Machine Learning
Prof. S. Harmeling

Wednesday, 13. October 2021
DUE 23:55h Tuesday, 19. October 2021

Exercise set #1

Solution should be submitted in teams of two. Please submit your solution online using the sciebo file-drop folder. The link is available in ILIAS. Please submit a single zip file with the following naming scheme: `username1-username2-ex#.zip` (e.g. `jadoe101-jodoe108-ex01.zip`). Allowed file extensions (of files within the zip file) are: `.pdf`, `.txt`, `.py` and `.ipynb`. Make sure the total file size does not exceed 10 MB. We will not accept photos of your solutions, scans produced by scanner apps are ok. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Monster vs. mouse**
   Recall the example: "While you tried to sleep, you hear some noise under the bed..." Following the lecture's notation,

$$\begin{aligned} n &= \text{some noise under your bed} \\ M &= \text{a monster under your bed} \\ m &= \text{a mouse under your bed} \\ e &= \text{something else (e.g. only air) under your bed} \end{aligned}$$

   express your beliefs about monsters, mice and noise by assigning numbers to $p(M)$, $p(m)$, $p(e)$, $p(n|M)$, $p(n|m)$, $p(n|e)$ and calculate the probability that given some noise under your bed, there is a monster, i.e. calculate $p(M|n)$ using Bayes' rule. Calculate the results with pencil and paper, use the `02-Monster-vs-mouse` notebook that you can find on sciebo to check your results.

   30 points

2. **Plausible inference**
   Without assumptions the plausibility of B is $p(B)$. Assuming that A is true, the plausibility of B is $p(B|A)$. Thus we can translate the phrase "if A is true, B becomes more plausible" into the inequality $p(B|A) \geq p(B)$. Assuming that inequality and using Bayes' rule and the basic laws of probability show the following:

   (a) $p(B|A) \geq p(B)$
   (b) $p(B|\neg A) \leq p(B)$
   (c) $p(A|B) \geq p(A)$
   (d) $p(A|\neg B) \leq p(A)$

   30 points

3. **Medical inference**
   Imagine you are an oncologist. Your current patient is a 45 year-old woman without symptoms, and with no history of cancer in her family, who came in for a routine checkup. Among patients with this background, 0.8% have breast cancer. But the routine mammogram of this patient is positive. Among women with this specific background who have breast cancer, 90% have a positive mammogram, while only 7% of these patients without breast cancer have positive mammograms.[1] Given these observations, what is the probability that your patient has breast cancer?

   20 points

[1]numbers based on K.-A. Phillips, G. Glendon, J.A. Knight; *Putting the risk of breast cancer into perspective.* New England J of Medicine **340** (1999), pp. 141–144

4. **Three prisoners problem**[2]

   Three prisoners, A, B and C, are in separate cells and sentenced to death. The governor has selected one of them at random to be pardoned. The warden knows which one is pardoned, but is not allowed to tell. Prisoner A begs the warden to let him know the identity of one of the others who is going to be executed. "If B is to be pardoned, give me C's name. If C is to be pardoned, give me B's name. And if I'm to be pardoned, flip a coin to decide whether to name B or C."

   The warden tells A that B is to be executed. Prisoner A is pleased because he believes that his probability of surviving has gone up from 1/3 to 1/2, as it is now between him and C. Prisoner A secretly tells C the news, who is also pleased, because he reasons that A still has a chance of 1/3 to be the pardoned one, but his chance has gone up to 2/3. What is the correct answer?

   20 points

[2] copied from `http://en.wikipedia.org/wiki/Three_Prisoners_problem`; think yourselves at least for five minutes before you look up the solution (if you don't want to find out yourself)!

Machine Learning Wednesday, 20. October 2021
Prof. S. Harmeling DUE 23:55 Tuesday, 26. October 2021

Exercise set #2

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Bayes net calculations**
Consider the following directed acyclic graphs (DAGs):

$$\begin{tikzcd} A \arrow[r] & B \arrow[r] & C \end{tikzcd}$$

$\mathcal{G}_1$

$$\begin{tikzcd} C \arrow[r] & D \\ A \arrow[u] \arrow[r] & B \arrow[u] \end{tikzcd}$$

$\mathcal{G}_2$

   (a) Let A, B and C be binary random variables. Calculate $p(B)$ and $p(C)$ for the Bayes net defined by $\mathcal{G}_1$ and the (conditional) probabilities $p(A) = 0.3$, $p(B|A) = 0.2$, $p(B|\neg A) = 0.4$, $p(C|B) = 0.7$ and $p(C|\neg B) = 0.5$.

   (b) Let A, B, C and D be binary random variables. Calculate $p(B)$ and $p(D)$ for the Bayes net defined by $\mathcal{G}_2$ and the (conditional) probabilities $p(A) = 0.3$, $p(B|A) = 0.2$, $p(B|\neg A) = 0.4$, $p(C|A) = 0.7$, $p(C|\neg A) = 0.6$, $p(D|B, C) = 0.9$, $p(D|B, \neg C) = 0.5$, $p(D|\neg B, C) = 0.3$ and $p(D|\neg B, \neg C) = 0.3$.

25 points

2. **The d-separation criterion**
Consider the following DAG $\mathcal{G}$:

$$\begin{tikzcd} & D \arrow[r] & E \arrow[r] & F \\ A \arrow[r] & B \arrow[u] \arrow[r] & C \arrow[u] & \end{tikzcd}$$

   (a) Write down the factorization of the joint distribution $p(A, B, C, D, E, F)$.

   (b) From your answer in (a) derive the marginal distribution $p(A, B, C, D, E)$.

   (c) Find a minimal set, that d-separates A and F (i.e. there is no other set with fewer elements, that also d-separates A and F). Prove that this is the case.

