Data Analytics and Machine Learning (I26)
Department of Informatics
Technical University of Munich

Ecorrection
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Maschinelles Lernen

Exam:	IN2064 / Endterm	**Date:**	Friday 24th February, 2023
Examiner:	Prof. Günnemann	**Time:**	17:00 – 19:00

	P 1	P 2	P 3	P 4	P 5	P 6	P 7	P 8	P 9
I									

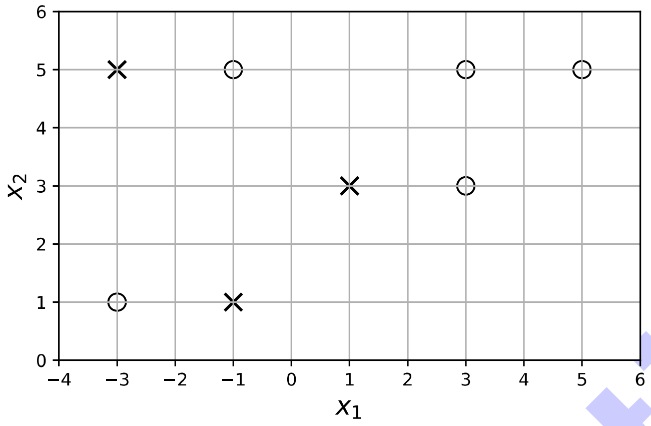
Working instructions

- This exam consists of **16 pages** with a total of **9 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 36 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources:
  - Two-sided DIN A4 sheet of handwritten notes (a print of digitally handwritten notes is allowed).
- **No other material (e.g. books, cell phones, calculators) is allowed!**
- Physically turn off all electronic devices, put them into your bag and close the bag.
- There is scratch paper at the end of the exam (after problem 9).
- Write your answers only in the provided solution boxes or the scratch paper.
- If you solve a task on the scratch paper, clearly reference it in the main solution box.
- All sheets (including scratch paper) have to be returned at the end.
- **Only use a black or a blue pen (no pencils, red or greens pens!)**
- **For problems that say “Justify your answer” you only get points if you provide a valid explanation.**
- **For problems that say “Derive” you only get points if you provide a valid mathematical derivation.**
- **For problems that say “Prove” you only get points if you provide a valid mathematical proof.**
- If a problem does not say “Justify your answer”, “Derive” or “Prove”, it is sufficient to only provide the correct answer.

Left room from ____________ to ____________ / Early submission at ____________

Problem 1 Decision trees (4 credits)

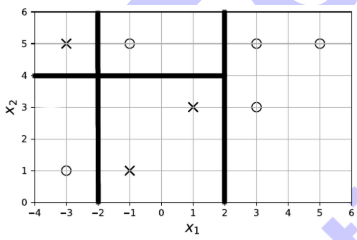
Consider the following two-dimensional classification dataset with the classes “0” (○ marker) and 1 (x marker).



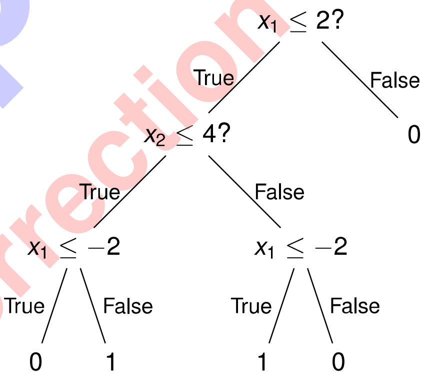
0 □ 1 □ 2 □

a) Draw a decision tree of maximum depth 3 that correctly classifies all datapoints. Each decision node must be of the form $x_d \leq c$ with $d \in \{1, 2\}$ and $c \in \mathbb{R}$. Also annotate each edge with “True” or “False” and each leaf node with “0” or “1”

The feature space can be partitioned as follows:



This corresponds to the following decision tree:



0 points if:

- Lines corresponding to nodes do not separate the input space into regions with the same class,
- or any decision node is not of the form $x_{1/2} \leq c$, $x_{1/2} < c$, $x_{1/2} \geq c$, $c \leq x_{1/2}$ etc., i.e. comparing a single variable to a single constant,
- or the tree has depth > 3.

