Introduction to Machine Learning

Lab 1

Rasmus Holm

2016-11-06

Contents

Assignment 1

1.3

```
#>   [,1] [,2] [,3] [,4] [,5]
#> 2  448 1003 1324  131 1179
#> 3 1003 1324  131  448 1179
#> 5  954 1276  724  240  106
#> 6  324 1041   11  562  176
#> 7  778  742 1090  890  621
#> 8 1324 1003 1201 1179  328
#>           predicted
#> actual     non-spam spam
#>   non-spam      695  242
#>   spam          193  240
```

1.4

```
#>   [,1]
#> 2  448
#> 3 1003
#> 5  954
#> 6  324
#> 7  778
#> 8 1324
#>           predicted
#> actual     non-spam spam
#>   non-spam      639  298
#>   spam          178  255
```

1.5

```
#>           predicted
#> actual     non-spam spam
#>   non-spam      640  297
#>   spam          177  256
```

1.6

```
#>   [,1] [,2] [,3] [,4] [,5]
#> 2  448 1003 1324  131 1179
#> 3 1003 1324  131  448 1179
#> 5  954 1276  724  240  106
#> 6  324 1041   11  562  176
#> 7  778  742 1090  890  621
#> 8 1324 1003 1201 1179  328
```

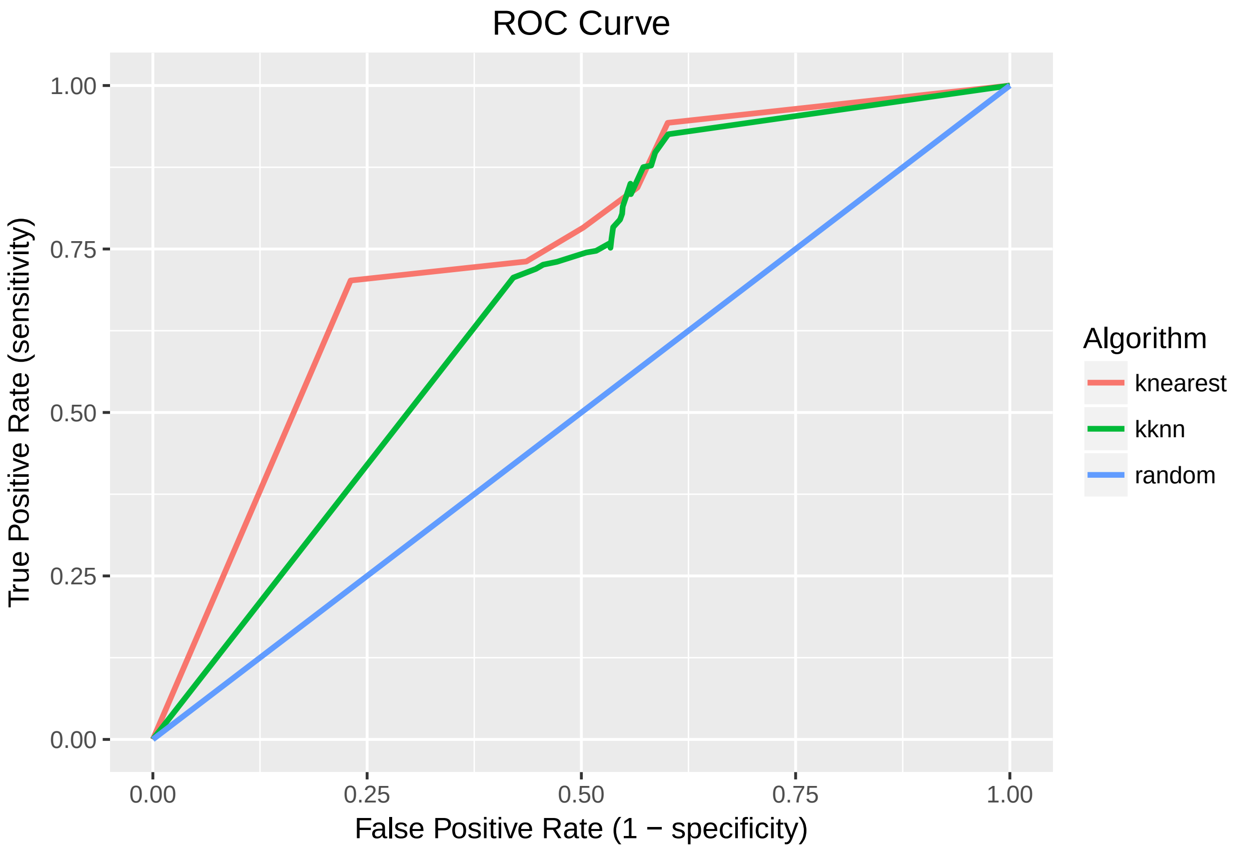

Figure 1: Receiver operating characteristic curves.

Assignment 2

2.2

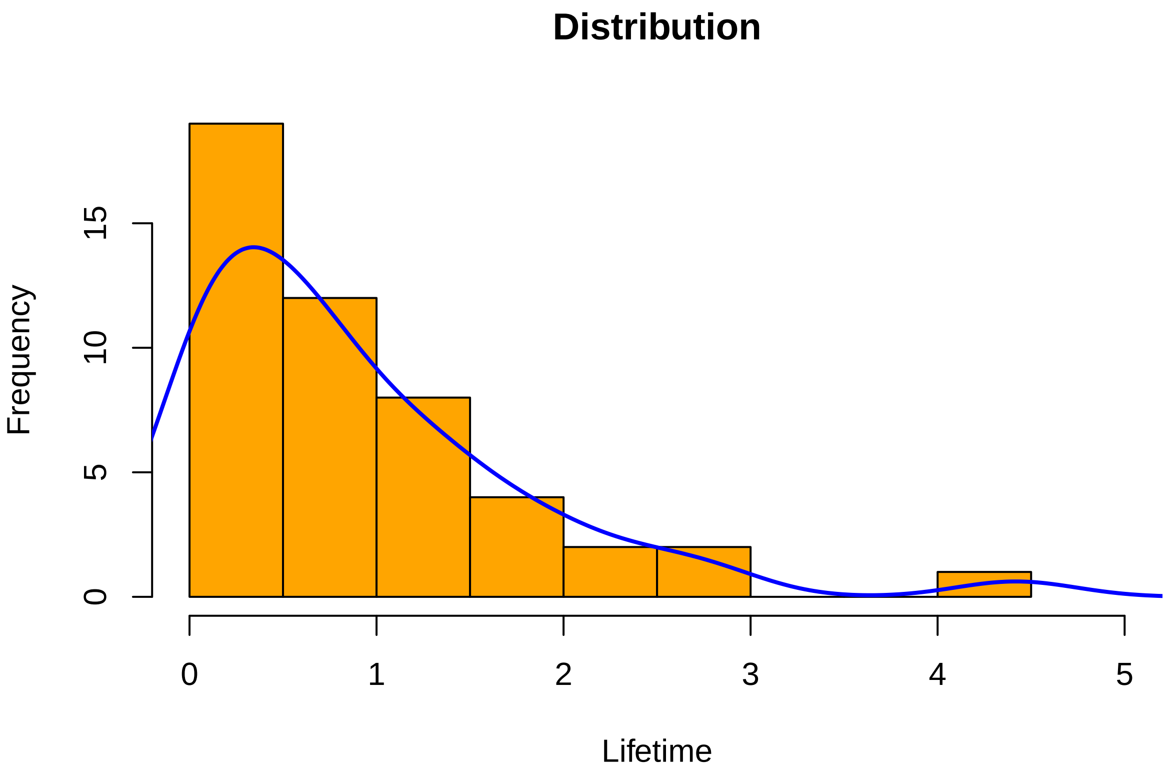

Figure 2: Data Distribution.

