Machine Learning A

708.064 11W 1sst KU

Exercises

Problems marked with * are optional.

1 Conditional Independence I [2 P]

a) [1 P] Give an example for a probability distribution $P(A, B, C)$ that disproves

$$P(A, B) = P(A)P(B) \rightarrow P(A, B|C) = P(A|C)P(B|C).$$

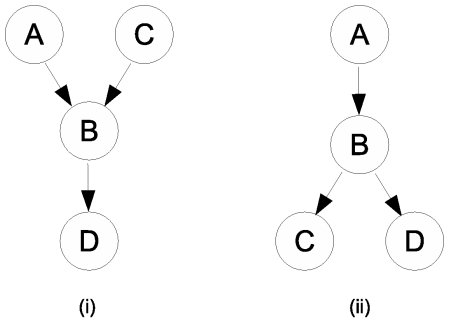
Illustrate the phenomenon of 'explaining away'.

b) [1 P] Give an example for a probability distributions $P(A, B, C)$ that disproves

$$P(A, B|C) = P(A|C)P(B|C) \rightarrow P(A, B) = P(A)P(B).$$

2 Conditional Independence II [2+1* P]

a) [2 P] Consider the two networks:



For each of them, determine whether there can be any other Bayesian network that encodes the same independence assertions.

b) [1* P] Give an example for a probability distribution P so that all independence assertions that hold in P can not be represented by any (single) Bayesian network.

3 Bayesian Networks [3 P]

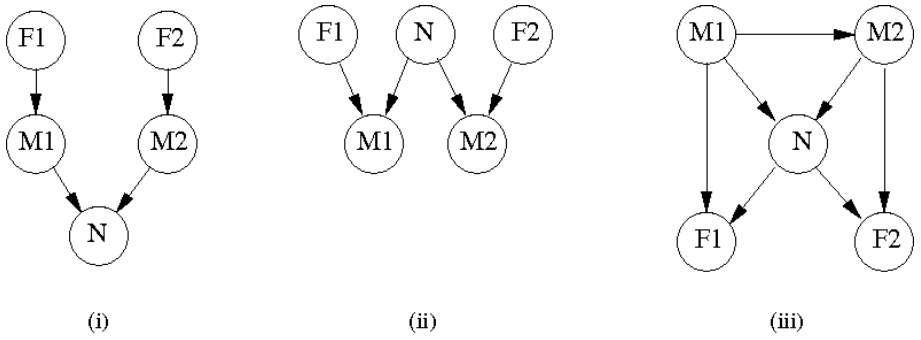


Figure 1: Three possible networks for the telescope problem.

Two astronomers in different parts of the world make measurements $M1$ and $M2$ of the number of stars N in some small region of the sky, using their telescopes. Normally, there is a small possibility e of error by up to one star in each direction. Each telescope can also (with a smaller probability f) be badly out of focus (events $F1$ and $F2$), in which case the scientists will undercount by three or more stars (or, if N is less than 3, fail to detect any stars at all). Consider the three networks illustrated in Figure 1.

a) [1 P] Which of these Bayesian networks are correct (but not necessarily efficient) representations of the preceding information?

b) [1 P] Which is the best network? Why?

c) [1 P] Write out a conditional distribution for $P(M1|N)$, for the case $N \in \{1, 2, 3\}$ and $M1 \in \{0, 1, 2, 3, 4\}$. Each entry in the conditional distribution should be expressed as a function of the parameters e and/or f.

4 d-separation [4+1* P]

a) Prove or disprove:

1. [1P] $(X \perp Y, W|Z)) \Rightarrow (X \perp Y|Z)$.
2. [1P] $(X \perp Y|Z)$ & $(X, Y \perp W|Z) \Rightarrow (X \perp W|Z)$.
3. [1P] $(X \perp Y, W|Z)$ & $(Y \perp W|Z) \Rightarrow (X, W \perp Y|Z)$.
4. [1* P] $(X \perp Y|Z)$ & $(X \perp Y|W) \Rightarrow (X \perp Y|Z, W)$.

