



Profs. Nicolas Flammarion and Martin Jaggi  
Machine Learning – CS-433 - IC  
Wednesday 13.01.2021  
from 16h15 to 19h15 in STCC  
Duration : 180 minutes













1

# Student One

SCIPER: 111111

Do not turn the page before the start of the exam. This document is double-sided, has 20 pages, the last ones are possibly blank. Do not unstaple.

- This is a closed book exam. No electronic devices of any kind.
- Place on your desk: your student ID, writing utensils, one double-sided A4 page cheat sheet (hand-written or 11pt min font size) if you have one; place all other personal items below your desk.
- You each have a different exam.
- Only answers in this booklet count. No extra loose answer sheets. You can use the last two pages as scrap paper.
- For the **multiple choice** questions, we give :
  - +2 points if your answer is correct,
  - 0 points if you give no answer or more than one,
  - −0.5 points if your answer is incorrect.
- For the **true/false** questions, we give :
  - +1 points if your answer is correct,
  - 0 points if you give no answer or more than one,
  - −1 points if your answer is incorrect.
- Use a **black or dark blue ballpen** and clearly erase with **correction fluid** if necessary.
- If a question turns out to be wrong or ambiguous, we may decide to nullify it.

Respectez les consignes suivantes   Observe this guidelines   Beachten Sie bitte die unten stehenden Richtlinien		
choisir une réponse   select an answer Antwort auswählen	ne PAS choisir une réponse   NOT select an answer NICHT Antwort auswählen	Corriger une réponse   Correct an answer Antwort korrigieren
  		 
ce qu'il ne faut <b>PAS</b> faire   what should <b>NOT</b> be done   was man <b>NICHT</b> tun sollte		
     		



## First part: multiple choice questions

For each question, mark the box corresponding to the correct answer. Each question has **exactly one** correct answer.

### Linear Regression

**Question 1** Under certain conditions, maximizing the log-likelihood is equivalent to minimizing mean-squared error for linear regression. The mean-squared error can be defined as  $\mathcal{L}_{mse}(\mathbf{w}) := \frac{1}{2N} \sum_{n=1}^N (y_n - \tilde{\mathbf{x}}_n^\top \mathbf{w})^2$  and  $y_n = \tilde{\mathbf{x}}_n^\top \mathbf{w} + \varepsilon_n$  is assumed for the probabilistic model. Which of following conditions is necessary for the equivalence?

- ☒ The noise parameter  $\varepsilon_n$  should have a normal distribution.
- ☐ The target variable  $y_n$  should have a normal distribution.
- ☐ The i.i.d. assumption on the variable  $\mathbf{w}$ .
- ☐ The conditional probability  $p(y_n | \tilde{\mathbf{x}}_n, \mathbf{w})$  should follow a Laplacian distribution.
- ☐ The noise parameter  $\varepsilon_n$  should have non-zero mean.

**Solution:** The i.i.d. assumption on  $\tilde{\mathbf{x}}$  is necessary to write the likelihood as a product of individual likelihoods, but not for the weights. The noise parameter is assumed to have a normal distribution with zero-mean. An assumption on the target variables is not used. The conditional probability should be Gaussian.

### SVMs versus Logistic Regression

Consider a classification problem on linearly separable data. We train an SVM model and a logistic regression model. For logistic regression (LR) we add a small regularization term (penalty on weights) in order to make the optimum well-defined. Each model gives us a margin. Consider a datapoint  $\mathbf{x}_0$  that is correctly classified and strictly outside both margins.

**Question 2** Which one of the following statements is **incorrect** ?

- ☒ There exists a direction in which we can slightly move  $\mathbf{x}_0$  without changing the LR decision boundary after retraining.
- ☐  $\mathbf{x}_0$  isn't a support vector.
- ☐ There exists a direction in which we can arbitrarily move  $\mathbf{x}_0$  without changing the SVM decision boundary after retraining.
- ☐ If we remove  $\mathbf{x}_0$  from the dataset and retrain, this will change the LR decision boundary.
- ☐ If we remove  $\mathbf{x}_0$  from the dataset and retrain, this will not change the SVM decision boundary.

**Solution:** The solution of the SVM problem only depends on the support vectors,  $\mathbf{x}_0$  is not one by definition, hence deleting it or moving it orthogonally to  $\mathbf{w}^*$  (solution of original SVM problem) will not change the solution. This observation is not true for logistic regression since it still puts some loss weight on correctly classified points such as  $\mathbf{x}_0$ .



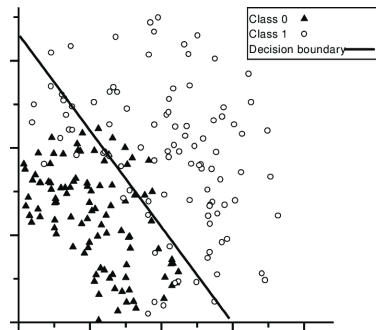
## Logistic Regression Assumptions

**Question 3** Binary logistic regression assumes a:

- ☐ Linear relationship between the input variables.
- ☐ Linear relationship between the observations.
- ☒ Linear relationship between the input variables and the logit (inverse of sigmoid) of the probability of the event that the outcome  $Y = 1$ .
- ☐ Linear relationship between the input variables and the probability of the event that the outcome  $Y = 1$ .

**Solution:** Linear relationship between the input variables and the logit of the outcome variable.  $\mathbf{x}^\top \mathbf{w} = \sigma^{-1}(\mathbb{P}(y = 1|x)) = \text{logit}(\mathbb{P}(y = 1|x))$

## Decision boundary



**Question 4** Which of these classifiers could have generated this decision boundary?

- ☐ 1-nearest-neighbor with  $L_2$  distance & SVM.
- ☒ SVM & logistic regression.
- ☐ logistic regression & 1-nearest-neighbor with  $L_2$  distance.
- ☐ None of the other options are correct.

