Machine Learning — Final Exam — SOLUTION

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	$\sum$

Do not write anything above this line

Name:

Student ID: Signature:

- Only write on the sheets given to you by supervisors. If you need more paper, ask the supervisors.
- All sheets (including scratch paper) have to be returned at the end.
- **Do not unstaple the sheets!**
- Wherever answer boxes are provided, please write your answers in them.
- Please write your student ID (*Matrikelnummer*) on every sheet you hand in.
- **Only use a black or a blue pen (no pencils, red or green pens!).**
- You are allowed to use your A4 sheet of handwritten notes (two sides). **No other materials (e.g. books, cell phones, calculators) are allowed!**
- Exam duration - 120 minutes.
- This exam consists of 14 pages, 19 problems. You can earn 52 points.

Probability distributions

For your reference, we provide the following probability density functions.

- Multivariate normal distribution

$$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right),$$

  where D is the dimensionality of $\boldsymbol{x}$.

- Beta distribution

$$\text{Beta}(\theta \mid a, b) = \begin{cases} \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \theta^{a-1}(1-\theta)^{b-1} & \text{if } \theta \in [0,1], \\ 0 & \text{else;} \end{cases}$$

  where $\Gamma(\cdot)$ is the gamma function.

- Exponential distribution

$$\text{Expo}(x \mid \lambda) = \begin{cases} \lambda \exp(-\lambda x) & \text{if } x \in [0, \infty), \\ 0 & \text{else.} \end{cases}$$

1 Probability Theory

Problem 1 [2 points] Given is the joint PDF of 3 continuous random variables $p(a, b, c)$. Write down how the following expressions can be obtained using the rules of probability.

1. $p(a)$
2. $p(c \mid a, b)$
3. $p(b \mid c)$

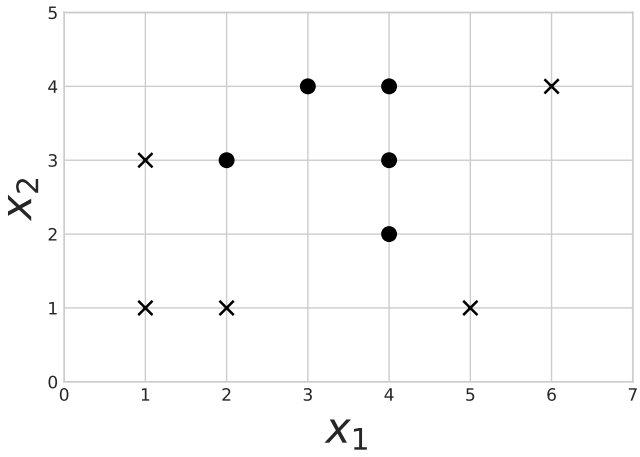
$$p(a) = \int \int p(a, b, c) dbdc$$

$$p(c \mid a, b) = \frac{p(a, b, c)}{p(a, b)} = \frac{p(a, b, c)}{\int p(a, b, c) dc}$$

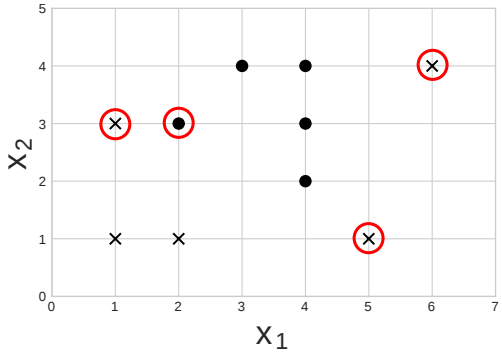
$$p(b \mid c) = \frac{p(b, c)}{p(c)} = \frac{\int p(a, b, c) da}{\int \int p(a, b, c) dadb}$$

2 KNN Classification

You are given the following dataset, where points of two different classes are denoted as • and ×.



Problem 2 [2 points] We perform 1-NN classification with Euclidean distance with leave-one-out cross validation on the data in the plot. Circle all points that are misclassified during this procedure.