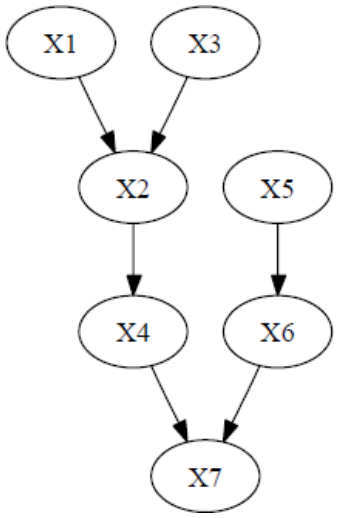


Figure 2: Bayesian network.

b) [1 P]

Apply d-seperation to determine whether the following conditional independence statements hold for the Bayesian network illustrated in Fig. 2.

1. $(X3 \perp X7)$
2. $(X3 \perp X7|X4)$
3. $(X3 \perp X5)$
4. $(X3 \perp X5|X7)$

5 Inference in Factor Graphs [2 P]

Consider a tree-structured factor graph, in which a given subset of the variable nodes form a connected subgraph (i.e., any variable node of the subset is connected to at least one of the other variable nodes via a single factor node). Show how the sum-product algorithm can be used to compute the marginal distribution over that subset.

6 Factor graphs: HMM model [5 P]

Implement the sum-product algorithm for factor graphs in MATLAB to infer the hidden states of a HMM for the following problem.

An agent is located randomly in a 10×10 grid world. In each of $T = 20$ time steps he either stays at his current location with a probability of 25% or

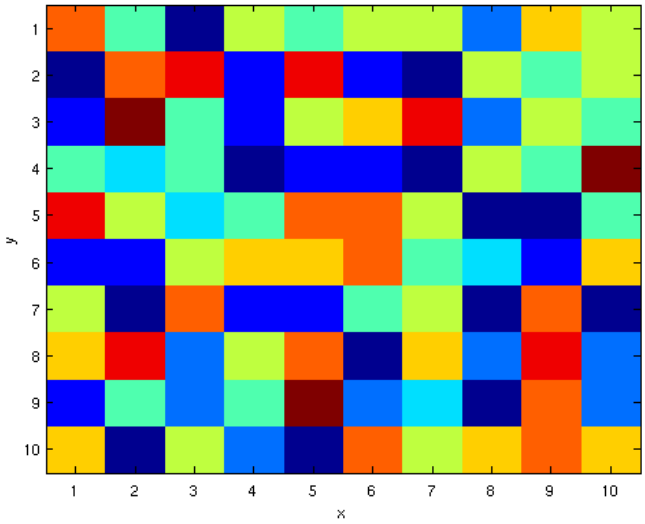


Figure 3: Grid world.

moves up, down, left or right, where each of these four actions is chosen with a probability of 18.75%. Each cell in the grid world is painted with a certain color that is chosen randomly from a set of k possible colors as shown in Fig. 3. This color is observed by the agent at each time step and reported to us. Only these color values are available to infer the initial location and the subsequent positions of the agent resulting in the Hidden Markov model (HMM) illustrated in Fig. 4.

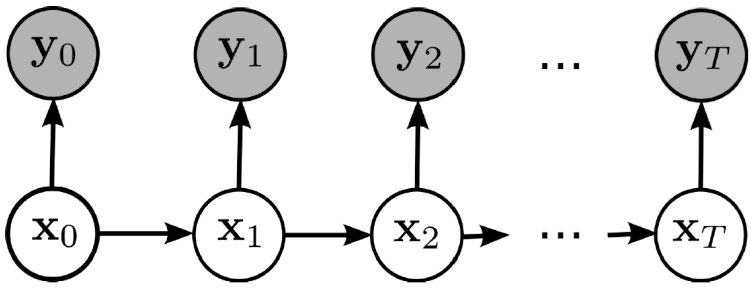


Figure 4: Hidden Markov model (HMM).

