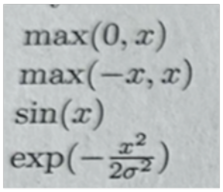
Exam 1 WS2024/2025

1. Multiple Choice (6 x 4 = 20 P)
Answer the following multiple choice questions. There is only one good answer per question. To make an answer, put an x in the box next to it. For each question, no or false answer is zero pointe, correct answer is full points.

(a) Which of the following is True about the Bayes-optimal classifier?
Its performance represents the theoretical upper bound of accuracy given the true data distributions.
Its performance is typically inferior to that of other classifiers in practice.
It disregards misclassification errors entirely.
It relies on approximations rather than true posterior probabilities.

(b) Which of the following is True for Principal Component Analysis (PA)?
It selects a subset of original features that contribute most to classification.
It projects the data onto a new set of uncorrelated variables ordered by variance.
It clusters the data points based on their similarity
It is primarily used for increasing the number of features in a dataset.

(c) Which of the following is a popular activation function used in neural networks to introduce non-linearity into the forward process?

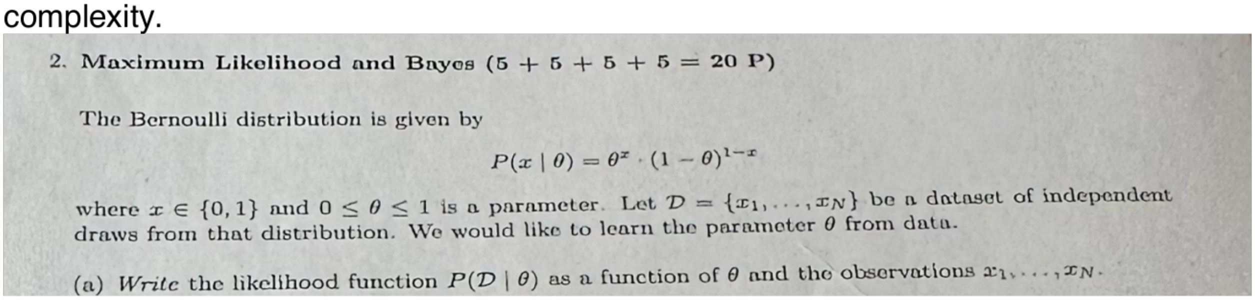


$\max(0, x)$
$\max(-x, x)$
$\sin(x)$
$\exp(-\frac{x^2}{2\sigma^2})$

(d) Which of the following is True for slack variables in Support Vector Machines (SVM)?
They help separate non-linearly separable data by mapping it to a higher-dimensional space.
They allow some misclassification to improve the model's flexibility in handling non-linearly separable data.
They make the margin smaller to ensure a tighter fit to the training data.Slacking in class improves the learning performance.

Which of the following is True about activation maximization as an explainability method?
Activation maximization finds input patterns that maximize the activation of a specific neuron or output.
Activation maximization finds network weights that maximize the activation of a specific neuron for a given input.
Activation maximization is used to compute per-feature importance scores for a prediction.

Activation maximization makes models more interpretable by reducing model complexity.



2. Maximum Likelihood and Bayes (5 + 5 + 5 + 5 = 20 P)

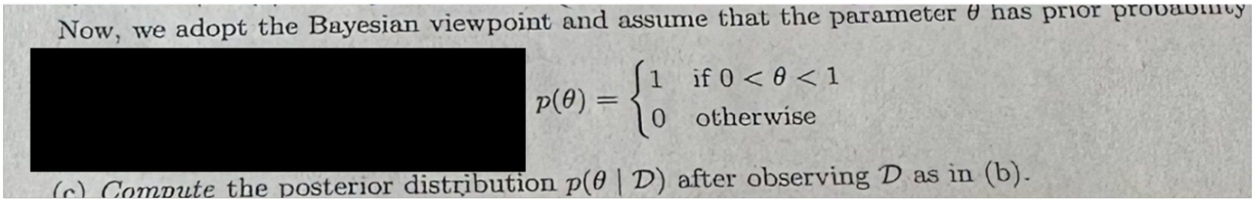
The Bernoulli distribution is given by

$$P(x \mid \theta) = \theta^x \cdot (1 - \theta)^{1-x}$$

where $x \in \{0, 1\}$ and $0 \leq \theta \leq 1$ is a parameter. Let $\mathcal{D} = \{x_1, \ldots, x_N\}$ be a dataset of independent draws from that distribution. We would like to learn the parameter θ from data.