   (d) Prove that $C \perp\!\!\!\perp_{\mathcal{G}} D|B$ holds.

25 points

3. **Three node networks**
   Prove all four cases from the lecture, i.e.:

(a) $A \to B \to C$	implies	$A \perp\!\!\!\perp C \mid B$
(b) $A \leftarrow B \leftarrow C$	implies	$A \perp\!\!\!\perp C \mid B$
(c) $A \leftarrow B \to C$	implies	$A \perp\!\!\!\perp C \mid B$
(d) $A \to B \leftarrow C$	implies	$A \perp\!\!\!\perp C$

   For this write out the factorization of $p(A, B, C)$ that is implied by the graph and show the wanted conditional independence.

   20 points

4. **Properties of expected value and variance**
   Let X and Y be two discrete, not necessarily independent, random variables and $a \in \mathbb{R}$ a real number.

   (a) Show that the expectated value is an linear operator, i.e.,
   $$\mathrm{E}(aX + Y) = a\,\mathrm{E}(X) + \mathrm{E}(Y)$$

   (b) Show that
   $$\mathrm{Var}(aX) = a^2\,\mathrm{Var}(X)$$

   10 points

5. **Mode of the Gaussian distribution**
   The mode of a continuous probability distribution is the point at which the probability density function attains its maximum value. Prove that the mean μ is also the mode of the Gaussian normal distribution
   $$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right).$$

   Hint: Try maximizing $\log \mathcal{N}(x|\mu, \sigma^2)$ and argue why it has the same maximum.