Modify the file `hmmMessagePassingTask.m` available for download on the course homepage[1] and implement the sum-product algorithm to solve this inference problem. Investigate and discuss the dependence of the solution on the number of different colors in the grid world (k). Hand in figures of representative results that show the actual agent trajectories, the most likely trajectories

[1] http://www.igi.tugraz.at/lehre/intern/MLA_WS1112_HW6.zip

and the probabilities of the agent positions at each time step as infered by the sum-product algorithm.

Present your results clearly, structured and legible. Document them in such a way that anybody can reproduce them effortless. Hand in printouts of your MATLAB code.

7 Message passing in Gaussian Bayesian networks [3 P]

Consider the Bayesian network illustrated in Fig. 5 with the probabilities

$$P(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0|\mathbf{a}_0, \mathbf{Q}_0) \quad (1)$$

$$P(\mathbf{x}_{t+1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t+1}|\mathbf{A}_t\mathbf{x}_t + \mathbf{a}_t|\mathbf{Q}_t) \quad (2)$$

$$P(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t|\mathbf{D}_t\mathbf{x}_t + \mathbf{d}_t|\mathbf{C}_t) \quad (3)$$

where $\mathcal{N}(\mathbf{x}|\mu, \mathbf{\Sigma})$ denotes the multivariate Gaussian distribution with mean μ and covariance $\mathbf{\Sigma}$.

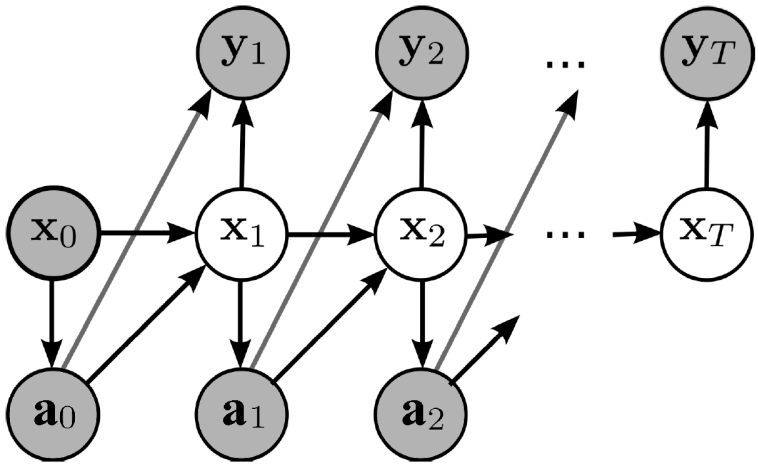


Figure 5: Gaussian Bayesian network.

Prove that the forward messages have again the form of a Gaussian distribution

$$\alpha(\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_t|\mathbf{S}_t^{-1}\mathbf{s}_t, \mathbf{S}_t^{-1}) \quad (4)$$

with mean $\mathbf{S}_t^{-1}\mathbf{s}_t$ and covariance $\mathbf{S}_t^{-1}$ as defined on the slides for the exercise hour (slide 23). (Hint: Use the three properties of Gaussian distributions discussed in the exercise hour.)

8 Factor graphs: Robot Localization [5 P]

Implement the sum-product algorithm for factor graphs in MATLAB for the Gaussian Bayesian network illustrated in Fig. 5 with the probabilities given in Eq. 1 - 3.