(a) *Write* the likelihood function $P(\mathcal{D} \mid \theta)$ as a function of θ and the observations $x_1, \ldots, x_N$.

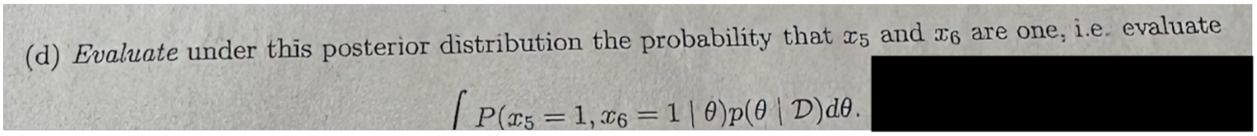
(b) Compute the maximum likelihood solution for Θ^ given the dataset D = {1,0,1, 1} and the probability P(x5 = 1, x6 = 1 I Θ^).



Now, we adopt the Bayesian viewpoint and assume that the parameter θ has prior probability

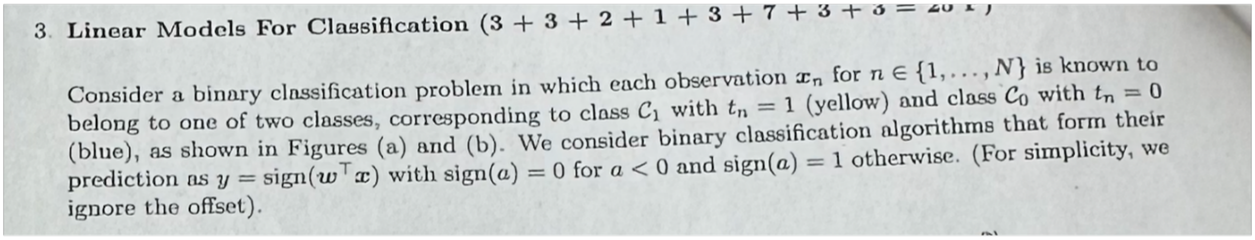
[redacted] $p(\theta) = \begin{cases} 1 & \text{if } 0 < \theta < 1 \\ 0 & \text{otherwise} \end{cases}$

(c) *Compute* the posterior distribution $p(\theta \mid \mathcal{D})$ after observing $\mathcal{D}$ as in (b).



(d) *Evaluate* under this posterior distribution the probability that x_5 and x_6 are one, i.e. evaluate

$$\int P(x_5 = 1, x_6 = 1 \mid \theta) p(\theta \mid \mathcal{D}) d\theta.$$ [redacted]



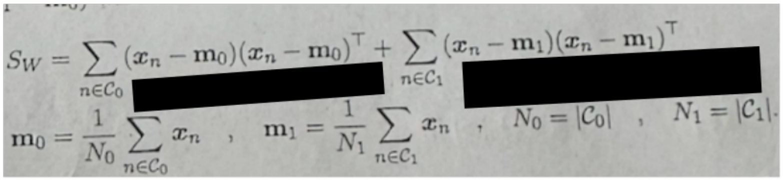
3. Linear Models For Classification (3 + 3 + 2 + 1 + 3 + 7 + 3 + 3 = 25 P)

Consider a binary classification problem in which each observation x_n for $n \in \{1, \ldots, N\}$ is known to belong to one of two classes, corresponding to class $\mathcal{C}_1$ with $t_n = 1$ (yellow) and class $\mathcal{C}_0$ with $t_n = 0$ (blue), as shown in Figures (a) and (b). We consider binary classification algorithms that form their prediction as $y = \text{sign}(\boldsymbol{w}^\top \boldsymbol{x})$ with $\text{sign}(a) = 0$ for $a < 0$ and $\text{sign}(a) = 1$ otherwise. (For simplicity, we ignore the offset).