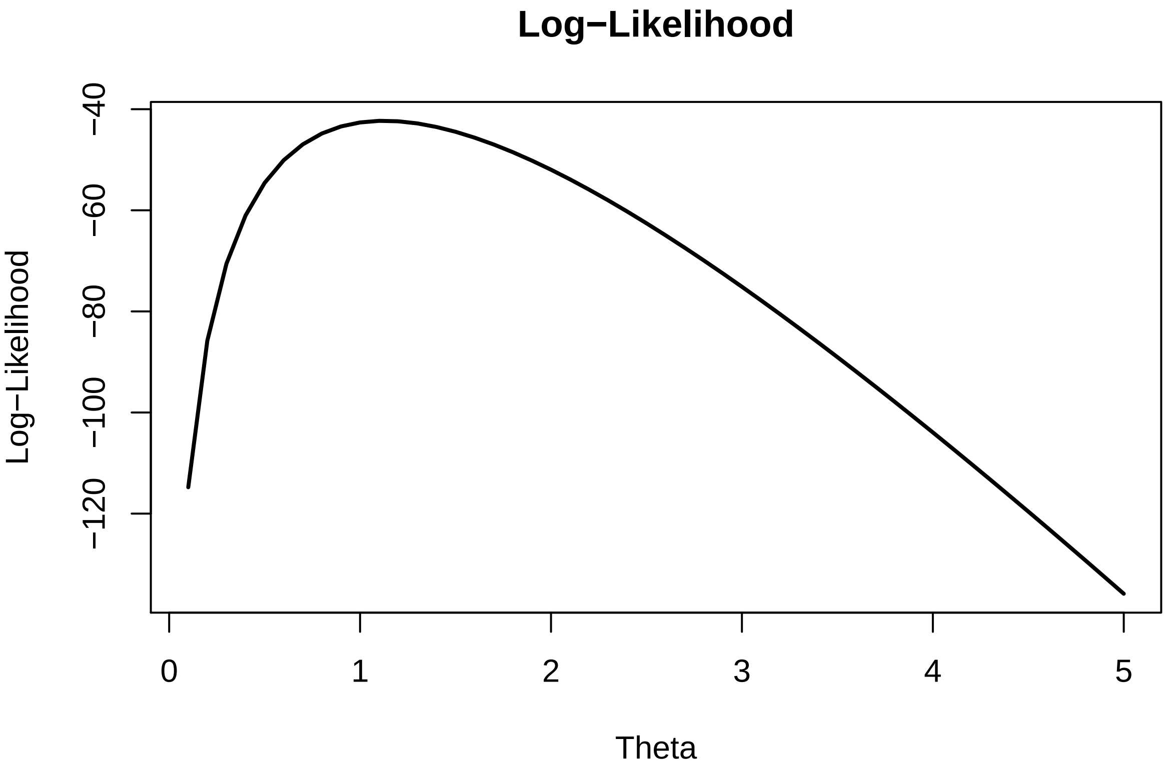

Figure 3: Likelihoods.