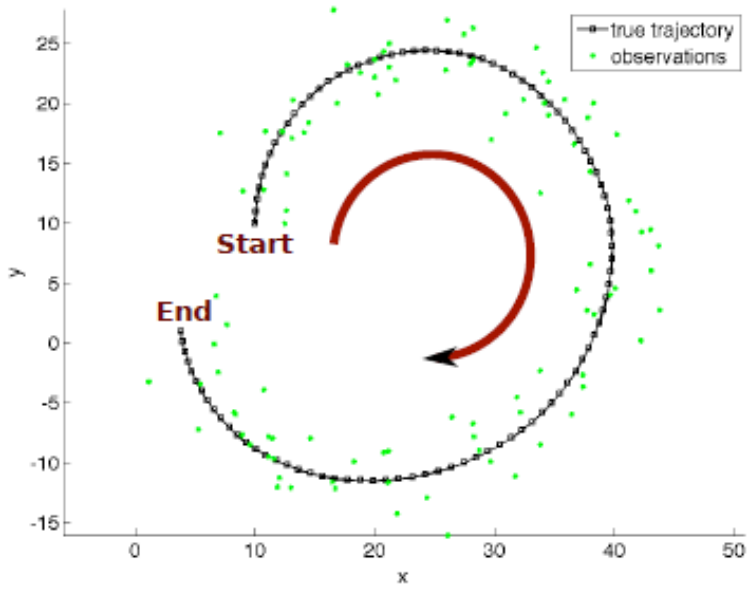


Figure 6: Robot control task.

Modify the files `mainTemplate.m`, `kalmanFilter.m` and `kalmanSmoother.m` available for download on the course homepage[2] and implement a) the Kalman filter algorithm and b) the Kalman smoother algorithm. The task is to infer the trajectory of a robot that starts at position $\mathbf{x}_0$ and moves according to the linear transition model

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t + \mathcal{N}(0|\mathbf{Q}) \tag{5}$$

for given actions $\mathbf{a}_t$ from the observations (green dots)

$$\mathbf{y}_t = \mathbf{D}\mathbf{x}_t + \mathbf{d} + \mathcal{N}(0|\mathbf{C}) \tag{6}$$

as illustrated in Fig. 6. The environment and the parameters for the linear state transition and observation model are already provided in the file `mainTemplate.m`.

Investigate and discuss the results for both algorithms. Hand in figures of representative results that show the actual robot trajectories and the most likely robot trajectories for both algorithms as infered by the sum-product algorithm.

[2] http://www.igi.tugraz.at/lehre/intern/MLA_WS1112_HW8.zip

Present your results clearly, structured and legible. Document them in such a way that anybody can reproduce them effortless. Hand in printouts of your MATLAB code.

9 Beta distribution [2* P]

Show that the mean, variance, and mode of the beta distribution are given respectively by

$$\begin{aligned} E[\mu] &= \frac{a}{a+b} \\ var[\mu] &= \frac{ab}{(a+b)^2(a+b+1)} \\ mode[\mu] &= \frac{a-1}{a+b-2}. \end{aligned}$$

10 EM Algorithm Applet [3 P]

Go to Olivier Michel's applet Gaussian Mixture Model EM Algorithm at `http://lcn.epfl.ch/tutorial/english/gaussian/html/index.html` and answer questions 1-6 [1 P]. Additionally do the following:

a) [1 P]

Create points from two intersecting lines and run the algorithm with option LineMix and 2 clusters. Click repeatedly on EM 1 Step to run the algorithm. Repeat this experiment with different cutting angles of the lines and analyze if this has an effect on the number of iterations until convergence.

b) [1 P]

Create points in a ring around the center with the button RingPts. Try to fit a mixture of lines model (with EM Run) to this distribution and analyze what happens when you change the number of lines. Do the same experiment with mixtures of Gaussians and describe what you find.

11 EM Algorithm for Mixtures of Lines [3 P]

Assume that the training examples $\mathbf{x}_n \in \mathbb{R}^2$ with $n = 1, ..., N$ were generated from a mixture of K lines

$$P(x_{n,2}|z_{n,k} = 1) = \mathcal{N}(x_{n,2}|\theta_{k,1}x_{n,1} + \theta_{k,2}, \sigma_k) \quad (7)$$

where

$$\mathcal{N}(x|\mu,\sigma) \;=\; \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{8}$$

and the hidden variable $z_{n,k} = 1$ if $\mathbf{x}_n$ is generated from line k and 0 otherwise. Derive the update equations for the M-step of the EM algorithm for the variables θ_k and σ_k.

