Assignment 1 : CS-E4830 Kernel Methods in Machine Learning 2020

The **deadline** for this assignment is **Thursday 06.02.2020 at 4pm**. If you have questions about the assignment, you can ask them in the 'General discussion' section on MyCourses. We will have a tutorial session regarding this assignment on 30.01.20 at 4:15 pm in TU1(1017), TUAS, Maarintie 8 (**check room**).

Please follow the **submission instructions** given in MyCourses: `https://mycourses.aalto.fi/course/view.php?id=24366&section=4`.

Pen & Paper exercise (12 points in total)

Question 1 (2 points): Recall from Lecture 1, the form for the polynomial kernel

$$K_1(x, y) = (\langle x, y\rangle + c)^m$$

where $c \geq 0$, m is a positive integer and $x, y \in \mathbb{R}^d$.

- Prove that $K_1(x, y)$ as defined above is a valid kernel

Solution

- Given $K_1(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y}\rangle + c)^m$ given that $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.
- Using Binomial Theorem : $(\langle \mathbf{x}, \mathbf{y}\rangle)^m$ can be expanded as follows:

$$\binom{m}{0}(\langle \mathbf{x}, \mathbf{y}\rangle)^m c^0 + \binom{m}{1}(\langle \mathbf{x}, \mathbf{y}\rangle)^{m-1} c^1 + \ldots + \binom{m}{m}(\langle \mathbf{x}, \mathbf{y}\rangle)^0 c^m$$

- Now, $\langle \mathbf{x}, \mathbf{y}\rangle$ is an inner product in $\mathbb{R}^d$, hence it is a kernel.
- Since product of kernels is a kernel, therefore $(\langle \mathbf{x}, \mathbf{y}\rangle)^2$, and $(\langle \mathbf{x}, \mathbf{y}\rangle)^3$ and so on are also kernels
- Furthermore, multiplying a kernel by a positive number is also a kernel
- Therefore, all the terms (except the last term c^m which is a constant), and hence their sum is a kernel. Call it $K_s(\mathbf{x}, \mathbf{y})$, and its corresponding feature map be $\phi(.)$ such that $K_s(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y})\rangle$
- Call the last term α, therefore we have $K_1(\mathbf{x}, \mathbf{y}) = K_s(\mathbf{x}, \mathbf{y}) + \alpha = \langle \phi(\mathbf{x}), \phi(\mathbf{y})\rangle + \alpha$
- Therefore, $K(., .)$ can be expressed as an inner product with the following feature map $\begin{bmatrix} \phi(\mathbf{x}) \\ \sqrt{\alpha} \end{bmatrix}$.
- Hence $K_1(., .)$ is a kernel.

Question 2 (4 points) Recall from lecture 2, in the context of binary classification, the Parzen window classifier assigns a test instance x based on the distance to the centroids in the following way :

$$h(x) = \begin{cases} +1 \text{ if } ||\phi(x) - c_-||^2 > ||\phi(x) - c_+||^2 \\ -1 \text{ otherwise.} \end{cases}$$

where c_- and c_+ represent the centroids in the feature space of the negative and positive classes respectively. Show by deriving appropriate expressions for α_i and b, that the above decision function can be written in the following form $h(x) = \text{sgn}\left(\sum_{i=1}^n \alpha_i k(x, x_i) + b\right)$ such that $k(x, x_i) = \langle \phi(x), \phi(x_i) \rangle$. Here sgn(.) represents the sign function, and n is the total number of training samples.

Solution

- We can rewrite $h(x) = \text{sgn}\left(\|\phi(x) - c_-\|^2 - \|\phi(x) - c_+\|^2\right)$
- using $\|x\|^2 = \langle x, x \rangle$ we have $h(x) = \text{sgn}\left(\langle \phi(x) - c_-, \phi(x) - c_- \rangle - \langle \phi(x) - c_+, \phi(x) - c_+ \rangle\right)$
- Now expanding the terms inside the sgn(.) using properties $\langle x, y+z \rangle = \langle x, y \rangle + \langle x, z \rangle$ we have

  $$\langle \phi(x) - c_-, \phi(x) - c_- \rangle - \langle \phi(x) - c_+, \phi(x) - c_+ \rangle = 2\langle \phi(x), c_+ \rangle - 2\langle \phi(x), c_- \rangle + \langle c_-, c_- \rangle - \langle c_+, c_+ \rangle$$

- As sgn(.) is unaffected by a positive multiple we can write

  $$h(x) = \text{sgn}\left(\langle \phi(x), c_+ \rangle - \langle \phi(x), c_- \rangle + 0.5\langle c_-, c_- \rangle - 0.5\langle c_+, c_+ \rangle\right)$$