   10 points

6. **Plotting distributions (programming task)**
   The goal of this task is to make yourself familiar with some continuous distributions and plotting pdfs and histograms. First, add the following import statements:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import beta, expon, gamma, laplace, norm
```

   For each of the following distributions, plot its pdf in the given interval and a histogram based on 1000 samples. Use `.pdf(x, *args)` to compute the probability densities for the given array of x values, e.g., `norm.pdf(x, a, b)`. Use `.rvs(*args, n)` to sample `n` values, e.g., `norm.rvs(a, b, n)`. For plotting you may either use `plotly` or `matblotlib`.

   (a) Normal distribution: $\mathcal{N}(5, 1.5)$. Plot interval: $[0, 10]$
   (b) Laplace distribution: $\mathrm{Laplace}(5, 1.5)$. Plot interval: $[-5, 15]$
   (c) Exponential distribution: $\mathrm{Exp}(0, 1/1.5)$. Plot interval: $[-1, 8]$
   (d) Gamma distribution: $\mathrm{Gamma}(5, 0, 1)$. Plot interval: $[0, 15]$
   (e) Beta distribution: $\mathrm{Beta}(2, 1.5, 0, 1)$. Plot interval: $[0, 1]$

   If you are interested, play around with the parameters of the distributions and see what influence they have on the probability densities.

   Important: Make sure you also submit the output produced by your Python code.

   10 points

Machine Learning — Wednesday, 27. October 2021
Prof. S. Harmeling — DUE 23:55 Tuesday, 02. November 2021

Exercise set #3

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **MLEs**
   Derive the maximum likelihood estimators for the following distributions. For this write down the log-likelihood function given n observations $x_1, \dots, x_n$ and determine the maximum with respect to the parameter.

   (a) Gaussian normal distribution $\mathcal{N}(x|\mu, \sigma^2)$ for μ given observations $x_1, \dots, x_n$.

   (b) Exponential distribution with probability density function $f(x|\lambda) = \lambda e^{-\lambda x}$ for $\lambda > 0$ given observations $x_i \geq 0$.

   (c) Gamma distribution with probability density function $g(x|\alpha, \lambda) = \frac{1}{\Gamma(\alpha)}\lambda^{\alpha} x^{(\alpha-1)} e^{-\lambda x}$ for parameter λ given observations $x_1, \dots, x_n \geq 0$ and a known $\alpha > 0$.

   Hint: The derivative of $\log(y)$ is $\frac{1}{y}$.

   45 points

2. **Dirichlet-multinomial model**
   Throwing a (not necessarily fair) K-sided die n times allows us to infer posteriors for the unknown probabilities. The data is $\mathcal{D} = (x_1, \dots, x_K)$ with x_j being the number of times you have seen side j. Assume a Dirichlet prior (with (hyper-)parameter vector α) for the parameter vector $\theta = (\theta_1, \dots, \theta_K)$ with $0 \leq \theta_j \leq 1$ and $\sum_j \theta_j = 1$ and a multinomial likelihood for your data, i.e.,

   $$p(\theta) = \mathrm{Dir}(\theta|\alpha) \qquad\qquad p(\mathcal{D}|\theta) = \mathrm{Mu}(x|n, \theta)$$

   Show that the posterior is also Dirichlet, i.e., show

   $$p(\theta|\mathcal{D}) = \mathrm{Dir}(\theta|\alpha + x)$$

   Hint: You do not have to calculate the normalization constant, i.e., prove that the posterior is proportional to a Dirichlet distribution with parameter $\alpha + x$.

   30 points

3. **Inference for a difference in proportions (programming task)**
   Recall the story from the lecture "Two sellers at Amazon have the same price. One has 90 positive and 10 negative reviews. The other one 2 positive and 0 negative. Who should you buy from?" Write down the posterior probabilities about the reliability (as in the lecture).

   (a) Calculate $p(\theta_1 > \theta_2|\mathcal{D}_1, \mathcal{D}_2)$ using quadrature, e.g., by using the function `dblquad` from `scipy.integrate`.

   (b) Calculate $p(\theta_1 > \theta_2|\mathcal{D}_1, \mathcal{D}_2)$ using Monte Carlo integration[1]. You can generate Beta distributed samples with the function `scipy.stats.beta.rvs(a,b,size)`.

   25 points

[1] `https://en.wikipedia.org/wiki/Monte_Carlo_integration`

Machine Learning — Wednesday, 03. November 2021
Prof. S. Harmeling — DUE 23:55 Tuesday, 09. November 2021

Exercise set #4

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Transformation of a variable: scaling and shifting**
   Assume x is a random variable with PDF $p_x(x)$. Define $y = ax + b$ being a scaled and shifted version of x for $a > 0$. Derive the PDF $p_y(y)$ using the transformation rule stated in Theorem 7.1 for the following two distributions:

   (a) $p_x(x) = \mathcal{U}(x|c, d)$, i.e., x is uniformly distributed on the interval $[c, d]$.

   (b) $p_x(x) = \mathcal{N}(x|\mu, \sigma^2)$, i.e., x is Gaussian distributed with mean μ and variance σ^2.

   25 points

2. **Bayesian linear regression**
   Assuming the model from the lecture

$$
\begin{aligned}
p(w) &= \mathcal{N}(w|w_0, V_0) \\
p(y|X, w) &= \mathcal{N}(y|Xw, \Sigma)
\end{aligned}
$$

   derive the mean and variance of the posterior $p(w|X, y) = \mathcal{N}(w|w_n, V_n)$. For this write out the exponent of the product of prior and likelihood and bring it to the form of a multivariate second-order polynomial. Then apply Lemma 6.4 (3) from lecture 6 to get expressions for η and Λ. Reordering gives you expressions for w_n and V_n.

   30 points

3. **Boston housing data (programming task)**
   Goal of this exercise is to predict the price of the houses in Boston. Load the dataset from the text file `boston.txt`[1] using the function `np.genfromtxt`. Use the first 100 rows for testing, the next 50 rows for validation, i.e., for tuning hyperparameters, and the rest of the dataset for fitting your linear model. You do not have to introduce any additional features. For each dataset report the test mean squared error.

   (a) Which column is the target? Which columns are the features?

   (b) Fit a linear model for predicting the price using the MLE estimate w_{MLE}.

   (c) Fit a linear model for predicting the price using the ridge regression estimate w_{RIDGE}. Find a good choice for the regularization strength using the validation dataset.

   15 points

[1] Adapted from `http://lib.stat.cmu.edu/datasets/boston`

4. **Polynomial linear regression (programming task)**
Goal of this task is to fit a polynomial through data points $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R} \times \mathbb{R}$. Assume that the outcome $y = (y_1, \ldots, y_n)^T$ follows a normal distribution $\mathcal{N}(y|Xw, \sigma^2 I)$, where

$$X = \begin{bmatrix} | & | & | & & | \\ 1 & x & x^2 & \ldots & x^d \\ | & | & | & & | \end{bmatrix}$$

(a) Write a function that generates the matrix X for $x = (x_1, \ldots, x_n)^T$.

(b) Implement the estimator w_{MLE}.

(c) Implement a function that calculates the error $\text{MSE}(w) = \frac{1}{n}(Xw - y)^T(Xw - y)$.

(d) Try to find a good polynomial degree $d < 20$ that leads to a small validation error, i.e., the error on the validation dataset. Plot your best solution together with the training data and compute the error on the test dataset.

(e) Plot the training and test errors against the degree of the polynomial. A paper-pencil plot on squared paper is fine. What do you observe?

(f) Implement the estimator w_{RIDGE}.

(g) Find a good combination of d and λ that gives you a small validation error. Is the test error smaller than the test error of the optimal solution from (d)?

If you have problems with this exercise, let us know! Training, validation and test data are available in sciebo.

30 points

Machine Learning
Prof. S. Harmeling

Wednesday, 10. November 2021
DUE 23:55 Tuesday, 16. November 2021

Exercise set #5

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Matrix differential calculus**
   The notation for this exercises follows the matrix differential calculus cheat sheet, which is available in the sciebo folder.

   (a) Let $x, b \in \mathbb{R}^m, \alpha \in \mathbb{R}$ and $A \in \mathbb{R}^{m\times m}$. Use matrix differential calculus to find the gradient of $\psi = x^T Ax + b^T x + \alpha$ with respect to x.

   (b) Let $x_i \in \mathbb{R}^m$ and $y_i \in \{0, 1\}$ for $1 \leq i \leq n$. The loss of a logistic regression problem can be written as $\tau = \sum_{i=1}^{n}(y_i - \sigma(x_i^T w + \beta))^2$, where $\sigma(\alpha) = \frac{1}{1+\exp(-\alpha)}$ is the sigmoid function. Find the derivative of τ with respect to $w \in \mathbb{R}^m$ and $\beta \in \mathbb{R}$ using the rules from matrix differential calculus.

   (c) Let $x, y \in \mathbb{R}^m, A, B \in \mathbb{R}^{m\times m}$ and $\sigma : \mathbb{R}^m \to \mathbb{R}^m$ the sigmoid function. Use matrix differential calculus to find the gradient of $\psi = ||y - A\sigma(Bx)||_2^2$ with respect to x.

   (d) Prove by induction that $\mathrm{d}(\phi^\alpha) = \alpha\phi^{\alpha-1}\mathrm{d}\phi$ for a real-valued function ϕ and $\alpha \in \mathbb{N}$.

   Hint: You can use without proof that $\mathrm{d}\sigma(\alpha) = \sigma(\alpha)(1 - \sigma(\alpha))\mathrm{d}\alpha$.

   40 points

2. **Bayesian linear regression (programming task)**
   Consider the following prior and likelihood:

$$p(w) = \mathcal{N}(w \,|\, \begin{bmatrix}0 & 0\end{bmatrix}^T, I)$$
$$p(y \,|\, X, w) = \mathcal{N}(y \,|\, Xw, I)$$

   Use this Python code to generate training data X and y:

```
mean = np.array([1., 2.])
cov = np.array([[1., 0.9], [0.9, 1.]])
x, y = np.random.multivariate_normal(mean, cov, 100).T
X = np.stack([np.ones_like(x), x], axis=1)
```

   Compute the posterior distribution over w. Then generate 100 samples from the posterior and plot all resulting linear models together with the training data.

   15 points

3. **Ordinary least squares**
   Derive the formula $w_{\text{ML}} = (X^T X)^{-1} X^T y$ from the lecture, by calculating the derivative of $p(y|X, w) = \mathcal{N}(y|Xw, \sigma^2 I)$ with respect to w, setting it to zero and solve it for w. You do not need to show that your solution is a maximum.

   30 points

4. **Cross-validation for ridge regression (programming task)**
   Implement a Python function

   ```
   ridgeCV(X, y, number_of_folds=5, lambda_grid=[0.001, 0.1, 1, 10, 100])
   ```

   that calculates the ridge regression estimate while performing cross-validation to determine the best regularization strength, where

   - `X` and `y` is the data that should be used for cross-validation,
   - `number_of_folds` is the number of splits used for cross-validation,
   - `lambda_grid` are the regularization strengths from which the best should be selected.

   The function should return a tuple `(w, best_lambda, best_score)`, where `w` are the fitted parameters corresponding to the best regularization strength `best_lambda` and `best_score` is the mean of the validation errors over the folds for the best regularization strength.

   Apply your method to the data from `boston.txt`. Use the first 100 rows for testing and the remaining rows for training with cross-validation.

   15 points

Machine Learning — Wednesday, 17. November 2021
Prof. S. Harmeling — DUE 23:55 Tuesday, 23. November 2021

Exercise set #6

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Equality constrained least squares problem**
Let $m \geq n \geq k$, $y \in \mathbb{R}^m, b \in \mathbb{R}^k$ and $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{k\times n}$ having full rank. Consider the following optimization problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \|Ax - y\|_2^2 \\ \text{s.t.} \quad & Bx = b. \end{aligned}$$

Find a matrix $P \in \mathbb{R}^{(n+k)\times(n+k)}$ and a vector $p \in \mathbb{R}^{n+k}$ such that solving

$$P \begin{bmatrix} x \\ \lambda \end{bmatrix} = p$$

gives a critical point for the optimization problem. You can assume that P is invertible.

Hint: Start with defining the Lagrangian function for this problem. After that calculate its gradient with respect to x and λ and set it to zero.

20 points

2. **Dual problem of ν-SVM for the non-separable case**
The ν-SVM is another realization of the SVM for the non-separable case. Instead of the parameter C it uses $\nu \in [0, 1]$ which is the proportion of support vectors that lie on the wrong side of the hyperplane. The *primal* problem of a ν-SVM is given as:

$$\begin{aligned} \min_{w,b,\xi,\rho} \quad & \frac{1}{2}||w||^2 - \nu\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq \rho - \xi_i \text{ for all } i \\ & \xi_i \geq 0 \text{ for all } i \\ & \rho \geq 0 \end{aligned}$$

Derive the dual problem by completing the following subtasks:

(a) Define the Lagrangian by introducing dual variables $\alpha_i \geq 0$, $\beta_i \geq 0$ and $\delta \geq 0$.

(b) Calculate the derivatives wrt w, b, ξ and ρ.

(c) Set the derivatives to zero and plug the resulting equations back into the Lagrangian to eliminate w, b, ξ and ρ.

(d) Do you get the constraints $\frac{1}{n} \geq \alpha_i \geq 0$ and $\sum_{i=1}^{n} \alpha_i \geq \nu$? Try to derive it, if it is missing!

(e) Write down the resulting maximization problem with constraints on α_i.

30 points

3. **Dropping the one**
   Consider the linearly separable case of a SVM, but replace in the constraints the 1 with 0, i.e.

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}||w||^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 0 \text{ for all } i \end{aligned}$$

   Consider a data set that is separable, what will happen to w and b? Will it still work? What might go wrong? What happens if you replace 0 by 0.5?

   10 points

4. **Implement your own SVM (programming task)**
   Implement your own linear SVM for the non-separable case. Please do not use any pre-built SVM implementations.

   (a) Start with writing down the quadratic programming formulation (see section 11 slide 31) for the primal problem. Describe how you choose $Q, c, A, b, A_{eq}, b_{eq}, x_l, x_u$.

   (b) Generate training data using the following code snippet:

```
import numpy as np
np.random.seed(123)
x1 = np.random.randn(2, 20)
x2 = np.random.randn(2, 20) + 2
y1, y2 = np.ones(20), -np.ones(20)
X_train = np.hstack([x1, x2]).T
y_train = np.hstack([y1, y2]).T
```

   (c) Solve the quadratic programming problem for the given training data. You can use the `quadprog` function from the lecture.

   (d) Visualize the training data and plot the decision boundary.

   40 points

Machine Learning
Prof. S. Harmeling

Wednesday, 24. November 2021
DUE 23:55 Tuesday, 30. November 2021

Exercise set #7

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Eigenvectors and eigenvalues**

   (a) Given a matrix $A \in \mathbb{R}^{n \times n}$ and one of its eigenvectors v, how do you obtain the corresponding eigenvalue λ?

   (b) Show that scaling an eigenvector v of matrix $A \in \mathbb{R}^{n \times n}$ by a constant $c \neq 0$ yields another eigenvector with the same eigenvalue λ.

   (c) For symmetric $A \in \mathbb{R}^{n \times n}$ with distinct eigenvalues $\lambda_1, \ldots, \lambda_n$ show that the corresponding eigenvectors $v_1, \ldots, v_n$ are orthogonal to each other.

   15 points

2. **Centering matrix**
   The Centering matrix is defined as $H = I_n - \frac{1}{n} 1_n 1_n^\top$ where I_n is the $n \times n$ identity matrix, 1_n is the n dimensional one-vector, i.e. $1_n = [1, 1, \ldots, 1]^\top$. Show:

   (a) H is symmetric, i.e. $H = H^\top$.

   (b) H is idempotent, i.e. $H = HH$.

   (c) Show that $H 1_n = 0_n$, i.e., 1_n is an eigenvector with eigenvalue 0.

   (d) What are the other eigenvalues of H?

   (e) For a data matrix $X \in \mathbb{R}^{d \times n}$ show that XH has mean zero, i.e. show $\frac{1}{n} X H 1_n = 0_d$.

   20 points

3. **Benchmarking the centering matrix (programming task)**
   The centering matrix is a very handy tool for theoretical derivations, however it is not a computational efficient way to center a data matrix as you will see by solving this exercise.

   (a) Write a Python function `center_with_matrix(data)` that subtracts the row-wise mean from the input array `data` by multiplying with the centering matrix.

   (b) Write a Python function `centering_with_numpy(data)` that performs the same operation using basic NumPy-functions.

   (c) Sample random data matrices with uniformly distributed entries with 10 rows and a different number of columns. Plot the number of columns of the data matrix against the elapsed runtime for both functions. Also add a legend to your plot.

   15 points

4. **PCA on iris data (programming task)**
   For this exercise we will use the iris data set which is available online[1]. The goal is to perform a principal component analysis (PCA) and derive with a new feature subspace. You may use NumPy/SciPy functions for the computations.

   (a) Write a Python function that calculates the covariance matrix

   $$\Sigma = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu})(x_i - \hat{\mu})^T$$

   of a dataset $X = [x_1, \dots, x_n] \in \mathbb{R}^{D \times n}$, where $\hat{\mu} \in \mathbb{R}^D$ is the mean of the data matrix. Do not use the function `np.cov`.

   (b) Implement a function `pca(X, d, whitening=False)` that performs PCA on the input data `X` and returns the projected (and optionally whitened) data `Y`, the matrix of eigenvectors `V`, and the eigenvalues `Lambda`. For the eigenvector decomposition you can use the function `np.linalg.eigh`.

   (c) Project the Iris data onto a two-dimensional feature space. Create scatter plots visualizing the projected data points before and after applying whitening.

   Important: This exercise assumes that the data points are stacked columnwise!

   30 points

5. **Inhomogeneous kernel function**
   For $a, b \in \mathbb{R}^2$, consider the inhomogeneous polynomial kernel function $k(a,b) = (a^\mathsf{T} b + 1)^p$.

   (a) Show that $\Phi(a) = [a_1^2, a_2^2, \sqrt{2}a_1a_2, \sqrt{2}a_1, \sqrt{2}a_2, 1]^T$ is a feature map for the kernel $k(a,b) = (a^\mathsf{T} b + 1)^2$, i. e., show that $k(a,b) = \Phi(a)^T\Phi(b)$.

   (b) What is the corresponding feature map for $k(a,b) = (a^\mathsf{T} b + 1)^3$? What is the dimensionality of that feature space?

   (c) What is the feature space dimension of $k(a,b) = (a^\mathsf{T} b + 1)^p$? Proof your answer.

   20 points

[1] `https://archive.ics.uci.edu/ml/datasets/iris`

Machine Learning — Wednesday, 01. December 2021

Prof. S. Harmeling — DUE 23:55 Tuesday, 07. December 2021

Exercise set #8

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Neighborhood graphs**
   Given a data set of seven data points $x_1 \dots x_7$, the following distance matrix has been calculated:

$$D = \begin{bmatrix} 0 & 5 & 7 & 8 & 7 & 6 & 7 \\ 5 & 0 & 4 & 6 & 6 & 10 & 9 \\ 7 & 4 & 0 & 2 & 6 & 8 & 5 \\ 8 & 6 & 2 & 0 & 4 & 6 & 7 \\ 7 & 6 & 6 & 4 & 0 & 3 & 6 \\ 6 & 10 & 8 & 6 & 3 & 0 & 4 \\ 7 & 9 & 5 & 7 & 6 & 4 & 0 \end{bmatrix}$$

   (a) Draw the ϵ-graph for $\epsilon = 5.5$.

   (b) Draw the k-nearest neighbor graph for $k = 2$.

   20 points

2. **ISOMAP (programming task)**
   We work on the Swiss Roll data set, which is a good example data set to study nonlinear dimensionality reduction techniques. The data set can be generated using the function `make_swiss_roll` from scikit-learn.

   (a) Generate a swiss roll data set of 800 points using the random seed 1234. Produce a 3D scatter plot of the swiss roll. Data points that are next to each other should have a similar color.

   (b) Perform a PCA of the swiss roll data. Show the scatter plot for the first two principal components and explain it briefly.

   (c) Implement ISOMAP only using basic Python/NumPy/SciPy functions. Use a k nearest neighbor graph and an ϵ graph to model the neighborhood relationship between the data points. Make sure that your weight matrix is symmetric.

   (d) Try to find a good value for k and ϵ. What might go wrong when you choose inappropriate values?

   (e) Reduce the number of dimensions of the swiss roll from three to two. Visualize your results using a colored scatter plot. Do you see any improvement over PCA?

   60 points

3. **Locally linear embedding**
   For embeddings $Y = [y_1, \dots, y_n] \in \mathbb{R}^{d \times n}$ and weights $W \in \mathbb{R}^{n \times n}$ show that

$$\sum_{i=1}^{n} ||y_i - \sum_{j=1}^{n} W_{ij} y_j||_2^2 = \mathrm{tr}(YMY^T),$$

   where

$$M = I_n - W - W^T + W^T W$$

   and I_n being the $n \times n$ identity matrix.

   20 points

Machine Learning
Prof. S. Harmeling

Wednesday, 08. December 2021
DUE 23:55 Tuesday, 14. December 2021

Exercise set #9

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Convergence of K-means**
   Show that K-means converges by considering the distortion measure,

$$J = \sum_n \sum_i z_n^i ||x_n - \mu_i||^2.$$

   (a) Show that for fixed μ_i the E-step, that reassigns z_n^i,

$$z_n^i = \begin{cases} 1 & \text{if } i = \arg\min_j ||x_n - \mu_j||^2 \\ 0 & \text{otherwise} \end{cases}$$

   minimizes J.

   (b) Show that for fixed assignments z_n^i, the M-step,

$$\mu_i = \frac{\sum_n z_n^i x_n}{\sum_n z_n^i}$$

   minimizes J.

   (c) Conclude that K-means converges.

   You can assume that $\arg\min_j ||x_n - \mu_j||^2$ always returns only a single index.

   30 points

2. **Implementation of K-means (programming task)**
   For this task we will again work with the iris data set. This time we aim for clustering the different iris species. For simplicity, we only work with two features, so please consider only the petal length and width.

   (a) Implement K-means in Python, see slide 14 of `ml-section14-gmm-em.pdf`.

   (b) Run your implementation using $K = 3$ means. Generate a figure similar to slide 15 of `ml-section14-gmm-em.pdf`. It does not have to be exactly the same style, but should show some EM iterations. A series of plots is also ok. You also don't have to plot the separating lines. It is fine if the points in the various clusters have different colors. The cluster centers could be e.g. plus-sign, or for simplicity, yet another color is also fine.

   25 points

3. **EM update for the weights of a Gaussian mixture model**
   The update formula for the weights

$$\pi_i = \frac{1}{N} \sum_n \tau_n^i$$

can be derived by considering the Lagrangian

$$\mathcal{L}(\theta, \lambda) = l(\theta|\mathcal{D}) + \lambda \left(\sum_k \pi_k - 1 \right).$$

   (a) Start by calculating the derivatives $\frac{\partial \mathcal{L}}{\partial \pi_i}$ and $\frac{\partial \mathcal{L}}{\partial \lambda}$.
   (b) What constraint on $\pi = (\pi_1, \ldots, \pi_K)^T$ do you get by setting $\frac{\partial \mathcal{L}}{\partial \lambda} = 0$?
   (c) Show that $\lambda = -N$ by setting $\frac{\partial \mathcal{L}}{\partial \pi_i} = 0$, multiplying with π_i on both sides and using the constraint from the previous part of the exercise.
   (d) Finally, plug $\lambda = -N$ into $\frac{\partial \mathcal{L}}{\partial \pi_i} = 0$ to obtain the update formula for π_i.

   Hint: The calculation of the derivative is very similar to the derivations done on slide 21 of `ml-section14-gmm-em.pdf`.

20 points

4. **EM algorithm for Gaussian mixture models (programming task)**
   In this exercise you will implement the EM algorithm for Gaussian mixture models. Consider the following Python code that generates training data $X \in \mathbb{R}^N$:

```
np.random.seed(42)
X = np.concatenate([np.random.normal(-1.5, 1.2, 50),
                    np.random.normal(3, 0.7, 100),
                    np.random.normal(6, 1, 150)])
```

   A Gaussian mixture model has a pdf of the form

$$p(x \mid \theta) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x \mid \mu_i, \sigma_i^2)$$

   with parameters $\theta = (\pi_1, \ldots, \pi_K, \mu_1, \ldots, \mu_K, \sigma_1^2, \ldots, \sigma_K^2)$.

   (a) Implement the E-step of the EM algorithm calculating the soft assignments for each data point

$$\tau_n^i = \frac{\pi_i \mathcal{N}(x_n|\mu_i, \sigma_i^2)}{\sum_j \pi_j \mathcal{N}(x_n|\mu_j, \sigma_j^2)}.$$

   (b) Implement the M-step of the EM algorithm recomputing the parameters

$$\mu_i = \frac{\sum_n \tau_n^i x_n}{\sum_n \tau_n^i},$$

$$\sigma_i^2 = \frac{\sum_n \tau_n^i (x_n - \mu_i)^2}{\sum_n \tau_n^i},$$

$$\pi_i = \frac{1}{N} \sum_n \tau_n^i.$$

   (c) Apply the algorithm on the training data using $K = 3$ clusters. Report the final means μ_i, variances σ_i^2, and mixing components π_i and plot the resulting pdf of the mixture model.

25 points

Machine Learning
Prof. S. Harmeling

Wednesday, 15. December 2021
DUE 23:55 Tuesday, 21. December 2021

Exercise set #10

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Linear multi-layer networks**
   The single-layer network $y(x) = b + Wx$ can only represent linear functions.
   (a) Is the multi-layer version
   $$y(x) = b_2 + W_2(b_1 + W_1 x)$$
   more expressive? Yes or no, can you prove it?
   (b) Is it more expressive if we add a component-wise non-linearity g between the layers,
   $$y(x) = b_2 + W_2\, g(b_1 + W_1 x)$$
   e.g., with $g(a) = \tanh(a)$? If yes, why? If not, why not?

20 points

2. **Nonlinear activation functions**
   (a) Show that the derivatives of the sigmoid activation function
   $$\sigma : \mathbb{R} \to \mathbb{R}, \quad \sigma(a) = \frac{1}{1 + e^{-a}}$$
   is given by
   $$\frac{\partial \sigma(a)}{\partial a} = \sigma(a)(1 - \sigma(a)).$$
   (b) Show that the Rectified linear unit (ReLU)
   $$g : \mathbb{R} \to \mathbb{R}, \quad g(a) = \max(a, 0)$$
   is not continuously differentiable at $a = 0$. What is its derivative for $a \in \mathbb{R}_{\neq 0}$?
   (c) Calculate the derivative of the exponential linear unit (ELU)
   $$h : \mathbb{R} \to \mathbb{R}, \quad h(a) = \begin{cases} a & a \geq 0 \\ e^a - 1 & a < 0 \end{cases}$$
   and show that it is continuously differentiable in $a = 0$.
   (d) The tanh activation function is defined as
   $$\tanh(a) : \mathbb{R} \to \mathbb{R}, \quad \tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}.$$
   Show that
   $$\tanh(a) = 2\sigma(2a) - 1.$$

20 points

3. **Weight space symmetries**
Consider the two-layer neural network:

$$f(x, W_1, b_1, W_2, b_2) = b_2 + W_2 \tanh(b_1 + W_1 x)$$

(a) Show $\tanh(-a) = -\tanh(a)$.

(b) Show $f(x, W_1, b_1, W_2, b_2) = f(x, -W_1, -b_1, -W_2, b_2)$

What other symmetries are there?

(c) Consider a permutation matrix P, i.e. PA permutes the rows of matrix A and Pv permutes the entries of vector v. Suppose you permute the rows of W_1, how do you have to change the other parameters, such that the output of f for a fixed x stays the same? Note that $PP^\top = P^\top P = I$.

(d) Consider a random sign changing matrix S which has ones and minus ones along the diagonal (and zeros everywhere else). What symmetries does this operation create? I.e. how do you have to change the other parameters if you replace W_1 by SW_1.

(e) Given parameters W_1, b_1, W_2, b_2, how many parameter settings are there that generate the same neural network output?

20 points

4. **Training a linear model on toy data (programming task)**
For this exercise you should implement a classification model using NumPy.

(a) Use this Python code to generate data $X \in \mathbb{R}^{4N \times 2}$ and $y \in \{-1, +1\}^{4N}$:

```
X = np.repeat(np.array([[0,0], [0,1], [1,0], [1,1]]), N, axis=0)
X = X + np.random.randn(4*N, 2)*0.2
y = np.repeat([1,-1,-1,1], N)
```

(b) Implement the model

$$f_{w,b}(x_i) = w^T \phi(x_i) + b$$

for inputs $x_i \in \mathbb{R}^2$, some feature mapping $\phi : \mathbb{R}^2 \mapsto \mathbb{R}^d$ and parameters $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$. Take a look at the generated data and choose ϕ (and d) accordingly.

(c) Define a function that computes the hinge loss

$$\ell(w, b) = \frac{1}{N} \sum_{i=1}^{N} \max(0, 1 - f_{w,b}(x_i) y_i)$$

where $y_i \in \{-1, +1\}$ denotes the label of datapoint x_i.

(d) Write a function that computes the derivative of the hinge loss with respect to w and b. Generate a training dataset with $N = 100$ and a test dataset with $N = 50$.

(e) Use gradient descent with a learning rate of 0.5 to train your model by minimizing the hinge loss for 100 steps.

(f) Compute the accuracy of your trained model on the test set. Let the sign of your model's output be the predicted class label. You should achieve a test accuracy of at least 90%.

40 points

Machine Learning — Wednesday, 12. January 2022
Prof. S. Harmeling — DUE 23:55 Tuesday, 18. January 2022

Exercise set #11

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Sampling a GP prior (programming task)**
   A Gaussian process is defined by a mean function $m : \mathbb{R} \to \mathbb{R}$ and a covariance (kernel) function $k : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. For all combinations of the mean functions $m_1(x) = 0$ and $m_2(x) = 0.1x^2$ and the covariance functions listed below draw three functions from the corresponding GP and plot them over the interval $[0, 10]$.

   (a) Linear kernel
   $$k_1(x, x') = x^T x'.$$

   (b) Squared-exponential kernel function
   $$k_2(x, x') = \sigma^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right),$$
   with band-width parameter $\ell > 0$ and covariance $\sigma > 0$.

   (c) Periodic kernel function
   $$k_3(x, x') = \sigma^2 \exp\left(-\frac{2}{\ell^2}\sin^2\left(\pi\frac{|x - x'|}{p}\right)\right),$$
   with period-length p, band-width parameter $\ell > 0$ and covariance $\sigma > 0$.

   Remark: Implement the kernel functions using Numpy, SciPy or Python functions. You are free to choose appropriate values for the hyper-parameters.

   25 points

2. **GP regression I**
   Consider the following covariance (kernel) function
   $$k(x, x') = 30x^T x' + 1 \text{ for } x, x' \in \mathbb{R}.$$

   (a) Prove that $k(x, x')$ is a positive semidefinite kernel function.

   (b) Consider Gaussian process regression using
   $$\begin{aligned} f &\sim \mathrm{GP}(m, k) && \text{prior} \\ y_*|x_*, f &\sim \mathcal{N}(f(x_*), \sigma_n^2) && \text{likelihood} \end{aligned}$$
   with zero mean function $m(x) = 0$, covariance function $k(x, x')$, and noise level σ_n^2. Now, given training data $X = \begin{bmatrix}-1 & 1\end{bmatrix}, y = \begin{bmatrix}1 & 1\end{bmatrix}^T$, test data $x_* = 0$, and noise level $\sigma_n^2 = 1$, calculate (by hand) the posterior predictive mean μ_* and variance σ_*^2. Show your steps!

   20 points

3. **GP regression II (programming task)**
This exercise teaches you how to apply Gaussian process regression to real-world data. The book is open-access and can be downloaded from the URL stated below[1].

   (a) Implement Algorithm 2.1 from [1] and the squared exponential covariance function (Equation 4.9 from [1]).

   (b) Now given the following training data

$$X^\mathsf{T} = \begin{bmatrix} 0.5 & 1.1 & 1.7 & 2.3 & 2.9 & 3.5 & 4.1 & 4.7 \end{bmatrix}$$
$$y^\mathsf{T} = \begin{bmatrix} 2.3 & 1.9 & 3.3 & 5.5 & 3.1 & 9.1 & 7.8 & 7.8 \end{bmatrix}$$

   and test data $x_\star = 3.4$, please use your implementation to calculate the predicted mean $\mu_\star$ and predicted variance $\sigma_\star^2$. Use a bandwidth of $\ell = 1.5$ for the given covariance function and a noise level of $\sigma_n = 0.2$.

   (c) Now consider 100 testpoints between 0.5 and 5. Calculate their predicted mean and variance. Plot the training data (points) and the predicted mean (continuous line) and the 95% confidence bands[2] (shaded area around mean) in one figure.

   (d) Modify the paramters l and σ_n. Write down a parameter setting that results in underfitting and also a parameter setting resulting in overfitting. Produce plots for these two scenarios like you did in (b).

55 points

References

[1] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning).* The MIT Press, 2005.

[1] `http://www.gaussianprocess.org/gpml/chapters/`

[2] The 95% confidence bands for a normal distributed prediction can be obtained by adding $\pm 2\sigma_\star$ to the predicted mean.

Machine Learning
Prof. S. Harmeling

Wednesday, 19. January 2022
DUE 23:55 Tuesday, 25. January 2022

Exercise set #12

Please submit your solutions in teams of two using the sciebo file-drop folder. The link is available in ILIAS. For the formatting please stick to the `submission_guideline.pdf` that you can find on sciebo. In the case of multiple uploads we will consider the latest. Uploads after the deadline will be deleted without further notice.

1. **Laplace approximation (from MacKay [1])**

   A photon counter is pointed at a remote star for one minute, in order to infer the rate of photons arriving at the counter per minute λ. Assuming the number of photons collected r has a Poisson distribution

$$p(r|\lambda) = \exp(-\lambda)\frac{\lambda^r}{r!}, \text{ with } \lambda > 0$$

   and assuming an improper[1] prior

$$p(\lambda) = \frac{1}{\lambda}$$

   perform a Laplace approximation for the posterior of λ.

   30 points

2. **Gaussian processes with probit likelihood (from Rasmussen and Williams [2])**
   For a binary Gaussian process classification model, show the equivalence of using

   (i) a noise-free latent process combined with a probit likelihood, i.e., choosing

$$p(y_i = 1|f_i) = \sigma_{\text{probit}}(f_i) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{f_i} \exp\left(-\frac{t^2}{2}\right) \mathrm{d}t$$

   and

   (ii) a latent process with Gaussian noise combined with a step-function likelihood. This can be expressed by introducing additional noisy latent variables $\tilde{f}_i$, which differ from f_i by Gaussian noise, and defining $p(y_i = 1|\tilde{f}_i)$ as follows:

$$p(\tilde{f}_i|f_i) = \mathcal{N}(\tilde{f}_i|f_i, \sigma^2) \qquad\qquad p(y_i = 1|\tilde{f}_i) = \begin{cases} 1 & \text{if } \tilde{f}_i \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

   Hint: Start with the expressions in (i) and integrate out $\tilde{f}_i$ to get an expression for $p(y_i = 1|f_i)$, which should look like σ_{probit}. What do you have to plug in for σ^2 in (ii) to exactly match (i)?

   30 points

[1]An improper prior is a prior that cannot be normalized to a probability distribution.

3. **Kernel design**
   Recall from the lecture that a *positive semidefinite kernel* is a function $k : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$, such that, for every set $\boldsymbol{x} = \{x_i\}_{i=1,\ldots,N}$ with $x_i \in \mathbb{X}\ \forall i$, the matrix $k_{\boldsymbol{xx}}$ with elements $k_{\boldsymbol{xx}(ij)} = k(x_i, x_j)$ is positive semidefinite. You can use the facts, that if $k_1(x, x')$ and $k_2(x, x')$ are both positive semidefinite kernels and $\alpha \in \mathbb{R}_{\setminus 0}$, then the functions $k(x, x') = \alpha^2 k_1(x, x')$, $k(x, x') = k_1(x, x') + k_2(x, x')$ and $k(x, x') = k_1(x, x') \cdot k_2(x, x')$ are also positive semidefinite kernels. For each of the following functions, determine whether or not they are kernels. Clearly state your assumptions and your derivation.

   (a) $k(x_1, x_2) = C$ for $C \in \mathbb{R}_{>0}$

   (b) $k(x_1, x_2) = x_1 x_2$ for $\mathbb{X} = \mathbb{R}$

   (c) $k(x_1, x_2) = x_1 + x_2$ for $\mathbb{X} = \mathbb{R}$

   (d) $k(x_1, x_2) = 5x_1^\intercal x_2$ for $\mathbb{X} = \mathbb{R}^D$

   (e) $k(x_1, x_2) = (x_1^\intercal x_2 + 1)^2$ for $\mathbb{X} = \mathbb{R}^N$

40 points

References

[1] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms.* Cambridge university press, 2003.

[2] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning).* The MIT Press, 2005.