Instituto Superior Técnico

Aprendizagem 2021/22

Exam 1 22 November 2021

1. [4 pts] **Bayes and tree learning**

Consider the following discrete dataset:

	y_1	y_2	class
$\mathbf{x}_1$	A	B	A
$\mathbf{x}_2$	B	C	A
$\mathbf{x}_3$	B	C	B
$\mathbf{x}_4$	B	A	B
$\mathbf{x}_5$	C	A	B

Showing your calculus,

a) [2v] Classify $\mathbf{x} = \binom{\text{B}}{\text{C}}$ using naive Bayes with the maximum a posteriori (MAP) assumption

$$p(\mathbf{x}|A) = p(x_1 = B|A) \times p(x_2 = C|A) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$$
$$p(\mathbf{x}|B) = p(x_1 = B|B) \times p(x_2 = C|B) = \frac{2}{3} \times \frac{1}{3} = \frac{2}{9}$$
$$p(B|\mathbf{x}) = \frac{2}{9} \times \frac{3}{5} = \frac{2}{15} > p(A|\mathbf{x}) = \frac{1}{4} \times \frac{2}{5} = \frac{1}{10}$$

Classification: B.

b) [1v] Consider a tree learned using ID3 (information gain). Which one of the variables is tested on the root of the tree?

$$IG(y1) = E(z) - E(z|y1), IG(y2) = E(z) - E(z|y2)$$
$$E(z|y1) = \frac{1}{5} \times 0 + \frac{1}{5} \times 0 + \frac{3}{5} \times -(\frac{1}{3} log \frac{1}{3} + \frac{2}{3} log \frac{2}{3}) = \frac{3}{5} \times 0.918 = 0.55$$
$$E(z|y2) = \frac{1}{5} \times 0 + \frac{2}{5} \times 0 + \frac{2}{5} \times 1 = \frac{2}{5} = 0.4$$

y2 since $E(z|y2) < E(z|y1)$, i.e. $IG(y2) > IG(y1)$

c) [1v] Consider a classifier that, among the training observations, wrongly classifies $\mathbf{x}_2$, $\mathbf{x}_3$ and $\mathbf{x}_4$. Plot the model's confusion matrix and identify the training sensitivity of class B.

$$TA = 1,\ \ TB = 1,\ \ FA = 2,\ \ FB = 1,\ \ sensitivity_B = TB/(TB + FA) = 1/3$$

2. [2 pts] **Perceptron**

Consider a perceptron with weights, and which processing unit implements the following function

$$f(\mathbf{x}) = \exp\left(\left(\sum_{j=0}^{D} w_j \times x_j\right)^2 + 1\right)$$

Determine the gradient descent-training rule for squared error

$$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n}(t_i - o_i)^2$$

Before advancing, we notice that $\frac{\partial}{\partial x}e^{f(x)} = e^x \times f'(x)$ and optionally $(a+b)^2 = a^2 - 2ab + b^2$

$$\begin{aligned}
\frac{\partial \mathrm{E}}{\partial w_j} = \frac{\partial \frac{1}{2}\sum_{i=1}^{n}(t_i - o_i)^2}{\partial w_j} &= \frac{1}{2}\sum_{i=1}^{n}\frac{\partial(t_i{}^2 - 2t_i o_i + o_i{}^2)}{\partial w_j} \\
&= \frac{1}{2}\sum_{i=1}^{n} -2t_i\frac{\partial o_i}{\partial w_j} + \frac{\partial o_i{}^2}{\partial w_j} = \frac{1}{2}\sum_{i=1}^{n} -2t_i\frac{\partial o_i}{\partial w_j} + 2o_i\frac{\partial o_i}{\partial w_j} = \\
&= \sum_{i=1}^{n}(o_i - t_i)\frac{\partial o_i}{\partial w_j} = \sum_{i=1}^{n}(o_i - t_i)\frac{\partial \exp((\boldsymbol{w}^T\mathbf{x}_i)^2 + 1)}{\partial w_j} \\
&= \sum_{i=1}^{n}(o_i - t_i) \times \exp((\boldsymbol{w}^T\mathbf{x}_i)^2 + 1) \times \frac{\partial((\boldsymbol{w}^T\mathbf{x}_i)^2 + 1)}{\partial w_j} \\
&= \sum_{i=1}^{n}(o_i - t_i) \times \exp((\boldsymbol{w}^T\mathbf{x}_i)^2 + 1) \times 2 \times \boldsymbol{w}^T\mathbf{x}_i\frac{\partial \boldsymbol{w}^T\mathbf{x}_i}{\partial w_j} \\
&= \sum_{i=1}^{n}(o_i - t_i) \times \exp((\boldsymbol{w}^T\mathbf{x}_i)^2 + 1) \times 2 \times \boldsymbol{w}^T\mathbf{x}_i x_{(i)j}
\end{aligned}$$

