Exercise 1 :

Solve Exercise 10.1 in PM.

Solution: The posterior distribution for C given $A = a$ and $B = b$ is

$$P(C \mid a, b) = \frac{P(C, a, b)}{P(C = 0)P(a|C = 0)P(b \mid C = 0) + P(C = 1)P(a|C = 1)P(b \mid C = 1)}.$$

Both terms in the denominator are zero due to $P(A \mid C = 0)$ and $P(B \mid C = 1)$, respectively.

Exercise 2 :

Consider a poker game consisting of two rounds, and where each player is initially dealt three cards. During the first round all three cards can be changed (*FC*), but during the second round at most two cards can be changed (*SC*). When deciding on whether to call or fold you can taken into account the number of cards changed by your opponent as well as your current hand (*MH*). After playing 20 games we have the results in Table 1, where *BH* shows who has the best hand.

Case number:	*BH*	*MH*	*FC*	*SC*
1	op	no	3	1
2	op	1a	2	1
3	draw	2 v	1	1
4	me	2 a	1	1
5	draw	fl	1	1
6	me	st	3	2
7	me	3 v	1	1
8	me	sfl	1	0
9	op	no	0	0
10	op	1 a	3	2
11	draw	2 v	2	1
12	me	2 v	3	2
13	op	2 v	1	1
14	op	2 v	3	0
15	me	2 v	3	2
16	draw	no	3	2
17	draw	2 v	1	1
18	op	fl	1	1
19	op	no	3	2
20	me	1 a	3	2

Table 1: Training data for constructing a poker classifier.

- Construct a naive Bayes classifier for the poker domain.
- Use the data cases to learn the parameters in the model; if you feel comfortable with the estimation procedure, you only need to estimate the probabilities required for solving the exercise below.

- What class does your classifier assign to a case with *MH=1a*, *FC=1*, and *SC=1*?

Solution:

See the Hugin network poker_model.net. The probabilities in the model have been calculated using simple frequency counting. For example, for $P(FC = 1|BH = op)$ we get

$$P(FC = 1|BH = op) = \frac{N(FC = 1, BH = op)}{N(BH = op)} = \frac{2}{8} = \frac{1}{4}.$$

By estimating the remaining entries in the conditional probability tables and inserting the evidence $MH = 1a, FC = 1, SC = 1$, we get the posterior probability $P(BH|MH = 1a, FC = 1, SC = 1) = (0.671_{op}, 0.329_{me}, 0_{dr})$.

Observe that if you are *only* interested in calculating the probability $P(BH|MH = 1a, FC = 1, SC = 1)$, then you need not estimate all the probabilities required to get a fully specified naive Bayesian network (this would require estimating $P(BH), P(MH|BH), P(FC|BH), P(SC|BH)$). Instead you only need to estimate probabilities for the configurations that are consistent with the configuration that you are conditioning on. Specifically,

$$\begin{aligned} &P(BH|MH = 1a, FC = 1, SC = 1)\\ &\quad = \frac{P(BH)P(MH = 1a|BH)P(FC = 1|BH)P(SC = 1|BH)}{\sum_{BH} P(BH)P(MH = 1a|BH)P(FC = 1|BH)P(SC = 1|BH)}, \end{aligned}$$

so you need not estimate probabilities that are not used in the calculations above (e.g. $P(MH = 2v|BH)$).

Exercise 3 :

You want to predict whether a person will pay back a loan based on the features *Income*, *Houseowner* and *Marital Status* of the person. Domains and distance functions on the domains of these features are defined as follows:

Income	*low*	*medium*	*high*
low	0	1	2
medium	1	0	1
high	2	1	0

Houseowner	*yes*	*no*
yes	0	1
no	1	0

Marital Status	*unmarried*	*married*	*divorced*
unmarried	0	1	1
married	1	0	1
divorced	1	1	0

Define the distance between two examples by the sum of the distances for the three features.

Your training examples are:

	Income	Houseowner	Marital Status	Pay back
1	high	yes	married	yes
2	high	yes	unmarried	yes
3	medium	no	divorced	no
4	low	yes	married	no
5	low	no	unmarried	no

Now you want to predict 'Pay back' for a new case with Income = high, Houseowner = no, Marital Status = divorced.

- What is the prediction obtained by the 1-nearest-neighbor rule?
- What is the prediction obtained by the 3-nearest-neighbor rule?
- What could be a sensible modification of the distance function such that you would get a different result from the 1-nearest-neighbor rule?