1 point if:

- Drawing the lines separates the input space into rectangles containing only samples from the same class

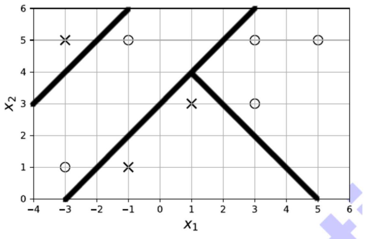
2 points if:

- The decision tree classifies all data correctly, all edge and leaf node annotations are correct and all decision nodes have the correct form.

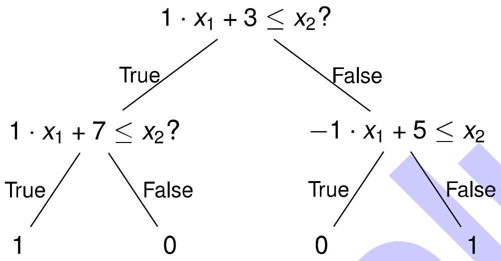
b) We now consider a modified form of decision trees that **only** allows for decision nodes of the form $a \cdot x_1 + b \leq x_2$ with $a, b \in \mathbb{R}$. In particular, nodes of the form $x_1 \leq c$ are **not allowed**.
Draw such a decision tree of maximum depth 2 that correctly classifies all datapoints. Also annotate each edge with “True” or “False” and each leaf node with “0” or “1”

□ 0
□
□ 1
□
□ 2

The feature space can be partitioned as follows:



This corresponds to the following decision tree:



$1 \cdot x_1 + 3 \leq x_2$?
True
False
$1 \cdot x_1 + 7 \leq x_2$?
$-1 \cdot x_1 + 5 \leq x_2$
True
False
True
False
1
0
0
1

0 points if:

- Lines corresponding to nodes do not separate the input space into regions with the same class,
- or any decision node is not of the form $f(x_1) \leq g(x_2)$, $f(x_1) > g(x_2)$, $g(x_2) \geq f(x_1)$ etc. with linear f, g
- or the tree has depth > 2.

1 point if:

- Drawing the lines separates the input space into polytopes containing only samples from the same class

2 points if:

- The decision tree is correct (including all edge and leaf node annotations) and all decision nodes have the form specified in the problem statement ($a \cdot x_1 + b \leq x_2$).

Problem 2 Probabilistic inference (3 credits)

Consider an infinite number of barns arranged on a regular grid $\mathbb{Z}^2$, with $\mathbb{Z}$ being the set of all integers. An owl starts exploring the world at an unknown location $\mathbf{x}^{(0)} \in \mathbb{Z}^2$. Each day, it moves in one of four directions, according to the following distribution:

$$\Pr\left[\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \begin{bmatrix}-1\\0\end{bmatrix} \mid \mathbf{x}^{(t)}\right] = \frac{2}{8} \qquad \Pr\left[\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \begin{bmatrix}+1\\0\end{bmatrix} \mid \mathbf{x}^{(t)}\right] = \frac{2}{8}$$

$$\Pr\left[\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \begin{bmatrix}0\\-1\end{bmatrix} \mid \mathbf{x}^{(t)}\right] = \frac{3}{8} \qquad \Pr\left[\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \begin{bmatrix}0\\+1\end{bmatrix} \mid \mathbf{x}^{(t)}\right] = \frac{1}{8}$$

0 □
1 □

a) After two days, you find the owl sleeping in $\mathbf{x}^{(2)} = \begin{bmatrix}6 & 8\end{bmatrix}^\top$. List all possible starting locations, i.e. all $\mathbf{s} \in \mathbb{Z}^2$ such that $\Pr\left[\mathbf{x}^{(2)} = \begin{bmatrix}6 & 8\end{bmatrix}^\top \mid \mathbf{x}^{(0)} = \mathbf{s}\right] > 0$.

$(6,8)$, $(7,9)$, $(7,7)$, $(5,9)$, $(5,7)$, $(8,8)$, $(4,8)$, $(6,10)$, $(6,6)$.

