Exercise 2

1. The dataset `prostcancer` contains medical data measured on 97 men who were about to receive a radical prostatectomy. The aim is to find out how the predictors influence the response **psa**, which is an important indicator for further development of the cancer. The variables are

psa	level of prostate specific antigen (response)
cavol	cancer volume
weight	prostate weight
age	age of patient
bph	benign prostatic hyperplasia amount
svi	seminal vesicle invasion
cp	capsular penetration
gleason	gleason score
pgg45	percent of Gleason score 4 or 5

Analyze these data by linear regression. Try to find a good model by stepwise variable selection and transformations of the variables (hint: taking logarithms may help). Make use of diagnostic plots such as scatterplots of the response **psa** versus the predictors, as well as residuals versus both the fitted values and the predictors. Try to get an impression about the validity of the model assumptions. Which variables influence the **psa**-level significantly (and in which direction)?

R-hints:

```
## Reading the dataset
prostate <- read.table("http://stat.ethz.ch/Teaching/Datasets/prostcancer.dat",
                       header = TRUE)

## Fit the full model
prost.full  <- lm(psa ~ cavol+weight+age+bph+svi+cp+gleason+pgg45,
                  data = prostate)

## Fit the empty model. This is not very useful in itself, but is required
## as a starting model for stepwise forward variable selection
prost.empty <- lm(psa ~ 1, data = prostate)

## Backward selection, starting from the full model
prost.bw    <- step(prost.full, direction = "backward")

## Forward selection, starting from the empty model
prost.fw    <- step(prost.empty, direction = "forward",
                    scope = psa ~ cavol+weight+age+bph+svi+cp+gleason+pgg45)

## Loading the package for all-subsets regression
library(leaps)
```

```
## All subsets model choice, compare to the stepwise methods
prost.alls  <- regsubsets(psa  ~ cavol+weight+age+bph+svi+cp+gleason+pgg45,
                           data = prostate)
```

`pairs(prostate, pch = ".")` may be a useful plot. `summary` summarizes results from lots of methods (including `step` and `regsubsets`).

2. **a)** The artificial dataset shown in Figure 1 is generated from a mixture of two normal distributions (see part b)). Imagine you wouldn't know anything about the distribution and you'd want to estimate the density by use of a kernel density estimator with Gaussian kernel. Guess a good bandwidth.

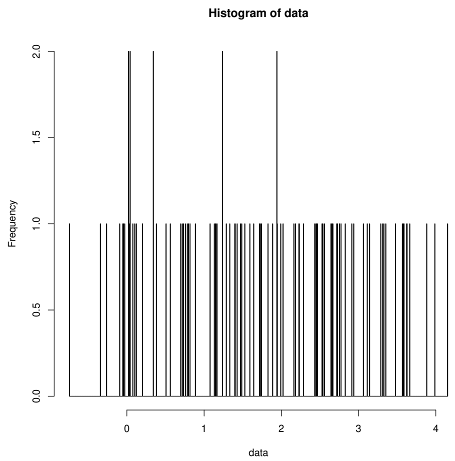

Figure 1: Artificial data from mixture of normal distributions

b) Carry out a simulation with `R` to evaluate the quality of bandwidths for kernel density estimation for a mixture of normal distributions where a point is generated by a $\mathcal{N}(0, 0.01)$-distribution with probability 0.2 and by a $\mathcal{N}(2, 1)$-distribution with probability 0.8. Repeat the following code 200 times for multiple bandwidth values: $h = 0.02,\ 0.1,\ 0.3,\ 0.6,\ 1,\ 1.5$, as well as h equal to your own guess from part a) and to a plug-in estimate of the optimal bandwidth (see below):

1. Generate a dataset of 100 points from the mixture distribution from above:

```
data <- numeric(100)
for(i in 1:100){
  p <- runif(1, min = 0, max = 1)
  if (p < 0.2)
    data[i] <- rnorm(1, mean = 0, sd = sqrt(0.01))
  else
    data[i] <- rnorm(1, mean = 2, sd = 1)
}
```

2. Compute the kernel density estimator with the function `density`. Consult `help(density)` to understand the syntax.

```
ke <- density(data, bw = 0.1, n = 61, from = -1, to = 5)
```

An estimated plug-in bandwidth is obtained by using `bw = "sj"`.

3. The goodness of the kernel estimate can be measured by averaging the squared difference between the kernel estimator and the true density values at the multiple datapoints $-0.2,\ 0,\ 0.2,\ 0.5,\ 1,\ 1.5,\ 2,\ 2.5,\ 3,\ 4$:

```
index <- c(9, 11, 13, 16, 21, 26, 31, 36, 41, 51)
ke$x[index]  ## -0.2  0.0  0.2  0.5  1.0  1.5  2.0  2.5  3.0  4.0

## Compute the true density at the given datapoints
dmix <- 0.2 * dnorm(ke$x[index], mean = 0, sd = sqrt(0.01)) +
  0.8 * dnorm(ke$x[index], mean = 2, sd = 1)

## Take the mean of the squared differences
quality <- mean((ke$y[index] - dmix)^2)
```

Note that the commands given above are suitable for a single execution, but not necessarily for the full simulation. For example, you will need a vector of the qualities of all simulation runs. You may define a matrix for the qualities to store the results for all different bandwidths. Consider `help(matrix), help(apply)`.

Compute and compare the averaged quality of the kernel estimators for the different kernel functions and bandwidths.

c) (Optional) Repeat the whole project with the Epanechnikov kernel, which can be obtained by specifying `kernel="epanechnikov"` in `density`.

Preliminary discussion: Friday, April 15, 2005.

Deadline: Friday, April 22, 2005.

Advice: Thursdays from 12.00-13.00, LEO C12.1, Leonhardstr. 27. Or contact either Maik Berchtold, `berchtld@stat.math.ethz.ch`, or Lukas Meier, `meier@stat.math.ethz.ch`.