2.3

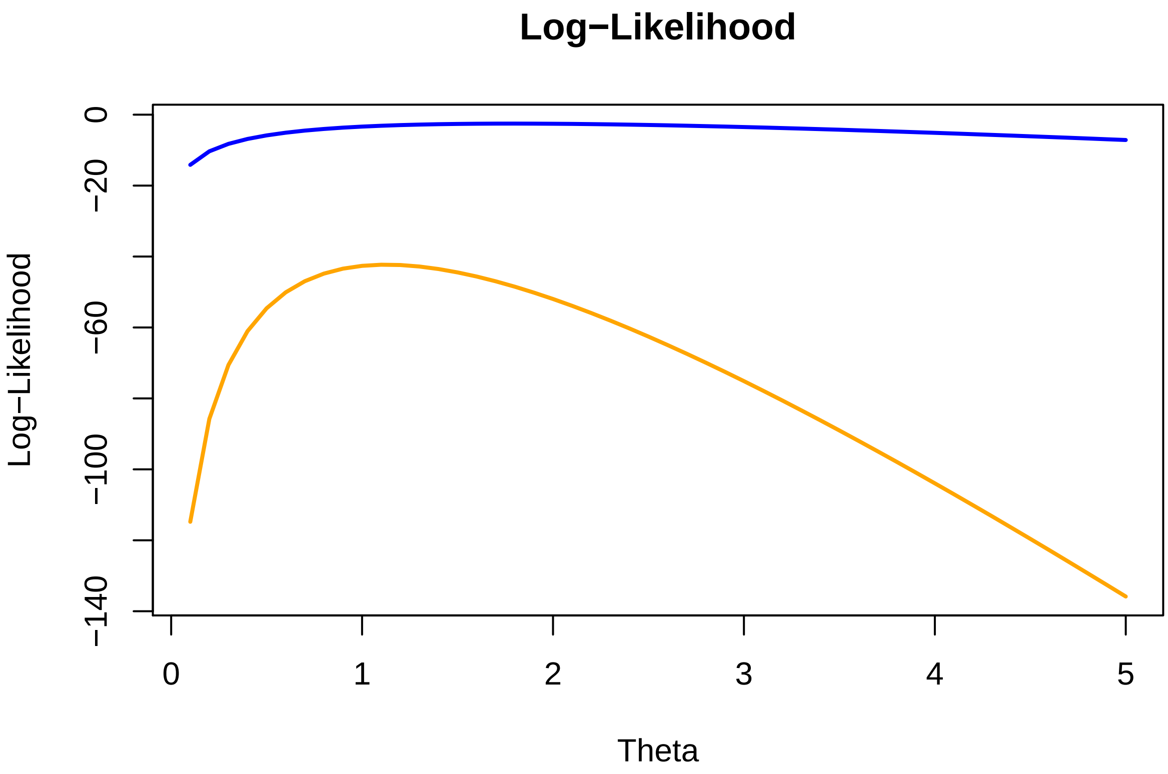

Figure 4: Likelihoods.

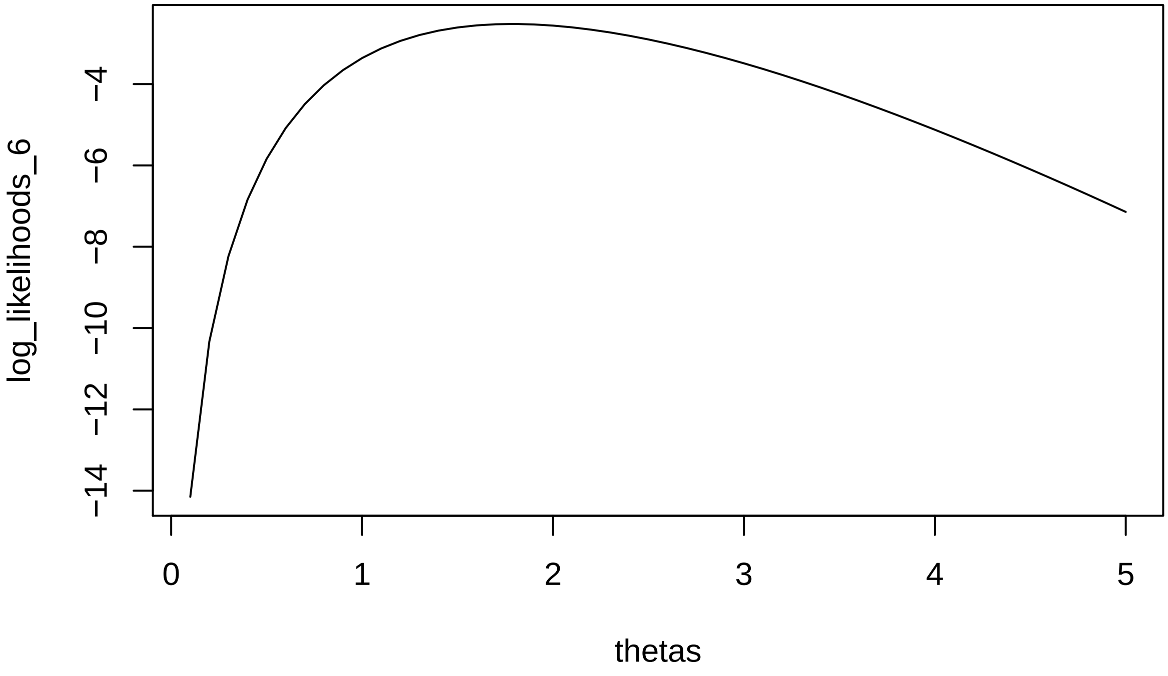

Figure 5: Likelihoods.