- 1 ✓ when getting all starting locations right
- $\frac{1}{2}$ ✓ when getting at least $4/9$ starting locations right
- 0 ✓ otherwise

0 □
1 □
2 □

b) **Derive** the maximum likelihood estimate for the starting location $\mathbf{x}^{(0)}$, i.e. $\operatorname{argmax}_{\mathbf{s}} \Pr\left[\mathbf{x}^{(2)} = \begin{bmatrix}6 & 8\end{bmatrix}^\top \mid \mathbf{x}^{(0)} = \mathbf{s}\right]$.

- $(6,8)$: $2 \cdot \frac{2}{8} \cdot \frac{2}{8} + 2 \cdot \frac{1}{8} \cdot \frac{3}{8} = \frac{14}{64}$
- $(5,9)$, $(7,9)$: $2 \cdot \frac{2}{8} \cdot \frac{3}{8} = \frac{12}{64}$
- $(5,7)$, $(7,7)$: $2 \cdot \frac{2}{8} \cdot \frac{1}{8} = \frac{4}{64}$
- $(4,8)$, $(8,8)$: $\frac{2}{8} \cdot \frac{2}{8} = \frac{4}{64}$
- $(6,10)$: $\frac{3}{8} \cdot \frac{3}{8} = \frac{9}{64}$
- $(6,6)$: $\frac{1}{8} \cdot \frac{1}{8} = \frac{1}{64}$

Starting location $\begin{bmatrix}6 & 8\end{bmatrix}^\top$ is the most likely.
Note that when we start to the left $(6,8)$ of $(6,8)$, we need to go to the right, which makes the starting location less likely. For instance, the likelihood of $(6,6)$ is smaller than that of $(6,10)$.

- 0.5 ✓ when all calculations are correct, but students calculate probabilities for reaching s from $\begin{bmatrix}6 & 8\end{bmatrix}$ instead of the other way around XOR forget to sum over multiple possible paths.
- 0 ✓ if they get both wrong.
- 1 ✓ when using correct ansatz, i.e. multiplying probability for first step from s with probability for second step and summing over all paths.
- 1.5 ✓ when derivation and results for at least $6/9$ starting locations are correct (it's ok if they end up with an incorrect MLE estimate due to previous mistakes)
- 2 ✓ when everything is correct

Problem 3 Linear regression (5 credits)

We want to perform regularized linear regression (without bias) on a dataset with N samples $\mathbf{x}_i \in \mathbb{R}^d$ with corresponding targets y_i (represented compactly as $\mathbf{X} \in \mathbb{R}^{N\times d}$ and $\mathbf{y} \in \mathbb{R}^N$). You assume that your targets are normal distributed, i.e.,

$$p(y_i|\mathbf{x}_i, \mathbf{w}) = \mathcal{N}(\mathbf{x}_i^\top \mathbf{w}, 1) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(y_i - \mathbf{x}_i^\top \mathbf{w}\right)^2\right). \tag{3.1}$$

To add regularization, you choose a Laplace prior on the parameters $\mathbf{w} \in \mathbb{R}^d$, i.e.,

$$p(\mathbf{w}) = \frac{1}{2\lambda} \prod_{i=1}^{d} \exp\left(-\frac{|w_i|}{\lambda}\right). \tag{3.2}$$

with hyperparameter $\lambda > 0$.

a) **Derive** the negative logarithm of the posterior distribution, i.e., $-\log p(\mathbf{w}|\mathbf{X}, \mathbf{y})$ (up to some normalization constant).

□ 0
□ 1
□ 2
□ 3
□ 4

1P for each line.

$$p(\mathbf{w}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})} \propto p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w}) \tag{3.3}$$

$$= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}} \exp\left(\frac{1}{2}(y_i - \mathbf{x}_i^T\mathbf{w})^2\right) \frac{1}{2\lambda} \prod_{i=1}^{d} \exp\left(-\frac{|w_i|}{\lambda}\right) \tag{3.4}$$

$$\propto \prod_{i=1}^{N} \exp\left(\frac{1}{2}(y_i - \mathbf{x}_i^T\mathbf{w})^2\right) \prod_{i=1}^{d} \exp\left(-\frac{|w_i|}{\lambda}\right) \tag{3.5}$$

$$\log p(\mathbf{w}|\mathbf{X}, \mathbf{y}) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \mathbf{x}_i^T\mathbf{w})^2 + -\frac{1}{\lambda} \sum_{i=1}^{d} |w_i| \tag{3.6}$$

b) What is the advantage/difference of having such a Laplace prior over a Gaussian prior? **Justify your answer!**

□ 0
□ 1

The Laplace distribution is more fat-tailed than a Gaussian due to the lower exponent, i.e., it penalizes larger values less while small values are penalized to a larger amount due to the sharp decrease around 0. As an optimal point estimate, the Laplace distribution results in the median while the Gaussian results in the mean. Thus, sparsity is encouraged. 0.5P for sparsity, 0.5P for reasoning

Problem 4 Optimization (6 credits)

Below, you are given two different functions and asked to **prove** convexity.

