EXAM IN STATISTICAL MACHINE LEARNING
STATISTISK MASKININLÄRNING

DATE: March 13, 2023

RESPONSIBLE TEACHER: Jens Sjölund

NUMBER OF PROBLEMS: 5

AIDING MATERIAL: Calculator, mathematical handbooks

PRELIMINARY GRADES:	grade 3	23 points
	grade 4	33 points
	grade 5	43 points

Some general instructions and information:

- Your solutions should be given in English.
- Only write on one page of the paper.
- Write your exam code and a page number on all pages.
- Do not use a red pen.
- Use separate sheets of paper for the different problems (i.e. the numbered problems, 1–5).

With the exception of Problem 1, all your answers must be clearly motivated! A correct answer without a proper motivation will score zero points!

Good luck!

Formula sheet for Statistical Machine Learning

Warning: This is not a complete list of formulas used in the course, some exam problems may require expressions not listed here. Furthermore, the formulas below are not self-explanatory, you need to be familiar with the expressions to interpret them.

The Gaussian distribution: The probability density function of the p-dimensional Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is

$$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{p/2}\sqrt{\det \boldsymbol{\Sigma}}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \qquad \mathbf{x} \in \mathbb{R}^p.$$

Sum of identically distributed variables: For identically distributed random variables $\{z_i\}_{i=1}^n$ with mean μ, variance σ^2 and average correlation between distinct variables ρ, it holds that $\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^n z_i\right] = \mu$ and $\mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^n z_i\right) = \frac{1-\rho}{n}\sigma^2 + \rho\sigma^2$.

Linear regression and regularization:

- The least-squares estimate of $\boldsymbol{\theta}$ in the linear regression model

$$y = \theta_0 + \sum_{j=1}^{p} \theta_j x_j + \epsilon$$

  is given by the solution $\widehat{\boldsymbol{\theta}}_{\text{LS}}$ to the normal equations $\mathbf{X}^{\mathsf{T}}\mathbf{X}\widehat{\boldsymbol{\theta}}_{\text{LS}} = \mathbf{X}^{\mathsf{T}}\mathbf{y}$, where

$$\mathbf{X} = \begin{bmatrix} 1 & -\mathbf{x}_1^{\mathsf{T}}- \\ 1 & -\mathbf{x}_2^{\mathsf{T}}- \\ \vdots & \vdots \\ 1 & -\mathbf{x}_n^{\mathsf{T}}- \end{bmatrix} \text{ and } \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \text{ from the training data } \mathcal{T} = \{\mathbf{x}_i, y_i\}_{i=1}^n$$

- Ridge regression uses the regularization term $\lambda\|\boldsymbol{\theta}\|_2^2 = \lambda\sum_{j=0}^p \theta_j^2$.
  The ridge regression estimate is $\widehat{\boldsymbol{\theta}}_{\text{RR}} = (\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{y}$.
- LASSO uses the regularization term $\lambda\|\boldsymbol{\theta}\|_1 = \lambda\sum_{j=0}^p |\theta_j|$.

Maximum likelihood: The maximum likelihood estimate is given by

$$\widehat{\boldsymbol{\theta}}_{\text{ML}} = \arg\max_{\boldsymbol{\theta}} \ln \ell(\boldsymbol{\theta})$$

where $\ln \ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ln p(y_i \,|\, \mathbf{x}_i; \boldsymbol{\theta})$ is the log-likelihood function (the last equality holds when the n training data points are modeled to be independent).

Logistic regression: The logistic regression combines linear regression with the logistic function to model the class probability

$$p(y = 1 \,|\, \mathbf{x}_i) = \frac{e^{\boldsymbol{\theta}^\mathsf{T} \mathbf{x}_i}}{1 + e^{\boldsymbol{\theta}^\mathsf{T} \mathbf{x}_i}}.$$

For multi-class logistic regression we use the *softmax* function and model

$$p(y = m \,|\, \mathbf{x}_i) = \frac{e^{\boldsymbol{\theta}_m^\mathsf{T} \mathbf{x}_i}}{\sum_{j=1}^{M} e^{\boldsymbol{\theta}_j^\mathsf{T} \mathbf{x}_i}}.$$