**Solution:** SVM & logistic regression. They are the only combination of (possibly) purely linear classifiers.

## Optimization Algorithm Complexity

**Question 5** Consider a linear regression problem with  $N$  datapoints and  $D$  features. Finding the optimal parameters by running grid search while trying  $P$  values for each feature has approximately *the same* computational complexity as running:

- ☐  $P$  iterations of Gradient Descent or  $NP$  iterations of Stochastic Gradient Descent.
- ☐  $NP^D$  iterations of Gradient Descent or  $P^D$  iterations of Stochastic Gradient Descent.
- ☒  $P^D$  iterations of Gradient Descent or  $NP^D$  iterations of Stochastic Gradient Descent.
- ☐  $PD$  iterations of Gradient Descent or  $NPD$  iterations of Stochastic Gradient Descent.

**Solution:**  $P^D$  iterations of Gradient Descent or  $NP^D$  iterations of Stochastic Gradient Descent has computational complexity of  $O(P^D ND)$ , which is the same as the complexity of grid search as there are  $P^D$  grid points and computing the cost function once has complexity  $O(ND)$ .



## Subgradients

**Question 6** Consider the function  $f(x) = -x^2$ . Which of the following statements are true regarding subgradients of  $f(x)$  at  $x = 0$ ?

- ☐ A subgradient does not exist as  $f(x)$  is differentiable at  $x = 0$ .
- ☐ A subgradient exists but is not unique.
- ☐ A subgradient exists and is unique.
- ☒ A subgradient does not exist even though  $f(x)$  is differentiable at  $x = 0$ .

**Solution:** The subgradient does not exist even though  $f(x)$  is differentiable at  $x = 0$ , as  $f(x)$  is concave.

DRAFT



## Train/Test Errors



**Question 7** The above figure was produced by changing a hyperparameter in a classifier on a non-linearly separable training dataset. For which classifier could this picture be produced and how was its hyperparameter changed?

- ☐ Logistic regression, increasing regularization parameter  $\lambda$ .
- ☐ Logistic regression, decreasing regularization parameter  $\lambda$ .
- ☒ K-nearest neighbor classifier, decreasing number of neighbors  $k$ .
- ☐ K-nearest neighbor classifier, increasing number of neighbors  $k$ .

**Solution:** Since the dataset is not linearly separable, zero training error is not possible for Logistic regression. For K-NN with  $k = 1$  training error is always equal to zero.

## Exponential Families

**Question 8** You are given two distributions over  $\mathbb{R}$ : Uniform on the interval  $[a, b]$  and Gaussian with mean  $\mu$  and variance  $\sigma^2$ . Their respective probability density functions are

$$p_U(y|a, b) := \begin{cases} \frac{1}{b-a}, & \text{for } a \leq y \leq b, \\ 0 & \text{otherwise} \end{cases} \quad p_G(y|\mu, \sigma^2) := \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)$$

Which one(s) belong to the exponential family?

- ☐ Only Uniform.
- ☐ Both of them.
- ☒ Only Gaussian.
- ☐ None of them.

**Solution:** Only Gaussian. Recall that a distribution belongs to the exponential family if it can be written in the form  $p(y|\eta) = h(y) \exp(\eta^\top \phi(y) - A(\eta))$ . For the Uniform distribution, since the support depends on the parameters  $\eta = (a, b)$ , it cannot be represented in the given form. Therefore Uniform distribution is not a member of the exponential family.

For the Gaussian distribution, we can equivalently rewrite

$$p_G(y|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) = \exp\left[(\mu/\sigma^2, -1/(2\sigma^2))(y, y^2)^\top - \frac{\mu^2}{2\sigma^2} - \frac{1}{2} \ln(2\pi\sigma^2)\right],$$

and it is a member of the exponential family with  $h(y) = 1$  and

$$\phi(y) = (y, y^2)^\top, \quad \eta = (\mu/\sigma^2, -1/(2\sigma^2)), \quad A(\eta) = \frac{\mu^2}{2\sigma^2} + \frac{1}{2} \ln(2\pi\sigma^2) = -\frac{\eta_1^2}{4\eta_2} - \frac{1}{2} \ln(-\eta_2/\pi).$$



## Kernels

Let us assume that a kernel  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  is said to be *valid* if there exists  $k \in \mathbb{N}$  and  $\Phi : \mathcal{X} \rightarrow \mathbb{R}^k$  such that for all  $(x, x') \in \mathcal{X} \times \mathcal{X}$ ,  $K(x, x') = \Phi(x)^\top \Phi(x')$ .

**Question 9** Which one of the following kernels is **not** valid ?

- ☐  $\mathcal{X} = \mathbb{N}$ ,  $K(x, x') = 2$ .
- ☐  $\mathcal{X} = \mathbb{R}^d$ ,  $K(x, x') = (x^\top x')^2$ .
- ☐  $\mathcal{X} = \mathbb{R}$ ,  $K(x, x') = \cos(x - x')$ .
- ☒ All of the proposed kernels are in fact valid.
- ☐  $\mathcal{X} = \mathbb{Q}$ ,  $K(x, x') = 2^{x+x'}$ .
- ☐  $\mathcal{X} = \mathbb{R}^d$ ,  $K(x, x') = x^\top A x'$ , where  $A$  is a  $d \times d$  symmetric positive semi-definite matrix.

**Solution:**

All the kernels are valid:

- $x^\top A x' = (Bx)^\top (Bx')$  where  $B^2 = A$  ( $B$  exists since  $A$  is p.s.d.)
- $2^{x+x'} = 2^x \times 2^{x'}$
- $(x^\top x')^2$ :  $x^\top x'$  is a valid kernel, hence the square is still a valid kernel (seen in class and in exercise session)
- $2 = \sqrt{2} \times \sqrt{2}$
- $\cos(x - x') = \langle (\cos(x), \sin(x)), (\cos(x'), \sin(x')) \rangle$

## Neural networks

Let  $f_{\text{MLP}} : \mathbb{R}^d \rightarrow \mathbb{R}$  be an  $L$ -hidden layer multi-layer perceptron (MLP) such that

$$f_{\text{MLP}}(\mathbf{x}) = \mathbf{w}^\top \sigma(\mathbf{W}_L \sigma(\mathbf{W}_{L-1} \dots \sigma(\mathbf{W}_1 \mathbf{x}))),$$

with  $\mathbf{w} \in \mathbb{R}^M$ ,  $\mathbf{W}_1 \in \mathbb{R}^{M \times d}$  and  $\mathbf{W}_\ell \in \mathbb{R}^{M \times M}$  for  $\ell = 2, \dots, L$ , and  $\sigma$  is an entry-wise activation function.

