STK2100: Solutions Week 13

Lars H. B. Olsen

07.04.2021

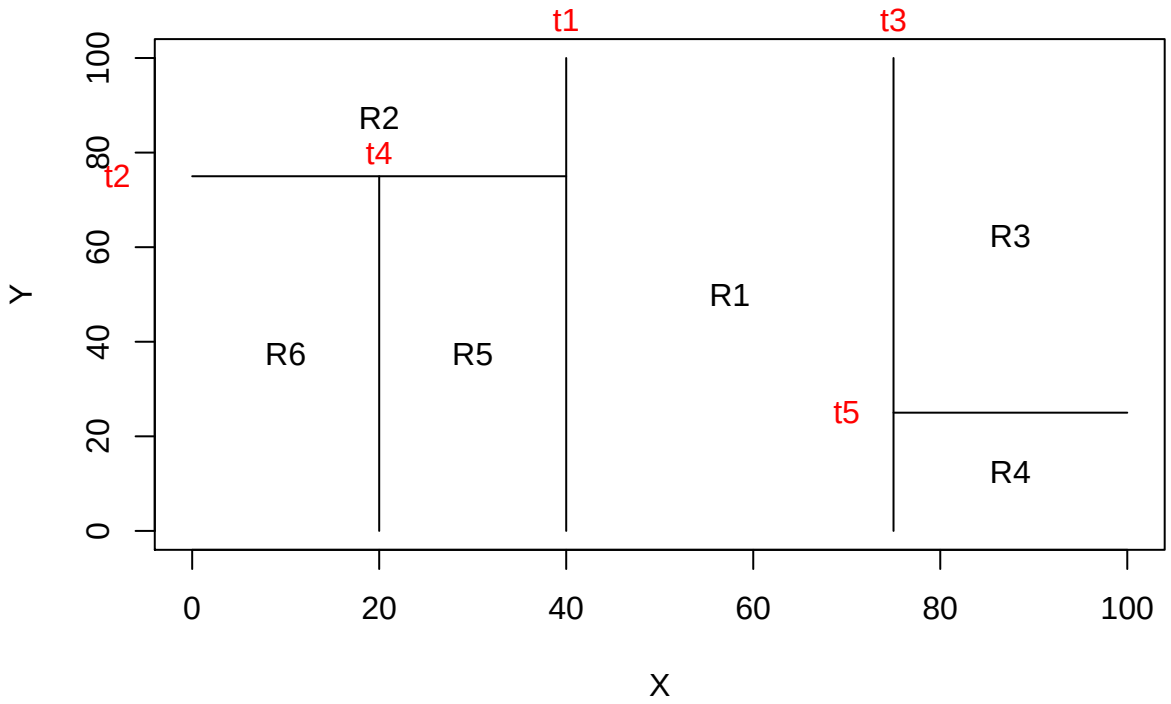
ISLR

Exercise 8.1

We are asked to draw an example (of your own invention) of a partition of two-dimensional feature space that could result from recursive binary splitting. Our example should contain at least six regions. Then draw a decision tree corresponding to this partition. Be sure to label all aspects of your figures, including the regions R1,R2,..., the cutpoints t1,t2,..., and so forth.

```
par(xpd = NA)
plot(NA, NA, type = "n", xlim = c(0, 100), ylim = c(0, 100), xlab = "X", ylab = "Y")
# t1: x = 40; (40, 0) (40, 100)
lines(x = c(40, 40), y = c(0, 100))
text(x = 40, y = 108, labels = c("t1"), col = "red")
# t2: y = 75; (0, 75) (40, 75)
lines(x = c(0, 40), y = c(75, 75))
text(x = -8, y = 75, labels = c("t2"), col = "red")
# t3: x = 75; (75,0) (75, 100)
lines(x = c(75, 75), y = c(0, 100))
text(x = 75, y = 108, labels = c("t3"), col = "red")
# t4: x = 20; (20,0) (20, 75)
lines(x = c(20, 20), y = c(0, 75))
text(x = 20, y = 80, labels = c("t4"), col = "red")
# t5: y=25; (75,25) (100,25)
lines(x = c(75, 100), y = c(25, 25))
text(x = 70, y = 25, labels = c("t5"), col = "red")

text(x = (40 + 75)/2, y = 50, labels = c("R1"))
text(x = 20, y = (100 + 75)/2, labels = c("R2"))
text(x = (75 + 100)/2, y = (100 + 25)/2, labels = c("R3"))
text(x = (75 + 100)/2, y = 25/2, labels = c("R4"))
text(x = 30, y = 75/2, labels = c("R5"))
text(x = 10, y = 75/2, labels = c("R6"))
```




```
#           [X<40]
#         |         |
#      [Y<75]     [X<75]
#      |     |    |     |
#   [X<20] R2    R1   [Y<25]
#   |     |             |     |
# R6    R5             R4    R3
```