Discriminant Analysis: The linear discriminant analysis (LDA) classifier models $p(y \,|\, \mathbf{x})$ using Bayes' theorem and the following assumptions

$$p(y = m \,|\, \mathbf{x}) = \frac{p(\mathbf{x} \,|\, m) p(y = m)}{\sum_{j=1}^{M} p(\mathbf{x} \,|\, j) p(y = j)} = \frac{\mathcal{N}\left(\mathbf{x} \,|\, \widehat{\boldsymbol{\mu}}_m, \, \widehat{\boldsymbol{\Sigma}}\right) \widehat{\pi}_m}{\sum_{j=1}^{M} \mathcal{N}\left(\mathbf{x} \,|\, \widehat{\boldsymbol{\mu}}_j, \, \widehat{\boldsymbol{\Sigma}}\right) \widehat{\pi}_j},$$

where

$$\begin{aligned}
\widehat{\pi}_m &= n_m / n \text{ for } m = 1, \dots, M \\
\widehat{\boldsymbol{\mu}}_m &= \frac{1}{n_m} \sum_{i: y_i = m} \mathbf{x}_i \text{ for } m = 1, \dots, M \\
\widehat{\boldsymbol{\Sigma}} &= \frac{1}{n - M} \sum_{m=1}^{M} \sum_{i: y_i = m} (\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_m)(\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_m)^\mathsf{T}.
\end{aligned}$$

For quadratic discriminant analysis (QDA), the model is

$$p(y = m \,|\, \mathbf{x}) = \frac{\mathcal{N}\left(\mathbf{x} \,|\, \widehat{\boldsymbol{\mu}}_m, \, \widehat{\boldsymbol{\Sigma}}_m\right) \widehat{\pi}_m}{\sum_{j=1}^{M} \mathcal{N}\left(\mathbf{x} \,|\, \widehat{\boldsymbol{\mu}}_j, \, \widehat{\boldsymbol{\Sigma}}_j\right) \widehat{\pi}_j},$$

where $\widehat{\boldsymbol{\mu}}_m$ and $\widehat{\pi}_m$ are as for LDA, and

$$\widehat{\boldsymbol{\Sigma}}_m = \frac{1}{n_m - 1} \sum_{i: y_i = m} (\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_m)(\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_m)^\mathsf{T}.$$

Classification trees: The cost function for tree splitting is $\sum_{\ell=1}^{|T|} n_\ell Q_\ell$ where T is the tree, $|T|$ the number of terminal nodes, n_ℓ the number of training data points falling in node ℓ, and Q_ℓ the impurity of node ℓ. Three common impurity measures for splitting classification trees are:

$$\begin{aligned}
\text{Misclassification error:} \quad & Q_\ell = 1 - \max_m \widehat{\pi}_{\ell m} \\
\text{Gini index:} \quad & Q_\ell = \sum_{m=1}^{M} \widehat{\pi}_{\ell m}(1 - \widehat{\pi}_{\ell m}) \\
\text{Entropy/deviance:} \quad & Q_\ell = -\sum_{m=1}^{M} \widehat{\pi}_{\ell m} \log \widehat{\pi}_{\ell m}
\end{aligned}$$

where $\widehat{\pi}_{\ell m} = \frac{1}{n_\ell} \sum_{i\colon \mathbf{x}_i \in R_\ell} \mathbb{I}(y_i = m)$

Loss functions for classification: For a binary classifier expressed as $\widehat{y}(\mathbf{x}) = \operatorname{sign}\{C(\mathbf{x})\}$, for some real-valued function $C(\mathbf{x})$, the margin is defined as $y \cdot C(\mathbf{x})$ (note the convention $y \in \{-1, 1\}$ here). A few common loss functions expressed in terms of the margin, $L(y, C(\mathbf{x}))$ are,

$$\begin{aligned}
\text{Exponential loss:} \quad & L(y, c) = \exp(-yc). \\
\text{Hinge loss:} \quad & L(y, c) = \begin{cases} 1 - yc & \text{for } yc < 1, \\ 0 & \text{otherwise.} \end{cases} \\
\text{Binomial deviance:} \quad & L(y, c) = \log(1 + \exp(-yc)). \\
\text{Huber-like loss:} \quad & L(y, c) = \begin{cases} -yc & \text{for } yc < -1, \\ \frac{1}{4}(1 - yc)^2 & \text{for } -1 \leq yc \leq 0, \\ 0 & \text{otherwise.} \end{cases} \\
\text{Misclassification loss:} \quad & L(y, c) = \begin{cases} 1 & \text{for } yc < 0, \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

1. This problem is composed of 10 true-or-false statements. You only have to classify these as either `true` or `false`. For this problem *(only!)* no motivation is required. Each correct answer scores 1 point and each incorrect answer scores -1 point (capped at 0 for the whole problem). Answers left blank score 0 points.

