STAA 577: HW3

Matthew Stoebe

Problem 1

$$P(Y = k \mid X = x) = \frac{\pi_k f_k(x)}{\sum_{j=1}^{K} \pi_j f_j(x)},$$

where $f_k(x)$ is the normal density $\mathcal{N}(\mu_k, \sigma^2)$:

Denominator is constant across classes, so maximizing the numerator is equivalent to maximizing the whole thing:

$$\pi_k f_k(x) \propto \pi_k \exp\left(-\frac{(x-\mu_k)^2}{2\sigma^2}\right).$$

Taking logarithm:

$$\log\left(\pi_k f_k(x)\right) = \log(\pi_k) - \frac{(x-\mu_k)^2}{2\sigma^2}$$

Expanding:

$$\log(\pi_k) - \frac{x^2 - 2\mu_k x + \mu_k^2}{2\sigma^2} = \log(\pi_k) + \frac{\mu_k x}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} - \frac{x^2}{2\sigma^2}.$$

Last term is constant for all classes and can be ignored in maximizaiton

$$\delta_k(x) = \log(\pi_k) + \frac{\mu_k x}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2}.$$

Problem 2

For class k with $X \sim \mathcal{N}(\mu_k, \sigma_k^2)$, the likelihood is:

$$f_k(x) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right).$$

The posterior probability is proportional to $\pi_k f_k(x)$. Taking the logarithm:

$$\delta_k(x) = \log(\pi_k) + \log(f_k(x)).$$

Which equals

$$\delta_k(x) = \log(\pi_k) - \frac{1}{2}\log(2\pi\sigma_k^2) - \frac{(x-\mu_k)^2}{2\sigma_k^2}.$$

Expand and Separate

$$\delta_k(x) = \log(\pi_k) - \frac{1}{2}\log(\sigma_k^2) - \frac{x^2 - 2\mu_k x + \mu_k^2}{2\sigma_k^2}.$$

$$= \log(\pi_k) - \frac{1}{2}\log(\sigma_k^2) - \frac{x^2}{2\sigma_k^2} + \frac{\mu_k x}{\sigma_k^2} - \frac{\mu_k^2}{2\sigma_k^2}.$$

This is Quadratic as you can see with the x^2 term

Problem 3a

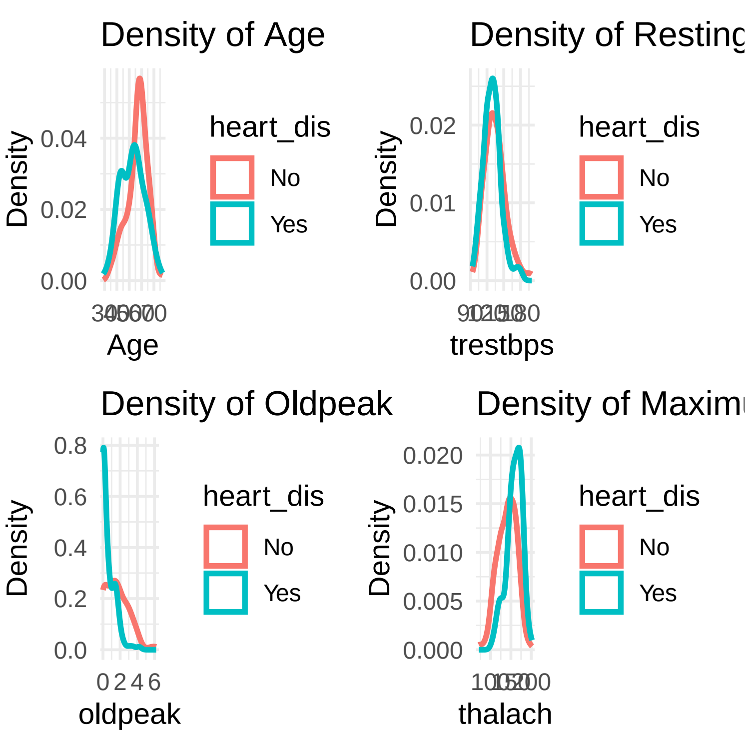
```
## Logistic Regression - Training Accuracy: 0.852
```

```
## Logistic Regression - Test Accuracy: 0.817
```

- training accuracy: (.852)
- test accuracy: (.817)

Problem 3b

Problem 3c



Problem 3d

```
## LDA - Training Accuracy: 0.84

## LDA - Test Accuracy: 0.817
```