Also, let  $f_{\text{CNN}} : \mathbb{R}^d \rightarrow \mathbb{R}$  be an  $L'$ -hidden layer convolutional neural network (CNN) such that

$$f_{\text{CNN}}(\mathbf{x}) = \mathbf{w}^\top \sigma(\mathbf{w}_{L'} \star \sigma(\mathbf{w}_{L'-1} \star \dots \sigma(\mathbf{w}_1 \star \mathbf{x}))),$$

with  $\mathbf{w} \in \mathbb{R}^d$ ,  $\mathbf{w}_\ell \in \mathbb{R}^K$  for  $\ell = 1, \dots, L'$  and  $\star$  denoting the one-dimensional convolution operator with zero-padding, i.e., output of the convolution has the same dimensionality as the input.

**Question 10** For each CNN neural network of the above form, there exists an MLP of the form  $f_{\text{MLP}}$  that can approximate  $f_{\text{CNN}}$  arbitrarily well, if

- ☒  $L = L'$  and  $M = d$ .
- ☐  $L > L'$  and  $M > K$ .
- ☐ another necessary condition different from these three.
- ☐  $\sigma$  is a sigmoidal activation function.

**Solution:** A convolution  $\mathbf{w} \star \mathbf{x}$  is a linear mapping between  $\mathbb{R}^d$  and  $\mathbb{R}^d$ . Hence, it can be equivalently represented by a matrix vector product  $\mathbf{W}\mathbf{x}$  with  $\mathbf{W} \in \mathbb{R}^{d \times d}$ . In the case of convolutions,  $\mathbf{W}$  will be a circulant matrix with shifted copies of the vector  $\mathbf{w}$  in its rows.



**Question 11** Let's assume  $\sigma$  is a tanh activation function. Thus, by flipping the signs of all of the weights leading in and out of a hidden neuron, the input-output mapping function represented by the network is unchanged. Besides, interchanging the values of all of the weights (i.e., by permuting the ordering of the hidden neurons within the layer) also leaves the network input-output mapping function unchanged. Suppose that, given the training data, SGD can find a solution with zero training loss, and the (absolute value) weights of such solution are non-zero and all unique. Choose the largest lower bound on the number of solutions (with zero training loss) achievable by  $f_{\text{MLP}}$  with  $L=1$  and  $M$  hidden units on this dataset.

☒  $M!2^M$

☐ 1

☐  $2^M$

☐  $M!$

**Solution:** By changing the signs of a particular group of weights, the input-output mapping function represented by the network is unchanged. Thus, any given weight vector will be one of a set of  $2^M$  equivalent weight vectors.

Similarly, interchanging the values of all of the weights also leaves the network input-output mapping function unchanged, thus for  $M$  hidden units, any given weight vector will belong to a set of  $M!$  equivalent weight vectors.

The network will therefore have an overall weight-space symmetry factor of  $M!2^M$ .

**Question 12** Regarding the weight updates in back-propagation,

☐ The output layer weights are not used for computing the error of the hidden layer.

☐ The weight changes are not proportional to the difference between the desired and actual outputs.

☐ A standard technique to initialize the weights is to set them exactly to 0.

☒ The weight change is also proportional to the input to the weight layer.

**Solution:** Based on the chain rule.



## Minimum-Norm Adversarial Examples

Consider a binary classification problem with a linear classifier  $f(\mathbf{x})$  given by

$$f(\mathbf{x}) = \begin{cases} 1, & \mathbf{w}^\top \mathbf{x} \geq 0, \\ -1, & \mathbf{w}^\top \mathbf{x} < 0, \end{cases}$$

where  $\mathbf{x} \in \mathbb{R}^3$ . Suppose that the weights of the linear model are equal to  $\mathbf{w} = (4, 0, -3)$ .

For the next two questions, we would like to find a *minimum-norm* adversarial example. Specifically, we are interested in solving the following optimization problem, for a given  $\mathbf{x}$ :

$$\min_{\boldsymbol{\delta} \in \mathbb{R}^3} \|\boldsymbol{\delta}\|_2 \quad \text{subject to} \quad \mathbf{w}^\top (\mathbf{x} + \boldsymbol{\delta}) = 0 \quad (\text{OP})$$

This leads to the point  $\mathbf{x} + \boldsymbol{\delta}$  that lies exactly at the decision boundary and the perturbation  $\boldsymbol{\delta}$  is the smallest in terms of the  $\ell_2$ -norm.

**Question 13** What is the minimum value of the optimization problem Eq. (OP) for the point  $\mathbf{x} = (-1, 3, 2)$ ?

- ☒ 2
- ☐ 3
- ☐ 1
- ☐ Other
- ☐ 1.5
- ☐ 0
- ☐  $\sqrt{2}$
- ☐ 4

**Solution:** The minimum  $\ell_2$ -distance to the hyperplane defined by  $\mathbf{w}$  is:

$$\frac{|\mathbf{w}^\top \mathbf{x}|}{\|\mathbf{w}\|_2} = \frac{|4 \cdot (-1) + 0 \cdot 3 + (-3) \cdot 2|}{\sqrt{4^2 + 0^2 + 3^2}} = \frac{10}{5} = 2. \quad (1)$$

This should be familiar from the SVM lecture or alternatively it can be also rederived from scratch by noting that the optimal  $\boldsymbol{\delta}^*$  has to be colinear with  $\mathbf{w}$ .





**Question 14** What is the optimum  $\delta^*$  that minimizes the objective in Eq. (OP) for the point  $\mathbf{x} = (-1, 3, 2)$ ?

