Examination

Linköping University, Department of Computer and Information Science, Statistics

Course code and name	TDDE01 Machine Learning
Date and time	2025-01-17, 8.00-13.00
Assisting teacher	Oleg Sysoev
Allowed aids	Contents of “help_materials” folder + your help file (if submitted to LISAM in due time)
Grades:	
	5=18-20 points
	4=14-17 points
	3=10-13 points
	U=0-9 points

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.

Note: seed 12345 should be used in all codes that assumes randomness unless stated otherwise!

To start work in RStudio, type this in the Terminal application:

```
module add courses/TDDE01
rstudio
```

To submit your report:

1. Create one file (allowed formats: DOC, DOCX, ODT, PDF)
2. Use Exam Client to submit, and choose Assignment 1 in the drop box
3. Attach your report
4. Submit.
5. “Request Received” status implies that your report is successfully submitted.

Assignment 1 (10p)

File **lakesurvey.csv** contains chemical measurements of different lakes in Sweden as well as their pH-level.

1. Perform principal component analysis excluding pH variable on the scaled original data. How much variation is explained by the first two principal components? Which features contribute to the first principal component mostly? By assuming that we keep only first two principal components, report an equation showing how the unscaled Cond variable can be approximated from the first two principal components. **(3p)**
2. Divide the original data into training, validation and test sets (40/30/30). Consider also a dataset with the three variables - first two principal components and pH variable – and split this dataset into training, validation and test by using same partitioning indices as in the original dataset. Scale the datasets appropriately. Estimate a) K-nearest neighbor models with pH as target and remaining variables as features by using original data partitioning and K=1, 10, 50 b) K-nearest neighbor models with pH as target and remaining variables as features by using the principal component data partitioning and K=1, 10, 50. Report training, validation and test MSE for these 6 models. Which K leads to the best model from the original data? Which K leads to the best model from the principal component dataset? What kind of data (original or principal component) results in the best prediction? Why might one need to check the test error of the best model as well? Why does K=1 result in zero training MSE? What target distribution do we implicitly assume by using MSE as a cost function? **(5p)**
3. Use the original data set and the principal component dataset from step 2, scale them, and compute the ratio between the Euclidian distance to the nearest observation from observation number 1 and Euclidian distance to the furthest observation from observation number 1, per dataset (exclude pH variable in the distance computations). Compare these ratios and comment what kind of machine learning phenomenon this comparison illustrates. **(2p)**

Assignment 2 (10p)

KERNEL MODELS – 6 Points

In the course, you have learned about kernel models for classification and regression. Kernel models can also be used for density estimation, i.e., to model a probability distribution or density function p(x_*). In particular,

$$p(x_*) = \frac{1}{n}\sum_{i=1}^{n} k(\frac{x_* - x_i}{h})$$

where the kernel function k() must integrate to 1. To ensure this, you will hereinafter consider k() to be the density function of a Gaussian distribution with mean equal to 0 and standard deviation equal to 1. You can get it by using the command `dnorm` in R.

(2 p) Run the code below to produce some learning data, which consist of 1000 samples from class 1 and 1000 samples from class 2. These points are stored in the variables data_class1 and data_class2.

Implement the kernel model presented above to estimate the density function of the data sampled from class 1. Do the same for class 2. Use only 800 samples from class 1 and 800 samples from class 2.

```
set.seed(123456789)

N_class1 <- 1000

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)

}
```

(3 p) Once you have obtained the kernel models for the class conditional density functions p(x_*|class=1) and p(x_*|class=2) in the previous question, you can use them to produce posterior class probabilities p(class|x_*) via Bayes theorem. Specifically,

p(class=1|x_*) = p(x_*|class=1) p(class=1) / [p(x_*|class=1) p(class=1) + p(x_*|class=2) p(class=2)]

Use these probabilities to compute the correct classification rate on 200 samples that you did not use before, 100 from class 1 and 100 from class 2. Use this classification rate to select the kernel width h from among the values 0.1, 0.2, …, 4.9, 5.

(1 p) Finally, use the 200 samples that you have not used so far to estimate the generalization error of the kernel model selected.

In summary, you should use 1600 samples to build kernel models of the class conditional density functions that you should convert into a probabilistic classifier via Bayes theorem. To select the kernel width, you

should use 200 samples as validation set. Finally, you should use 200 samples to estimate the generalization error of the model selected. **Comment your code.**

NEURAL NETWORKS - 4 Points

In this assignment, you are asked to use the R package `neuralnet` to train a NN to learn the trigonometric sine function. To produce the learning data, sample 50 points uniformly at random in the interval [0, 10] and, then, apply the sine function to each point.

(3 p) Your task is to estimate the generalization mean squared error of a NN with a single hidden layer of 10 units for the regression task described above. To this end, use cross-validation with 2 folds. Initialize the weights of the NN to random values in the interval [−1, 1]. Stop the training when the partial derivatives of the error function are below a threshold value of 0.001. Recall that cross-validation works as follows:

1. Divide the learning data into approximately equal sized folds D1 and D2
2. Train the regressor on D1 and test it on D2
3. Train the regressor on D2 and test it on D1
4. Report the average error on the test folds

(1 p) Finally, answer to the following question: What is the NN that you should return to the user? The one learned from D1? The one learned from D2? Either of them? None?

Help: Check the argument `threshold` in the documentation. Use the function `predict()` to compute the output of the trained NN for a given input vector. Use the default values for the arguments not mentioned here. Feel free to use the following template. **Comment your code.**

```
library(neuralnet)

set.seed(1234567890)

Var <- runif(50, 0, 10)

tr <- data.frame(Var, Sin=sin(Var))

tr1 <- tr[1:25,] # Fold 1

tr2 <- tr[26:50,] # Fold 2
```