Exam STK2100 spring 2019

Solutions are directly copied from https://www.uio.no/studier/emner/matnat/math/STK2100/v19/eksamen/solution_stk2100_2019.pdf.

Exercises 1

a)

There is no protecting factor, as all regression coefficients are larger than 0. The intercept includes the baseline effects, in this case the log-odds for a female, non-smoker who does not drink and has age = 0. The latter is the reason why the intercept does not have a "physical" meaning in this case. A solution is to center age, so the baseline is a female with average age. Since the regression coefficient is not significant, it may also be excluded from the model (although, strictly speaking, in this way the problem is removed, more than solved...).

b)

The correct answer is model 2, as it has the best performance on the test set in terms of the ROC-curve (valid explanations: largest area under the curve, closest curve to the top-left corner, farthest curve from the intercept). The ROC curve is a graphical tool to visualize the performance of a classification model, and it displays the sensitivity and (1 minus) specificity when moving the threshold used to discriminate between the two response classes. Sensitivity = true positive / (true positive + false negative), where true positive are the observations correctly identified as positive by the model, and false negative the observations incorrectly classified as negative by the model. Specificity = true negative / (false positive + true negative), where true negative are the observations correctly identified as negative by the model, and false positive the observations incorrectly classified as positive by the model.

c)

The plot shows how many times a classification based on the selected model is better than a random choice when classifying a specific amount of observations. In particular, for the highlighted point, this means that if we want to classify 35% of the observations, if we select those to which the model gives the highest probability to be, let us say, positive, we identify 1.6 times more positive than when repeating the procedure at random.

d)

The values in the table can be computed as:

- $a_{21} = 47 * \exp(-0.96141)/(1 + \exp(-0.96141)) \approx 13$
- $a_{22} = 152 * \exp(-0.96141 + 1.11964)/(1 + \exp(-0.96141 + 1.11964)) \approx 82$
- $a_{11} = 47 - 13 = 34$
- $a_{12} = 152 - 82 = 70$

e)

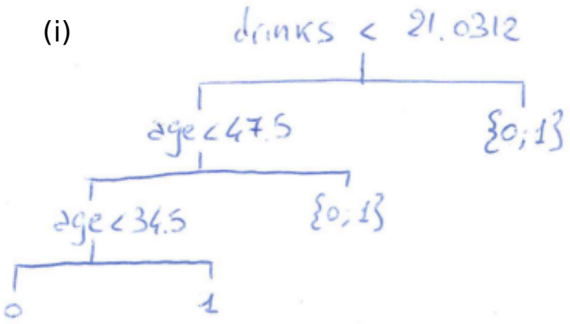
The predicted response for a 40-year old man who smokes and drinks, on average, 16 ounces of alcoholic drinks per week is 0 (no oral cancer). It could be found following the path: left (drink less than 21.0312 ounces per week), left (younger than 47.5 years), right (older than 34.5 years).

Cross-validation is a procedure to estimate the prediction error, often used to select the tuning parameter(s) of a model (here the number of leaves). It consists in splitting the observations in K approximately equal-sized folds and using in turn one fold as an independent test set, in which the performance of a model (or, more generally, a method) trained on the remaining $K - 1$ folds is evaluated. The K estimates of the performance measure (e.g., deviance) are then summarized (e.g., by averaging) into a single measure. If the goal is finding the best parameter, the procedure is performed for each parameter and that corresponding to the best performance is selected.