2.4

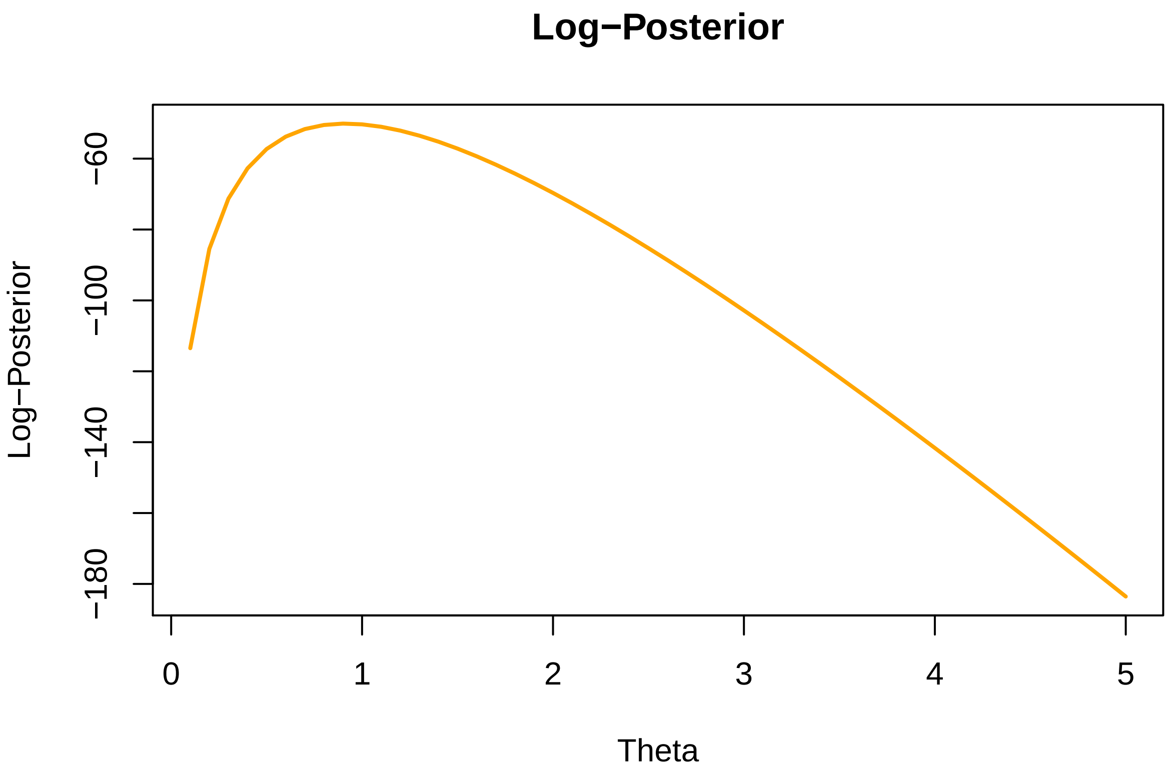

Figure 6: Likelihoods.

