TDDE01 Machine Learning

Lab1

Ruben Hillborg

November 15, 2018

Assignment 1. Spam classification with nearest neighbors

This assignment is about classifying mails as spam or not spam. A data file containing 2470 e-mails manually classified as spam or regular e-mail was given. This file also contained information about the frequencies of different words in the e-mails.

1.1 Setup

The data was imported and randomly split into two equally distributed data sets, one for training and one for testing.

1.2 Logistic regression (50% probability)

A model was fitted to the test set using logistic regression (the glm() funciton in R) and the probabilities for the mails being spam was calculated using the predict() function. The mails were then classified as spam based on the classification principle:

$$\hat{Y} = 1 \; if \; p(Y = 1|X) > 0.5, \; otherwise \; \hat{Y} = 0$$

The resulting confusion matrix and the missclassification rate when using this model on the **training** set can be seen below:

```
##            Truth
## Prediction   0   1
##          0 803  81
##          1 142 344
```

```
## [1] "Missclassification rate: 0.162774"
```

And the results when using this model on the **test** data set:

```
##            Truth
## Prediction   0   1
##          0 791  97
##          1 146 336
```

```
## [1] "Missclassification rate: 0.177372"
```

As can be seen from the results, the model does an ok job at predicting spam e-mails. I wouldn't want to use it for my e-mails though. In both cases there are ~140 e-mails falsly predicted as spam, which means that those (probably) important e-mails would be sent to the spam folder instead of your inbox. It works a bit better on the training data than on the test data, which isn't suprising since the model was fitted to the training data.

1.3 Logistic regression (90% probability)

Here the same logistic regression model was used again, but the classification principle was changed to:

$\hat{Y} = 1 \; if \; p(Y = 1|X) > 0.9, \; otherwise \; \hat{Y} = 0$

The resulting confusion matrix and the missclassification rate when using this model on the **training** set can be seen below:

```
##            Truth
## Prediction   0   1
##          0 944 419
##          1   1   6
```

```
## [1] "Missclassification rate: 0.306569"
```

And the results when using this model on the **test** data set:

```
##            Truth
## Prediction   0   1
##          0 939 424
##          1   6   1
```

```
## [1] "Missclassification rate: 0.312409"
```

The results clearly show that this model isn't very reliable. Almost all e-mails were predicted to be not spam. Only a total of 7 e-mails in each data set are predicted as spam with a probability > 90%, and 6 of those are false-positives on the testing set. From this we can gather that the logistic regression model works ok, but predicts e-mails as spam with high uncertainty.

1.4 K-nearest-neighbors (K=30)

To see if a better model for classifying the spam e-mails could be found, the knn algorithm was used with K = 3. An e-mail was classified as spam the same way as in 1.2 (> 0.5). The results on the **training** set can be seen below:

```
##           Truth
## Predicted   0   1
##         0 807  98
##         1 138 327
```

```
## [1] "Missclassification rate: 0.172263"
```

And the results when using this model on the **test** data set:

```
##           Truth
## Predicted   0   1
##         0 672 187
##         1 265 246
```

```
## [1] "Missclassification rate: 0.329927"
```

1.5 K-nearest-neighbors (K=1)

Lastly I tried knn with K=1.

```
##           Truth
## Predicted   0   1
##         0 945   0
```

```
##         1   0 425
```

```
## [1] "Missclassification rate: 0.000000"
```

And the results when using this model on the **test** data set:

```
##          Truth
## Predicted   0   1
##         0 640 177
##         1 297 256
```

