# Examination

## Linköping University, Department of Computer and Information Science, Statistics

| | |
|---|---|
| Course code and name | 732A99/732A68 Machine Learning |
| Date and time | 2020-01-15, 14.00-19.00 |
| Assisting teacher | Oleg Sysoev |
| Allowed aids | See “732A99_TDDE01_exam_regulations.PDF” |
| | A=19-20 points plus passed oral defense |
| Grades: | B=16-18 points plus passed oral defense |
| | F= 16-20 points plus failed oral defense |
| | C= 16-20 points without oral defense |
| | C=11-15 points with or without oral defense |
| | D=9-10 points with or without oral defense |
| | E=7-8 points with or without oral defense |
| | F=0-6 points with or without oral defense |

**Provide a detailed report that includes plots, conclusions and interpretations. Give motivated answers to the questions. If an answer is not motivated, the points are reduced. Provide all necessary codes in the appendix.**

**Use seed 12345 when randomness is present unless specified otherwise.**

## Assignment 1 (10p)

The data file **default.csv** contains information about the credit card payments of clients in the bank as well as information about whether the customer is reliable (default_payment=0, i.e. No) or unreliable (default_payment=1, i.e. Yes). The following variables are also available for the analysis:

- Limit_bal: Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.
- Sex (0 = male; 1 = female).

- Education (0 = graduate school; 1 = university; 2 = high school; 3 = others).
- Marriage (0 = married; 1 = single; 2 = others).
- Age (year).

1. Use first 20 observations to fit Nearest Shrunken Centroid (NSC) model in which default_payment is used as target and the remaining variables are used as features, the threshold value is defined by cross-validation, and priors are specified as (0.2, 0.8). Report the optimal threshold value, the selected number of features and the cross-validation error. Interpret the centroid plot for the optimal model. Mention at least two important reasons of why this model is actually not appropriate for these data. Finally, report the fitted probabilistic model for threshold=0 (hint: use model$sd to get standard deviations per feature) **(4p)**
2. Divide the original data into training, validation and test (40/30/30) by using the standard codes provided in the lecture slides. Implement a function that uses training data, depends on parameter vector **w** and computes the minus log-likelihood of the logistic regression model in which default_payment is target variable, Age/100 and Gender are the features. Compare the minus log-likelihood values for $\boldsymbol{w} = (w_0, w_{age}, w_{sex})$ equal to
   a. (0,1,0)
   b. (0,0,1)
   c. (1,1,1)

   What can be stated by comparing these minus log-likelihood values? **(3p)**
3. Use function from step 2 and function optim() with initial values (1,1,1) to find the optimal values of **w**. Report equation of the decision boundary corresponding to the optimal model and the test misclassification error. Compare the training and test errors and make necessary conclusions **(3p)**
   a. **Remark:** if you don't manage to solve step 2, you may use function glm() to derive the optimal model here, but the points will be reduced

## Assignment 2 (10p)

MIXTURE MODELS – 5 POINTS

You are asked to modify the EM algorithm that you implemented for the lab assignments. Recall that in the lab assignment, you sampled the training data in three steps: First, you sampled one component and, then, you sampled one point from the component and, finally, you discarded the component label. So, your training data contained no component labels. This is an example of unsupervised learning, since the supervisor (component labels) is absent. On the other hand, classification is an example of supervised learning. In between these two extremes, we have semi-supervised learning, where the component label is known for some training points and unknown for the rest. You are asked to adapt your implementation of the EM algorithm to semi-supervised learning. To do so, first replace the lines that generated the training data in the lab with the following lines:

```
true_k <- array(dim = n) # true component labels

# Producing the training data
```

```
for(n in 1:N) {
  k <- sample(1:3,1,prob=true_pi)
  true_k[n] <- k * sample(0:1,1,prob = c(0.7,0.3))
  for(d in 1:D) {
    x[n,d] <- rbinom(1,1,true_mu[k,d])
  }
}
```

In other words, true_k[n]=0 if the component label is unknown for the n-th training data. Otherwise, true_k[n] contains the component label. The array true_k is part of the training data. So, you know the component labels for some of the training points (in the lines above, for around 30 % of the training points).

Your task is to modify your EM implementation to make use of the information in true_k during learning. Then, you should run your code when 0 %, 10 % and 30 % of the training points have a label. Compare the results obtained and explain why they differ.

## KERNEL METHODS – 5 POINTS

In the slides 11 and 12 of the lecture on kernel methods, you can see how to produce a probabilistic classifier by using kernel density estimation and Bayes theorem. You are asked to implement such a classifier. First, you have to produce the learning data by running the code below, which samples 1500 points from class 1 and 1000 points from class 2. These points are stored in the variables data_class1 and data_class2. Second, you have to use the Gaussian kernel and, thus, you have to select an appropriate kernel width h. Choose a value that you deem appropriate. Choose the value manually and disregard overfitting issues. Explain your choice. Finally, compute the posterior distribution of class 1 for the points in the interval [-5, 25], e.g. for the points in seq(-5, 25, 0.1). Visualize this distribution with a plot like the second plot in slide 12.

```
N_class1 <- 1500
N_class2 <- 1000
data_class1 <- NULL
for(i in 1:N_class1){
  a <- rbinom(n = 1, size = 1, prob = 0.3)
  b <- rnorm(n = 1, mean = 15, sd = 3) * a + (1-a) * rnorm(n = 1, mean = 4, sd = 2)
  data_class1 <- c(data_class1,b)
}
```

```
data_class2 <- NULL

for(i in 1:N_class2){

  a <- rbinom(n = 1, size = 1, prob = 0.4)

  b <- rnorm(n = 1, mean = 10, sd = 5) * a + (1-a) * rnorm(n = 1, mean = 15, sd = 2)

  data_class2 <- c(data_class2,b)

}
```