MTAT.03.227 Machine Learning
Spring 2013 / Exercise session VIII
Nominal score: 10p
Maximum score: 15p
Deadline: 16th of April 8:15 EET

1. Quantile-quantile plots are the best way to compare samples from two different distributions. In brief, if `x` and `y` are sample vectors of same size coming from two different distributions then the command `plot(sort(x), sort(y))` produces qq-plot. By sorting the values, we compute the quantiles of the distribution. Hence, the plot visualises how the corresponding quantiles are related. If `x` and `y` come from the same distribution, the resulting graph should ideally follow the line $x = y$. Due to random fluctuations, the plot deviates from a straight line. In the following, we take the frequentistic viewpoint and ask how big deviations occur on average if `x` and `y` are from the same distribution. Obtained knowledge allows us to interpret the output of unknown qq-plots.

   (a) The simplest way to analyse average-case behaviour is by simulation. That is, we draw many sample pairs `x` and `y` from the same distribution and observe how do qq-plot look like on average. As the fluctuations might depend on the data distribution, we will experiment with uniform, normal and exponential distributions (`runif`, `rnorm`, `rexp`). For a basic experiment, take 100 samples for `x` and `y` and draw the qq-plot. For each distribution, repeat this experiment 100 times and draw points on the same graph with `points` function. As a result, you should get three different fluctuation patterns. (**1p**)

   (b) If the distribution to compare with is known, then the use of theoretical quantiles reduces the amount of noise in the graph. For instance, if we want to compare a sample vector `y` of size 100 with uniform distribution, we can compute`x<-qunif(c(1:100)/100)`. Let us now estimate the effect of fluctuations in more detail. Draw $1,000$ sample vectors`y` of size 100 for all distributions. But this time store empirical quantiles $q_{0.01}, \ldots q_{0.50}, \ldots, q_{1.00}$ for each draw into a row of a 1000×100 matrix `Q`. As a result, every column `Q[*,i]` represents individual measurements of a quantile q_x and you can compute compute 2.5%, 50% and 97.5% empirical quantiles for each q_x. As a result, you have measured how much a quantile q_x of `y` can fluctuate in 95% cases. Draw these confidence intervals on a separate plot for all three distributions. (**1p**)

   (c) In general, finding analytical confidence intervals for the qq-plot is intractable unless you compare a sample against known distribution with simple enough cumulative distribution function. For instance, if `y` of size n has uniform distribution and `z<-sort(y)` then z_k is

distributed according to beta distribution with parameters k and $n+1-k$. Hence, for uniform distribution we can provide analytical confidence intervals by solving the equations:

$$\Pr\left[z_k \leftarrow \mathcal{B}(k, n+1-k) : z_k \leq z_0\right] = 2.5\%$$
$$\Pr\left[z_k \leftarrow \mathcal{B}(k, n+1-k) : z_k \leq z_1\right] = 97.5\% \ .$$

For that you can use function `qbeta` with appropriate parameters. Draw the corresponding graph and compare with empirical results you obtained in the (b) part. How many points of qq-plot should lie outside confidence intervals if `y` has uniform distribution? (**1p**)

Tip: Note that `qq.plot` function in the `car` package can draw confidence intervals if you specify the theoretical distribution.

(d) Load the data from the file `unknown-data.Rdata` and report which of the samples are drawn from uniform distribution. Can you assign confidence measure to you conclusions, i.e., to specify your decision making procedure so that in 95% cases you would declare that the sample comes from the uniform distribution? (**1p**)

Hint: You can always measure the confidence of your decision procedure bay generating 1000 samples from uniform distribution and then counting how many do you accept.

2. The following exercise illustrates the basic mechanism behind Bayesian inference. For clarity, let us consider a coin with bias α:

$$\Pr\left[x_i = 1|\alpha\right] = \alpha \qquad \Pr\left[x_i = 0|\alpha\right] = 1-\alpha \ .$$

Assume that only eleven values are plausible: $\alpha \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$. Let corresponding probability vector be $p_{0.0}, \ldots, p_{1.0}$.

(a) Write a function `UpdateBeliefs` that takes in a priori probabilities and single observation and uses the Bayes formula to compute posterior probabilities $q_{0.0}, \ldots, q_{1.0}$. (**1p**)

(b) Write a function `UpdateBeliefs2` that takes in a priori probabilities and vector of observations and uses Bayes formula to compute posterior probabilities $q_{0.0}, \ldots, q_{1.0}$. (**1p**)

(c) Show that iterative belief updates with `UpdateBeliefs` give exactly the same result as a batch update with `UpdateBeliefs2`. For that try it on 10 different observation vectors of length 15 and on two different prior assignments (**1p**)

(d) Although orthodox Bayesianism excludes the question of correctness, we can still experiment whether our posterior distribution converges to true value given enough observations. Let the true value of α be 0.5. Consider a uniform prior and sceptical prior $p_{0.0} = 0.455, p_{1.0} =$

0.455 and $q_{0.1} = \ldots = q_{0.09} = 0.01$. Draw 1024 samples and compute posterior distributions after observing $4, 16, 64, 256, 1024$ samples. Interpret results. (**1p**)

3. There are many methods for detecting language of a text document. In this exercise, we derive a model-based approach for this task. That is, we first define a naive probabilistic model, which specifies how Estonian and English words are created. Then we estimate model parameters from the data and finally we use Bayes formula to determine the language.

