Machine Learning 1 Gedächtnisprotokoll

Wintersemester 2020/21, Ersttermin

Hinweis: Da die Klausur online durchgeführt wurde, wurden die Aufgaben randomisiert. Die Aufgabentypen waren dieselben, lediglich die Werte oder beim Multiple-Choice-Teil die Aussagen anders. Dieses Gedächtnisprotokoll enthält nur meine Aufgaben.

Please note that the exercises were randomized. The topics of the tasks were the same for all participants, but the numbers and multiple choice questions were different.

1 Multiple Choice

[5 Points] Which of the following is **True**: In the context of model selection, risk of overfitting is particularly high when:

a. The model is a low-bias estimator.

b. The model is a high-bias estimator.

c. The model is a low-variance estimator.

d. The model is a high-variance estimator.

[5 Points] Which of the following is **True**: A biased estimator is sometimes used to:

a. Reduce the risk of underfitting the data.

b. Make the estimation procedure more sensitive to the observed data.

c. Reduce the estimation error for high-dimensional data.

d. None of the above. We should always favor an unbiased estimator.

[5 Points] Which of the following is **True**: A Mixture Model:

a. Is a class of unsupervised models designed in a way that the objective function is always convex.

b. Should only be used if the Product of Experts takes too long to train or does not converge.

c. Requires each mixture element to be Gaussian-distributed.

d. Can be trained using the expectation maximization algorithm.

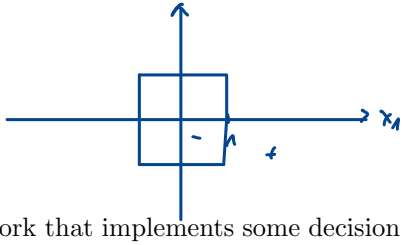
[5 Points] Which of the following is **True**: In learning theory, the Hoeffding's inequality can be used to obtain:

a. An upper-bound to the generalization error of a trainable ML classifier.

b. An upper-bound to the generalization error of a fixed classifier.

c. The exact generalization error of a trainable ML classifier.

d. The exact generalization error of a fixed classifier.



2 Neural Networks

Assume you would like to build a neural network that implements some decision boundary in $\mathbb{R}^d$. For this, you have at your disposal neurons of the type

$$a_j = \text{step}\left(\sum_i a_i w_{ij} + b_j\right)$$

where $\sum_i$ sums over the indices of the incoming neurons, and where the step function is given by $\text{step}(t) = 1_{t\geq 0}$ i.e. one when the input is positive, and zero otherwise.Denote by a_1 and a_2 the two input neurons (initialized to the value x_1 and x_2 respectively). Denote by a_3, a_4, a_5, a_6 the hidden neurons, and by a_7 the output neuron.

[10 Points] Give the weights and biases associated to a neural network with the structure above and that implements the function $f(x) = \text{step}(\min(|x_1|, |x_2|) - 1)$.

$w13 = \underline{1}$, $w23 = \underline{0}$, $b3 = \underline{-1}$
$w14 = \underline{-1}$, $w24 = \underline{0}$, $b4 = \underline{-1}$
$w15 = \underline{0}$, $w25 = \underline{1}$, $b5 = \underline{-1}$
$w16 = \underline{0}$, $w26 = \underline{-1}$, $b6 = \underline{-1}$

$w37 = \underline{1}$, $w47 = \underline{1}$, $w57 = \underline{1}$, $w67 = \underline{1}$, $b7 = \underline{-1}$

[5 Points] Assuming a similar neural network architecture composed of one layer of hidden neurons, explain what would be the number of required hidden neurons if not taking two dimensions as input, but d dimensions, and replacing x_1, x_2 by $x_1, x_2, ..., x_d$ in the formula above.

We define the objective to minimize to be the square error between the output neuron a_7 and some target variable t, i.e

$$E = (a_7 - t)^2$$

.

[5 Points] Assume you observe the data point $x = (2, 3)$ with target $t = 0$. Give the value of the partial derivative $\frac{\partial E}{\partial w_{13}}$ for this data point.

3 Parameter Estimation

Clients are lining up in a post office. We record the time $t_1, ..., t_N$ in minutes required to serve the N consecutive clients. We distinguish between two types of clients, those that are coming to send a packet, and those that are coming to send a letter (and whose service is typically twice faster). Service times for all clients are independent, and drawn from an exponential distributions with rate dependent on whether the client sends a packet or a letter:

$$p(t_i|\theta) = \theta \exp(-\theta t_i) \quad \text{(packet)}$$
$$q(t_i|\theta) = 2\theta \exp(-2\theta t_i) \quad \text{(letter)}$$

and where θ is a parameter between 0 and ∞ to be learned.

Consider six clients, the first two wanted to send a packet, and stayed at the post office for 2 and 5 minutes respectively. The last four clients wanted to send a letter and were served in 1 minute each.

[5 Points] State the likelihood function measuring the joint probability of observing all these events.