3 Probabilistic Inference

Problem 3 [5 points] You model a coin flip F as a Bernoulli distribution with a parameter θ

$$p(F \mid \theta) = \text{Bern}(F \mid \theta) = \theta^{\mathbb{I}[F=T]}(1-\theta)^{\mathbb{I}[F=H]}.$$

That is, the probability of landing tails (T) is θ, and probability of heads (H) is $(1-\theta)$ respectively.

Your prior on θ is a Beta$(6, 4)$ distribution

$$p(\theta) = \text{Beta}(\theta \mid 6, 4).$$

You observe $(M + N)$ coin flips, out of which M are tails and N are heads. After you do maximum a posteriori estimation of θ, you obtain the result $\theta_{MAP} = 0.75$.

Name any possible values of M and N that can lead to such result. Show your work.

Because Beta is a conjugate prior to binomial likelihood, the posterior is also a Beta

$$p(\theta \mid \mathcal{D}) = \text{Beta}(\theta \mid 6 + M, 4 + N)$$

The mode of a Beta(a, b) distribution is

$$\theta_{MAP} = \frac{a-1}{a+b-2}$$

(If you don't remember this fact, it can be easily shown by taking the derivative of the log of Beta PDF and setting it to zero).

$$\begin{aligned}
\frac{d}{d\theta}\log \text{Beta}(\theta \mid a, b) &= \frac{d}{d\theta}[(a-1)\log\theta + (b-1)\log(1-\theta)] \\
&= \frac{a-1}{\theta} - \frac{b-1}{1-\theta} \stackrel{!}{=} 0 \\
(a-1)(1-\theta) - \theta(b-1) &= 0 \\
\theta_{MAP} &= \frac{a-1}{a+b-2}
\end{aligned}$$

Which in our case means

$$\theta_{MAP} = \frac{6+M-1}{6+M+4+N-2} = \frac{M+5}{M+N+8}$$

so it must hold that

$$\begin{aligned}
\frac{M+5}{M+N+8} &\stackrel{!}{=} \frac{3}{4} \\
4(5+M) &= 3(M+N+8) \\
M &= 3N+4
\end{aligned}$$

any pair (M, N) that satisfies this equation will produce the required result.
For example $N = 1$ and $M = 7$ works.

4 Regression

John Doe is a data scientist, and he wants to fit a polynomial regression model to his data. For this, he needs to choose the degree of the polynomial that works best for his problem.

Unfortunately, John hasn't attended IN2064, so he writes the following code for choosing the optimal degree of the polynomial:

```
X, y = load_data()
best_error = -1
best_degree = None

for degree in range(1, 50):
    w = fit_polynomial_regression(X, y, degree)
    y_predicted = predict_polynomial_regression(X, w, degree)
    error = compute_mean_squared_error(y, y_predicted)
    if (error <= best_error) or (best_error == -1):
        best_error = error
        best_degree = degree

print("Best degree is " + str(best_degree))
```

Assume that the functions are implemented correctly and do what their name suggests.
(e.g., `fit_polynomial_regression` returns the optimal coefficients `w` for a polynomial regression model with the given degree.)

Problem 4 [3 points] What is the output of this code? Explain in 1-2 sentences why this code doesn't do what it's supposed to do.

Output: `Best degree is 49`
(`range(1, 50)` in Python goes to 49, but 50 is also accepted as a correct answer)

Error on the training set *always* goes down when we use a higher degree polynomial (unless it's already 0, then it stays at 0).

Problem 5 [2 points] Describe in 1-2 sentences a possible way to fix the problem with this code. (You don't need to write any code, just describe the approach.)

Split data into train and validation sets. Choose the degree that achieves the lowest mean squared error on the validation set (not on the training set!).

Remark: Regularization does not help at all in this case. No matter which λ you choose, higher degree polynomial is still able to fit the training data better.