2.5

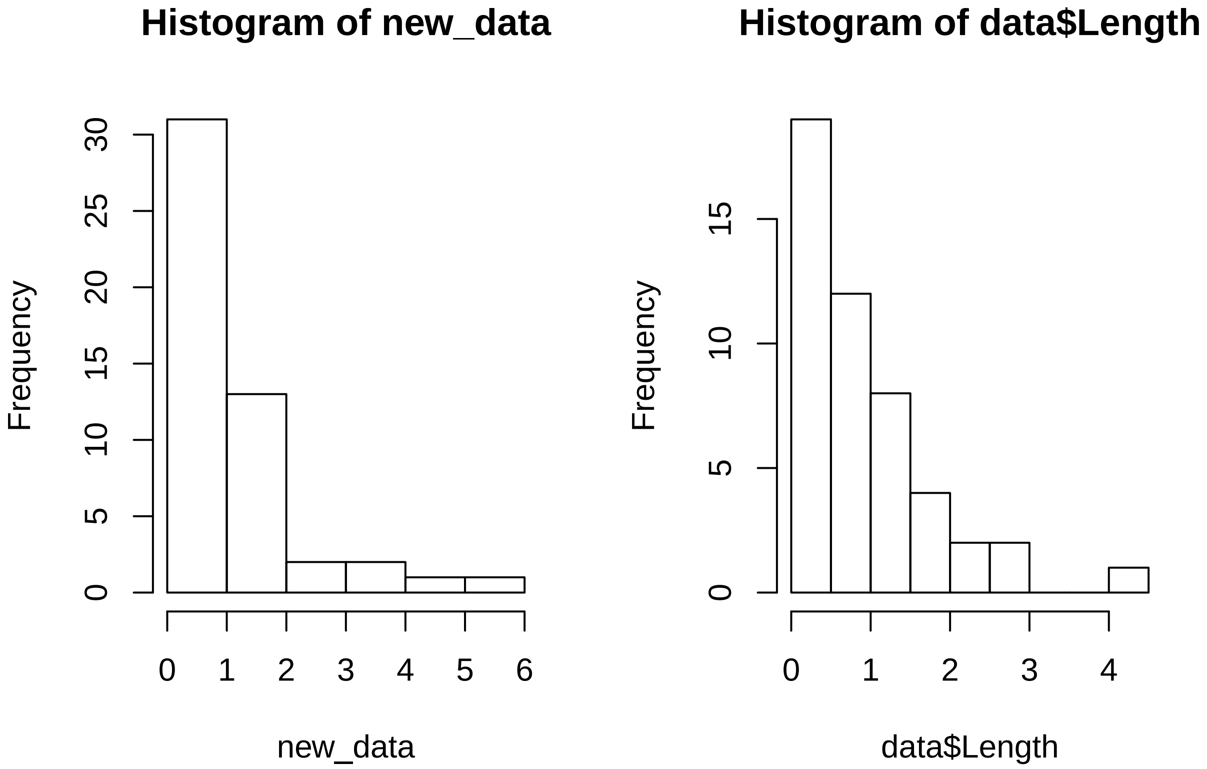

Figure 7: Likelihoods.

Appendix

Code for Assignment 1

```r
library(kknn)
library(caret)
library(ggplot2)
library(reshape)

data <- read.csv("../data/spambase.csv", sep=",", header=TRUE)

n <- nrow(data)
set.seed(12345)
id <- sample(1:n, floor(n * 0.5))

train <- data[id,]
test <- data[-id,]

train_labels <- factor(train[, ncol(train)], levels=c(0, 1), labels=c("non-spam", "spam"))
test_labels <- factor(test[, ncol(test)], levels=c(0, 1), labels=c("non-spam", "spam"))

single_threshold <- 0.5

knearest <- function(data, k, newdata) {
    stopifnot(k > 0)

    train_labels <- data[, ncol(data)]
    train <- as.matrix(data[, -ncol(data)])
    train <- train / sqrt(rowSums(train^2))

    test_labels <- newdata[, ncol(newdata)]
    test <- as.matrix(newdata[, -ncol(newdata)])
    test <- test / sqrt(rowSums(test^2))

    cosine_sim <- train %*% t(test)
    cosine_dis <- 1 - cosine_sim

    ordering <- as.matrix(t(apply(cosine_dis, 2, order))[, 1:k])
    print(head(ordering))

    predicted <- as.matrix(apply(ordering, 1, function(x) {
        mean(train_labels[x])
    }))

    predicted
}

predicted_result <- function(predicted, actual, threshold) {
    predicted <- as.numeric(predicted > threshold)
    predicted <- factor(predicted, levels=c(0, 1), labels=c("non-spam", "spam"))
    table(actual, predicted)
}
predicted <- knearest(train, 5, test)
```

```r
predicted_result(predicted, test_labels, single_threshold)
predicted <- knearest(train, 1, test)
predicted_result(predicted, test_labels, single_threshold)
kknn.fit <- kknn(Spam ~ ., train=train, test=test, distance=2, k=5)
predicted <- fitted(kknn.fit)
predicted_result(predicted, test_labels, single_threshold)
threshold <- seq(0.05, 0.95, by=0.05)

predicted_knearest <- knearest(train, 5, test)

kknn.fit <- kknn(Spam ~ ., train=train, test=test, distance=2, k=5)
predicted_kknn<- fitted(kknn.fit)

knearest_sensitivity <- rep(0, length(threshold))
knearest_specificity <- rep(0, length(threshold))

kknn_sensitivity <- rep(0, length(threshold))
kknn_specificity <- rep(0, length(threshold))

for (i in 1:length(threshold)) {
    knearest_prediction <- predicted_result(predicted_knearest, test_labels, threshold[i])
    knearest_sensitivity[i] <- sensitivity(knearest_prediction)
    knearest_specificity[i] <- specificity(knearest_prediction)

    kknn_prediction <- predicted_result(predicted_kknn, test_labels, threshold[i])
    kknn_sensitivity[i] <- sensitivity(kknn_prediction)
    kknn_specificity[i] <- specificity(kknn_prediction)
}

knearest_x <- c(0, rev((1 - knearest_specificity)), 1)
knearest_y <- c(0, rev(knearest_sensitivity), 1)

kknn_x <- c(0, rev((1 - kknn_specificity)), 1)
kknn_y <- c(0, rev(kknn_sensitivity), 1)

knearest_data <- data.frame(x=knearest_x, y=knearest_y,
                            label=rep("knearest", length(knearest_x)))

kknn_data <- data.frame(x=kknn_x, y=kknn_y,
                        label=rep("kknn", length(kknn_x)))

reference_line <- data.frame(x=seq(0, 1, 0.05), y=seq(0, 1, 0.05),
                             label=rep("random", length(knearest_x)))


complete_data <- melt(rbind(knearest_data, kknn_data, reference_line), id=c("x", "y"))
names(complete_data)[4] <- "Algorithm"
ggplot() + ggtitle("ROC Curve") +
    xlab("False Positive Rate (1 - specificity)") +
    ylab("True Positive Rate (sensitivity)") +
    geom_line(data=complete_data, aes(x=x, y=y, color=Algorithm), size=1) +
    scale_x_continuous(limits = c(0, 1)) + scale_y_continuous(limits=c(0, 1)) +
    theme(plot.title=element_text(hjust=0.5))
```