- training accuracy: (.84)
- test accuracy: (.817)

Problem 3e

```
## QDA - Training Accuracy: 0.885

## QDA - Test Accuracy: 0.8
```

- training accuracy: (.885)
- test accuracy: (.8)

Problem 3f

Problem 3g

Problem 3h

```
## KNN (k = 14) - Test Accuracy: 0.817
```

- test accuracy: (0.817)

Problem 4

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio  black lstat
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83  4.03
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63  2.94
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90  5.33
```

```
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12  5.21
##   medv
## 1 24.0
## 2 21.6
## 3 34.7
## 4 33.4
## 5 36.2
## 6 28.7
```

```
## Boston Logistic Regression - Training Accuracy: 0.905
```

```
## Boston Logistic Regression - Test Accuracy: 0.97
```

```
## Boston LDA - Training Accuracy: 0.848
```

```
## Boston LDA - Test Accuracy: 0.939
```

```
## Boston KNN (k = 5) - Test Accuracy: 0.97
```

- It appears that the knn model performs the best in terms of test accuracy. LDA is the second best, and Logistic Regression is the worst. It may be worth continuing this analysis and tuning the number of neighbors used in KNN to opimize our prediction. This would require a third holdout set to prevent data leakage and train set hacking.

Appendix

```r
library(knitr)
# install the tidyverse library (do this once) install.packages('tidyverse')
library(tidyverse)
library(patchwork)
# set chunk and figure default options
knitr::opts_chunk$set(echo = FALSE, message = FALSE, warning = FALSE, fig.width = 4,
    fig.height = 4, tidy = TRUE)
trainingdata <- read.csv("heart_training.csv")
testdata <- read.csv("heart_test.csv")

factor_vars <- c("sex", "cp", "fps", "exang", "restecg")
for (v in factor_vars) {
    if (v %in% names(trainingdata))
        trainingdata[[v]] <- as.factor(trainingdata[[v]])
    if (v %in% names(testdata))
        testdata[[v]] <- as.factor(testdata[[v]])
}


logit_model <- glm(target ~ age + sex + cp + trestbps + thalach + exang + oldpeak +
    ca + thal, data = trainingdata, family = binomial)
```

```r
train_pred_prob <- predict(logit_model, type = "response")
train_pred_class <- ifelse(train_pred_prob > 0.5, 1, 0)

train_accuracy <- mean(train_pred_class == trainingdata$target)
cat("Logistic Regression - Training Accuracy:", round(train_accuracy, 3), "\n")


test_pred_prob <- predict(logit_model, newdata = testdata, type = "response")
test_pred_class <- ifelse(test_pred_prob > 0.5, 1, 0)
test_accuracy <- mean(test_pred_class == testdata$target)
cat("Logistic Regression - Test Accuracy:", round(test_accuracy, 3), "\n")

heart_dis <- rep("Yes", nrow(trainingdata))
heart_dis[trainingdata$target == 0] <- "No"
trainingdata$heart_dis <- heart_dis


p_age <- ggplot(trainingdata, aes(x = age, color = heart_dis)) + geom_density(size = 1) +
    labs(title = "Density of Age", x = "Age", y = "Density") + theme_minimal()

p_trestbps <- ggplot(trainingdata, aes(x = trestbps, color = heart_dis)) + geom_density(size = 1) +
    labs(title = "Density of Resting BP", x = "trestbps", y = "Density") + theme_minimal()

p_oldpeak <- ggplot(trainingdata, aes(x = oldpeak, color = heart_dis)) + geom_density(size = 1) +
    labs(title = "Density of Oldpeak", x = "oldpeak", y = "Density") + theme_minimal()

p_thalach <- ggplot(trainingdata, aes(x = thalach, color = heart_dis)) + geom_density(size = 1) +
    labs(title = "Density of Maximum Heart Rate", x = "thalach", y = "Density") +
    theme_minimal()

combined_plot <- (p_age | p_trestbps)/(p_oldpeak | p_thalach)
print(combined_plot)

library(MASS)  # package containing lda function


trainingdata$target <- as.factor(trainingdata$target)
testdata$target <- as.factor(testdata$target)

lda_model <- lda(target ~ age + sex + cp + trestbps + thalach + exang + oldpeak +
    ca + thal, data = trainingdata)

lda_train_pred <- predict(lda_model)$class
lda_train_accuracy <- mean(lda_train_pred == trainingdata$target)
cat("LDA - Training Accuracy:", round(lda_train_accuracy, 3), "\n")

lda_test_pred <- predict(lda_model, newdata = testdata)$class
lda_test_accuracy <- mean(lda_test_pred == testdata$target)
cat("LDA - Test Accuracy:", round(lda_test_accuracy, 3), "\n")


qda_model <- qda(target ~ age + sex + cp + trestbps + thalach + exang + oldpeak +
    ca + thal, data = trainingdata)
```