5 Classification

We want to create a generative binary classification model for classifying *nonnegative* one-dimensional data. This means, that the labels are binary ($y \in \{0, 1\}$) and the samples are $x \in [0, \infty)$.

We place a uniform prior on y

$$p(y = 0) = p(y = 1) = \frac{1}{2}.$$

As our samples x are nonnegative, we use exponential distributions (and not Gaussians) as class conditionals:

$$p(x \mid y = 0) = \text{Expo}(x \mid \lambda_0) \qquad \text{and} \qquad p(x \mid y = 1) = \text{Expo}(x \mid \lambda_1),$$

where $\lambda_0 \neq \lambda_1$. Assume, that the parameters λ_0 and λ_1 are known and fixed.

Problem 6 [1 point] What is the name of the posterior distribution $p(y \mid x)$? You only need to provide the name of the distribution (e.g., "normal", "gamma", etc.), not estimate its parameters.

Bernoulli.

Remark: y can only take values in $\{0, 1\}$, so obviously Bernoulli is the only possible answer.

Problem 7 [5 points] What values of x are classified as class 1?
(As usual, we assume that the classification decision is $y_{predicted} = \arg\max_k p(y = k \mid x)$)

Sample x is classified as class 1 if $p(y = 1 \mid x) > p(y = 0 \mid x)$. This is the same as saying

$$\frac{p(y = 1 \mid x)}{p(y = 0 \mid x)} \overset{!}{>} 1 \quad \text{or equivalently} \quad \log \frac{p(y = 1 \mid x)}{p(y = 0 \mid x)} \overset{!}{>} 0.$$

$$\begin{aligned}
\log \frac{p(y = 1 \mid x)}{p(y = 0 \mid x)} &= \log \frac{p(x \mid y = 1)p(y = 1)}{p(x \mid y = 0)p(y = 0)} \\
&= \log \frac{p(x \mid y = 1)}{p(x \mid y = 0)} \\
&= \log \frac{\lambda_1 \exp(-\lambda_1 x)}{\lambda_0 \exp(-\lambda_0 x)} \\
&= \log \frac{\lambda_1}{\lambda_0} + \lambda_0 x - \lambda_1 x = \log \frac{\lambda_1}{\lambda_0} + (\lambda_0 - \lambda_1)x
\end{aligned}$$

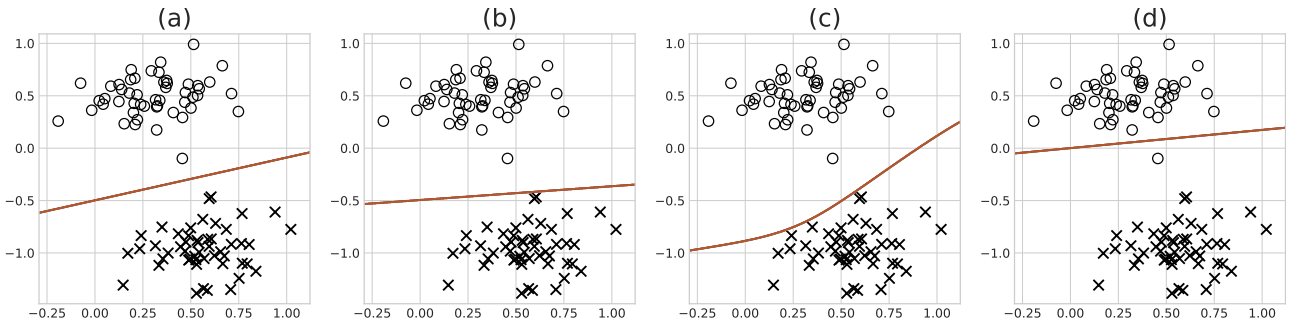
The inequality that we are interested in solving is

$$\begin{aligned}
(\lambda_0 - \lambda_1)x + \log \frac{\lambda_1}{\lambda_0} &> 0 \\
(\lambda_0 - \lambda_1)x &> \log \frac{\lambda_0}{\lambda_1}
\end{aligned}$$