0 □ □ 1 □ □ 2 □ □ 3 □

a) Prove that the subsequent function is convex in $\mathbf{x} \in \mathbb{R}^d_{>0}$, i.e., over the set of vectors solely consisting of positive entries $x_i > 0$ for all $i = 1, \ldots, d$:

$$f(\mathbf{x}) = \sum_{i=1}^{d} x_i \log x_i$$

Hint: *Remember that a matrix* $\mathbf{A} \in \mathbb{R}^{d \times d}$ *is positive semi-definitive, if* $\forall z \in \mathbb{R}^d : \mathbf{z}^\top \mathbf{A} \mathbf{z} \geq 0$

0.5P for recognizing proof strategy as showing positive semi-definites of Hessian. 1P for deriving the correct form of the Hessian

$$\nabla^2 f(\mathbf{x})_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j} = \begin{cases} 0 & \text{if } i \neq j \\ \frac{1}{x_i} & \text{otherwise} \end{cases} \tag{4.1}$$

Therefore, for an arbitrary $z \in \mathbb{R}^n$

1P for correctly proving positive definiteness.

$$\begin{aligned} z^\top \nabla^2 f(x) z &= \sum_{i=1}^{n} z_i (\sum_{j=1}^{n} \nabla^2 f(x)_{ij} z_j) && (4.2) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} z_i \nabla^2 f(x)_{ij} z_j && (4.3) \\ &= \sum_{i=1}^{n} \frac{1}{x_i} z_i^2 && (4.4) \\ &\geq 0 && (4.5) \end{aligned}$$

Line 4.4 follows from Equation 4.1 and the last inequality from $x_i > 0$.

0.5P for connecting positive definiteness with convexity.

Because the Hessian of $f(x)$ is pos. semi-definitive, $f(x)$ is convex.

b) Let $f_1 : \mathbb{R}^d \to \mathbb{R}$ and $f_2 : \mathbb{R}^d \to \mathbb{R}$ be two convex functions. **Prove** that

□ 0
□
□ 1
□
□ 2
□
□ 3

$$h(\mathbf{x}) = \max\{f_1(\mathbf{x}), f_2(\mathbf{x})\}$$

is a convex function.
Note: For this, you are not allowed to use any convexity rules from the lecture without proving them.

0.5P for recognizing proof strategy as showing definition of convexity holds.
Assume $x, y \in \mathbb{R}^d$, $\lambda \in [0, 1]$. We show that the definition of a convex function holds.
Each line in the following proof is worth 0.5P. Students may make more steps at once. However, if one just writes down e.g. the first and last line of the below proof, that are not enough intermediate steps! The solution has to be derived in a way such that each next step is reasonable und understandable (use your judgement). If many steps are done at once, there must be a verbal (or similar) explanation of why that should be valid. Only give points up to this point where it is clear that the student understood what they did.

$$\begin{aligned}
h(\lambda x + (1-\lambda)y) &= \max\{f_1(\lambda x + (1-\lambda)y), f_2(\lambda x + (1-\lambda)y)\} && (4.6)\\
&\leq \max\{\lambda f_1(x) + (1-\lambda)f_1(y), \lambda f_2(x) + (1-\lambda)f_2(y)\} && (4.7)\\
&\leq \max\{\lambda f_1(x), \lambda f_2(y)\} + \max\{(1-\lambda)f_1(y), (1-\lambda)f_2(y)\} && (4.8)\\
&= \lambda \max\{f_1(x), f_2(x)\} + (1-\lambda)\max\{f_1(y), f_2(y)\} && (4.9)\\
&= \lambda h(x) + (1-\lambda)h(y) && (4.10)
\end{aligned}$$

Line 4.7 follows from the convexity of f_1 and f_2. Line 4.9 holds, because λ is non-negative. This proves $h(x)$ is convex.
Note: Using variables without declaring them, e.g. y or λ, results in -0.5P (in total not for each variable).