- ☐  $(1, -1, 0)$   
☐  $(0, -1, 1)$   
☐  $(-2, 0, 0)$   
☐  $(1.2, 0, 1.6)$   
☒ Other  
☐  $(0, 2, 0)$   
☐  $(-1.2, 0, 1.6)$

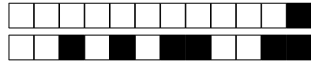
**Solution:** The optimal  $\delta^*$  has to be colinear with  $\mathbf{w}$ , this gives us:

$$\mathbf{w}^\top \delta^* = -\mathbf{w}^\top \mathbf{x} = 10 \quad (2)$$

$$\mathbf{w}^\top (\alpha \mathbf{w}) = 10 \quad (3)$$

$$\alpha = 10 / \|\mathbf{w}\|_2^2 = 10/25 = 0.4 \quad (4)$$

This leads to  $\delta^* = \alpha \mathbf{w} = 0.4 \cdot (4, 0, -3) = (1.6, 0, -1.2)$ . Therefore, the correct answer is "*Other*".



**Question 15** Let  $\mathcal{R}_p(f, \varepsilon)$  be the  $\ell_p$  adversarial risk of a classifier  $f : \mathbb{R}^d \rightarrow \{\pm 1\}$ , i.e.,

$$\mathcal{R}_p(f, \varepsilon) = \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}} \left[ \max_{\tilde{\mathbf{x}} : \|\mathbf{x} - \tilde{\mathbf{x}}\|_p \leq \varepsilon} \mathbb{1}_{\{f(\tilde{\mathbf{x}}) \neq y\}} \right],$$

for  $p = 1, 2, \infty$ . Which of the following relationships between the adversarial risks is true?

- ☐  $\mathcal{R}_2(f, \varepsilon) \leq \mathcal{R}_1(f, 2\varepsilon)$
- ☒  $\mathcal{R}_\infty(f, \varepsilon) \leq \mathcal{R}_2(f, \sqrt{d}\varepsilon)$
- ☐  $\mathcal{R}_\infty(f, \varepsilon) \leq \mathcal{R}_1(f, \varepsilon)$
- ☐  $\mathcal{R}_\infty(f, \varepsilon) \leq \mathcal{R}_2(f, \varepsilon/d)$

**Solution:** The  $\ell_\infty$  ball of radius  $\varepsilon/\sqrt{d}$  is inscribed by the  $\ell_2$  ball of radius  $\varepsilon$ , i.e.,  $\max_{\|\mathbf{x}\|_\infty \leq \varepsilon/\sqrt{d}} \|\mathbf{x}\|_2 = \varepsilon$ . Hence,  $\{\tilde{\mathbf{x}} : \|\mathbf{x} - \tilde{\mathbf{x}}\|_\infty \leq \varepsilon\} \subseteq \{\tilde{\mathbf{x}} : \|\mathbf{x} - \tilde{\mathbf{x}}\|_2 \leq \sqrt{d}\varepsilon\}$ , and therefore, since the maximum inside of the expectation is taken over a larger set, we have that  $\mathcal{R}_2(f, \sqrt{d}\varepsilon) \geq \mathcal{R}_\infty(f, \varepsilon)$

**Question 16** Identify the correct statement.

- ☐ None of the other options are correct.
- ☐ After training, and when the size of the vocabulary is large, a Skip-gram model would have higher space requirements than a GloVe model. We assume both models have the same number of dimensions (features), vocabulary, and are trained on the same corpus.
- ☒ Language models can be trained using either a multi-class (number of classes equal to the vocabulary size) classifier or a binary classifier to generate text.
- ☐ Language Models are useless for classification tasks in Natural Language Processing as they are only suited for text generation.

**Solution:** Language models can use multi-class (predict the next word) or a binary classifier (predict if the next word is real or fake) to predict the next word. Language models can be used to generate features. GloVe has higher space overhead as we need to store the co-occurrence matrix.

**Question 17** Consider a matrix factorization problem of the form  $\mathbf{X} = \mathbf{W}\mathbf{Z}^\top$  to obtain an item-user recommender system where  $x_{ij}$  denotes the rating given by  $j^{\text{th}}$  user to the  $i^{\text{th}}$  item. We use Root mean square error (RMSE) to gauge the quality of the factorization obtained. Select the correct option.

- ☐ Given a new item and a few ratings from existing users, we need to retrain the already trained recommender system from scratch to generate robust ratings for the user-item pairs containing this item.
- ☐ Regularization terms for  $\mathbf{W}$  and  $\mathbf{Z}$  in the form of their respective Frobenius norms are added to the RMSE so that the resulting objective function becomes convex.
- ☐ For obtaining a robust factorization of a matrix  $\mathbf{X}$  with  $D$  rows and  $N$  elements where  $N \ll D$ , the latent dimension  $K$  should lie somewhere between  $D$  and  $N$ .
- ☒ None of the other options are correct.

**Solution:** Latent dimensions is kept less than or equal to  $\min\{N, D\}$ . Regularization terms are added to prevent overfitting. Using SGD on the matrix entries corresponding to the new item should suffice to include it in the already trained recommender system.



## Cost functions

**Question 18** Which statement is true for the Mean Squared Error (MSE) loss  $\text{MSE}(\mathbf{x}, y) := (f_{\mathbf{w}}(\mathbf{x}) - y)^2$ , with  $f_{\mathbf{w}}$  a model parametrized by the weights  $\mathbf{w}$ ?

- ☒ MSE is not necessarily convex with respect to the weights of the model  $\mathbf{w}$ .
- ☐ MSE is more robust to outliers than Mean Absolute Error (MAE).
- ☐ For any ML task you are trying to solve, minimizing MSE will provably yield the best model.

**Solution:** Given a large error  $f_w(x) - y$ , MSE will penalize it more than MAE, making it less robust to outliers - MSE is obviously convex w.r.t. the model's output but we cannot assume any convexity w.r.t. the weights of the model (the model can be non-convex as in deep learning models) - The best model is defined based on the downstream application. In most cases, the loss we optimize is a proxy to optimize a key metric such as number of sales, engagement, precision ...