- Now using the definitions of c_+ and c_- we can expand each term using properties of inner products as follows

  $$\begin{aligned} \langle \phi(x), c_+ \rangle &= \left\langle \phi(x), \frac{1}{m_+} \sum_{i \in \mathcal{I}^+} \phi(x_i) \right\rangle \\ &= \frac{1}{m_+} \left\langle \phi(x), \sum_{i \in \mathcal{I}^+} \phi(x_i) \right\rangle && \because \langle ax, y \rangle = a\langle x, y \rangle \\ &= \frac{1}{m_+} \sum_{i \in \mathcal{I}^+} \langle \phi(x), \phi(x_i) \rangle && \because \langle x, y+z \rangle = \langle x, y \rangle + \langle x, z \rangle \\ &= \frac{1}{m_+} \sum_{i \in \mathcal{I}^+} k(x, x_i) \end{aligned}$$

  Similarly $\langle \phi(x), c_- \rangle = \frac{1}{m_-} \sum_{i \in \mathcal{I}^-} k(x, x_i)$

- A similar process for $\langle c_-, c_- \rangle$

$$\begin{aligned}
\langle c_-, c_- \rangle &= \left\langle \frac{1}{m_-} \sum_{i \in \mathcal{I}^-} \phi(x_i), \frac{1}{m_-} \sum_{j \in \mathcal{I}^-} \phi(x_j) \right\rangle \\
&= \frac{1}{m_-^2} \left\langle \sum_{i \in \mathcal{I}^-} \phi(x_i), \sum_{j \in \mathcal{I}^-} \phi(x_j) \right\rangle \\
&= \frac{1}{m_-^2} \sum_{i \in \mathcal{I}^-} \sum_{j \in \mathcal{I}^-} \langle \phi(x_i), \phi(x_j) \rangle \\
&= \frac{1}{m_-^2} \sum_{i,j \in \mathcal{I}^-} k(x_i, x_j)
\end{aligned}$$

  Similarly $\langle c_+, c_+ \rangle = \frac{1}{m_+^2} \sum_{i,j \in \mathcal{I}^+} k(x_i, x_j)$

- We can now take $b = 0.5(\langle c_-, c_- \rangle - \langle c_+, c_+ \rangle)$, therefore

$$b = \frac{1}{2m_-^2} \sum_{i,j \in \mathcal{I}^-} k(x_i, x_j) - \frac{1}{2m_+^2} \sum_{i,j \in \mathcal{I}^+} k(x_i, x_j)$$

- Now combining all previous results we have

$$h(x) = \operatorname{sgn} \left(\frac{1}{m_+} \sum_{i \in \mathcal{I}^+} k(x, x_i) - \frac{1}{m_-} \sum_{i \in \mathcal{I}^-} k(x, x_i) + b \right)$$

- As $\mathcal{I} = \mathcal{I}^+ \cup \mathcal{I}^-$ and $n = m_+ + m_-$ we can combine the integrals by introducing a coefficient term α_i like

$$h(x) = \operatorname{sgn} \left(\sum_{i=1}^{n} \alpha_i k(x, x_i) + b \right)$$

  Where

$$\alpha_i = \begin{cases} \frac{1}{m_+}, & y_i = +1 \\ \frac{-1}{m_-}, & y_i = -1 \end{cases}$$

Hence derived

Question 3 (3 points) For $x, y \in \mathbb{R}$, check if $K_2(x, y) = \cos(x + y)$ is a valid kernel function.

Solution : Recall from Lecture 2 that a kernel function need to be positive definite. A symmetric function $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is positive definite if $\forall n \geq 1, \forall (a_1, \dots, a_n) \in \mathbb{R}^n, \forall (x_1, \dots, x_n) \in \mathcal{X}^n$,

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j k(x_i, x_j) \geq 0$$

The above should hold even for a single point, i.e., $a^2 k(x, x) \geq 0$ for all $x \in \mathcal{X}$ and $a \in \mathbb{R}$

Now, take $x = \frac{\pi}{2}$ and $a = 1$. In this case, it is $1^2 \times \cos\left(\frac{\pi}{2} + \frac{\pi}{2}\right) = -1 < 0$. Hence it is not a kernel. Of course, there could be many other counter-examples.