[5 Points] Give the optimal parameter θ in the maximum likelihood sense.

[5 Points] Give the expected time (according to the learned model) taken to serve the next three clients, each of them coming with a letter.

We now take a Bayesian view on the problem. We consider the prior distribution for the parameter θ to be

$$p(\theta) = \exp(-\theta)$$

[5 Points] Give the equation for the posterior distribution $p(\theta|\mathcal{D})$ where $\mathcal{D}$ denotes the dataset of observations we have made, and give the parameter θ for which this posterior distribution is maximized. (Hint: you don't need to develop $p(\mathcal{D})$.)

4 Explanations

Shapley values provide a way of attributing a prediction on the input features. The Shapley values $\phi_1, ..., \phi_d$ measuring the contribution of each feature are:

$$\phi_i = \sum_{\mathcal{S}: i \notin \mathcal{S}} \frac{|\mathcal{S}|!(d - |\mathcal{S}| - 1)!}{d!} [f(x_{\mathcal{S} \cup \{i\}}) - f(x_{\mathcal{S}})]$$

where $(x_{\mathcal{S}})_{\mathcal{S}}$ are all possible subsets of features contained in the input x. Shapley values assume a reference point which provides a replacement value for features that are removed. In this exercise, we define the reference point to be $\tilde{x} = 0$, i.e. we set features to zero when removing them.

[10 Points] Compute the Shapley values $\phi_1, \phi_2 \phi_3$ for the function $f(x) = x_1 + \max(x_2, x_3)$ evaluated at the data point $x = (1, 2, 3)$.

Another approach for explaining a prediction that does not involve evaluating the function multiple times is based on Taylor expansions. For example, we can choose a reference point $\tilde{x} = tx$ with $0 < t < 1$, and compute

$$f(x) = f(\tilde{x}) + \sum_{i=1}^{d} [\nabla f(\tilde{x})]_i \cdot (x - \tilde{x}_i) + ...$$

where ... denotes the higher-order terms. Interestingly, when the function is positive homogeneous, i.e. $\forall_{t \geq 0}$ f(tx) = tf(x) , the Taylor expansion simplifies, and in the limit of $t \to 0$, we get the scores $\phi_i = [\nabla f(x)]_i \cdot x_i$ also known as Gradient x Input.

[5 Points] Show that the function $f(x)$ used above in the Shapley value exercise is positive homogeneous, i.e. show that:

$$f(tx) = tf(x)$$

for all $x \in \mathbb{R}^3$ and $t \geq 0$.

[5 Points] Compute an explanation of the same prediction as in the Shapley values exercise, but this time using Gradient x Input, i.e. compute the explanation scores $\phi_i = [\nabla f(x)]_i \cdot x_i$.

5 Constrained Optimization

Consider an unsupervised dataset $x_1, ..., x_N \in \mathbb{R}^d$. We would like to learn a hyperplane that separates the data from the origin and that is as far as possible from the origin. This orientation of the hyperplane can be characterized by a vector $w \in \mathbb{R}^d$ to which the hyperplane is orthogonal, and this vector can be found by constrained optimization.
The first part of the exercise will consist of writing a function that receives the dataset as input (as a numpy array X of dimensions $N \times d$) and returns the vector w (a numpy array of dimension d).
Your implementation should make use of the cvxopt library internally. Specifically, your function should consist of two parts, first, prepare the data structures required by cvxopt, then run cvxopt.solvers.qp (cf. https://cvxopt.org/userguide/coneprog.html for the documentation) and return the desired output based on the output of the cvxopt function.

[15 Points] Write the desired function when the vector w is obtained by first solving the dual optimization problem:

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j x_i^{\mathsf{T}} x_j - \sum_{i=1}^{N}\alpha_i \quad \text{s.t.} \quad \forall_{i=1}^{N}\alpha_i \geq 0$$

and then computing $w = \sum_{i=1}^{N}\alpha_i x_i$.

$\alpha : n$

$\frac{1}{2}\alpha^T\alpha X^T X - 1^T\alpha$ $\quad -1^T\alpha \leq 0$

$\frac{1}{2}\alpha^T \frac{X}{P} X^T\alpha$

[5 Points] Write a function that takes the output of the previous function and finds the point in the dataset that is closest to the hyperplane.

cvxopt.solvers.qp(*P, q* [, *G, h* [, *A, b* [, *solver* [, *initvals*]]]])

Solves the pair of primal and dual convex quadratic programs

$$\begin{array}{ll} \text{minimize} & (1/2)x^T Px + q^T x \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

```
solve(X):
    N,d = X.shape
    P = matrix( X @ X.T )
    q = matrix( -1 * np.ones(N) )
    G = matrix( -1 * np.eye(N) )
    h = matrix( np.zeros(N) )
return qp( P, q, G, h )
```

```
def A(X, w)
    dists = np.abs( X.dot( w ) ) / np.linalg.norm(w)
    idx = np.argmin( dists )
    return X[:idx]
```