So, we can write our update rule as follows:

$$w_j{}' = w_j - \eta\frac{\partial \mathrm{E}}{\partial w_j} = w_j - 2\eta\sum_{i=1}^{n}(o_i - t_i) \times \exp((\boldsymbol{w}^T\mathbf{x}_i)^2 + 1) \times \boldsymbol{w}^T\mathbf{x}_i x_{(i)j}$$

3. [4 pts] **Neural networks**

Consider a network with 4 inputs, a hidden layer with 2 units using a $ReLU$ activation function, and 2 output units with the softmax activation function.
Connection weights and biases are initialized as 1.
Consider a cross-entropy loss, and a stochastic gradient descent update for the training example:

$$\{\mathbf{x} = [1\ 1\ 1\ 1]^T, \mathbf{z} = [0\ 1]^T\}$$

a) [1v] Do forward propagation

$$\mathbf{W}^{[1]} = \begin{pmatrix}1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1\end{pmatrix},\quad \mathbf{b}^{[1]} = \begin{pmatrix}1\\1\end{pmatrix}, \mathbf{W}^{[2]} = \begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix},\quad \mathbf{b}^{[2]} = \begin{pmatrix}1\\1\end{pmatrix}$$

$$\mathbf{x}^{[1]} = ReLU\begin{pmatrix}5\\5\end{pmatrix} = \begin{pmatrix}5\\5\end{pmatrix},\qquad \mathbf{x}^{[2]} = softmax\begin{pmatrix}11\\11\end{pmatrix} = \begin{pmatrix}0.5\\0.5\end{pmatrix}$$

b) [1v] Compute $\delta^{[2]}$

$$\delta^{[2]} = \frac{\partial E}{\partial \mathbf{z}^{[2]}} = \frac{\partial}{\partial \mathbf{z}^{[2]}}\left(-\sum_{i=1}^{d}\mathbf{t}_i\, log\left(\mathbf{x}_i{}^{[2]}\right)\right) = \mathbf{x}^{[2]} - \mathbf{t} = \begin{pmatrix}0.5\\-0.5\end{pmatrix}$$

c) [1v] Compute $\delta^{[1]}$

$$\delta^{[1]} = \left(\frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{x}^{[1]}}\right)^T \cdot \delta^{[2]} \circ \frac{\partial \mathbf{x}^{[1]}}{\partial \mathbf{z}^{[1]}} = \mathrm{W}^{[2]T} \cdot \delta^{[2]} \circ \frac{\partial\, ReLU(\mathbf{z}^{[1]})}{\partial \mathbf{z}^{[1]}} = \begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix} \cdot \begin{pmatrix}0.5\\-0.5\end{pmatrix} \circ \begin{pmatrix}1\\1\end{pmatrix} = \begin{pmatrix}0\\0\end{pmatrix}$$

d) [1v] Update $\mathrm{W}^{[2]}$ using a learning rate of 0.1

$$\mathrm{W}^{[2]} = \mathrm{W}^{[2]} - \eta\frac{\partial E}{\partial \mathrm{W}^{[2]}} = \mathrm{W}^{[2]} - \eta\delta^{[2]}(\mathbf{x}^{[1]})^T = \begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix} - 0.1\begin{pmatrix}0.5\\-0.5\end{pmatrix}(5\quad 5) = \begin{pmatrix}0.75 & 0.75\\ 1.25 & 1.25\end{pmatrix}$$