## Linear regression

**Question 19** Which statement is true for linear regression?

- ☒ A linear regression model can be expressed as an inner product between feature vectors and a weight vector.
- ☐ Linear regression, when using 'usual' loss functions, works fine when the dataset contains many outliers.
- ☐ A good fit with linear regression implies a causal relationship between inputs and outputs.

**Solution:** Outliers should be handled as they can strongly affect the best linear fit given by linear regression. - Linear regression always assumes a linear relationship between inputs and outputs. -  $f_w(x) = w_0x_0 + w_1x_1 + \dots + w_nx_n = x^\top w$ . - Correlation does not imply causation.

## Generative adversarial networks

**Question 20** Every time you open the website `thispersondoesnotexist.com`, you see a fake picture of a person that was sampled from the distribution learned by a GAN. Which part of the GAN is deployed on this server?

- ☐ The discriminator.
- ☒ The generator.
- ☐ The discriminator and the generator.
- ☐ Only the last layer of the discriminator.

**Solution:** Only the generator needs to be deployed. The discriminator is only used for training.

## Principal component analysis

In principal component analysis, the left singular vectors  $\mathbf{U}$  of a data matrix  $\mathbf{X}$  of shape  $(d \text{ features}, n \text{ datapoints})$  are used to create a new data matrix  $\mathbf{X}' = \mathbf{U}^\top \mathbf{X}$ .



**Question 21** Which property always holds for the matrix  $\mathbf{X}'$ ?

- ☐  $\mathbf{X}'$  is a square matrix.
- ☐ The mean of any row  $\mathbf{X}'_i$  is 0.
- ☐  $\mathbf{X}'$  has only positive values.
- ☒ For any two rows  $i, j$  ( $i \neq j$ ) from  $\mathbf{X}'$ , the dot product between the rows  $\mathbf{X}'_i$  and  $\mathbf{X}'_j$  is 0.

**Solution:** The principal components of any matrix are orthogonal.

**Question 22** To achieve dimensionality reduction, we keep only certain rows of the matrix  $\mathbf{X}'$ . We keep those rows that have:

- ☐ the lowest variance.
- ☒ the highest variance.
- ☐ smallest L2 norm.
- ☐ L2 norm closest to 1.

**Solution:** We keep the components with high variance. They contain most useful information.

DRAFT



## Second part: true/false questions

For each question, mark the box (without erasing) TRUE if the statement is **always true** and the box FALSE if it is **not always true** (i.e., it is sometimes false).

**Question 23** (Generalized Linear Models) Deep neural networks with logistic loss for binary classification are generalized linear models.

☐ TRUE ☒ FALSE

**Solution:** False. In generalized linear models the loss function  $L(\mathbf{w})$  is typically convex, and for deep neural networks with several layers it is non-convex.

**Question 24** (Weight initialization) The choice of weight initialization will not impact the optimization behavior of the neural network.

☐ TRUE ☒ FALSE

**Solution:** The weight initialization are crucial since poorly initialized networks cannot be trained (otherwise the activations may either vanish or explode).

**Question 25** (Support Vector Machines)

**Reminder:** The hard-margin problem for linearly separable points in  $\mathbb{R}^d$  is

$$\min_{\mathbf{w} \in \mathbb{R}^d \text{ s.t. } \forall i, y_i \mathbf{w}^\top \mathbf{x}_i \geq 1} \|\mathbf{w}\|.$$

Let  $m > 1$  be a fixed number. Then there exists  $\lambda > 0$  such that for every sample  $S$  of  $m$  examples which are linearly separable, the hard-SVM and the soft-SVM (with parameter  $\lambda$ ) solutions will return exactly the same weight vector.

☐ TRUE ☒ FALSE

**Solution:** For simplicity, consider  $m = 2$ ,  $d = 1$  and  $S = \{(x_1, y_1), (x_2, y_2)\}$  (the general case can simply be derived from this simple case). Let  $a > 0$  and consider  $x_1 = a$ ,  $y_1 = 1$ ,  $x_2 = -a$ ,  $y_2 = -1$ . The solution of the hard-SVM problem is then  $w_{hard} = 1/a$  and the solution of the  $\lambda$ -soft-SVM is

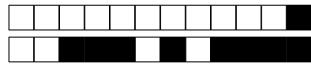
$$w_{\lambda\text{-soft}} = \underset{w}{\operatorname{argmin}} [1 - aw]_+ + \frac{\lambda}{2} \|w\|^2 = \begin{cases} 1/a, & \text{if } \lambda \leq a^2. \\ a/\lambda, & \text{otherwise.} \end{cases} \quad (5)$$

Hence, as we could expect,  $w_{\lambda\text{-soft}} = w_{hard}$  for  $\lambda$  small enough. However there does not exist a fixed  $\lambda$  that recovers the hard-margin solution for all  $a > 0$  since taking  $a \rightarrow 0$  would contradict this.

**Question 26** (Linear Regression) You are given samples  $\mathcal{S} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$  where  $\mathbf{x}_n \in \mathbb{R}^D$  and  $y_n$  are scalar values. You are solving linear regression using normal equations. You will always find the optimal weights with 0 training error in case of  $N \leq D$ .

☐ TRUE ☒ FALSE

**Solution:** False. Think of the case where your data samples are  $\{((0, 0), 0), ((0, 0), 1)\}$ .



**Question 27** (Linear or Logistic Regression) Suppose you are given a dataset of tissue images from patients with and without a certain disease. You are supposed to train a model that predicts the probability that a patient has the disease. It is preferable to use logistic regression *over* linear regression.

☒ TRUE ☐ FALSE

**Solution:** True. The predictions of linear regression are not bounded in the interval  $[0,1]$ , whereas logistic regression provides the probability of a label (patient has disease) for a given input (tissue image).

**Question 28** (Minima) Convex functions over a convex set have a unique global minimum.

☐ TRUE ☒ FALSE

**Solution:** False. Strictly convex functions have a unique global minimum, but this is not the case for convex functions in general.

**Question 29** (Neural networks) Training only the first layer of a deep neural network using the logistic loss is equivalent to training a logistic regression over a transformed feature space.