Problem 5 Deep Learning (5 credits)

The following code snippets all contain **exactly one error**. Your task is to spot the mistakes and explain how to fix it. **Justify your answer!**
We omitted variable initializations to avoid clutter. Assume that all variables were appropriately initialized.

0 □ □ 1 □ □ 2 □

a) Given an input $\mathbf{x} \in \mathbb{R}^d$, the subsequent class implements the ReLU layer $f(x) = \max(0, x)$ and the corresponding backward pass.

```
import numpy as np

class ReLU:
    def forward(self, inputs):
        self.cache = inputs
        out = np.maximum(inputs, 0)
        return out
    def backward(self, d_out):
        inputs = self.cache
        d_inputs = d_out * (inputs < 0)
        return d_inputs

relu = ReLU()
z = relu.forward(x)
d_x = relu.backward(1.0)
```

The backward pass through the ReLU layer is wrong (1p).

$$\frac{\partial \mathrm{ReLU}(x)}{\partial x} = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases} \tag{5.1}$$

The fix (1p) should thus be:

```
d_inputs = d_out * (inputs > 0)
```

b) We have trained a model to perform multiclass classification over c classes on a dataset $\mathbf{X} \in \mathbb{R}^{n\times d}$ and one-hot encoded targets $y \in \{0,1\}^{n\times c}$ with $\sum_{j=1}^{c} y_{i,j} = 1 \quad \forall i \in [1, \dots, n]$. The model is defined as: outputs = ReLU($x@w_1 + b_1$)@$w_2 + b_2$. The model was trained to minimize the Cross Entropy between the one-hot encoded target and the prediction. Now, we want to obtain the normalized class probabilities as well as the predicted class.

□ 0 □ 1 □ 2 □ 3

```
import torch

model.eval()
outputs = model.forward(x)
classprobs = torch.sigmoid(outputs)
y_hat = torch.argmax(classprobs, axis=1)
for i, (cp, yh) in enumerate(zip(classprobs, y_hat)):
    print(f"predicted class {yh} for sample {i} with probability {cp[yh]}")
```

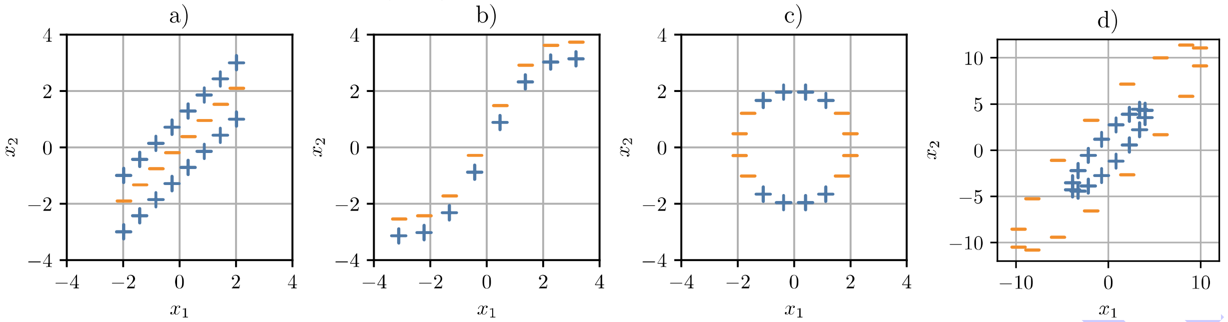
The sigmoid over a multiclass output will not give normalized class probabilities (1p). Instead, the softmax should be taken over the output (1p for softmax, 1p for axis kw):

```
classprobs = torch.softmax(outputs, dim=-1)
```

Problem 6 Linear classification (4 credits)

0 □
1 □
2 □
3 □
4 □

We want to perform binary classification on four different datasets, $t \in \{a, b, c, d\}$, each consisting of N_t samples $\mathbf{x}_i \in \mathbb{R}^2$ with labels $y_i \in \{-, +\}$:



You already came up with transformations $\phi_1, ..., \phi_4$ that transform the respective datasets such that they are linearly separable:

$$\phi_1(\mathbf{x}) = \hat{x}_1 \hat{x}_2$$

$$\hat{\mathbf{x}} = \mathbf{x} \begin{bmatrix} \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} \\ \sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{bmatrix} \tag{6.1}$$

$$\phi_2(\mathbf{x}) = x_2 - \sin(x_1) - x_1 \tag{6.2}$$

$$\phi_3(\mathbf{x}) = \left\| \begin{bmatrix} \frac{x_1}{2} \\ x_2 - x_1 \end{bmatrix} \right\|_2 \tag{6.3}$$

$$\phi_4(\mathbf{x}) = |x_1 - x_2| \tag{6.4}$$

Unfortunately, you forgot which transform belongs to which dataset. Assign the transformations $\phi_1, \phi_2, \phi_3, \phi_4$ to the datasets a, b, c, d such that the transformed datasets are linearly separable. **Justify your answer!**

We give four points, half a point for each matching and half a point for each justification.
The correct pairs are:

- (a, 4): subtracting x_1 from x_2 maps each point to either -1, 0, or 1. The absolute value then maps -1 to 1. So, afterward the - are at 0 and all plus are mapped to 1.
- (b, 2): Similar to a, by subtracting $\sin(x_1)$ we get a similar result to a but with - being the upper line. Thus, subtracting x_1 also maps the data points either to 0 (-) or 1 (+) depending on their class.
- (c, 1): Here we first need to rotate the dataset by 45° (thus the rotation matrix), then the data points are clearly distinguished by the sign of the product of both coordinates.
- (d, 3): Here the classes differ by radius in an ellipsoid. The ellipsoid is stretched in x direction and correlated in y direction. The transformations in (3) reverse the transformations such that the euclidean norm (L2) norm distinguishes both circles.

Problem 7 Support Vector Machines and Kernels (4 credits)

You are given a dataset with N datapoints $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$ representing the class of datapoint i. We use the augmentation trick $\mathbf{x} \mapsto \tilde{\mathbf{x}} = (\mathbf{x}, 1)$ to turn the affine decision function of an SVM classifier $h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ (with explicit bias term) into a linear function $\tilde{h}(\mathbf{x}) = \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}$ with $\tilde{\mathbf{w}} = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$.

Now, we want to solve the adapted (maximum-margin) optimization problem

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}\tilde{\mathbf{w}}^\top \tilde{\mathbf{w}} \\ \text{subject to} \quad & y_i \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}_i - 1 \geq 0 \qquad i = 1, \ldots, N \end{aligned}$$

a) What is the Lagrangian function $L(\tilde{\mathbf{w}}, \boldsymbol{\alpha})$ associated to the above problem, with $\alpha_i \geq 0$ corresponding to the Lagrangian multipliers.

□ 0
□ 1

$$L(\tilde{w}, \alpha) = \frac{1}{2}\tilde{w}^\top \tilde{w} - \sum_{i=1}^N \alpha_i (y_i \tilde{w}^\top \tilde{x}_i - 1) \tag{7.1}$$

0.5P for correctly coming up with the first term in $L(\tilde{w}, \alpha)$ and 0.5P for correctly comming up the second term.

b) **Derive** the corresponding dual function $g(\boldsymbol{\alpha})$. It suffices to simplify $g(\boldsymbol{\alpha})$ such that it does not contain any minimization or maximization term.

□ 0
□ 1
□ 2
□ 3

Hint: *The Lagrangian function $L(\tilde{\mathbf{w}}, \boldsymbol{\alpha})$ is convex in $\tilde{\mathbf{w}}$.*

0.5P for recognizing that to minimize $L(\tilde{\mathbf{w}}, \boldsymbol{\alpha})$ w.r.t. $\tilde{\mathbf{w}}$, we have to set the gradient w.r.t. $\tilde{\mathbf{w}}$ to 0.

The dual function is defined as $g(\alpha) = \min_w L(\tilde{w}, \alpha)$. Because $L(\tilde{w}, \alpha)$ is convex, it suffices to set its derivative w.r.t. $\tilde{w}$ to zero.

1P for correctly calculating $\nabla_{\tilde{w}} L(\tilde{w}, \alpha)$.