(3 half-moon plots)

(a) Draw in Figure (a) the weight vector w (as an arrow) and the corresponding decision boundary (as a line) of a classifier trained according to the mean separation criterion. The orange and blue crosses show the empirical mean of the classes. Hint: The mean separation criterion aims to maximize the
distances between the class means when projected onto w

(b) Draw in Figure (b) the weight vector w (as an arrow) and the corresponding decision boundary (as a line) of Fisher's Linear Discriminant (LDA). The solution of Fisher's Linear Discriminant is given
by w = S^{-1}_{w} (m1 – m0) where



$$S_W = \sum_{n\in\mathcal{C}_0} (\boldsymbol{x}_n - \mathbf{m}_0)(\boldsymbol{x}_n - \mathbf{m}_0)^\top + \sum_{n\in\mathcal{C}_1} (\boldsymbol{x}_n - \mathbf{m}_1)(\boldsymbol{x}_n - \mathbf{m}_1)^\top$$

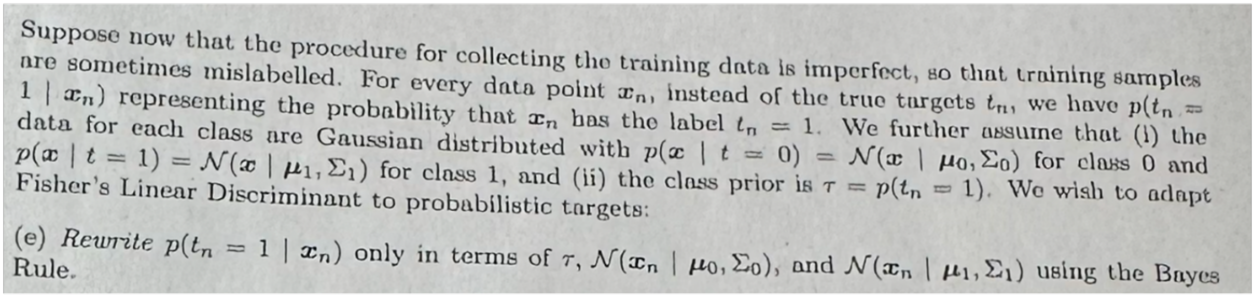
[redacted] [redacted]

$$\mathbf{m}_0 = \frac{1}{N_0}\sum_{n\in\mathcal{C}_0} \boldsymbol{x}_n \;, \quad \mathbf{m}_1 = \frac{1}{N_1}\sum_{n\in\mathcal{C}_1} \boldsymbol{x}_n \;, \quad N_0 = |\mathcal{C}_0| \;, \quad N_1 = |\mathcal{C}_1| .$$

The within-class variance is the sum of the unnormalized variance of the projected data of each class.
The contour plots visualize the covariance structure of the classes