   Hint: It is often better to only answer problems where you are confident. You do not need to answer all questions.

   i. Ridge regression has lower bias than unregularized linear regression.

   ii. The trees in a random forest classifier can be trained independently of each other.

   iii. A logistic regression model is trained by minimizing the squared error.

   iv. Training a neural network using stochastic gradient descent always produces the same result regardless of the initialization.

   v. Increasing the number of layers in a neural network always reduces the expected new data error.

   vi. After training, a parametric model no longer needs the training data to make predictions.

   vii. The irreducible error σ^2 is the difference between the training error E_{train} and the expected new data error E_{new}.

   viii. Cross-validation is primarily used to tune parameters such as the coefficients of a linear regression model.

   ix. In general, decision trees with multiple splits result in nonlinear decision boundaries.

   x. In QDA, the marginal distribution $p(\mathbf{x})$ of the inputs $\mathbf{x}$ is assumed to be a Gaussian mixture model. (10p)

k	$u_k^{(\mathrm{A})}$	$u_k^{(\mathrm{B})}$
1	59.8	1434
2	59.2	1449
3	58.4	1442
4	58.5	1413

Table 1: Stock prices u_k (in SEK) for companies A and B at subsequent dates indexed by k.

2. Let u_k denote the price of a stock at a date indexed with k. Suppose that you are trading in stocks and want to predict next day's price u_{k+1} based on the current price u_k.

   (a) Consider the model

   $$u_{k+1} = \theta u_k + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

   Based on the data for company A in Table 1, use ridge regression to estimate the parameter θ for each of the different regularization strengths $\lambda \in \{10, 100, 1000\}$. What happens in the limits $\lambda \to 0$ and $\lambda \to \infty$? (5p)

   (b) For the ridge regression model in question 2a (which is used to make predictions for company A), why would it not be appropriate to use the data from company B to determine the regularization strength? (1p)

   (c) Consider the alternative model

   $$\ln\left(\frac{u_{k+1}}{u_k^v}\right) = w + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, 0.007^2).$$

   Rewrite this expression such that the exponent v and intercept w can be estimated by solving a linear regression problem $\mathbf{y} \approx X\boldsymbol{\theta}$. State the corresponding output vector $\mathbf{y}$, design matrix X and parameter vector $\boldsymbol{\theta}$ when using the data for company A in Table 1. Note that you do *not* have to compute the resulting fit. (4p)

Note: Questions 2(a)-2(c) can be solved independently.

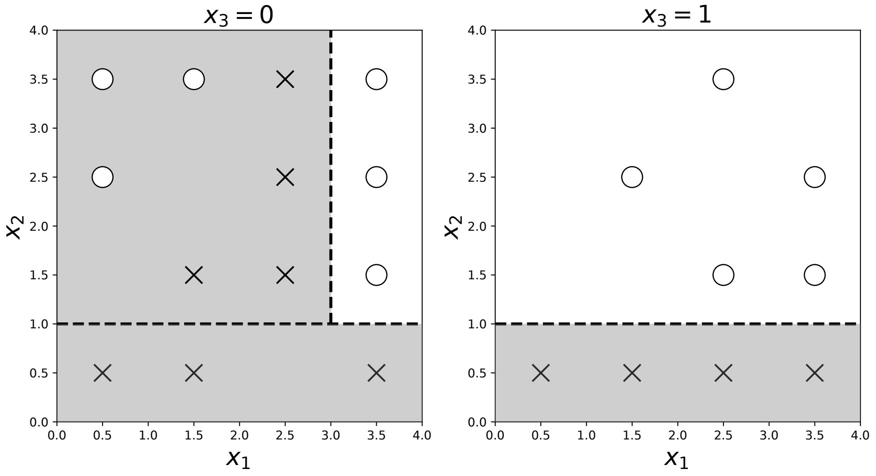


Figure 1: Data set with three-dimensional feature space and partition corresponding to an unknown decision tree.

3. Consider a classification problem with three inputs x_1, x_2 and x_3, and two class labels $\circ$ and $\times$. We consider the data set depicted in Fig. 1, which has already been partitioned into different regions according to an unknown decision tree. The features x_1 and x_2 of the data points take half-integer values and x_3 is either 0 or 1; e.g., the data point in the top left corner of the left subplot has the coordinates $(x_1 = 0.5, x_2 = 3.5, x_3 = 0)$.