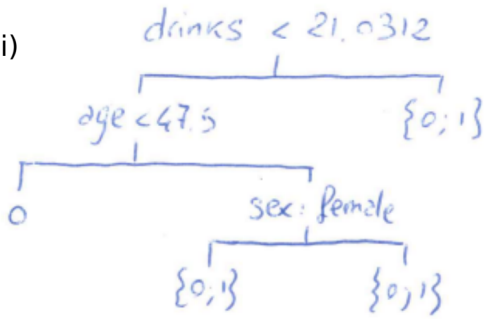
Here cross-validation is used to select the tuning parameter. In order to do this, it is important to evaluate the results obtained with a specific value of the tuning parameter in an independent set, to avoid overfitting. Having a moderately low number of observations (less than 200, see point d), it is not reasonable to split them in separated training and test sets. Cross-validation is used because it allows to take advantage of all observations to find the tuning parameter while keeping the independence between training and test set.

There are at least 4 possible ways to prune the tree while keeping the right response for the described subject. All the following choices are correct:

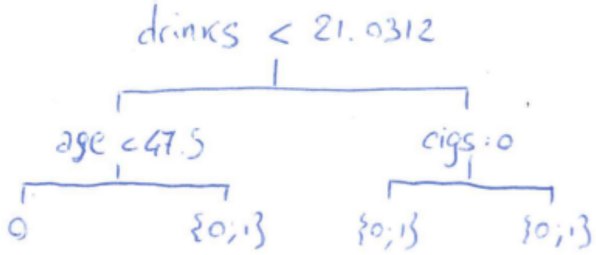
(i)



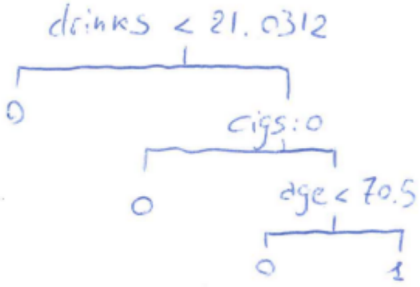
(ii)



(iii)



(iv)



Comment from Lars: I think (i) is incorrect, the "0" and "1" should be flipped. Can argue that some of the other trees should have "$\{0, 1\}$" instead of a single value.

Exercises 3

a)

In local regression methods h is a tuning parameter that controls the width of the window used to locally estimate the function. Smaller values of h correspond to smaller windows, and consequently to a reduced number of points used to estimate the value of interest. Larger values of h, instead, correspond to larger windows, and consequently to estimates that follow less the local behavior of the curve, as they also take into account farthest points.

Therefore, smaller values of h mean smaller bias and larger variance (the estimated curve follows the local behavior of the data, with the risk, if h is too small, of modeling random fluctuations). In contrast, larger values mean increased bias but reduced variance.

The concept of bias-variance trade-off is crucial to define the best prediction model. Basically, one desires a model which explains well the data (low bias) without following their random fluctuations (low variance). Unfortunately the two aspects are in contrast: decreasing the bias results in an increase of the variance, and the other way around. Both extremes, low bias but too much variance (overfitting) and low variance but too much bias (underfitting), should be avoided, and therefore the best balance (trade-off) should be found.

b)

The "curse of dimensionality" defines the phenomenon for which the number of observations close to a point decreases exponentially with the increase of the number of dimensions. As a consequence, non-parametric estimations have to rely on fewer points, with an obvious decrease in precision, or farther points, which are less likely to be similar to the point under investigation. A possible solution is to concentrate the information in fewer dimensions, for example by applying techniques such as principal component analysis.

Extra exercises

Exercise 14

a)

Bayes classifier is a classifier that minimizes the probability of misclassification (i.e. error rate). By using Bayes' theorem, we have

$$\Pr(Y|X) = \frac{\Pr(Y,X)}{\Pr(X)} = \frac{\Pr(X|Y)\Pr(Y)}{\Pr(X)}.$$

We are given that

$$\Pr(X|Y=k) = \text{Poisson}(\lambda_k) = \frac{(5+5k)^x e^{-(5+5k)}}{x!} = \frac{5^x(1+k)^x e^{-5(1+k)}}{x!}$$

and $\pi_k = \Pr(Y=k) = 1/K$, for $k = 1, 2, 3$ and where $K = 3$.

