732A96/TDDE15 ADVANCED MACHINE LEARNING

EXAM 19/10-2017

TEACHERS

Jose M. Peña. Phone: 0700895280.

GRADES

- For 732A96 (A-E means pass):
  - A=19-20 points
  - B=17-18 points
  - C=12-16 points
  - D=10-11 points
  - E=8-9 points
  - F=0-7 points
- For TDDE15 (3-5 means pass):
  - 5=18-20 points
  - 4=12-17 points
  - 3=8-11 points
  - U=0-7 points

The total number of points is rounded to the nearest integer. In each question, full points requires clear and well motivated answers.

ALLOWED MATERIAL

Hard copy of Bishop's book, and the content of the folder given_files in the exam system.

INSTRUCTIONS

The answers to the exam should be submitted in a single PDF file using the communication client. You can make a PDF from LibreOffice (similar to Microsoft Word). You can also use Markdown from RStudio. Include important code needed to grade the exam (inline or at the end of the PDF file). Submission starts by clicking the button ”Skicka in uppgift” in the communication client. Then, follow the instructions. Note that the system will let you know that the exam has been submitted, but will not tell you that it was received. This is ok and your solution has actually been received.

Do not ask question through the communication client. The teachers will be reachable by phone, and they will visit the room too.

1. Graphical Models (5 p)

- Learn a Bayesian network (BN) from the Asia dataset that is included in the `bnlearn` package. To load the data, run `data("asia")`. Learn both the structure and the parameters. Use any learning algorithm and settings that you consider appropriate. Identify a d-separation in the BN learned and show that it indeed corresponds to an independence in the probability distribution represented by the BN. To do so, you may want to use exact or approximate inference with the help of the `bnlearn` and `gRain` packages. (2.5 p)
- There are 29281 directed and acyclic graphs (DAGs) with five nodes. Compute approximately the fraction of these 29281 DAGs that are essential. An essential DAG is a DAG that is not Markov equivalent to any other DAG. The simplest way to solve the exercise may be by re-using the code that you produced for the lab. For this to work, you have to figure out how to determine if a DAG is essential by just looking at its CPDAG (a.k.a. essential graph). (2.5 p)

2. Hidden Markov Models (5 p)

- Consider a robot moving along a straight corridor. The corridor is divided into 100 segments. The corridor has three doors: The first spans segments 10, 11 and 12, the second spans segments 20, 21 and 22, and the third spans segments 30, 31, and 32. In each time step, the robot moves to the next segment with probability 0.9 and stays in the current segment with probability 0.1. You do not have direct observation of the robot. However, the robot is equipped with a sensor that is able to detect whether the robot is or is not in front of a door. The accuracy of the sensor is 90 %. Initially, the robot is in any of the 100 segments with equal probability. You are asked to build a hidden Markov model (HMM) to model the robot's behavior. You may want to use the `HMM` package. (2.5 p)
- Give a sequence of observations $x_{1:t}$ such that $p(z_t|x_{1:t})$ is unimodal, i.e. there is a segment that is more likely to contain the robot than the rest of the segments. Recall that initially the robot is in any of the 100 segments with equal probability. Use the `HMM` package to show that the sequence really solves the exercise. (2.5 p)

  Note that the function `which.max` only returns the first maximum, not all of them. Thus, you may want to use the function below, which does return all the maxima.

  ```
  which.maxima<-function(x){
    return(which(x==max(x)))
  }
  ```

3. Gaussian Processes

The file `KernelCode.R` distributed with the exam contains code to construct a `kernlab` function for the Squared Exponential covariance function:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{r^2}{2\ell^2}\right)$$

where $r = |\mathbf{x} - \mathbf{x}'|$.

(a) Let $f \sim GP(0, k(\mathbf{x}, \mathbf{x}'))$ a priori and let $\sigma_f^2 = 1$. Simulate and plot 5 realizations from the prior distribution of f over the grid `xGrid = seq(-1,1,by=0.1)` for the two length scales $\ell = 0.2$ and $\ell = 1$ (in two separate figures). Compute

i. $\mathrm{Corr}(f(0), f(0.1))$ and

ii. $\mathrm{Corr}(f(0), f(0.5))$

for the two length scales. Discuss the results and connect your discussion to the concept of smoothness of f. [Hint: you will probably need the `mvtnorm` package.] (2 p)

(b) The file GPdata.RData contains two variables `y` and `x`. Load the variables into memory with the R command `load("GPdata.RData")`. Compute the posterior distribution of f in the model

$$y = f(x) + \varepsilon, \quad \varepsilon \sim N(0, 0.2^2).$$

You should do this for both length scales $\ell = 0.2$ and $\ell = 1$. Set $\sigma_f = 1$. Your answer should be in the form of a scatter plot of the data overlayed with curves for

i. the posterior mean of f

ii. 95% probability intervals for f

iii. 95% prediction intervals for a new data point y

Explain the difference between the results from ii) and iii). Discuss the differences in results from using the two length scales. Do you think a GP with a squared exponential kernel is a good model for this data? If not, why?

Use the `gausspr` function in the kernlab package for i), but not for ii) and iii).

[Hint: $Cov(f) = K(X_\star, X_\star) - K(X_\star, X)\left[K(X, X) + \sigma_n^2 I\right]^{-1} K(X, X_\star)$ and remember that `%*%` does matrix multiplication and `solve` computes inverses in R] (2 p)

(c) (No need to do any computations here). Discuss how a Bayesian would handle the case where the kernel hyperparameters are unknown. What if the noise variance is unknown? (1 p)

4. State Space Models (5 p)

- Consider the following state space model (SSM):

$$\begin{aligned} p(x_t|x_{t-1}) &= \mathcal{N}(x_t|x_{t-1}+1, 1) \\ p(z_t|x_t) &= \mathcal{N}(z_t|x_t, 5) \\ p(x_0) &= \mathcal{N}(x_0|50, 10) \end{aligned}$$

  Implement and simulate the SSM above for $T = 10000$ time steps to obtain a sequence of observations $z_{1:T}$ and hidden states $x_{1:T}$. Implement the Kalman filter as it appears in the course slides or in the book by Thrun et al. Note that the SSM above specifies standard deviations 1, 5 and 10 for the transition, emission and initial models. However, the Kalman filter in the slides and in the book is described in terms of variances instead.

  Run the Kalman filter on the observations $z_{1:T}$. Report the mean and standard deviation of the errors for the $T = 10000$ time steps. The error for time t is defined as $abs(x_t - E[x_t])$, where the expectation is with respect to the filtered distribution (a.k.a. belief function). (2 p)
- Repeat the exercise above with the particle filter. You may want to re-use the code you produced for the lab. (2 p)
- Compare the performance (i.e. mean error and runtime) of the Kalman and particle filters on the same sequence of observations when using 10, 50 and 100 particles. Explain the differences in performance between the filters. (1 p)

  To measure runtime, you may want to use the code below.

```
start_time <- Sys.time()
# Your code
end_time <- Sys.time()
end_time - start_time
```

Good luck !