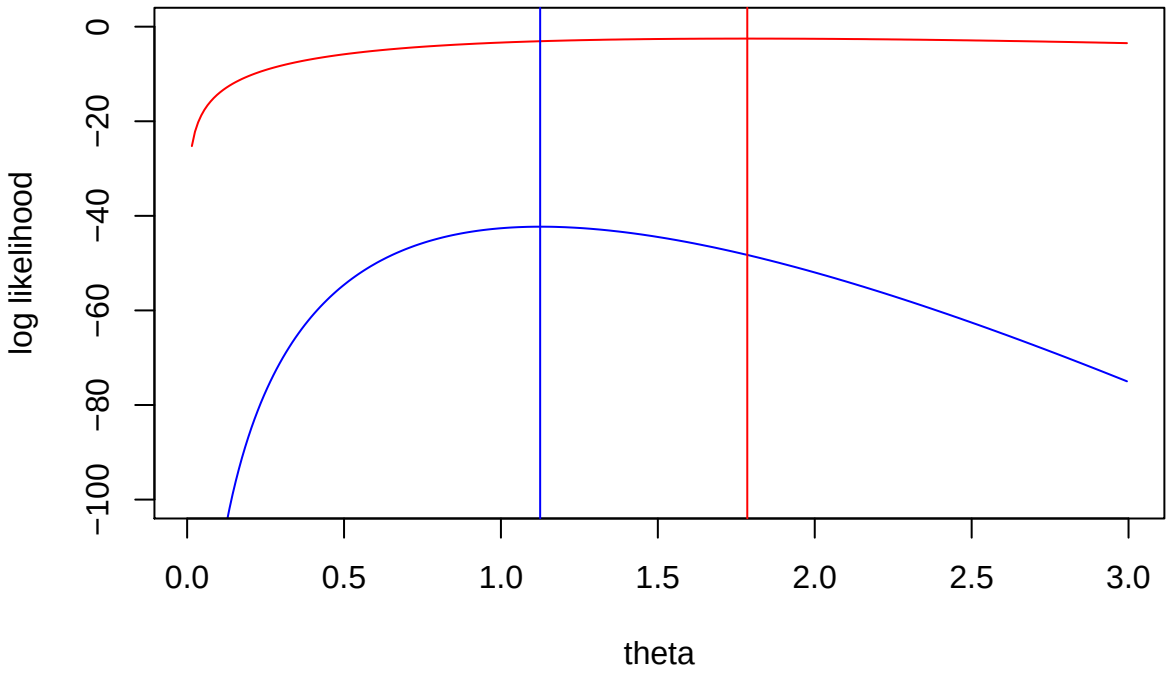
```
## [1] "Missclassification rate: 0.345985"
```

