ML HW3

Atul Parida
Discussed with Jash Jignesh Veragiwala (1006185) and Anutham Mukunthan (1006202).

1006184

Q1) The answer is provided in the Jupyter notebook attached inside the zip folder. Please maintain the folder contents for the code to correctly run.

Q2)

Properties of a valid kernel:
1. Kernel matrix is positive semidefinite.
2. Kernel matrix is symmetric.

Given : $\mathrm{K}_1(\mathrm{x}, \mathrm{z})$ and $\mathrm{K}_2(\mathrm{x}, \mathrm{z})$ are valid kernels

1. K(x, z) = K_1 (x, z) K_2(x, z)

If K_1(x, z) and K_2(x, z) are valid kernels, then their product K(x, z) = K_1(x, z) * K_2(x, z) is also a valid kernel. This holds true because the product of two positive semidefinite and symmetric matrices is another positive semidefinite and symmetric matrix.

2. K(x, z) = aK_1 (x, z) + bK_2 (x, z), where a, b > 0 are real numbers.

If a and b are positive real numbers, and K_1(x, z) and K_2(x, z) are valid kernels, then the combination K(x, z) = a * K_1(x, z) + b * K_2(x, z) is a valid kernel as well. This is because a linear combination of positive semidefinite matrices with positive coefficients maintains positive semidefiniteness.

3. K(x, z) = aK_1 (x, z) - bK_2 (x, z), where a, b > 0 are real numbers.

This kernel may not necessarily be valid.

Let K_1 (x, z) = x^T * z and K2(x, z) = (x^T * z)^2. Both K_1 and K_2 are valid linear and polynomial kernels.

If a, b = 2, 1, and x and z be a row and column vector respectively:

K(x, z) = 2K_1 (x, z) - K_2(x, z) = 2(x^T * z) - (x^T * z)^2 = 2(0) - (1)^2 = -1.

The eigenvalues may be negative, and hence the kernels may not be positive semidefinite. Hence, the corresponding kernel may not be a valid kernel.

4. K(x, z) = f(x)f(z), where f : Rn → R is any real-valued function of x

The expression K(x, z) = f(x) * f(z) can also represent a valid kernel, where f: R^n → R is any real-valued function of the input features. This is valid because we can find a feature map ф(x) such that K(x, z) = ф(x)^T ф(z), which satisfies the Mercer's condition for valid kernels.1. K(x, z) = K_1 (x, z) K_2 (x, z) is a valid kernel.

Q3) The answer is provided in the Jupyter notebook attached inside the zip folder. Please maintain the folder contents for the code to correctly run.

Q4)

To derive the SGD weight update rule, partial derivatives of error function w.r.t. each weight parameter is needed and weight must be updated in direction opposite to gradient.

Output equation:

$$y = w_o + \sum_{i+1}^{n} (w_i * x_i + w_i * {x_i}^2)$$

Error function:

$$E_d = \frac{1}{2} * \sum_{j} (y_j - y_{j*})^2$$

$$y_{j*} = w_o + \sum_{\mathrm{i}=1}^{\mathrm{n}} w_{\mathrm{i}} * x_{\mathrm{i}} + w_{\mathrm{i}} * {x_{\mathrm{i}}}^2$$

$$E_d = \frac{1}{2} * \sum_{j} (y_j - (w_o + \sum_{\mathrm{i}=1}^{\mathrm{n}} w_{\mathrm{i}} * x_{\mathrm{i}} + w_{\mathrm{i}} * {x_{\mathrm{i}}}^2))^2$$

Partial derivative:

$$\frac{\partial E_d}{\partial w_{\mathrm{i}}} = \frac{1}{2} * \nabla w \sum_{j} (y_j - (w_o + \sum_{\mathrm{i}=1}^{\mathrm{n}} w_{\mathrm{i}} * x_{\mathrm{i}} + w_{\mathrm{i}} * {x_{\mathrm{i}}}^2))^2$$

As dy/dw = 0 and dw0/dw = 0:

$$\frac{\partial E_d}{\partial w_{\mathrm{i}}} = \left(y_j - w_o - \sum_{\mathrm{i}=1}^{\mathrm{n}} w_{\mathrm{i}} * x_{\mathrm{i}} - w_{\mathrm{i}} * {x_{\mathrm{i}}}^2\right) * \left(\sum_{\mathrm{i}=1}^{\mathrm{n}} - x_{\mathrm{i}} - {x_{\mathrm{i}}}^2\right)$$

$$so, {\mathrm{w}_{\mathrm{i}}}^{\mathrm{k}+1} = {\mathrm{w}_{\mathrm{i}}}^{\mathrm{k}} - \eta_k * \frac{\partial E_d}{\partial w_{\mathrm{i}}}$$

$${\mathrm{w}_{\mathrm{i}}}^{\mathrm{k}+1} = \mathrm{w}^k + \eta_k * (y_j - w_o - \sum_{\mathrm{i}=1}^{\mathrm{n}} w_{\mathrm{i}} * x_{\mathrm{i}} - w_{\mathrm{i}} * {x_{\mathrm{i}}}^2) * (\sum_{\mathrm{i}=1}^{\mathrm{n}} - x_{\mathrm{i}} - {x_{\mathrm{i}}}^2)$$

$${\mathrm{w}_{\mathrm{i}}}^{\mathrm{k}+1} = \mathrm{w}^k + \eta_k * (y_j - y_{j*}) * (\sum_{\mathrm{i}=1}^{\mathrm{n}} - x_{\mathrm{i}} - {x_{\mathrm{i}}}^2)$$

Update rule for w_{o}:

$$\frac{\partial E_d}{\partial w_{\mathrm{o}}} = \frac{\partial (y_j - y_{j*})^2}{2 * \partial w_{\mathrm{o}}}$$

$$\frac{\partial E_d}{\partial w_{\mathrm{o}}} = \frac{\partial (y_j - (w_o + \sum_{\mathrm{i}=1}^{\mathrm{n}} w_{\mathrm{i}} * x_{\mathrm{i}} + w_{\mathrm{i}} * {x_{\mathrm{i}}}^2))^2}{2 * \partial w_{\mathrm{o}}} = y_{j*} - y_j$$

$${\mathrm{w}_{\mathrm{o}}}^{\mathrm{k}+1} = {\mathrm{w}_{\mathrm{o}}}^{\mathrm{k}} - \eta_k * \frac{\partial E_d}{\partial w_{\mathrm{o}}}$$

$${\mathrm{w}_{\mathrm{o}}}^{\mathrm{k}+1} = {\mathrm{w}_{\mathrm{o}}}^{\mathrm{k}} - \eta_k * (y_{j*} - y_j)$$

Q5)

In a Naive Bayes classifier, we make the assumption that the features are conditionally independent given the class label. This means that we treat the probability of a particular combination of features as the product of the probabilities of each feature occurring independently. For a problem with 2 features (x1 and x2), there are 4 possible combinations of features (00, 01, 10, and 11). The Naive Bayes classifier would require estimating 4 parameters to represent the probability of each feature combination occurring for each class label.

To generalize, for a problem with n features (x1, x2, ..., xn), the Naive Bayes classifier would need to estimate 4n parameters. These parameters correspond to the probabilities of each feature occurring for each class label. So, in total, the classifier would have 4n + 1 parameters, considering the 4*n possible combinations of features and one additional parameter for the probability of the class label.

Number of features	Number of parameters
2	9
3	13
4	17