We have to be careful, because if $(\lambda_0 - \lambda_1) < 0$, diving by it will flip the inequality sign. Hence the answer is

$$\begin{cases}
x \in \left(\frac{\log \lambda_0 - \log \lambda_1}{\lambda_0 - \lambda_1}, \infty\right) & \text{if} \quad \lambda_0 > \lambda_1; \\
x \in \left[0, \frac{\log \lambda_0 - \log \lambda_1}{\lambda_0 - \lambda_1}\right) & \text{if} \quad \lambda_0 < \lambda_1.
\end{cases}$$

Problem 8 [3 points] We fitted 4 different classification algorithms on the same dataset and obtained 4 different decision boundaries.



Match each classifier to the respective decision boundary. Explain each of the answers with 1 sentence.

#	**Classification algorithm**	**Plot**
1.	Logistic regression	
2.	Hard margin linear SVM	
3.	Soft margin SVM with polynomial kernel	
4.	Perceptron	

3. Soft margin SVM with polynomial kernel - plot (c). The only method that has a nonlinear decision boundary.

1. Logistic regression - plot (d). Linear decision boundary + the only of the remaining methods that permit misclassifying some training samples.

2. Hard margin linear SVM - plot (a). Linear decision boundary + no points are misclassified + has the highest margin.

4. Perceptron - plot (b). All points are correctly classified, but the margin is smaller than in SVM case (plot (a)).

6 Optimization

You are given the following objective function

$$f(x_1, x_2) = 0.5x_1^2 + x_2^2 + 2x_1 + x_2 + \cos(\sin(\sqrt{\pi})).$$

Problem 9 [2 points] Compute the minimum (x_1^*, x_2^*) of $f(x_1, x_2)$ analytically.

As f is a sum of convex functions, it is convex (in x_1 and x_2).

To find the minimum, simply compute the gradient and set it to zero

$$\nabla f(x_1, x_2) = \begin{pmatrix} x_1 + 2 \\ 2x_2 + 1 \end{pmatrix} \stackrel{!}{=} \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$\implies \begin{pmatrix} x_1^* \\ x_2^* \end{pmatrix} = \begin{pmatrix} -2 \\ -\frac{1}{2} \end{pmatrix}.$$

Problem 10 [2 points] Perform 2 steps of gradient descent on $f(x_1, x_2)$ starting from the point $(x_1^{(0)}, x_2^{(0)}) = (0, 0)$ with learning rate $\tau = 1$.

We already know the gradient from Problem 9. The first update

$$\begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \end{pmatrix} = \begin{pmatrix} x_1^{(0)} \\ x_2^{(0)} \end{pmatrix} - \tau \begin{pmatrix} x_1^{(0)} + 2 \\ 2x_2^{(0)} + 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - 1 \begin{pmatrix} 0 + 2 \\ 0 + 1 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \end{pmatrix}.$$

The second update

$$\begin{pmatrix} x_1^{(2)} \\ x_2^{(2)} \end{pmatrix} = \begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \end{pmatrix} - \tau \begin{pmatrix} x_1^{(1)} + 2 \\ 2x_2^{(1)} + 1 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \end{pmatrix} - 1 \begin{pmatrix} -2 + 2 \\ -2 + 1 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix}.$$

Problem 11 [2 points] Will the gradient descent procedure from Problem 10 ever converge to the true minimum (x_1^*, x_2^*)? Why or why not? If the answer is no, how can we fix it?

By performing one more iteration of GD we observe that

$$\begin{pmatrix} x_1^{(3)} \\ x_2^{(3)} \end{pmatrix} = \begin{pmatrix} x_1^{(2)} \\ x_2^{(2)} \end{pmatrix} - \tau \begin{pmatrix} x_1^{(2)} + 2 \\ 2x_2^{(2)} + 1 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix} - 1 \begin{pmatrix} -2+2 \\ 0+1 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \end{pmatrix} = \begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \end{pmatrix}$$

That is, we are stuck iterating between $\boldsymbol{x^{(1)}}$ and $\boldsymbol{x^{(2)}}$ forever.

