993    0.01160  -1.718   0.0859 .
## Depression_score  0.06851    0.01435   4.774  1.8e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 398.47  on 457  degrees of freedom
## Residual deviance: 334.73  on 453  degrees of freedom
## AIC: 344.73
##
## Number of Fisher Scoring iterations: 5
```

```r
predicted_probs <- predict(model, type = "response")
predicted_class <- ifelse(predicted_probs > 0.3, 1, 0)
table(True = depression$Diagnosis, Predicted = predicted_class)
```

```
##     Predicted
## True   0   1
##    0 349  37
##    1  38  34
```

```r
mean(predicted_class == depression$Diagnosis)  # Classification accuracy
```

```
## [1] 0.8362445
```

```r
1 - mean(predicted_class == depression$Diagnosis)  # Training error rate
```

```
## [1] 0.1637555
```

```r
conf_mat <- table(True = depression$Diagnosis, Predicted = predicted_class)
sensitivity <- conf_mat["1", "1"] / sum(conf_mat["1", ])
sensitivity
```

```
## [1] 0.4722222
```

### c

```r
set.seed(123)

n <- nrow(depression)
train_index <- sample(1:n, size = n * 0.7)

train_data <- depression[train_index, ]
test_data  <- depression[-train_index, ]
```

```r
model <- glm(Diagnosis ~ Gender + Guardian_status + Cohesion_score + Depression_score,
             data = train_data,
             family = binomial)
```

```r
test_probs <- predict(model, newdata = test_data, type = "response")
test_pred <- ifelse(test_probs > 0.3, 1, 0)
```

```r
test_actual <- test_data$Diagnosis
test_pred <- factor(test_pred, levels = levels(test_actual))
```

```r
conf_matrix <- table(True = test_actual, Predicted = test_pred)
print(conf_matrix)
```

```
##     Predicted
## True   0   1
##    0 111   7
##    1  13   7
```

### d

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

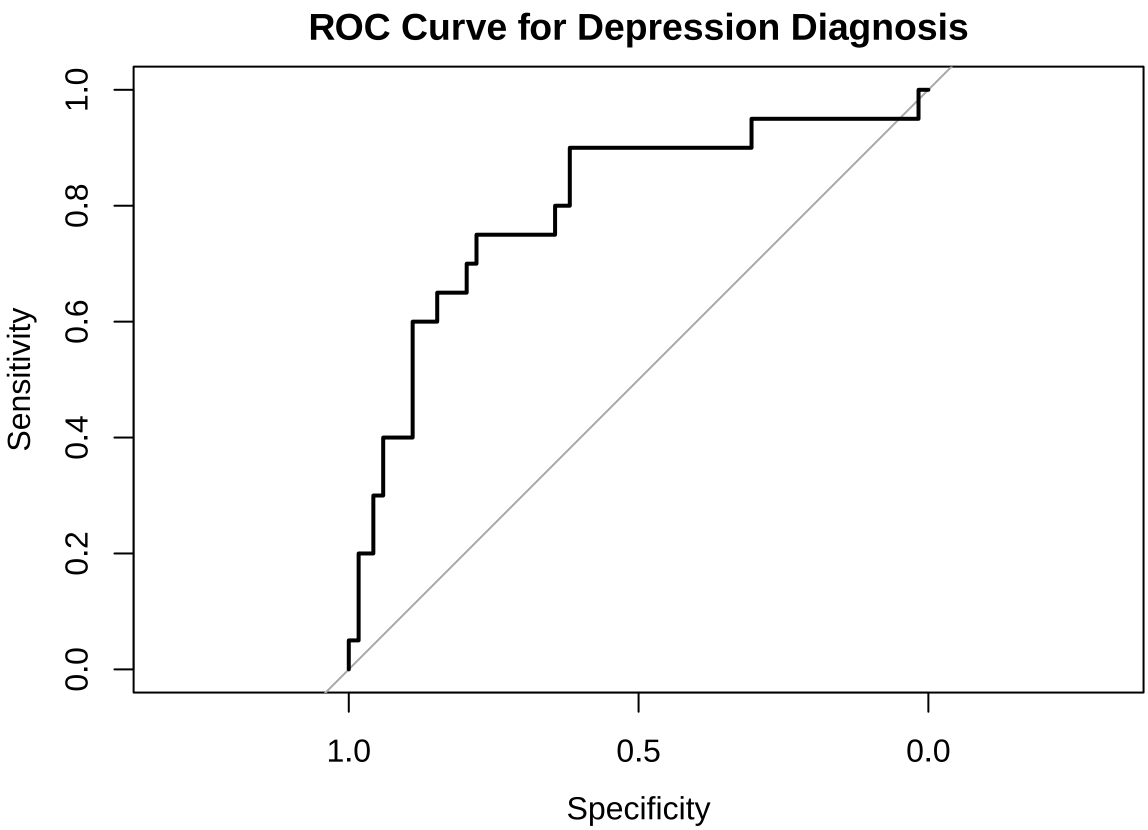
```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
roc_obj <- roc(test_data$Diagnosis, test_probs)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
plot(roc_obj, main = "ROC Curve for Depression Diagnosis")
```




the curve ventures off closer to the top left so i would say the model is decent.