Solution:

The distance of the new case to the 5 training examples is:

1	0 + 1 + 1 = 2
2	0 + 1 + 1 = 2
3	1 + 0 + 0 = 1
4	2 + 1 + 1 = 4
5	2 + 0 + 1 = 3

Thus:

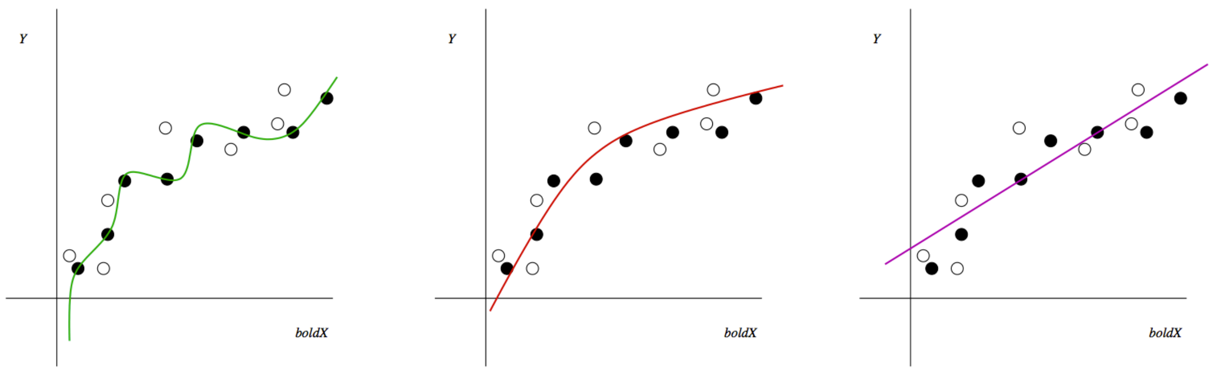
- The prediction from the 1-nearest-neighbor rule is Pay back = no, because training example 3 has smallest distance to the new case.
- The prediction from the 3-nearest-neighbor rule is Pay back = yes, because 2 out of the 3 nearest neighbors (training cases 1,2,3) have Pay back = yes.
- A sensible modification of the distance function could be to give lower weight to the distance contributed by the marital status attribute, as by the income attribute. For example, if we give a weight of 10 to the income attribute, a weight of 5 to the houseowner attribute, and a weight of 1 to the marital status attribute, then the distances are

1	$10 \cdot 0 + 5 \cdot 1 + 1 = 6$
2	$10 \cdot 0 + 5 \cdot 1 + 1 = 6$
3	$10 \cdot 1 + 5 \cdot 0 + 0 = 10$
4	$10 \cdot 2 + 5 \cdot 1 + 1 = 26$
5	$10 \cdot 2 + 5 \cdot 0 + 1 = 21$

  Now there is a tie between training cases 1 and 2 for being the nearest neighbor, but both have Pay back = yes, so now the 1-nearest neighbor rule would predict Pay back = yes.

Exercise 4 :

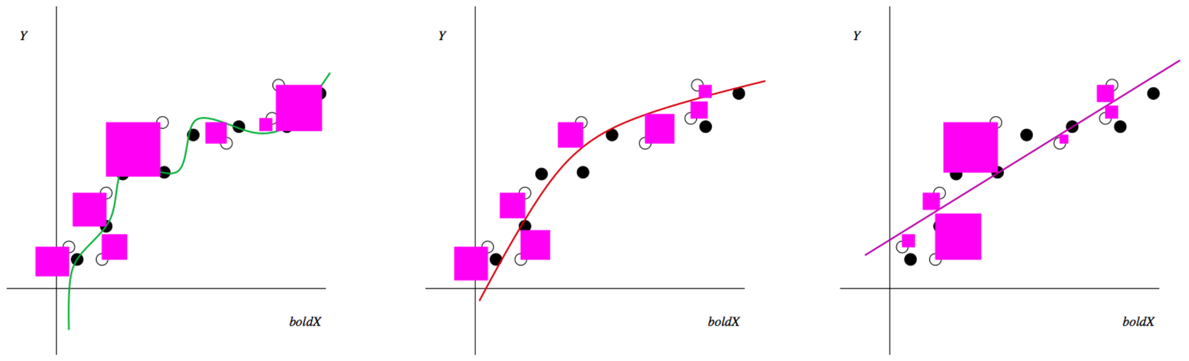
The following graphs show three regression models learned from training examples (filled dots):



The open dots represent a set of future observations (or a validation set). Which of the three models has the smallest SSE on these future observations? No exact computations required – try to read it (approximately) off the graphs!