☐ TRUE ☒ FALSE

**Solution:** False. Training only the first layer of a deep neural network is a non-convex problem, while the logistic regression is convex.

**Question 30** (Adversarial perturbations for linear models) Suppose you are given a linear classifier with the logistic loss. Is it true that generating the optimal adversarial perturbations by maximizing the loss under the  $\ell_2$ -norm constraint on the perturbation is an NP-hard optimization problem?

☐ TRUE ☒ FALSE

**Solution:** False. There is a simple closed-form solution for the optimal adversarial perturbation (as seen in the exercises).

**Question 31** (FastText supervised Classifier) The FastText supervised classifier can be modeled as a one-hidden-layer neural network.

☒ TRUE ☐ FALSE

**Solution:** FastText Supervised Classifier consists of two layers - embedding layer and linear classifier. Embedding layer can be thought of as another linear layer if the input is thought of as encoding word frequency.

**Question 32** (SVD) The set of singular values of any rectangular matrix  $\mathbf{X}$  is equal to the set of eigenvalues for the square matrix  $\mathbf{X}\mathbf{X}^\top$ .

☐ TRUE ☒ FALSE

**Solution:** False. The eigenvalues of  $\mathbf{X}\mathbf{X}^\top$  are the squares of the singular values of  $\mathbf{X}$ .



### Third part, open questions

Answer in the space provided! Your answer must be justified with all steps. Leave the check-boxes empty, they are used for the grading.

#### K-Means

We will analyze the  $K$ -means algorithm and show that it always converge. Let us consider the  $K$ -means objective function:

$$\mathcal{L}(\mathbf{z}, \boldsymbol{\mu}) = \sum_{n=1}^N \sum_{k=1}^K z_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2,$$

where  $z_{nk} \in \{0, 1\}$  with  $\sum_{k=1}^K z_{nk} = 1$  and  $\boldsymbol{\mu}_k \in \mathbb{R}^D$  for  $k = 1, \dots, K$  and  $n = 1, \dots, N$ .

**Question 33:** (3 points.) How would you choose the  $\{z_{nk}\}_{n,k=1}^{N,K}$  to minimize  $\mathcal{L}(\mathbf{z}, \boldsymbol{\mu})$  for given  $\{\boldsymbol{\mu}_k\}_{k=1}^K$ ? Compute the closed-form formula for the  $z_{nk}$ . To which step of the  $K$ -means algorithm does it correspond?

☐ 0 ☐ 1 ☐ 2 ☒ 3

**Solution:** Notice that minimising  $\mathcal{L}(\mathbf{z}, \boldsymbol{\mu})$  for given  $\{\boldsymbol{\mu}_k\}_{k=1}^K$  reduces to minimizing  $\sum_{k=1}^K z_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$  for each  $n \in \{1, \dots, N\}$  **independently**. Notice that this sum is minimized for  $z_{nk} = 1$  if  $k = \arg \min_{1 \leq j \leq K} \|\mathbf{x}_n - \boldsymbol{\mu}_j\|_2^2$  and 0 otherwise. Hence the closed-form update is, for each  $n$ :

$$z_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_{1 \leq j \leq K} \|\mathbf{x}_n - \boldsymbol{\mu}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

This step corresponds to the **assignment step** and boils down to finding, for each point  $\mathbf{x}_n$ , the centroid  $\boldsymbol{\mu}_k$  which is closest.

**Question 34:** (3 points.) How would you choose  $\{\boldsymbol{\mu}_k\}_{k=1}^K$  to minimize  $\mathcal{L}(\mathbf{z}, \boldsymbol{\mu})$  for given  $\{z_{nk}\}_{n,k=1}^{N,K}$ ? Compute the closed-form formula for the  $\boldsymbol{\mu}_k$ . To which step of the  $K$ -means algorithm does it correspond?

☐ 0 ☐ 1 ☐ 2 ☒ 3

**Solution:** Notice that minimising  $\mathcal{L}(\mathbf{z}, \boldsymbol{\mu})$  for given  $\{z_{nk}\}_{n,k=1}^{N,K}$  reduces to minimizing  $\sum_{n=1}^N z_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$  for each  $k \in \{1, \dots, K\}$  **independently**. This sum is a function of  $\boldsymbol{\mu}_k$  which is quadratic and positive. It is therefore minimum when its gradient vanishes. Setting the gradient to 0 leads to  $2 \sum_{n=1}^N z_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k) = 0$ , hence the update for each  $k$  is:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^N z_{nk} \mathbf{x}_n}{\sum_{n=1}^N z_{nk}}$$

This step corresponds to the **update step** and boils down to computing the center of mass of each cluster  $k$ .

**Question 35:** (2 points.) Using the two previous questions, show that the  $K$ -means algorithm always converges. Does it converge to a global minimum of the function  $\mathcal{L}$ ? Justify your answer.

☐ 0 ☐ 1 ☒ 2







## Shift of the Eigenvalue Spectrum

**Question 36:** (1 points.) Let  $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$  be two symmetric matrices. Assume that  $\mathbf{v} \in \mathbb{R}^n$  is an eigenvector for both matrices with associated eigenvalues  $\lambda_A$  and  $\lambda_B$  respectively. Show that  $\mathbf{v}$  is an eigenvector of the matrix  $\mathbf{A} + \mathbf{B}$ . What is the corresponding eigenvalue?

0 ☒ 1

**Solution:**

$$(\mathbf{A} + \mathbf{B})\mathbf{v} = \mathbf{A}\mathbf{v} + \mathbf{B}\mathbf{v} = \lambda_A\mathbf{v} + \lambda_B\mathbf{v} = (\lambda_A + \lambda_B)\mathbf{v}$$

**Question 37:** (2 points.) We consider now the ridge regression problem:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2N} \sum_{n=1}^N [y_n - \mathbf{x}_n^\top \mathbf{w}]^2 + \lambda \|\mathbf{w}\|_2^2,$$

where the data  $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$  are such that the feature vector  $\mathbf{x}_n \in \mathbb{R}^D$  and the response variable  $y_n \in \mathbb{R}$ .