   (a) Let us model the word creation in the language in the following way:

   - The first letter is chosen according to fixed frequency table T_{start}.
   - The following letters are chosen according to the previous letter. If the previous letter is 'a' then you choose it according to the frequency table T_a. For letter 'b' you use a different frequency table T_b and so on.

   Define a model. Formalise this informal description in terms of probabilities and conditional probabilities. Express the probability that a specific word is in the language. (**1p**)

   Remark: You get 0.5 points if instead of the analytical formula you define a function which generates data according to this model description.

   (b) Estimate all necessary probabilities $\Pr[\text{'a'}|\mathsf{Start}], \ldots, \Pr[\text{'a'}|\text{'a'}], \ldots$ by using the training sets for English and Estonian language given as file `word-samples.Rdata`. Do not try to be smart—compute probability as the ratio between good cases and all cases. Put these parameters into the formal model to compute probabilities

   $$\Pr[word|\mathsf{Estonian}] \qquad \Pr[word|\mathsf{English}]$$

   and then use Bayes formula

   $$\Pr[\mathsf{Estonian}|word] = \frac{\Pr[word|\mathsf{Estonian}]\Pr[\mathsf{Estonian}]}{\Pr[word]}$$

   to guess the language of a word on test samples `est.test.set` and `eng.test.set`. Why the procedure does not work? (**1p**)

   Hint: The number of samples is not the problem. You can assume that there are enough samples to estimate all parameters with high accuracy. The same problem could have manifested even if there would have been millions of word examples.

   (c) Let us make the original model more robust. Assume that on each time-point the text writer can deviate from the language model with probability α. When this happens, he will just choose the letter randomly. After that the writing procedure continues as before.

Express conditional probabilities $\Pr[\text{'a'}|\mathsf{Start}], \ldots, \Pr[\text{'a'}|\text{'a'}], \ldots$ of this new process in terms of old model. Use them to compute

$$\Pr[word|\mathsf{Estonian}] \qquad \Pr[word|\mathsf{English}]$$

and then use Bayes formula

$$\Pr[\mathsf{Estonian}|word] = \frac{\Pr[word|\mathsf{Estonian}]\Pr[\mathsf{Estonian}]}{\Pr[word]}$$

to guess the language of a word on test samples. Choose α based on your gut feeling and report how well the algorithm works. (**2p**)

4. Implement spam classifier based on Naive Bayes classifier. That is consider a model where the probability that an individual word occurs on the email depends only if it is spam email or not.

   (a) Assume that $x_1, \ldots, x_n$ are the counts of specific words. Use lecture material to define a model that assigns probabilities

   $$\Pr[x_1, \ldots, x_n|\boldsymbol{p}, \boldsymbol{q}, \mathsf{Spam}]$$
   $$\Pr[x_1, \ldots, x_n|\boldsymbol{p}, \boldsymbol{q}, \mathsf{Not\text{-}spam}]$$

   for each count vector $(x_1, \ldots, x_n)$. Keep in mind that words can now occur multiple of times. (**1p**)

   (b) Assume that prior probability of receiving spam and not spam letters are:

   $$\Pr[\mathsf{Spam}] = 90\% \qquad \Pr[\mathsf{Not\text{-}spam}] = 10\% \ .$$

   Derive a corresponding posterior estimates

   $$\Pr[\mathsf{Spam}|x_1, \ldots, x_n] \qquad \Pr[\mathsf{Not\text{-}spam}|x_1, \ldots, x_n]$$

   by using Bayes formula. (**0.5p**)

   (c) Use the Concept Drift Dataset 1 of ECUE Spam Dataset, which is available from `http://www.dit.ie/computing/staff/sjdelany/datasets/`, to estimate necessary conditional probabilities. The dataset represents individual emails as list of words and word count pairs mixed with other features. You can ignore the following features. Train the classifier on training set and evaluate its performance on test set. It is enough to train and test it on a single month (**2p**). However, you get an extra **0.5p** if you measure the performance degradation over the months and plot resulting graph with 95% confidence intervals.

   Hint: The methodology for computing confidence intervals of classification accuracy was thoroughly studied in the third homework.