Solution:

Indicating the squared errors of each validation example by a colored square (cf. slide 09.22) gives:



From this it appears (subject to exact verification by measurement!) that the SSE of the leftmost model is much higher than that of the other two, and that the one of the middle model is a little lower than that of the right (linear) model.

Exercise 5 :

Suppose you are working on a spam detection system. You formulated the problem as a classification task where "Spam" is the positive class and "not Spam" is the negative class. Your training set contains m = 1000 emails.

	Predicted Spam	Predicted not Spam
Actual Spam	8	2
Actual not Spam	16	974

Calculate the Accuracy, Recall, Specificity and Precision in this example.

How do your answers change if we now consider "Spam" the negative class and "not Spam" the positive class?

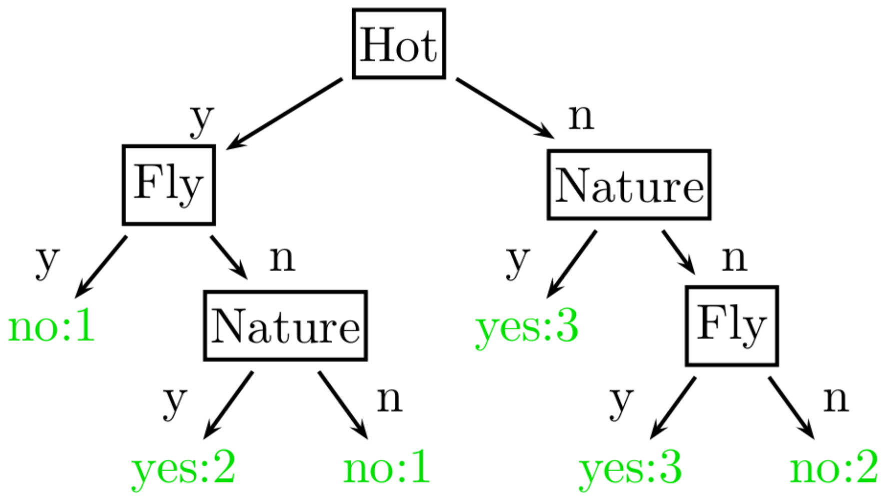
Solution:

- Accuracy $\frac{8+974}{1000} = 0.982$
- Recall $\frac{8}{8+2} = 0.80$
- Specificity $\frac{974}{974+16} = 0.9838$
- Precision $\frac{8}{8+16} = 0.3333$

If we consider "Spam" the negative class and "not Spam" the positive class then the original Specificity becomes Recall and vice versa, so new Recall is $\frac{974}{974+16} = 0.9838$ and new Specificity is $\frac{8}{8+2} = 0.80$. Accuracy remains the same, Accuracy $\frac{974+8}{1000} = 0.982$. Precision becomes $\frac{974}{976} = 0.9980$ (In tutorial note the difference in Precision values).

Exercise 6 :

Based on 12 training examples the following decision tree was learned:



Here, for example, the label "yes:2" at the end of the branch Hot=y, Fly=n, Nature=y means that there were two examples in the training set with Hot=y, Fly=n, Nature=y, and both examples had class label Likes=yes. Thus, the decision tree has 100% accuracy on the training data (all leaves are class pure).

Now suppose you have the following 5 examples (not used in the construction of the tree), which you want to use as a validation set:

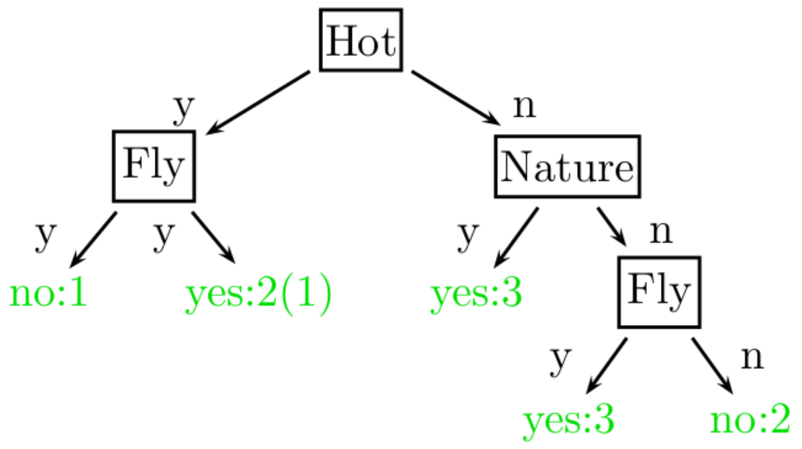
Culture	Fly	Hot	Music	Nature	Likes
no	no	yes	yes	yes	no
no	no	yes	no	no	yes
yes	yes	no	no	no	yes
yes	no	yes	no	yes	yes
no	no	no	no	no	no