We can then compute the marginal distribution of X. We get

$$\begin{aligned}
\Pr(X) &= \sum_{k=1}^{3} \Pr(X|Y=k)\Pr(Y=k) \\
&= \frac{1}{3}\sum_{k=1}^{3} \Pr(X|Y=k) \\
&= \frac{1}{3}\left(\frac{10^x e^{-10}}{x!} + \frac{15^x e^{-15}}{x!} + \frac{20^x e^{-20}}{x!}\right) \\
&= \frac{5^x e^{-10}}{3(x!)}\left(2^x + 3^x e^{-5} + 4^x e^{-10}\right).
\end{aligned}$$

We can know insert all of these values into Bayes' theorem to obtain the following conditional distribution

$$\begin{aligned}
\Pr(Y=k|X) &= \frac{\Pr(X|Y=k)\Pr(Y=k)}{\Pr(X)} \\
&= \frac{\frac{5^x(1+k)^x e^{-5(1+k)}}{x!}\frac{1}{3}}{\frac{5^x e^{-10}}{3(x!)}\left(2^x + 3^x e^{-5} + 4^x e^{-10}\right)} \\
&= \frac{(1+k)^x e^{5(1-k)}}{2^x + 3^x e^{-5} + 4^x e^{-10}}.
\end{aligned}$$

Minimizing the probability of misclassification is equal to maximizing the probability of correct classification. Thus, we get the following Bayes classifier:

$$\text{argmax}_k \Pr(Y=k|X) = \text{argmax}_k \left\{ \frac{(1+k)^x e^{5(1-k)}}{2^x + 3^x e^{-5} + 4^x e^{-10}} \middle| x \right\} = \text{argmax}_k \left\{ (1+k)^x e^{5(1-k)} \middle| x \right\}$$

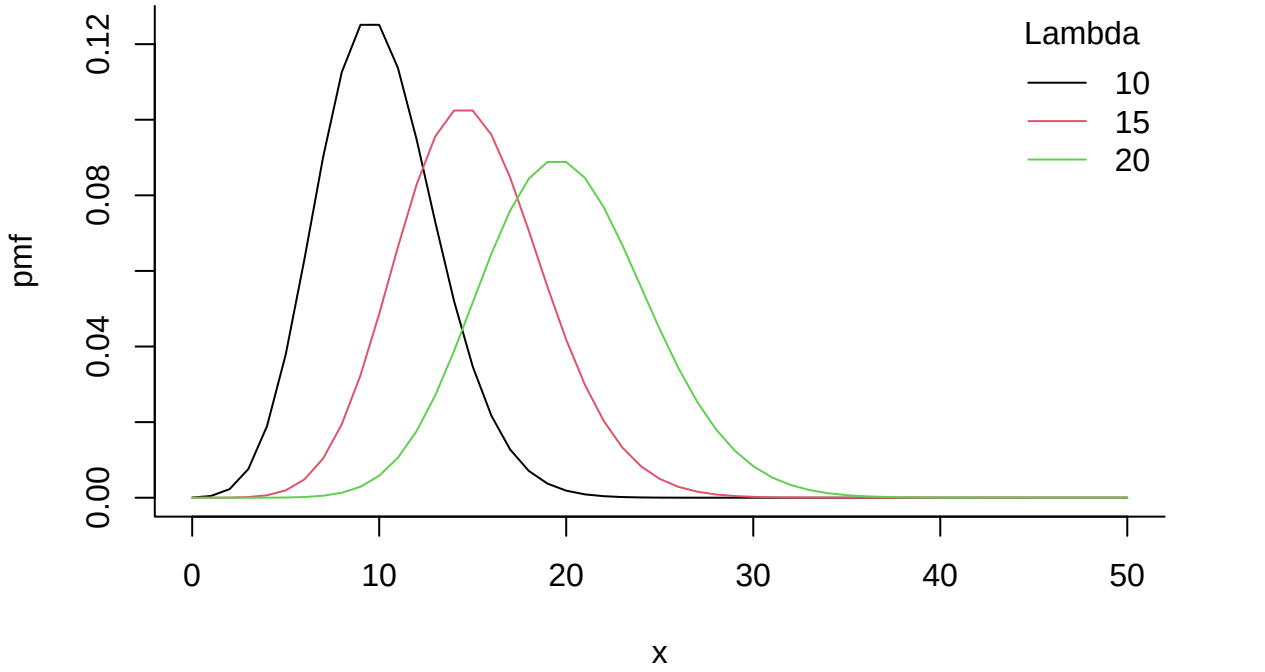
b)

Bayes classifier is a classifier that minimizes the probability of misclassification (i.e. error rate). So, error rate of Bayes classifier:

$$\Pr(Y \neq \hat{Y}|X) = 1 - \Pr(Y = \hat{Y}|X) = 1 - \max \Pr(Y=k|X).$$

Start by looking at the three probability mass functions. We see that there is an overlap from around 5 to 25, hence, we expect a higher error rate between these values.