We can fix this by decreasing the learning rate (adaptive stepsize, etc.).

Problem 12 [2 points] Given two convex functions $g_1 : \mathbb{R} \to \mathbb{R}$ and $g_2 : \mathbb{R} \to \mathbb{R}$, prove or disprove that the function $h(x) = g_1(g_2(x))$ is also convex.

Disprove by counterexample:

Consider $g_1(x) = -x$ and $g_2(x) = x^2$. Both g_1 and g_2 are convex, but $h(x) = -x^2$ is concave.

7 Kernels

Problem 13 [3 points] Prove or disprove that the function $k : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$, defined as

$$k(\boldsymbol{x}, \boldsymbol{y}) = x_1 y_1 - x_2 y_2$$

is a valid kernel.

A valid kernel $k(\boldsymbol{x}, \boldsymbol{y})$ by definition must correspond to an inner product $\langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{y}) \rangle$ for some feature map $\varphi : \mathbb{R}^2 \to \mathcal{H}$.

By the positive definiteness property of inner product it must hold that

$$\langle \boldsymbol{z}, \boldsymbol{z} \rangle \geq 0 \quad \text{for all } \boldsymbol{z} \in \mathcal{H}.$$

Consider $\boldsymbol{x} = (0, 1)^T$.

$$k(\boldsymbol{x}, \boldsymbol{x}) = 0 - 1 = -1 < 0,$$

which violates the positive definiteness property, hence k does not correspond to an inner product, hence k is not a valid kernel.

Alternative solution: By Mercer's theorem a valid kernel must produce a PSD Gram matrix $\mathbf{K}$ *(a.k.a. kernel matrix) when applied to any dataset* $\mathbf{X} = \{\boldsymbol{x}_1, ...\boldsymbol{x}_N\}$.

Consider the case with a single sample $\boldsymbol{x} = (0, 1)^T$.

In this case $k(\boldsymbol{x}, \boldsymbol{x}) = -1$ *is the* 1×1 *Gram matrix. A negative scalar is an example of negative-definite matrix in* $\mathbb{R}^{1\times1}$*, which according to Mercer's theorem means that* k *is not a valid kernel.*

8 Deep Learning

Problem 14 [1 point] In feed-forward neural networks, we apply elementwise nonlinear activation functions $\sigma(\cdot)$ at each layer. What is the purpose of doing this? What happens if we don't?

The purpose is to give the NN an ability to approximate nonlinear functions.

If we don't apply nonlinearities the NN can be expressed as a linear function of the form $\tilde{\boldsymbol{W}}\boldsymbol{x}$.

Problem 15 [1 point] Suppose that we use the following activation function in a simple feedforward neural network (NN)

$$\sigma(x) = e^{|x|}.$$

Describe in 1-2 sentences what problem will likely arise when training our NN using gradient descent.

For all $x \in \mathbb{R}$ we have $\sigma(x) \geq 1$ (with equality only for $x = 0$), and $|\frac{d}{dx}\sigma(x)| = e^{|x|}$.

Therefore repeated applications of σ will likely lead to exploding gradients.

9 Dimensionality Reduction

Problem 16 [5 points] Outline the main steps of applying PCA to reduce the dimensionality of a dataset $\mathbf{X} \in \mathbb{R}^{N \times D}$ from D to K.