0.5P for correctly calculating the solution of $\nabla_{\tilde{w}} L(\tilde{w}, \alpha) = 0$, i.e., $\tilde{w}^* = \sum_{i=1}^N \alpha_i y_i \tilde{x}_i$

$$\nabla_{\tilde{w}} L(\tilde{w}, \alpha) = \tilde{w} - \sum_{i=1}^N \alpha_i y_i \tilde{x}_i = 0 \quad \rightarrow \tilde{w}^* = \sum_{i=1}^N \alpha_i y_i \tilde{x}_i \tag{7.2}$$

1P for plugging the derived $\tilde{w}^*$ back into $g(\alpha)$.

Plugging the derived $\tilde{w}^*$ back into $g(\alpha)$ gives the solution

$$g(\alpha) = L(\tilde{w}^*, \alpha) \tag{7.3}$$

$$= \frac{1}{2}||\sum_{i=1}^N \alpha_i y_i \tilde{x}_i||_2^2 - \sum_{i=1}^N \alpha_i \Big(y_i \big(\sum_{j=1}^N \alpha_j y_j \tilde{x}_j\big)^\top \tilde{x}_i - 1\Big) \tag{7.4}$$

Up to here should be enough.

Problem 8 PCA (3 credits)

0 □
1 □
2 □
3 □

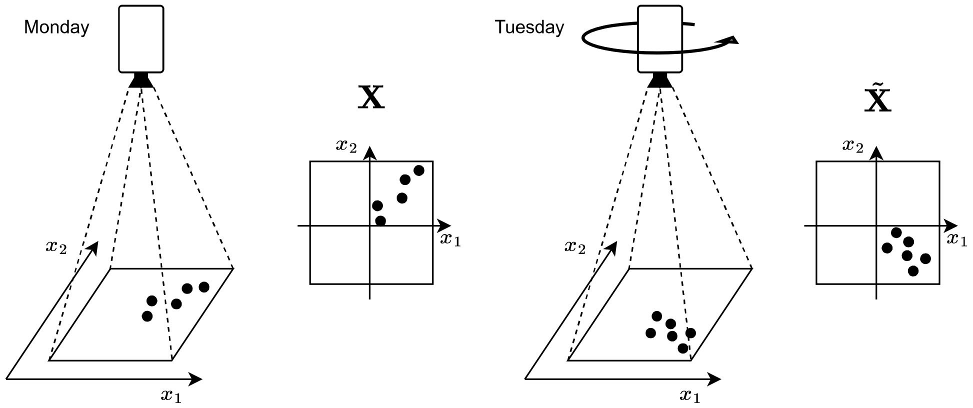
On Monday, you experimented with growing bacteria and took a photo of the result. You recorded the positions of bacteria as illustrated below. Each position is a two-dimensional coordinate, with the origin in the middle of the camera's frame. The positions are saved in a data matrix $\mathbf{X} \in \mathbb{R}^{N\times 2}$. On Tuesday, you repeated the experiment but did not set up the camera at the same angle. Tuesday's measurements are denoted with $\tilde{\mathbf{X}} \in \mathbb{R}^{M\times 2}$.

Since you assume the positions will follow the **same distribution** every day, you want to rotate the data recorded on Tuesday to **match the direction and shape** of the data from Monday. Unfortunately, the only data processing technique you know is PCA. Fortunately, this is enough to solve this problem. **Propose a solution and justify your answer.**

You have a function `PCA(D)` at your disposal, which takes data matrix $\mathbf{D} \in \mathbb{R}^{a\times b}$ and returns $\mathbf{\Gamma} \in \mathbb{R}^{b\times b}$ corresponding to the principal components, and $\mathbf{\Lambda} \in \mathbb{R}^{b}$ corresponding to the eigenvalues. You also know the commands for basic matrix manipulation: addition, subtraction, multiplication and transpose.

Assume that PCA always gives you the desired eigenvectors, that is, ignore the potential sign flips in $\mathbf{\Gamma}$.

Note: Figure below is just for illustration purposes. The angle and the values N and M are not given.



New data is assumed to be rotated by some random rotation matrix $\mathbf{R}$. If the data wasn't rotated it would have the same principal components.