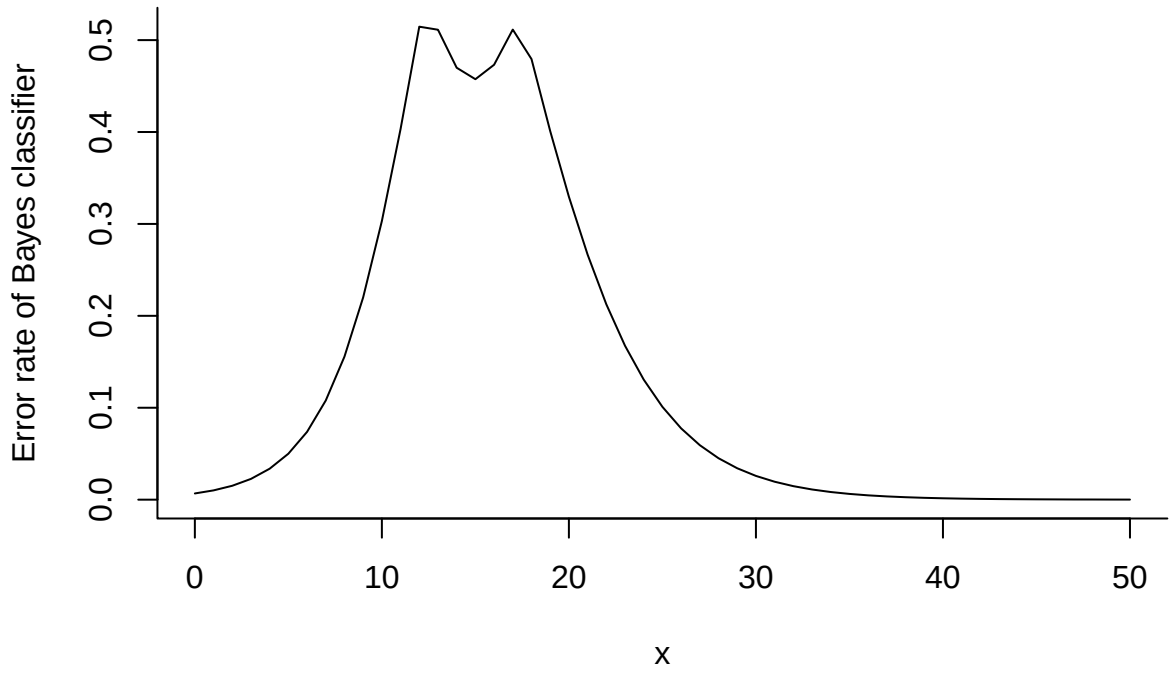
```r
# Start by looking at the three distributions
x.grid = 0:50
matplot(x.grid, cbind(dpois(x.grid, lambda = 10),
                      dpois(x.grid, lambda = 15),
                      dpois(x.grid, lambda = 20)),
        type = "l", lty = 1, bty = "l", xlab = "x", ylab = "pmf")
legend("topright", title = "Lambda", legend = c(10, 15, 20),
       lty = 1, col = 1:3, bty = "n")
```




```r
# Theoretical error rate of Bayes classifier
theoretical.Bayes.error.rate = function(x,K) {
  prob.mat = data.frame(k = 1:K, prob = NA)
  for (k in 1:K) {
    prob.mat[k, "prob"] = ((1 + k)^x)*exp(5 - 5*k)/(2^x +3^x*exp(-5) + 4^x*exp(-10))
  }
  theo.error.rate = 1 - prob.mat[which.max(prob.mat[,"prob"]), "prob"]
  return(theo.error.rate)
}

theoretical.Bayes.error.rate.vec =
  Vectorize(theoretical.Bayes.error.rate, vectorize.args = c("x"))

# Plot the theoretical error rate of Bayes classifier.
y.grid = theoretical.Bayes.error.rate.vec(x.grid, 3)
plot(x = x.grid, y = y.grid, type = "l", bty = "l",
     xlab = "x", ylab = "Error rate of Bayes classifier")
```