Compute the closed-form solution  $\mathbf{w}_{\text{ridge}}^*$  of this problem, providing the required justifications. State the final result using the data matrix  $\mathbf{X} \in \mathbb{R}^{N \times D}$ .

0  1 ☒ 2

**Solution:**

- Convex problem so we set the gradient to 0
- $\frac{1}{N} \sum_{n=1}^N [\mathbf{x}_n^\top \mathbf{w} - y_n] \mathbf{x}_n + 2\lambda \mathbf{w} = 0$ , therefore  $\mathbf{w} = [\sum_{n=1}^N [\mathbf{x} \mathbf{x}_n^\top] + 2N\lambda I]^{-1} \sum_{n=1}^N [\mathbf{x} y_n] = [\mathbf{X}^\top \mathbf{X} + 2N\lambda I]^{-1} \mathbf{X}^\top \mathbf{y}$ .

**Question 38:** (2 points.) Using the two previous questions, explain why computing  $\mathbf{w}_{\text{ridge}}^*$  is numerically more stable than computing the Least-squares solution  $\mathbf{w}^*$  (the solution obtained for  $\lambda = 0$ ), i.e., why the solution can be still reliably computed even if  $\mathbf{X}^\top \mathbf{X}$  is numerically singular.

0  1 ☒ 2

**Solution:** We have that

$$\mathbf{w}_{\text{ridge}}^* = [\mathbf{X}^\top \mathbf{X} + 2N\lambda I]^{-1} \mathbf{X}^\top \mathbf{y} \quad \text{and} \quad \mathbf{w}^* = [\mathbf{X}^\top \mathbf{X}]^{-1} \mathbf{X}^\top \mathbf{y}$$

- Therefore using the first question, each eigenvalue of  $\mathbf{X}^\top \mathbf{X} + 2N\lambda I$  is lifted by an amount of  $2N\lambda$  compared to the eigenvalues of  $\mathbf{X}^\top \mathbf{X}$ .
- Then even if  $\mathbf{X}^\top \mathbf{X}$  is singular  $\mathbf{X}^\top \mathbf{X} + 2N\lambda I$  will be invertible and  $\mathbf{w}_{\text{ridge}}^*$  is well defined

**Question 39:** (3 points.) In the lecture on bias-variance decomposition we have seen that the true error can be decomposed into noise, bias and variance terms. What happens to the three terms for ridge regression when the regularization parameter  $\lambda$  grows? Explain your answer.

0  1  2 ☒ 3

**Solution:**

- Noise term stays constant (it only depends on the data and not of the algorithm)
- Bias term increases (if  $\lambda = \infty$  very large bias)
- Variance term decreases (if  $\lambda = \infty$  then no variance)



## Kernel PCA

In this exercise, we will see how to combine the Principal Component Analysis (PCA) and the kernel method into an algorithm known as kernel PCA.

We are given  $n$  observations in a low dimensional space  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^L$  and we consider a kernel  $k$  and its associated features map  $\phi : \mathbb{R}^L \mapsto \mathbb{R}^H$  which satisfies:

$$k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_{\mathbb{R}^H},$$

where  $\langle \cdot, \cdot \rangle_{\mathbb{R}^H}$  is the standard scalar product of  $\mathbb{R}^H$ .

We define the empirical covariance matrix and the empirical covariance matrix of the mapped observations as:

$$\Sigma := \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top \quad \text{and} \quad \Sigma^H := \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^\top.$$

The kernel matrix  $\mathbf{K}$  is defined by:

$$\mathbf{K}_{i,j} := k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathbb{R}^H}.$$

We also define the data matrix and the corresponding matrix of the mapped data as:

$$\mathbf{X} := \begin{pmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{pmatrix} \in \mathbb{R}^{n \times L} \quad \text{and} \quad \Phi := \begin{pmatrix} \phi(\mathbf{x}_1)^\top \\ \vdots \\ \phi(\mathbf{x}_n)^\top \end{pmatrix} \in \mathbb{R}^{n \times H}.$$

Finally we denote the eigenpairs (eigenvalues and eigenvectors) of  $\Sigma^H$  by  $\{(\lambda_i, \mathbf{v}_i)\}_{i=1}^H$  and those of  $\mathbf{K}$  by  $\{(\rho_j, \mathbf{w}_j)\}_{j=1}^n$ . We also assume that the vectors  $\mathbf{v}_i$  and  $\mathbf{w}_j$  are normalized. Thus:

$$\Sigma^H \mathbf{v}_i = \lambda_i \mathbf{v}_i, \quad \|\mathbf{v}_i\|_2 = 1 \quad \text{and} \quad \mathbf{K} \mathbf{w}_j = \rho_j \mathbf{w}_j, \quad \|\mathbf{w}_j\|_2 = 1.$$

Let us remind that we assume in the kernel setting that we can compute  $k(\mathbf{x}, \mathbf{y})$  but that we cannot directly compute  $\phi(\mathbf{x})$ .

What we would like to do is to first map the data into the high-dimensional space using the features map  $\phi$  and then to apply the standard PCA algorithm in the high-dimensional space  $\mathbb{R}^H$ . This would amount to:

- Computing the empirical covariance matrix  $\Sigma^H$  of the mapped data  $\phi(\mathbf{x}_i)$ .
- Computing the eigenvectors  $\mathbf{v}_1, \dots, \mathbf{v}_N$  associated with the  $N$  largest eigenvalues of  $\Sigma^H$ .
- Computing the projection  $\Pi(\phi(\mathbf{x}_i)) \in \mathbb{R}^L$  for each data point onto these eigenvectors, where the  $j$ -th component of the projection is given by:

$$\Pi_j(\phi(\mathbf{x}_i)) = \langle \phi(\mathbf{x}_i), \mathbf{v}_j \rangle_{\mathbb{R}^H}. \quad (\text{Proj})$$



**Question 40:** (1 points.) Explain why we cannot directly apply the algorithm explained above.

☐ 0 ☒ 1

**Solution:** We assume that we cannot directly compute  $\phi(\mathbf{x})$ . The complexity would be too high. Instead, we will now apply the kernel trick to this problem:

**Question 41:** (2 points.)

Write the empirical covariance matrices  $\Sigma$  and  $\Sigma^H$  in function of the design matrix  $\mathbf{X}$  and the features matrix  $\Phi$ . What are the sizes of these matrices  $\Sigma$  and  $\Sigma^H$ ?