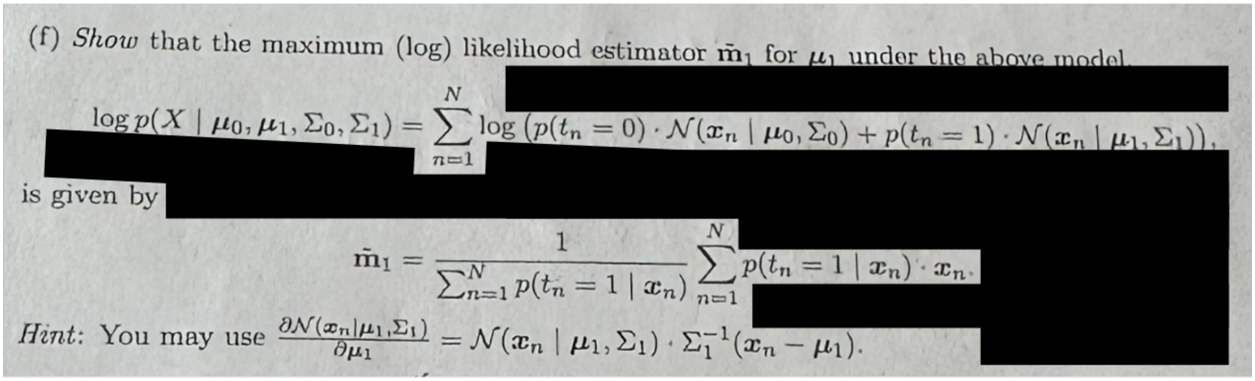
(c) Explain the objective that Fisher's Linear Discriminant tries to optimize.

(d) State a classification algorithm, discussed in the lecture, that will be able to classify the above data without any misclassification.



Suppose now that the procedure for collecting the training data is imperfect, so that training samples are sometimes mislabelled. For every data point $\boldsymbol{x}_n$, instead of the true targets t_n, we have $p(t_n = 1 \mid \boldsymbol{x}_n)$ representing the probability that $\boldsymbol{x}_n$ has the label $t_n = 1$. We further assume that (i) the data for each class are Gaussian distributed with $p(\boldsymbol{x} \mid t = 0) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_0, \Sigma_0)$ for class 0 and $p(\boldsymbol{x} \mid t = 1) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_1, \Sigma_1)$ for class 1, and (ii) the class prior is $\tau = p(t_n = 1)$. We wish to adapt Fisher's Linear Discriminant to probabilistic targets:

(e) *Rewrite* $p(t_n = 1 \mid \boldsymbol{x}_n)$ only in terms of τ, $\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_0, \Sigma_0)$, and $\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_1, \Sigma_1)$ using the Bayes Rule.



(f) *Show* that the maximum (log) likelihood estimator $\tilde{\mathbf{m}}_1$ for $\boldsymbol{\mu}_1$ under the above model [redacted]

$$\log p(X \mid \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \Sigma_0, \Sigma_1) = \sum_{n=1}^{N} \log\left(p(t_n = 0)\cdot\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_0, \Sigma_0) + p(t_n = 1)\cdot\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_1, \Sigma_1)\right),$$

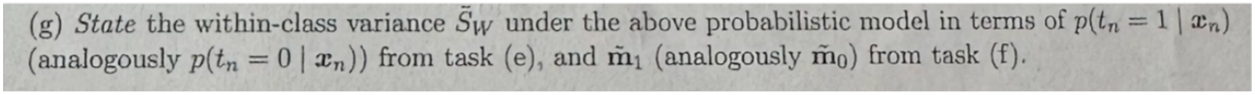
[redacted]

is given by [redacted]

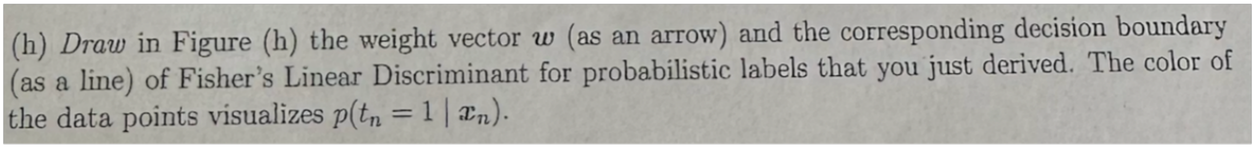
$$\tilde{\mathbf{m}}_1 = \frac{1}{\sum_{n=1}^{N} p(t_n = 1 \mid \boldsymbol{x}_n)} \sum_{n=1}^{N} p(t_n = 1 \mid \boldsymbol{x}_n)\cdot\boldsymbol{x}_n .$$