   (a) Consider the partition in Fig. 1 and sketch the corresponding classification tree with minimum total number of splits. (2p)

   (b) Add one more split in the remaining mixed region (the upper left region in the left subplot) to the decision tree, utilizing the Gini index. Consider only the potential splits at $x_1 = 1$ and $x_1 = 2$. What can be an advantage of using the Gini index over the misclassification rate? (3p)

   (c) *Bagging* and *random forest* are both ensemble methods. Briefly describe their relationship and differences. Compared to a single flexible model (e.g., k-NN with a small value of k, or a tree that is grown deep), how does bagging improve the performance (from the perspectives of bias and variance)? (3p)

(d) Boosting is an ensemble method that can reduce bias in high-bias base models. When using decision trees as base models for boosting, how is the depth of the tree typically chosen? Explain why! Finally, argue in terms of bias and variance if there is a risk with having many base models. (2p)

Note: Question 3(a)-3(d) can be solved independently.

4. (a) Consider a classification problem with the input $\mathbf{x} \in \mathbb{R}^p$ and output $y \in \{1, \ldots, M\}$, where M is the total number of classes. To solve the classification problem using Linear Discriminant Analysis (LDA), we assume that $p(\mathbf{x} \,|\, y = m) = \mathcal{N}(\mathbf{x}|\mu_m, \Sigma)$. Let $p(y = m) = \pi_m$ and show that the classifier

$$\widehat{y} = \underset{m=\{1,\ldots,M\}}{\arg\max}\ p(y = m \,|\, \mathbf{x}),$$

where

$$p(y \,|\, \mathbf{x}) = \frac{p(\mathbf{x} \,|\, y)p(y)}{\sum_{m=1}^{M} p(\mathbf{x} \,|\, m)p(m)},$$

is equivalent to the classifier

$$\widehat{y} = \underset{m=\{1,\ldots,M\}}{\arg\max}\ \delta_m(\mathbf{x})$$

with score function:

$$\delta_m(\mathbf{x}) = \log \pi_m + \mathbf{x}^{\mathsf{T}}\Sigma^{-1}\mu_m - \frac{1}{2}\mu_m^{\mathsf{T}}\Sigma^{-1}\mu_m.$$

(3p)

(b) Given the score function in (a), derive the decision boundary between two classes a and b, and simplify the expression to show that the decision boundary is linear. (2p)

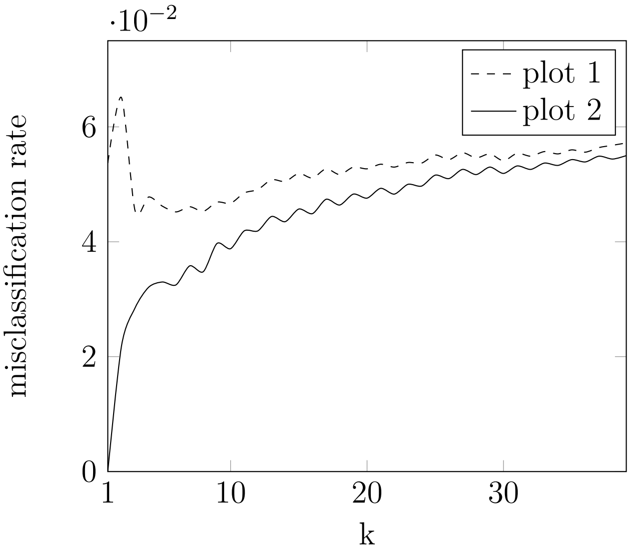
(c) You are given access to the following training data:

x_1	x_2	y
0.5	1	○
0	2	○
1.5	2	○
1	0	○
1.75	1.25	○
2	0	×
2.3	1.5	×
3	0	×
0	2.5	×

Construct a k-NN classifier and predict the output $\hat{y}_\star$ for an unseen test data point $x_\star = [2, 1.75]^\top$. Use the L1 distance as the metric and let $k = 3$. *Hint: The L1 distance between two vectors* $\mathbf{z}$ *and* $\mathbf{w}$ *of length* n *is defined as* $\sum_{i=1}^{n} |z_i - w_i|$. (2p)

(d) Suppose we want to train a k-NN classifier on a sufficiently large dataset. Provide a brief description of 10-fold cross-validation and explain how it can be used to systematically choose the appropriate value of k. (2p)