Question 4 (3 points) For $x, y \in \mathcal{X} = (-1, 1)$, prove that $K_3(x, y) = \frac{1}{1-xy}$ is a valid kernel

Solution : Since $x, y \in \mathcal{X} = (-1, 1)$, therefore, $K_3(x, y) = \frac{1}{1-xy} = 1 + xy + (xy)^2 + (xy)^3 + \dots$. Each of individual terms (apart from 1) is a kernel, either by definition or product of kernels is a kernel property. Hence the sum of kernels is also kernel. The sum of a kernel and a constant (the first term 1) is also a kernel as well. Hence $K_3(x, y)$ is a kernel.

Computer Exercise (8 points in total)

Solve the computer exercise in JupyterHub (`https://jupyter.cs.aalto.fi`). The instructions for that are given in MyCourses: `https://mycourses.aalto.fi/course/view.php?id=20602&section=3`.

Gaussian-Kernel using matrix operations (*vectorization*)

The Gaussian-Kernel between two d-dimensional vectors $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ is defined as:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{1}$$

$$\tag{2}$$

with $\sigma > 0$.

Note that:

- $\|\mathbf{z}\|^2 = \sum_{k=1}^{d} \mathbf{z}_k = \mathbf{z}^T \mathbf{z}$
- In the following we assume $i \in \{1, \dots, n_A\} = \mathcal{I}_A$ and $j \in \{1, \dots, n_B\} = \mathcal{I}_B$, where n_A, n_B are the number of feature vectors $(\mathbf{x}_i, \mathbf{x}_j)$ in two sample sets A and B. Those vectors are stored *row-wise* in the matrices $\mathbf{X}_A \in \mathbb{R}^{n_A \times d}$ and $\mathbf{X}_B \in \mathbb{R}^{n_B \times d}$.

So lets start:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{3}$$
$$= \exp\left(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma^2}\right) \tag{4}$$
$$= \exp\left(-\frac{\mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\mathbf{x}_j + \mathbf{x}_j^T\mathbf{x}_j}{2\sigma^2}\right) \tag{5}$$

- $\mathbf{x}_i^T\mathbf{x}_i \quad \forall_{i\in\mathcal{I}_A}$: Diagonal of a linear kernel between all examples of set A:

  $$\mathbf{z}_A = diag(\mathbf{X}_A\mathbf{X}_A^T) = (\mathbf{X}_A \circ \mathbf{X}_A)^T\mathbf{1} \in \mathbf{R}^{n_A\times 1}, \tag{6}$$

  where $\mathbf{1} = [1, 1, \ldots, 1]^T \in \{1\}^d$ vector of ones, and $\circ$ is the Hadamard product. In Python using NumPy we can write:

  ```
  z_A = np.sum(X_A * X_A, axis=1) # shape = (n_A,)
  z_A = z_A[:, np.newaxis]        # shape = (n_A, 1)
  ```

- $\mathbf{x}_j^T\mathbf{x}_j \quad \forall_{j\in\mathcal{I}_B}$, same as above, but for sample set B.
- $\mathbf{x}_i^T\mathbf{x}_j \quad \forall_{(i,j)\in\mathcal{I}_A\times\mathcal{I}_B}$: Linear kernel between the examples of set A and B:

  $$\mathbf{X}_A\mathbf{X}_B^T \in \mathbb{R}^{n_A\times n_B}. \tag{7}$$

  In Python ($\geq$ 3.6) using NumPy ($\geq$ 1.10):

  ```
  XX_AB = X_A @ X_B.T
  ```

We can now calculate the values $\forall_{(i,j)\in\mathcal{I}_A\times\mathcal{I}_B}$ of the difference norm (nominator) as follows:

$$\Delta = \underbrace{\mathbf{z}_A\mathbf{1}_{n_B}^T}_{(n_A\times 1)(1\times n_B)=(n_A\times n_B)} - 2 * \mathbf{X}_A\mathbf{X}_B^T + \underbrace{\mathbf{1}_{n_A}\mathbf{z}_B^T}_{(n_A\times 1)(1\times n_B)=(n_A\times n_B)} \tag{8}$$

Subsequently we can calculate:

$$\mathbf{K}_{gauss} = \exp\left(-\frac{\Delta}{2\sigma^2}\right), \tag{9}$$

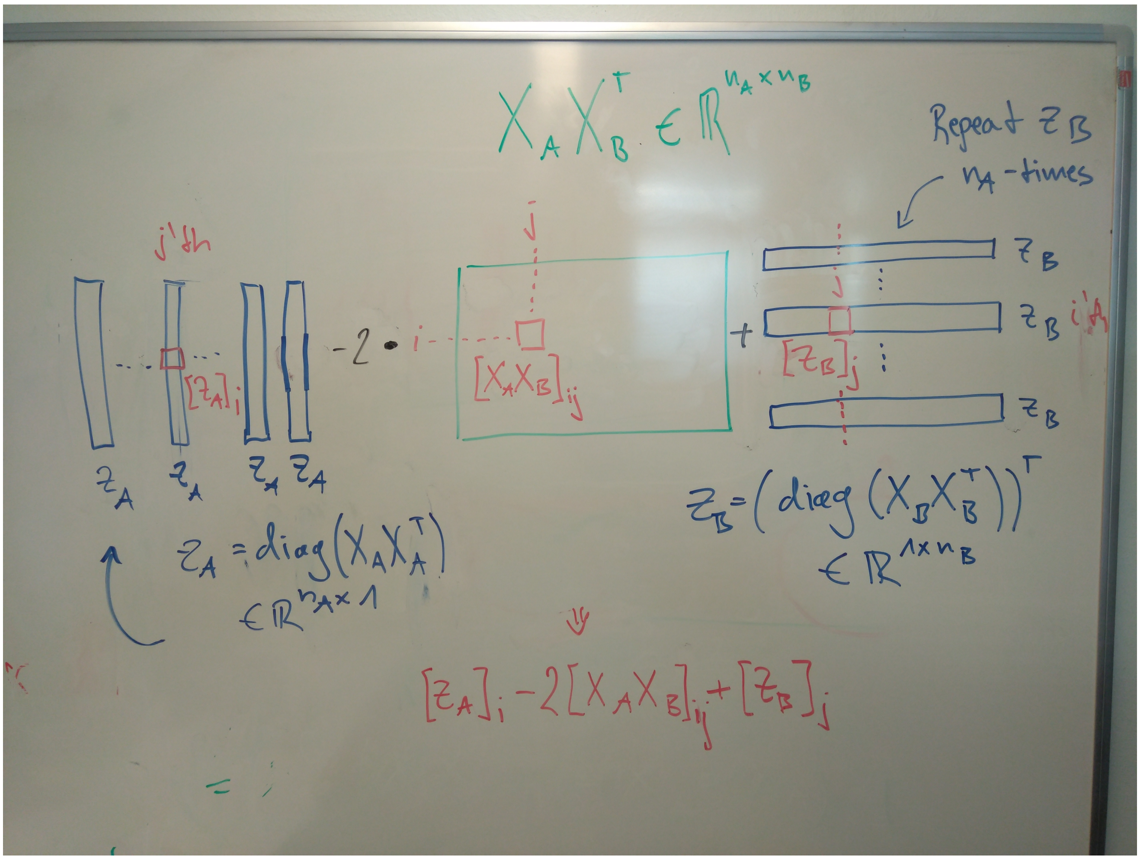
assuming that the exp function operates element-wise.

```
DELTA = z_A @ np.ones((1, n_B)) - 2 * XX_AB + np.ones((n_A, 1)) @ z_B.T
DELTA = DELTA / (-2 * sigma**2)
K_gauss = np.exp(DELTA)
```

If we make use of the broadcasting rules in NumPy, we can even more simply the expression for Δ:

```
DELTA = z_A - 2 * XX_AB + z_B
...
```

However, you should **really check what you do (and read the documentation)**, i.e. make `z_A` and `z_B` explicitly vectors.



Parzen Window Classifier

Here some notes on the implementation:

Positive and negative training examples: In a *supervised* machine learning task, we are given as set of training tuples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, each containing a feature vector $\mathbf{x}_i \in \mathbf{R}^d$ and a corresponding label y_i. In the case of a binary classification task we typically have $y_i \in \{0, 1\}$ or $y_i \in \{-1, 1\}$. Here, an example i belongs to the positive class if its label y_i is positive, i.e. $y_i == 1$. The negative class examples can be defined equally.

Calculating the bias-term:

$$
\begin{aligned}
b &= \frac{1}{2n_-^2} \sum_{i,j \in I^-} \kappa(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{2n_+^2} \sum_{i,j \in I^+} \kappa(\mathbf{x}_i, \mathbf{x}_j) && (10) \\
&= \frac{1}{2n_-^2} \sum_{i \in I^-} \sum_{j \in I^-} \kappa(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{2n_+^2} \sum_{i \in I^+} \sum_{j \in I^+} \kappa(\mathbf{x}_i, \mathbf{x}_j) && (11) \\
&= b_- - b_+, && (12)
\end{aligned}
$$

where the separate bias-terms can be expressed in Python using NumPy as follows:

- b_-: `b_n = np.sum(KX_train[I_n][:, I_n]) / (2. * n_n ** 2)`
- b_+: `b_p = np.sum(KX_train[I_p][:, I_p]) / (2. * n_p ** 2)`
- b: `self.b = b_n - b_p`