[redacted]

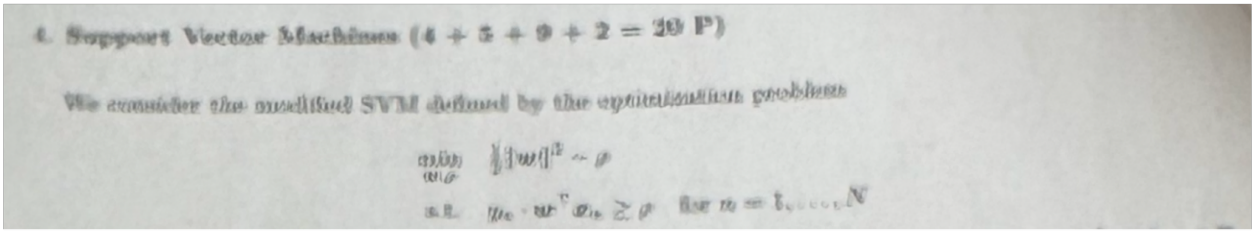
Hint: You may use $\frac{\partial\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_1, \Sigma_1)}{\partial\boldsymbol{\mu}_1} = \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_1, \Sigma_1)\cdot\Sigma_1^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_1)$. [redacted]



(g) *State* the within-class variance $\tilde{S}_W$ under the above probabilistic model in terms of $p(t_n = 1 \mid \boldsymbol{x}_n)$ (analogously $p(t_n = 0 \mid \boldsymbol{x}_n)$) from task (e), and $\tilde{\mathbf{m}}_1$ (analogously $\tilde{\mathbf{m}}_0$) from task (f).



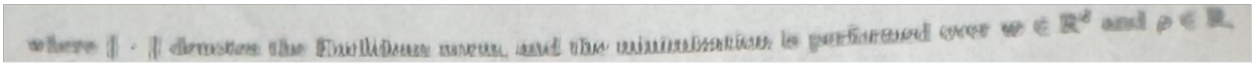
(h) *Draw* in Figure (h) the weight vector $\boldsymbol{w}$ (as an arrow) and the corresponding decision boundary (as a line) of Fisher's Linear Discriminant for probabilistic labels that you just derived. The color of the data points visualizes $p(t_n = 1 \mid \boldsymbol{x}_n)$.



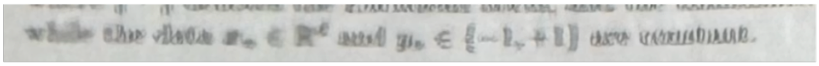
4. **Support Vector Machines (4 + 5 + 9 + 2 = 20 P)**

We consider the modified SVM defined by the optimization problem

$$\min_{w,\rho} \ \frac{1}{2}\|w\|^2 - \rho$$
$$\text{s.t.} \quad y_n \cdot w^\top x_n \geq \rho \quad \text{for } n = 1, \ldots, N$$



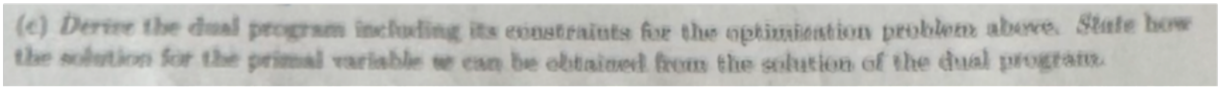
where $\|\cdot\|$ denotes the Euclidean norm, and the minimization is performed over $w \in \mathbb{R}^d$ and $\rho \in \mathbb{R}$,



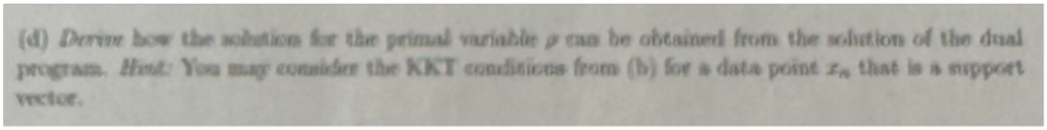
while the data $x_n \in \mathbb{R}^d$ and $y_n \in \{-1, +1\}$ are constant.