4. [4 pts] **Local learning and PCA**

Consider the following observations in a Euclidean space:

	y_1	y_2	z
$\mathbf{x}_1$	3	1	0.1
$\mathbf{x}_2$	1	2	0.4
$\mathbf{x}_3$	3	3	0.3
$\mathbf{x}_4$	1	0	0.7

a) [2.5v] Estimate the quantity $z = f(\mathbf{x})$ where $\mathbf{x} = [1,1]^T$ and f is given by:

i. [1.25v] a kNN model with $k = 3$, a median estimator and uniform weights.

$3NN(\mathbf{x}) = \{\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_4}\},\ \hat{z} = median(0.1, 0.4, 0.7) = 0.4$

ii. [1.25v] a linear regression model with $\phi(\mathbf{x}) = \|\mathbf{x}\|_2{}^2$ and $\mathbf{w} = (-0.2,0.5)^T$

$\phi\begin{pmatrix}1\\1\end{pmatrix} = 2,\ \hat{z} = -0.2 + 0.5 \times 2 = 0.8$

b) [1.5v] The following covariance matrix and eigenvalues were produced for the given dataset:

$$C = \begin{pmatrix}1.333 & 0.667\\0.667 & 1.667\end{pmatrix},\quad \lambda_1 = 2.187,\quad \lambda_2 = 0.813$$

Project the input bivariate data space into a univariate data space by applying PCA with the most informative component.

$$C\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

Solving these simple equations yields $\mathbf{u}_1 = \begin{pmatrix}0.615\\0.788\end{pmatrix}$ and $\mathbf{u}_2 = \begin{pmatrix}-0.788\\0.615\end{pmatrix}$.

$$\Phi = (0.615 \quad 0.788)\begin{pmatrix}3 & 1 & 3 & 1\\1 & 2 & 3 & 0\end{pmatrix} = (2.634 \quad 2.192 \quad 4.211 \quad 0.615)$$

5. [4 pts] **RBFs and clustering**

Consider the following training set described by three vectors

$$\mathbf{x}_1 = \begin{pmatrix}1\\2\\0\\0\end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix}3\\2\\0\\0\end{pmatrix}, \mathbf{x}_3 = \begin{pmatrix}0\\0\\0\\2\end{pmatrix},$$

and the corresponding binary targets:

$$t_1 = 1, t_2 = 1, t_3 = -1$$

a) [1 pts] Identify the k-means clustering solution and number of iterations towards convergence assuming $k = 2$ and cluster centers initialized as

$$\boldsymbol{c}_1 = \begin{pmatrix}1\\2\\0\\0\end{pmatrix}, \boldsymbol{c}_2 = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

Hint: *no computation of distance values for k-Means is required!*

1st iteration: $C_1 = \{\mathbf{x}_1, \mathbf{x}_2\}, C_2 = \{\mathbf{x}_3\}$ with $\boldsymbol{c}_1 = \begin{pmatrix}2\\2\\0\\0\end{pmatrix}$ and $\boldsymbol{c}_2 = \begin{pmatrix}0\\0\\0\\2\end{pmatrix}$. 2nd iteration: convergence.

b) [3 pts] Determine the parameters of the RBF network. Use an RBF with k-Means clustering from (a) and $\sigma = 1$ for the clusters and one output unit implemented as the original Rosenblatt perceptron. Initialize all weights of the perceptron to -1 (including the bias). Use a learning rate of one for simplicity. Apply the perceptron learning algorithm (the original Rosenblatt model with sign()) for one epoch.