Thus, the covariance of the original data can be factorized as $\mathbf{\Gamma\Lambda\Gamma}^\top$ while the covariance of the rotated data can be factorized as $\mathbf{R}^\top\mathbf{\Gamma\Lambda\Gamma}^\top\mathbf{R}$.

We only need to find $\mathbf{R}$ and then rotate. To do that perform the following:

1. $\mathbf{\Gamma}_1 = \text{PCA}(\mathbf{X})$
2. $\mathbf{\Gamma}_2 = \text{PCA}(\tilde{\mathbf{X}})$
3. $\mathbf{R}^\top = \mathbf{\Gamma}_2\mathbf{\Gamma}_1^\top = \mathbf{R}^\top\mathbf{\Gamma}_1\mathbf{\Gamma}_1^\top$
4. $\tilde{\mathbf{X}} \leftarrow \tilde{\mathbf{X}}\mathbf{R}^\top$

Important, we have to multiply with $\mathbf{R}^\top$ to revert the rotation. It's not necessary to center the data.

1pt if projecting both $\mathbf{X}$ and $\tilde{\mathbf{X}}$ to the same space with PCA or if computing eigendecomposition with function PCA for both matrices.

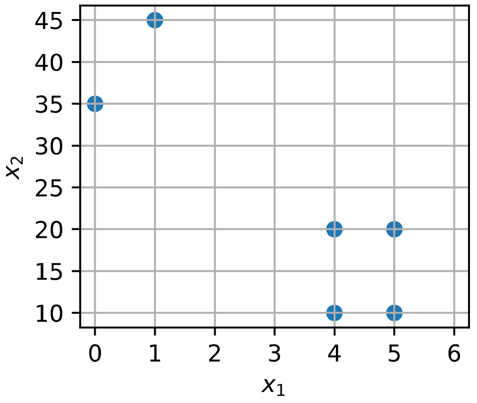
Projecting only one without followup — 0pt.

Rotating as described in task or by first projecting to same space and inverse transforming — 3pt.

Having wrong rotation matrix (up to transpose) or forgetting to "uncenter" data — remove 0.5pt.

Problem 9 Clustering (2 credits)

You are given the following two-dimensional dataset $\mathbf{X} \in \mathbb{R}^{6\times 2}$:

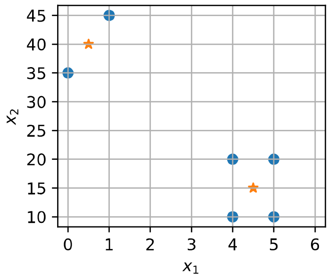


□ 0
□ 1

a) What are the globally optimal cluster centers μ that minimize the k-means objective with $K = 2$

$$J(\mathbf{X}, \mathbf{Z}, \mu) = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbf{z}_{ik} ||\mathbf{x}_i - \mu_k||_2^2 \tag{9.1}$$

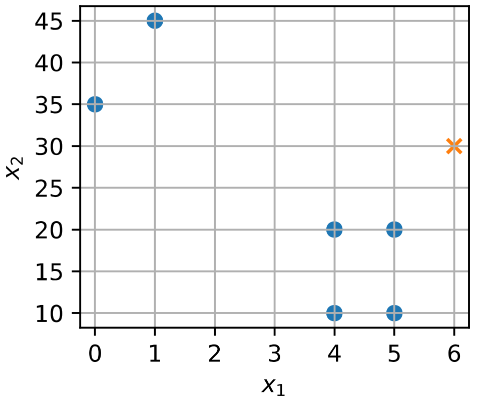
with the assignment to the closest cluster centers $\mathbf{Z}$.



(0.5, 40) ½ ✓ and (4.5, 15) ½ ✓ Due to an error in the exercise, any possible choice of K is valid.

□ 0
□ 1

b) Now assume you want to infer the corresponding cluster for a new datapoint without updating the cluster centers μ. To what cluster center in μ does the new point (x) correspond to? **Justify your answer!**



The new point belongs to the cluster with mean (0.5, 40), i.e., the cluster consisting of example (0, 35) and (1, 45). $\sqrt{(6-4.5)^2 + (30-15)^2} = 15.07 > 11.4 = \sqrt{(6-0.5)^2 + (30-40)^2}$ ✓ Due to an error in the exercise, any possible choice of K is valid.

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.