Based on these validation examples, we perform post-pruning of the decision tree:

- First check whether the 'Nature' node reached by Hot=y, Fly=n should be pruned (eliminated)
- If yes, what does the new tree look like after pruning?
- Continue the pruning process by checking for other nodes whether they should be pruned (in a bottom-up order; the next candidate for pruning could be the 'Fly' node reached by Hot=n, Nature=n).

Solution:

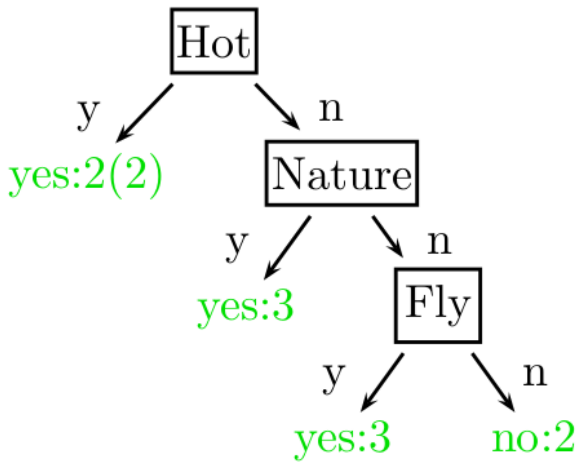
3 of the validation examples (1,2,and 4) go into the Hot=y, Fly=n branch. 2 of these (1,2) will be misclassified based on the 'Nature' feature. If the 'Nature' node is pruned, then it will be replaced by a leaf labeled with 'yes' (because the majority of the *original training examples* are 'yes' cases). With the resulting tree, then only validation example 1 will be misclassified. Thus, the performance on the validation set improves, we would perform the pruning to obtain:



The ' yes:2(1)' label means: cases ending in this leaf are predicted 'yes', and this is correct for 2 training examples, but there is also one 'no' training example.

Next we check whether the rightmost 'Fly' node should also be pruned. Two validation examples (3,5) reach this node. Both of them are correctly classified based on the 'Fly' feature. If that node was eliminated, then one of the two would have to be mis-classified. Therefore, the accuracy on the validation set is higher when the 'Fly' node is kept.

Finally, we can check whether pruning the 'Fly' node reached by Hot=y should be pruned. This node is still only reached by the three examples 1,2,4, two out of which are correctly classified. If the node was pruned, then the resulting leaf could be labeled either 'yes' or 'no' (because there are then 2 training examples for each of these labels). Assuming that we would label the resulting leaf 'yes', we would still obtain 2 out of 3 correct classifications for these three validation examples. Thus, by pruning, we obtain a tree with the same accuracy, but one that is a bit smaller than the previous. This is a borderline situation, and depending on how the pruning algorithm exactly resolve this, one might decide not to prune 'Fly', or prune this node also and end up with the tree:



Exercise 7 :

Look at this example, introduced in the lecture. Assuming the points are measurements from some experiment and the functions are fitting functions to predict further measurements. The left one is a linear function and the right one is a polynomial function. According to Ockham's razor, which one of these functions is the better pick. Justify your answer briefly (in 2-3 sentences), highlighting why Ockhams razor prefers one of the functions and what the problem of overfitting would be in this case.

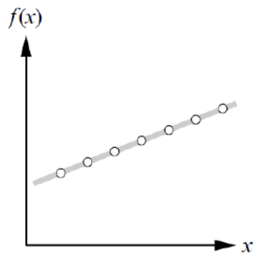


Figure 1: The linear function

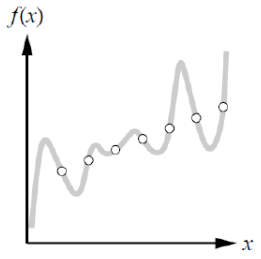


Figure 2: The polynomial function

Solution: Ockham's razor would pick the Linear Function. It prefers the simpler solution, and in this case, the polynomial function is more complex than the linear function. The polynomial function overfitts the past measurements, being correct for the current data points, but not following the general curve, so it will likely delivering worse predictions for future measurements.