(e) After tuning the k-NN classifier using 10-fold cross-validation, you plot misclassification rates of the model with respect to different k-values on the train and validation set, respectively. This is shown in the figure below. However, the labels of your plots are missing! Determine which of the two plots that shows the misclassification rate on the training dataset for various k-values, and which plot shows the corresponding misclassification rates on the validation dataset. Motivate your answer. (1p)



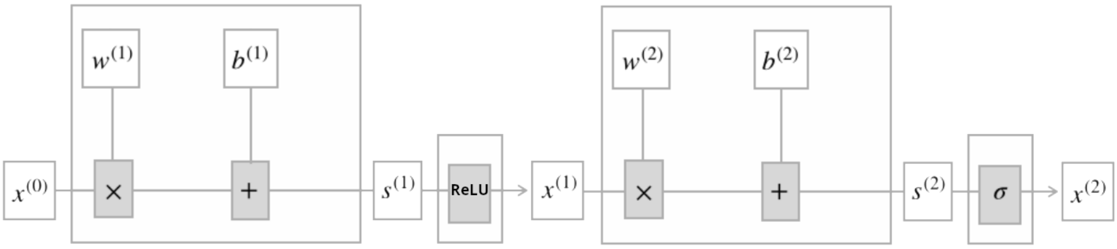


Figure 2: A neural network.

5. (a) Consider the neural network depicted in Figure 2 which has the following parameters:

$$w^{(1)} = \begin{pmatrix} 0.1 & 0.2 \\ 0.6 & -1.4 \\ -2.0 & 1.3 \end{pmatrix}, \qquad b^{(1)} = \begin{pmatrix} -0.3 \\ 0.1 \\ -1.05 \end{pmatrix},$$

$$w^{(2)} = \begin{pmatrix} 0.2 & 1.2 & -0.5 \\ -1.1 & 0.5 & 1.0 \end{pmatrix}, \qquad b^{(2)} = \begin{pmatrix} 0.2 \\ -0.1 \end{pmatrix}.$$

The first activation function is a ReLU, given by

$$\text{ReLU}(z) = \begin{cases} z, & \text{if } z \geq 0, \\ 0, & \text{if } z < 0, \end{cases}$$

and the second activation function is a sigmoid, given by

$$\sigma(z) = \frac{1}{1 + \exp(-z)}.$$

Note that the activation functions are applied elementwise. Let the input be $x^{(0)} = \begin{pmatrix} -1.0 \\ 0.5 \end{pmatrix}$.

i. Compute and write down the prediction $x^{(2)}$ as well as the intermediate results $s^{(1)}$, $x^{(1)}$, and $s^{(2)}$. Provide two digits after the decimal. (4p)

ii. How many layers does this neural network have? Based on the output, is this neural network used for regression or classification? (1p)

(b) Consider the data set $\mathcal{X} = \{6, 2, 4, 5, 1, 3\}$. We want to find a representative scalar $\theta \in \mathbb{R}$ which describes the data best. This would result in a model $f_\theta(x) = \theta$ to match our standard notation. As a loss function, we use the squared loss, i.e., $L(x, \theta) = (x - \theta)^2$. Note that we do not have a target/output y here. The objective function is given by $J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \theta)^2$ and the optimal estimate is $\theta^\star = \frac{1}{6}\sum_{i=1}^{6} x_i = 3.5$. However, we want to learn θ using an iterative optimization approach.

i. Learn the parameter $\theta \in \mathbb{R}$ using Stochastic Gradient Descent. Use a learning rate $\gamma = 0.3$ and an initial $\theta^{(0)} = 10$. Conduct two epochs and use a mini-batch size of three. After each update, write down the gradient value $\nabla J(\theta^{(t)})$ and the current parameter estimate $\theta^{(t)}$. Do not shuffle the data in each epoch! Provide two digits after the decimal. (4p)

Alternative of i. Only consider this sub-task if you have problems with the previous sub-task i. Alternatively[1] you can learn the parameter $\theta \in \mathbb{R}$ using Gradient Descent. Use a learning rate $\gamma = 0.3$ and an initial $\theta^{(0)} = 10$. Conduct six iterations. After each iteration, write down the gradient value $\nabla J(\theta^{(t)})$ and the current parameter estimate $\theta^{(t)}$. Do not shuffle the data! Provide two digits after the decimal. (2p)

ii. What are the problems of using a too large/small learning rate γ? (1p)

[1] Note that this alternative yields only half the points! We only consider one version of i., namely the one with the highest point score. Doing i. and the alternative version of i. will not give you more points in total!