Assignment 2

```
data = read_excel("machines.xlsx")
lifetimes = data[[1]]
```

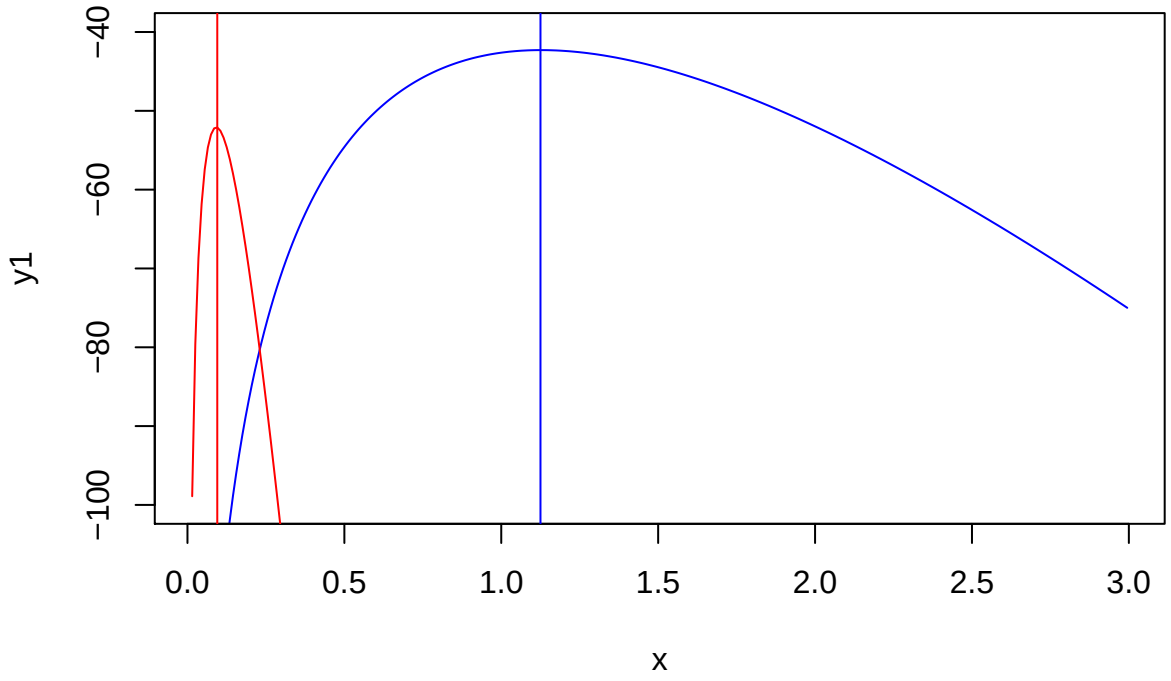
We assume that the probability model for the data is $p(x|\theta) = \theta e^{(-\theta x)}$ for $\mathbf{X}$=*Length*. The observations are independent and identically distributed (iid). Since the data is the time until an event happens, the data is exponentially distributed (which can also be seen on the model we assumed).

```
log_likelihood = function(theta, x) {
  n = length(x)
  n*log(theta) - theta*sum(x)
}

max_log_likelihood = function(x) {
  n = length(x)
  n/sum(x)
}

x = seq(0.015, 3, 0.01)
y1 = log_likelihood(x, lifetimes)
y2 = log_likelihood(x, lifetimes[1:6])

plot(x, y1, type="l", col="blue", ylim = c(-100, 0), xlab = "theta", ylab = "log likelihood")
lines(x, y2, col="red")
theta_hat1 = x[which.max(y1)]
theta_hat2 = x[which.max(y2)]
abline(v=theta_hat1, col="blue")
abline(v=theta_hat2, col="red")
```