☐ 0 ☐ 1 ☒ 2

**Solution:**

$$\Sigma := \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top = \frac{1}{n} \mathbf{X}^\top \mathbf{X} \in \mathbf{R}^{L \times L},$$

$$\Sigma^H := \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^\top = \frac{1}{n} \Phi^\top \Phi \in \mathbf{R}^{H \times H}$$

**Question 42:** (1 points.)

Write the kernel matrix  $\mathbf{K}$  as a function of the features matrix  $\Phi$ . What is the size of this matrix?

☐ 0 ☒ 1

**Solution:**

$$\mathbf{K}_{i,j} := k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathbb{R}^H} = \Phi \Phi^\top \in \mathbf{R}^{n \times n}$$

**Question 43:** (2 points.) We are given an eigenvector  $\mathbf{w}_i$  of the matrix  $\mathbf{K}$  with associated eigenvalue  $\rho_i$ . Show that the vector  $\Phi^\top \mathbf{w}_i$  is an eigenvector of  $\Sigma^H$ . What is the associated eigenvalue?

☐ 0 ☐ 1 ☒ 2

**Solution:**

$$\Sigma^H \Phi^\top \mathbf{w}_i = \frac{1}{n} \Phi^\top \Phi \Phi^\top \mathbf{w}_i = \frac{1}{n} \Phi^\top \mathbf{K} \mathbf{w}_i = \frac{\rho_i}{n} \Phi^\top \mathbf{w}_i$$

**Question 44:** (2 points.) Let us define  $\tilde{\mathbf{v}}_i := \Phi^\top \mathbf{w}_i$ . The vectors  $\tilde{\mathbf{v}}_i$  are not normalized (i.e., their norms are not necessarily equal to one). Give a formula for the unit vector  $\mathbf{v}_i$  (with respect to the norm defined by the scalar product  $\langle \cdot, \cdot \rangle_{\mathbb{R}^H}$ ) which is aligned with the vector  $\tilde{\mathbf{v}}_i$ ?

☐ 0 ☐ 1 ☒ 2

**Solution:**

$$\|\tilde{\mathbf{v}}_i\|^2 = \|\Phi^\top \mathbf{w}_i\|^2 = \mathbf{w}_i^\top \Phi \Phi^\top \mathbf{w}_i = \mathbf{w}_i^\top \mathbf{K} \mathbf{w}_i = \rho_i \|\mathbf{w}_i\|^2 = \rho_i.$$

Therefore:

$$\mathbf{v}_i = \frac{1}{\sqrt{\rho_i}} \Phi^\top \mathbf{w}_i.$$


**Question 45:** (2 points.)

Let us assume that we have selected the  $N$  largest eigenvalues of  $\mathbf{K}$  and computed the associated  $N$  eigenvectors  $\mathbf{w}_1, \dots, \mathbf{w}_N$ . We are given a vector  $\mathbf{x} \in \mathbb{R}^L$ . Derive a formula which computes the projection  $\Pi(\phi(\mathbf{x}))$  of  $\phi(\mathbf{x})$  onto the vectors  $\mathbf{v}_1, \dots, \mathbf{v}_N$  using only the kernel and objects defined on  $\mathbb{R}^L$ . You should ensure that your final result does not explicitly contain any  $\phi$  or any vector or matrix in  $\mathbb{R}^H$ .

☐ 0 ☐ 1 ☒ 2

**Solution:**

$$\begin{aligned}\Pi_i(\phi(\mathbf{x})) &= \langle \phi(\mathbf{x}), \mathbf{v}_i \rangle = \frac{1}{\sqrt{\rho_i}} \phi(\mathbf{x})^\top \Phi^\top \mathbf{w}_i \\ &= \frac{1}{\sqrt{\rho_i}} \sum_{j=1}^n (\mathbf{w}_i)_j \phi(\mathbf{x})^\top \phi(\mathbf{x}_j) \\ &= \frac{1}{\sqrt{\rho_i}} \sum_{j=1}^n (\mathbf{w}_i)_j k(\mathbf{x}, \mathbf{x}_j).\end{aligned}$$

**Question 46:** (2 points.) Finally, rewrite the (computationally expensive) algorithm explained at the beginning of the exercise in terms of the derived results, as a method that could actually be implemented. The input should be the samples  $(\mathbf{x}_1, \dots, \mathbf{x}_n)$ , the kernel  $k$  and the number  $N$  of principal components. The outputs are the projected data points  $(\Pi_1(\phi(\mathbf{x}_1)), \dots, \Pi_N(\phi(\mathbf{x}_i)))^\top$  for  $i = 1, \dots, n$  defined in Equation (Proj).

☐ 0 ☐ 1 ☒ 2

**Solution:**

- Input:  $(\mathbf{x}_1, \dots, \mathbf{x}_n)$ , the kernel  $k$  and the number  $N$  of principal components
- Compute the kernel matrix  $\mathbf{K}$ . Find the  $N$  principal components of  $\mathbf{K}$  by doing a PCA on  $\mathbf{K}$ .
- Output:

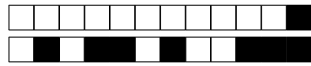
$$\Pi_i(\phi(\mathbf{x}_q)) = \frac{1}{\sqrt{\rho_i}} \sum_{j=1}^n (\mathbf{w}_i)_j k(\mathbf{x}_q, \mathbf{x}_j).$$

In the matrix form (optional):

$$(\Pi_i(\phi(\mathbf{x}_q)))_{q=1}^n = \frac{1}{\sqrt{\rho_i}} \mathbf{K} \mathbf{w}_i = \sqrt{\rho_i} \mathbf{w}_i.$$



DRAFT



DRAFT