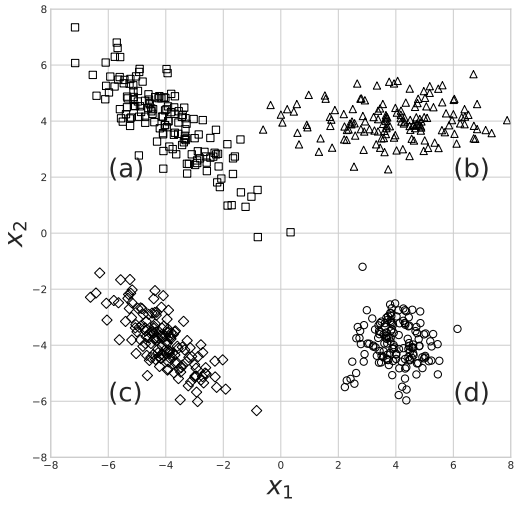
1. Center the data
$$\tilde{\boldsymbol{X}} = \boldsymbol{X} - \mathbf{1}_N \underbrace{(\frac{1}{N}\mathbf{1}_N^T \boldsymbol{X})}_{\bar{\boldsymbol{x}}}.$$
2. Compute the covariance matrix
$$\boldsymbol{\Sigma} = \frac{1}{N}\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}}.$$
3. Compute the eigendecompoisiton of $\boldsymbol{\Sigma}$
$$\boldsymbol{\Sigma} = \boldsymbol{\Gamma}^T\boldsymbol{\Lambda}\boldsymbol{\Gamma}^T.$$
4. Choose the top-K eigenvectors $\boldsymbol{\Gamma_K}$ (the ones that correspond to the K largest eigenvalues) of the covariance matrix $\boldsymbol{\Sigma}$.
5. Project the data
$$\boldsymbol{Y} = \tilde{\boldsymbol{X}}\boldsymbol{\Gamma_K}.$$

10 Clustering

Problem 17 [3 points] The dataset displayed on the right has been generated using a Gaussian mixture model with $K = 4$ components, each with its own mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$.

Match the covariance matrices in the table on the left with their corresponding Gaussian components in the plot on the right. Explain each of the answers with 1 sentence.

$\boldsymbol{\Sigma}_k$	**Cluster**
$\begin{bmatrix} 2 & -1.7 \\ -1.7 & 2 \end{bmatrix}$	
$\begin{bmatrix} 0.9 & -0.8 \\ -0.8 & 1.2 \end{bmatrix}$	
$\begin{bmatrix} 3 & 0 \\ 0 & 0.5 \end{bmatrix}$	
$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$	



1. plot (a) - Has same shape as 2 (plot (c)), but has higher variance in all directions.
2. plot (c) - Has same shape as 1 (plot (a)), but has lower variance in all directions.
3. plot (b) - Aligned with the axes (offdiagonal entries are 0); more variance along the x_1 dimension.
4. plot (d) - Isotropic Gaussian - same variance in all directions.

11 Variational Inference

Problem 18 [6 points] The distribution $p(\boldsymbol{x})$ is defined by a Gaussian mixture model

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

We are trying to approximate $p(\boldsymbol{x})$ with an isotropic Gaussian distribution $q(\boldsymbol{x} \mid \boldsymbol{m})$ by minimizing *forward* Kullback-Leibler divergence $\mathbb{KL}(p \parallel q)$.

The variational distribution $q(\boldsymbol{x} \mid \boldsymbol{m})$ is parameterized by the mean $\boldsymbol{m}$ and has a fixed identity covariance matrix $\boldsymbol{I}$.

$$q(\boldsymbol{x} \mid \boldsymbol{m}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{m}, \boldsymbol{I})$$

Derive the closed-form expression for the optimal mean $\boldsymbol{m}^*$ of the variational distribution $q(\boldsymbol{x} \mid \boldsymbol{m})$, that minimizes the KL divergence

$$\boldsymbol{m}^* = \arg\min_{\boldsymbol{m}} \mathbb{KL}(p(\boldsymbol{x}) \parallel q(\boldsymbol{x} \mid \boldsymbol{m})).$$

Show your work.

Write down the KL divergence

$$\mathbb{KL}(p \parallel q) = -\int p(\boldsymbol{x}) \log q(\boldsymbol{x}) d\boldsymbol{x} + \int p(\boldsymbol{x}) \log p(\boldsymbol{x}) d\boldsymbol{x}.$$

The second term doesn't depend on $q(\boldsymbol{x})$, so we can absorb it into const.