(a) State the Lagrange...

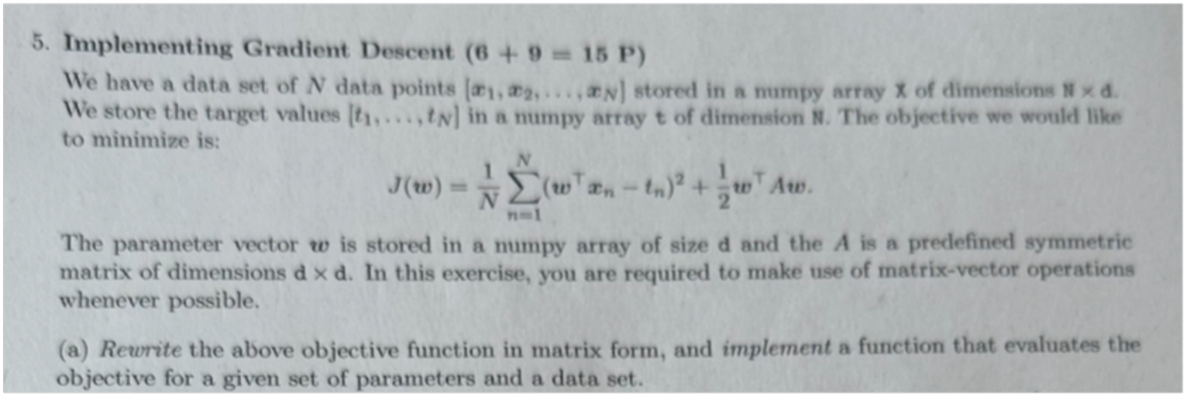
(b) State the KKT conditions



(c) *Derive* the dual program including its constraints for the optimization problem above. *State* how the solution for the primal variable w can be obtained from the solution of the dual program.



(d) *Derive* how the solution for the primal variable ρ can be obtained from the solution of the dual program. *Hint:* You may consider the KKT conditions from (b) for a data point x_n that is a support vector.



5. **Implementing Gradient Descent (6 + 9 = 15 P)**

We have a data set of N data points $[x_1, x_2, \ldots, x_N]$ stored in a numpy array X of dimensions N × d. We store the target values $[t_1, \ldots, t_N]$ in a numpy array t of dimension N. The objective we would like to minimize is:

$$J(w) = \frac{1}{N}\sum_{n=1}^{N}(w^\top x_n - t_n)^2 + \frac{1}{2}w^\top A w.$$

The parameter vector w is stored in a numpy array of size d and the A is a predefined symmetric matrix of dimensions d × d. In this exercise, you are required to make use of matrix-vector operations whenever possible.

(a) *Rewrite* the above objective function in matrix form, and *implement* a function that evaluates the objective for a given set of parameters and a data set.

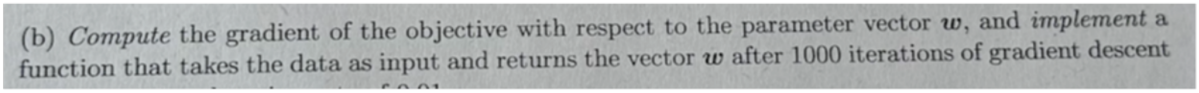
```
import numpy

def evaluate(w,X,t,A):

    N,d = X.shape()




return J
```



(b) *Compute* the gradient of the objective with respect to the parameter vector w, and *implement* a function that takes the data as input and returns the vector w after 1000 iterations of gradient descent with a constant learning rate of 0.01

```
import numpy

def gradient_descent(X,t,A):

    N,d = X.shape()

    w = numpy.random.rand()




return w
```