Calculating the decision function $g(\mathbf{x})$: Let $\mathbf{x}$ be a *new* examples, i.e. it has not been used for training:

$$g(\mathbf{x}) = \sum_{i=1}^{n_{\text{train}}} \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) + b \tag{13}$$

$$= \mathbf{k}(\mathbf{x})\boldsymbol{\alpha} + b, \tag{14}$$

where:

- $\mathbf{k}(\mathbf{x}) = [\kappa(\mathbf{x}, \mathbf{x}_1), \kappa(\mathbf{x}, \mathbf{x}_2), \ldots, \kappa(\mathbf{x}, \mathbf{x}_n)] \in \mathbb{R}^{1 \times n_{\text{train}}}$: Single row of the test-train kernel matrix: $\mathbf{K}_{\text{test vs. train}} \in \mathbb{R}^{n_{\text{test}} \times n_{\text{train}}}$. In Python:

```
# Lets predict for test example x_s
g_xs = KX_test_train[s, :] @ self.alphas + self.b  # shape = (1,)

# Lets predict for all test examples at ones       # shape = (n_test, 1)
g_X = KX_test_train @ self.alphas + self.b
```

- $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T \mathbf{R}^n$: Dual variables

Optional task: Inspect classifier's decision boundary

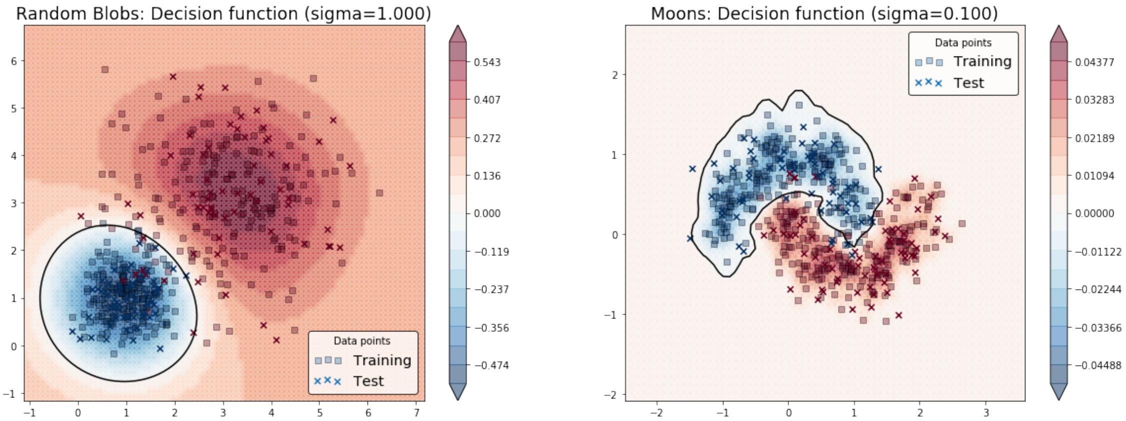


Figure 1: Gaussian kernel

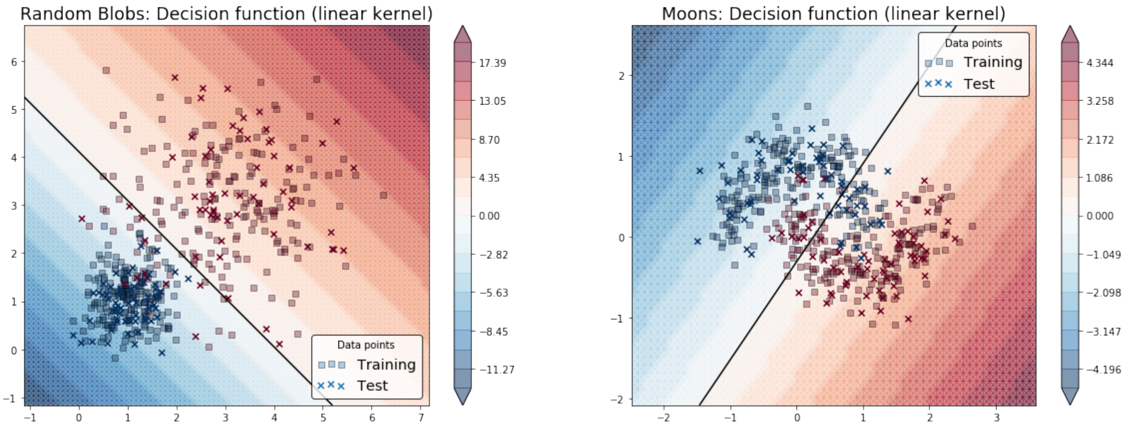


Figure 2: Linear kernel