```r
qda_train_pred <- predict(qda_model)$class
qda_train_accuracy <- mean(qda_train_pred == trainingdata$target)
cat("QDA - Training Accuracy:", round(qda_train_accuracy, 3), "\n")

qda_test_pred <- predict(qda_model, newdata = testdata)$class
qda_test_accuracy <- mean(qda_test_pred == testdata$target)
cat("QDA - Test Accuracy:", round(qda_test_accuracy, 3), "\n")

library(dplyr)
library(class)

training_knn <- trainingdata %>%
    dplyr::select(age, sex, cp, trestbps, thalach, exang, oldpeak, ca, thal)

testing_knn <- testdata %>%
    dplyr::select(age, sex, cp, trestbps, thalach, exang, oldpeak, ca, thal)

num_vars <- c("age", "trestbps", "oldpeak", "thalach")

training_knn <- training_knn %>%
    mutate(across(all_of(num_vars), scale))

testing_knn <- testing_knn %>%
    mutate(across(all_of(num_vars), scale))

set.seed(420)

knn_pred <- knn(train = training_knn, test = testing_knn, cl = trainingdata$target,
    k = 14)

knn_test_accuracy <- mean(knn_pred == testdata$target)
cat("KNN (k = 14) - Test Accuracy:", round(knn_test_accuracy, 3), "\n")

head(Boston)

trn_samples <- sample(1:nrow(Boston), 440, replace = FALSE)
training_Boston <- Boston[trn_samples, ]
testing_Boston <- Boston[-trn_samples, ]
training_Boston$crimMedian <- training_Boston$crim > median(training_Boston$crim)
testing_Boston$crimMedian <- testing_Boston$crim > median(training_Boston$crim)

logit_boston <- glm(crimMedian ~ . - crim - crimMedian, data = training_Boston, family = binomial)

train_pred_boston <- ifelse(predict(logit_boston, type = "response") > 0.5, TRUE,
    FALSE)
test_pred_boston <- ifelse(predict(logit_boston, newdata = testing_Boston, type = "response") >
    0.5, TRUE, FALSE)

train_acc_boston <- mean(train_pred_boston == training_Boston$crimMedian)
test_acc_boston <- mean(test_pred_boston == testing_Boston$crimMedian)

cat("Boston Logistic Regression - Training Accuracy:", round(train_acc_boston, 3),
    "\n")
```

```r
cat("Boston Logistic Regression - Test Accuracy:", round(test_acc_boston, 3), "\n")

lda_boston <- lda(crimMedian ~ . - crim - crimMedian, data = training_Boston)

lda_train_pred <- predict(lda_boston)$class
lda_test_pred <- predict(lda_boston, newdata = testing_Boston)$class

lda_train_acc <- mean(lda_train_pred == training_Boston$crimMedian)
lda_test_acc <- mean(lda_test_pred == testing_Boston$crimMedian)

cat("Boston LDA - Training Accuracy:", round(lda_train_acc, 3), "\n")
cat("Boston LDA - Test Accuracy:", round(lda_test_acc, 3), "\n")

library(class)

train_knn_boston <- training_Boston %>%
    dplyr::select(-crim, -crimMedian)
test_knn_boston <- testing_Boston %>%
    dplyr::select(-crim, -crimMedian)

train_knn_boston_scaled <- scale(train_knn_boston)
test_knn_boston_scaled <- scale(test_knn_boston, center = attr(train_knn_boston_scaled,
    "scaled:center"), scale = attr(train_knn_boston_scaled, "scaled:scale"))

knn_boston_pred <- knn(train = train_knn_boston_scaled, test = test_knn_boston_scaled,
    cl = training_Boston$crimMedian, k = 5)
knn_boston_test_acc <- mean(knn_boston_pred == testing_Boston$crimMedian)
cat("Boston KNN (k = 5) - Test Accuracy:", round(knn_boston_test_acc, 3), "\n")
```