Plugging in the (Gaussian) density of $q(\boldsymbol{x})$

$$= -\int p(\boldsymbol{x}) \left(-\frac{D}{2} \log 2\pi - \frac{1}{2} \log |\boldsymbol{I}| - \frac{1}{2} (\boldsymbol{x} - \boldsymbol{m})^T \boldsymbol{I}^{-1} (\boldsymbol{x} - \boldsymbol{m}) \right) d\boldsymbol{x} + \text{const.}$$

Absorbing the constant terms

$$= -\int p(\boldsymbol{x}) \left(-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{m})^T (\boldsymbol{x} - \boldsymbol{m}) \right) d\boldsymbol{x} + \text{const.}$$

This is just an expectation w.r.t. $p(\boldsymbol{x})$. By linearity of expectation

$$= \frac{1}{2} (\mathbb{E}_p[\boldsymbol{x}] - \boldsymbol{m})^T (\mathbb{E}_p[\boldsymbol{x}] - \boldsymbol{m}) + \text{const.}$$
$$= \frac{1}{2} \boldsymbol{m}^T \boldsymbol{m} - \mathbb{E}_p[\boldsymbol{x}]^T \boldsymbol{m} + \text{const.}$$

Compute the gradient w.r.t. $\boldsymbol{m}$

$$\nabla_{\boldsymbol{m}} \mathbb{KL}(p \parallel q) = \nabla_{\boldsymbol{m}} \left(\frac{1}{2} \boldsymbol{m}^T \boldsymbol{m} - \mathbb{E}_p[\boldsymbol{x}]^T \boldsymbol{m} + \text{const.} \right)$$
$$= \boldsymbol{m} - \mathbb{E}_p[\boldsymbol{x}]$$

Setting the gradient to zero, we obtain the solution

$$\boldsymbol{m}^* = \arg\min_{\boldsymbol{m}} \mathbb{KL}(p \parallel q) = \mathbb{E}_p[\boldsymbol{x}]$$

Now we need to get a closed form expression for $\mathbb{E}_p[\boldsymbol{x}]$.

Like in Problem 1 of HW 10, using the law of iterated expectations

$$\begin{aligned}\mathbb{E}[\boldsymbol{x}] &= \mathbb{E}_{p(\boldsymbol{z})}\left[\mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{z})}[\boldsymbol{x}|\boldsymbol{z}]\right] \\ &= \sum_{k=1}^{K} \pi_k \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{z})}[\boldsymbol{x}|\boldsymbol{z}]\end{aligned}$$

plugging in the mean of multivariate Gaussian, we get

$$= \sum_{k=1}^{K} \pi_k \boldsymbol{\mu}_k$$

(Alternatively, we can compute $\mathbb{E}_p[\boldsymbol{x}]$ by writing the integral and then pushing the expectation under the sum.)

Therefore, the answer is

$$\boldsymbol{m}^* = \arg\min_{\boldsymbol{m}} \mathbb{KL}(p \parallel q) = \mathbb{E}_p[\boldsymbol{x}] = \sum_{k=1}^{K} \pi_k \boldsymbol{\mu}_k$$

Problem 19 [2 points] When doing variational inference, we usually minimize the *reverse* KL divergence, $\mathbb{KL}(q \parallel p)$. Why do we use this objective instead of minimizing the forward KL divergence, $\mathbb{KL}(p \parallel q)$? Why is it possible to minimize $\mathbb{KL}(p \parallel q)$ in Problem 18?

In context of variational inference we usually want to approximate an *intractable* distribution p. That means, that we cannot compute expectations w.r.t. p (e.g. we cannot compute $\mathbb{E}_p[\boldsymbol{x}]$).

In Problem 18 $p(\boldsymbol{x})$ is tractable, that is we can easily evaluate its moments, such as $\mathbb{E}_p[\boldsymbol{x}]$.