c)

```
set.seed(1)

# Simulate y
simulated.data = data.frame(y = sample(x = 1:3, size = 1000, replace = T))

# Simulate X
simulated.data[(simulated.data[,"y"] == 1), "x"] =
  rpois(sum(simulated.data[,"y"] == 1), 10)
simulated.data[(simulated.data[,"y"] == 2), "x"] =
  rpois(sum(simulated.data[,"y"] == 2), 15)
simulated.data[(simulated.data[,"y"] == 3), "x"] =
  rpois(sum(simulated.data[,"y"] == 3), 20)

# Bayes classifier
Bayes.classifier = function(x,K) {
  prob.mat = data.frame(k = 1:K, prob = NA)
  for (k in 1:K) {
    prob.mat[k, "prob"] = ((1 + k)^x)*exp(5 - 5*k)/(2^x +3^x*exp(-5) + 4^x*exp(-10))
}
  y.hat = prob.mat[which.max(prob.mat[,"prob"]), "k"]
  return(y.hat)
}

# Compute y.hat based on Bayes classifier.
Bayes.classifier.vec = Vectorize(Bayes.classifier, vectorize.args = c("x"))
simulated.data[,"y.hat.Bayes"] = Bayes.classifier.vec(simulated.data[,"x"], 3)
simulated.data[,"is.pred.correct"] =
  as.numeric(simulated.data[,"y.hat.Bayes"] == simulated.data[,"y"])
```

```r
# Take a look at the data frame
head(simulated.data, 10)
```

```
##    y  x y.hat.Bayes is.pred.correct
## 1  1  7           1               1
## 2  3 18           3               1
## 3  1  8           1               1
## 4  2 11           1               0
## 5  1  5           1               1
## 6  3 17           2               0
## 7  3 17           2               0
## 8  2 15           2               1
## 9  2 17           2               1
## 10 3 14           2               0
```

```r
# Overall error rate
error.rate = 1 -
  sum(simulated.data[,"y"] == simulated.data[,"y.hat.Bayes"])/nrow(simulated.data)
show(error.rate)
```

```
## [1] 0.345
```

```r
# Error rate per x value
empirical.error.rate.mat = data.frame(
  x = sort(unique(simulated.data[,"x"])),
  n = as.numeric(table(simulated.data[,"x"])),
  n.correct.pred = NA)

for (i in 1:nrow(empirical.error.rate.mat)) {
  x.target = empirical.error.rate.mat[i, "x"]
  empirical.error.rate.mat[i, "n.correct.pred"] =
    sum(simulated.data[(simulated.data[,"x"] == x.target), "is.pred.correct"])
}

empirical.error.rate.mat[,"error.rate"] =
  1 - empirical.error.rate.mat[,"n.correct.pred"]/empirical.error.rate.mat[,"n"]

# Take a look at the data frame
head(empirical.error.rate.mat, 15)
```

```
##     x  n n.correct.pred error.rate
## 1   1  2             2 0.00000000
## 2   2  3             3 0.00000000
## 3   3  4             4 0.00000000
## 4   4  8             8 0.00000000
## 5   5 15            15 0.00000000
## 6   6 30            28 0.06666667
## 7   7 38            34 0.10526316
## 8   8 44            32 0.27272727
## 9   9 57            47 0.17543860
## 10 10 72            48 0.33333333
## 11 11 60            31 0.48333333
## 12 12 70            32 0.54285714
## 13 13 56            25 0.55357143
## 14 14 67            42 0.37313433
## 15 15 59            35 0.40677966
```

```r
# Plot the error rate as a function of x.
plot(x = empirical.error.rate.mat[,"x"], y = empirical.error.rate.mat[,"error.rate"],
     type = "l", xlab = "x", ylab = "Error rate of Bayes classifier", bty = "l")
points(x = x.grid, y = y.grid, type = "l", lty = 2, col = "red")
legend("topright", c("Theoretical error rate","Error rate from simulation"),
       lty = c(2,1), col = c("red","black"), bty = "n", cex = 0.8)
```