RBFs are centered on the centroids with $\sigma = 1$:

$$\phi_1 = exp\left(-\frac{||\mathbf{x} - \boldsymbol{c}_1||^2}{2}\right), \quad \phi_2 = exp\left(-\frac{||\mathbf{x} - \boldsymbol{c}_2||^2}{2}\right), \quad \Phi = \begin{pmatrix} 1 \\ \phi_1(\mathbf{x}) \\ \phi_2(\mathbf{x}) \end{pmatrix}$$

The hidden layer space with bias:

$$\Phi_1 = \begin{pmatrix} 1 \\ exp(-1/2) \\ exp(-9/2) \end{pmatrix} = \begin{pmatrix} 1 \\ 0.6065 \\ 0.0111 \end{pmatrix}, \Phi_2 = \begin{pmatrix} 1 \\ exp(-1/2) \\ exp(-17/2) \end{pmatrix} = \begin{pmatrix} 1 \\ 0.6065 \\ 0.0002 \end{pmatrix}, \Phi_3 = \begin{pmatrix} 1 \\ exp(-12/2) \\ exp(0) \end{pmatrix} = \begin{pmatrix} 1 \\ 0.0025 \\ 1 \end{pmatrix}$$

Using original perceptron rule, δ is either -2, 2, or 0:

$$o_1 = -1, \quad \delta_1 = 2; \quad \mathbf{w}^{new} = \boldsymbol{w}^{old} + \boldsymbol{\eta}(t_1 - o_1)\mathbf{x}_i = \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix} + 1 \cdot 2 \cdot \begin{pmatrix} 1 \\ 0.6065 \\ 0.0111 \end{pmatrix} = \begin{pmatrix} 1 \\ 0.213 \\ -0.9778 \end{pmatrix}$$

$$o_2 = +1, \quad \delta_2 = \ 0; \quad \text{no update}$$

$$o_3 = -1, \quad \delta_3 = -2; \mathbf{w}^{new} = \boldsymbol{w}^{old} + \boldsymbol{\eta}(t_3 - o_3)\mathbf{x}_i = \begin{pmatrix} 1 \\ 0.213 \\ -0.9778 \end{pmatrix} + 1 \cdot -2 \cdot \begin{pmatrix} 1 \\ 0.0025 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 0.2080 \\ -2.9778 \end{pmatrix}$$

6. [2 pts] **Model complexity**

Assuming a binary classification problem with inputs of dimension $d > 3$. Consider three models:

i. a fixed feature transformation $\phi: \mathbb{R}^d \to \mathbb{R}^6$ followed by a perceptron.

ii. a learnable feature transformation $\phi: \mathbb{R}^d \to \mathbb{R}^2$ that depends on 4 parameters, followed by a perceptron

iii. a multilayer perceptron with one hidden layer of size three (i.e. architecture $[d, 3, 2]$).

Order the ~~four~~ three models in terms of their expected risk of overfitting.

Justify each decision with a short sentence.

In model 1 the feature transformation is fixed, so it contributes with no parameters. The perceptron is applied after the transformation to 6 dimensional inputs. So in total, 6+1=7 parameters.

In model 2 the transformation requires the learning of 4 parameters. The perceptron is applied after the transformation to 2 dimensional inputs, i.e. 2+1=3 parameters. Adding both parts: 4+3=7 parameters.

In model 3 we have the following parameter matrices: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}$. The total number of parameters is $d \times 3 + 3 \times 1 + 2 \times 3 + 2 \times 1 = 3d + 3 + 6 + 2 = 3d + 11$.

Regardless of the value of d, model 3 will have a higher number of free parameters than the other two.

With that said, the risk of overfitting is larger for more complex models. Models are more complex if they have a larger VC dimension, which can be approximated by the number of parameters.

Models 1 and 2 are at an equivalent risk and model 2 is at a higher risk of overfitting.

END