12 Clustering: Face recognition [3 P]

In this homework example you should cluster images derived from the Olivetti face database with the cluster algorithms 'k-means' and 'k-medoids'. You can download the dataset and its description from the course homepage.[3]

- Normalize the pixels in each 50 by 50 image (each column of the design matrix) to have mean 0 and variance 0.1.
- Apply k-means and k-medoids to the dataset. For k-means use the MATLAB function `kmeans`. For k-medoids you have to write your own MATLAB function. For both algorithms use the squared Euclidean distance. Analyze the dependence of the average squared error on the number of clusters.
- Analyze for $k \in \{2, 5, 10, 30\}$ the cluster centers and outliers (points with the largest squared errors) for both algorithms. Explain the differences in the results.

13 Clustering: Image compression [3 P]

Apply the k-means algorithm for lossy image compression by means of vector quantization.

- Download the 512×512 image `mandrill.tif`.[4] Each pixel represents a point in a three dimensional (r,g,b) color space. Each color dimension encodes the corresponding intensity with an 8 bit integer.
- Cluster the pixels in color space using $k \in \{2, 4, 8, 16, 32, 64, 128\}$ clusters and replace the original color values with the indices of the closest cluster centers. Determine the compression factor for each value of k and relate it to the quality of the image. Apply an appropriate quality measure of you choice.

[3] http://www.igi.tugraz.at/lehre/intern/MLA_WS1112_HW12.zip
[4] http://www.igi.tugraz.at/lehre/intern/MLA_WS1112_HW13.zip

14 Mixture distributions [2* P]

Consider a density model given by a mixture distribution

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k p(\mathbf{x}|k) \tag{9}$$

and suppose that we partition the vector $\mathbf{x}$ into two parts so that $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$. Show that the conditional density $p(\mathbf{x}_b|\mathbf{x}_a)$ is itself a mixture distribution and find expressions for the mixing coefficients and for the component densities.

15 Parameter Learning for Naive Bayes Classifiers [3* P]

Implement an algorithm for learning a naive Bayes classifier and apply it to a spam email data set. You are required to use MATLAB for this assignment. The spam dataset is available for download on the course homepage[5].

a) [1/2* P]

Write a function called `nbayes_learn.m` that takes a training dataset for a binary classification task with binary attributes and returns the posterior Beta distributions of all model parameters (specified by variables a'_i and b'_i for the ith model parameter) of a naive Bayes classifier given a prior Beta distribution for each of the model parameters (specified by variables a_i and b_i for the ith model parameter).

b) [1/2* P]

Write a function called `nbayes_predict.m` that takes a set of test data vectors and returns the most likely class label predictions for each input vector based on the posterior parameter distributions obtained in a).

c) [1* P]

Use both functions to conduct the following experiment. For your assignment you will be working with a data set that was created a few years ago at the Hewlett Packard Research Labs as a testbed data set to test different spam email classification algorithms.

1. Verify the naive Bayes assumption for all pairs of input attributes.

[5] http://www.igi.tugraz.at/lehre/intern/MLA_WS1112_HW15.zip

2. Train a naive Bayes model on the first 2500 samples (using Laplace uniform prior distributions) and report the classification error of the trained model on a test data set consisting of the remaining examples that were not used for training.
3. Repeat the previous step, now training on the first {10, 50, 100, 200, ... , 500} samples, and again testing on the same test data as used in point 1 (samples 2501 through 4601). Report the classification error on the test dataset as a function of the number of training examples. Hand in a plot of this function.
4. Comment on how accurate the classifier would be if it would randomly guess a class label or it would always pick the most common label in the training data. Compare these performance values to the results obtained for the naive Bayes model.

d) [1* P] Train a feedforward neural network with one sigmoidal output unit and no hidden units with backpropagation (use the algorithm `traingdx` and initialize the network with small but nonzero weights) on the first {10, 50, 100, 200, ... , 500} samples and test on the same test data as used in point 1 (samples 2501 through 4601). Report the classification error on the test dataset as a function of the number of training examples and compare the results to the one obtained for the naive Bayes classifier. Hand in a plot of this function.

Present your results clearly, structured and legible. Document them in such a way that anybody can reproduce them effortless.