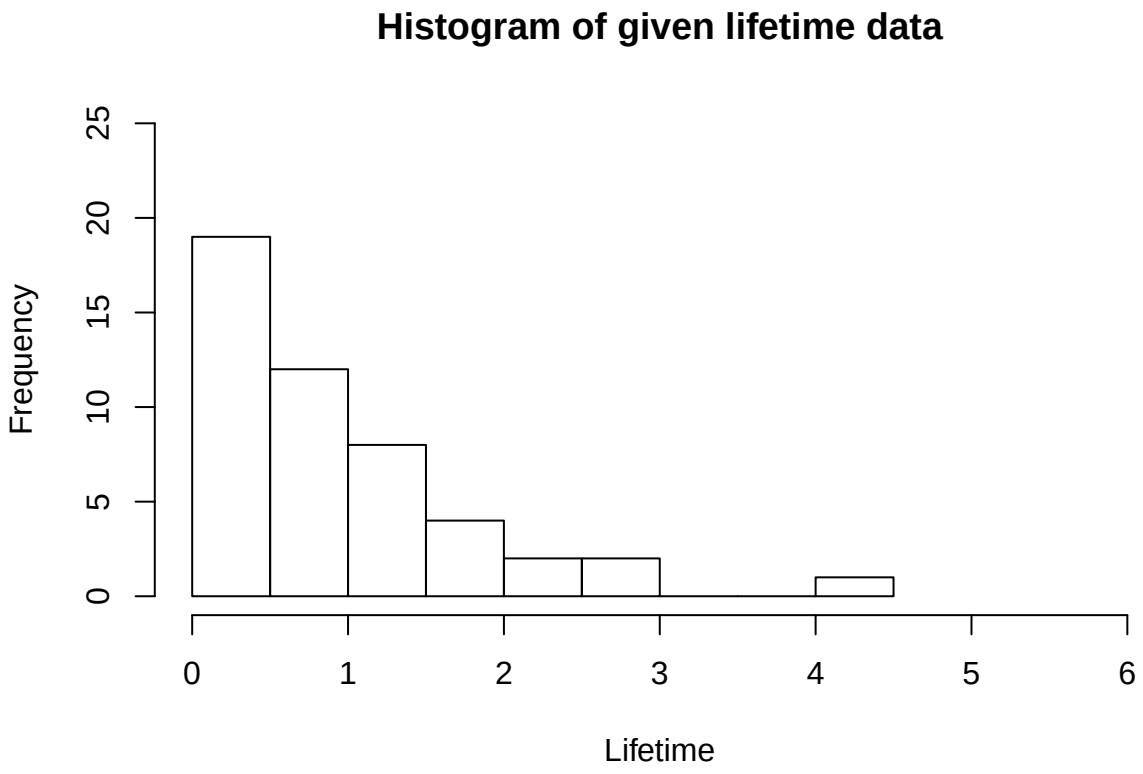


The first estimated theta is probably more reliablie, since it's based on 48 data values and theta_hat2 is only based on 6 values.

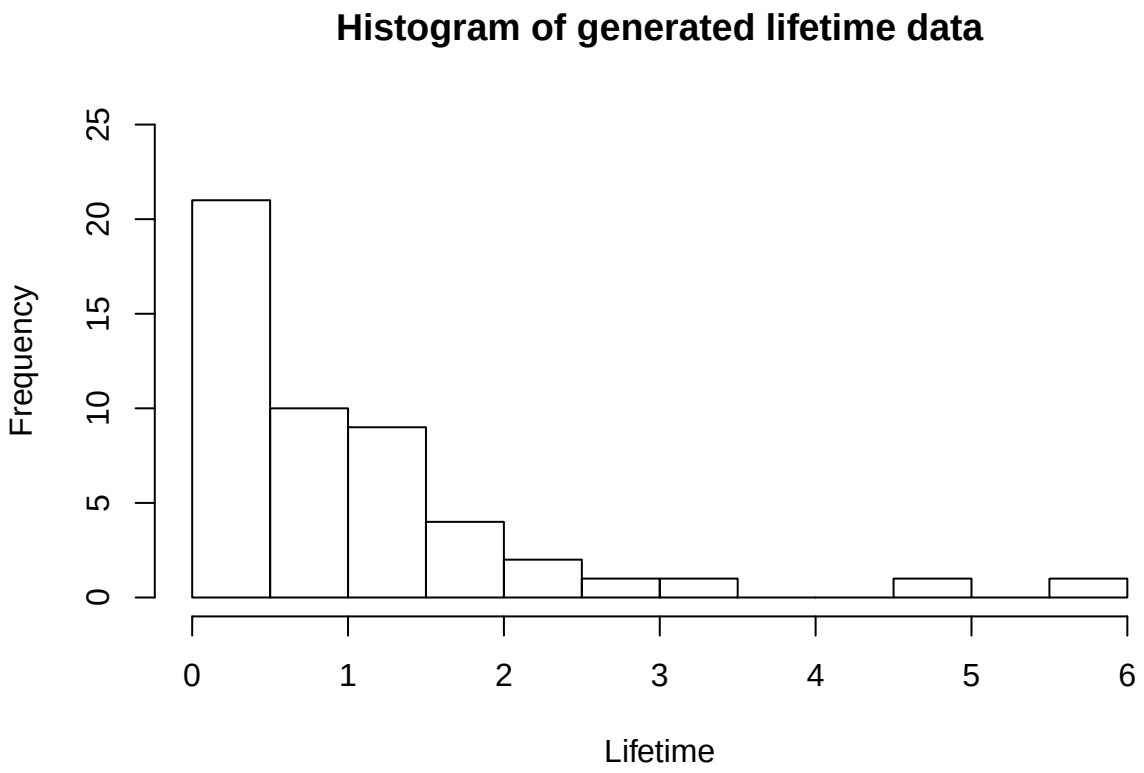
```
bayesian_log_likelihood = function(theta, x) {
  n = length(x)
  lambda = 10
  n*log(lambda) + n*log(theta) - theta*(sum(x)+lambda*n)
}

y3 = bayesian_log_likelihood(x, lifetimes)

plot(x, y1, type="l", col="blue", ylim = c(-100, -40))
lines(x, y3, col="red")
theta_hat1 = x[which.max(y1)]
theta_hat3 = x[which.max(y3)]
abline(v=theta_hat1, col="blue")
abline(v=theta_hat3, col="red")
```




```
set.seed(12345)
rdata = rexp(50, theta_hat1)
hist(lifetimes,
     main = "Histogram of given lifetime data",
     xlab = "Lifetime",
     xlim = c(0, 6),
     ylim = c(0, 25),
     breaks = 10)
```