Code for Assignment 2

```r
length_histogram <- hist(data$Length, plot=FALSE)
multiplier <- length_histogram$counts / length_histogram$density
multiplier <- max(multiplier[which(!is.nan(multiplier))])
length_density <- density(data$Length)
length_density$y <- length_density$y * multiplier

log_likelihood <- function(x, theta) {
    log(theta * exp(-theta * x))
}

thetas <- seq(0.1, 5, by=0.1)

log_likelihoods <- sapply(thetas, function(x) {
    sum(log_likelihood(x=data$Length, theta=x))
})

best_theta <- thetas[which.max(log_likelihoods)]
plot(length_histogram, col="orange", main="Distribution",
     xlab="Lifetime", ylab="Frequency", xlim=c(0, 5))
lines(length_density, col="blue", lwd=2)
plot(thetas, log_likelihoods, main="Log-Likelihood",
     xlab="Theta", ylab="Log-Likelihood", type="l", lwd=2)
log_likelihoods_6 <- sapply(thetas, function(x) {
    sum(log_likelihood(x=data$Length[1:6], theta=x))
})

ylim <- c(min(min(log_likelihoods), min(log_likelihoods_6)),
          max(max(log_likelihoods), max(log_likelihoods_6)))
plot(thetas, log_likelihoods, col="orange",
     main="Log-Likelihood", xlab="Theta", ylab="Log-Likelihood",
     type="l", ylim=ylim, lwd=2)
lines(thetas, log_likelihoods_6, col="blue", lwd=2)
plot(thetas, log_likelihoods_6, type="l")
prior <- function(theta, lambda=10) {
    lambda * exp(-lambda * theta)
}

log_posteriors <- sapply(1:length(thetas), function(i) {
    log_likelihoods[i] + log(prior(thetas[i]))
})
plot(thetas, log_posteriors, col="orange",
     main="Log-Posterior", xlab="Theta", ylab="Log-Posterior",
     type="l", lwd=2)
set.seed(12345)
new_data <- rexp(50, best_theta)
par(mfrow=c(1, 2))
hist(new_data)
hist(data$Length)
```