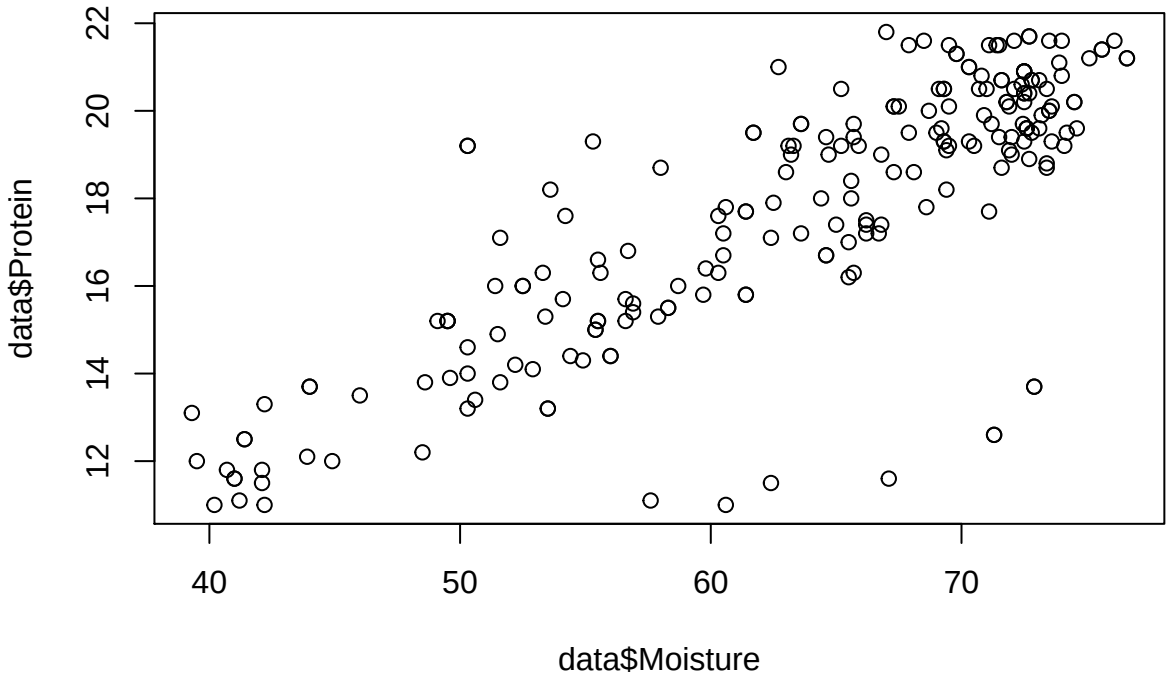


```r
hist(rdata,
     main = "Histogram of generated lifetime data",
     xlab = "Lifetime",
     xlim = c(0, 6),
     ylim = c(0, 25),
     breaks = 10)
```



Assignment 4

```r
data = read_excel("tecator.xlsx")

plot(data$Moisture, data$Protein)
```



Yes this data looks like they are described good with a linear model.