Machine Learning (Homework 3)

Due date : 12/29

1 Gaussian Process for Regression (40%)

In this exercise, you will implement Gaussian process for regression. You are given a file 2_data.mat, the same dataset which we use in Bayesian linear regression, which contains two vectors: $\mathbf{x}: \{x_1, x_2, \ldots, x_{100}\},\ 0 \leq x_i \leq 2$, $\mathbf{t}: \{t_1, t_2, \ldots, t_{100}\}$ where $\mathbf{x}$ represents the input data, $\mathbf{t}$ represents the corresponding target data. There are $N = 100$ data samples. Let us split them into training set (the first 60 points) and test set (the last 40 points). A regression function $y(\cdot)$ is used to express the target value by

$$t_n = y(x_n) + \epsilon_n$$

where the noisy signal ϵ_n is Gaussian distributed $\epsilon_n \sim \mathcal{N}(0, \beta^{-1})$ with $\beta^{-1} = 1$.

1. Please implement the Gaussian process by using the exponential-quadratic kernel function given by

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left\{-\frac{\theta_1}{2}\|\mathbf{x}_n - \mathbf{x}_m\|^2\right\} + \theta_2 + \theta_3 \mathbf{x}_n^\top \mathbf{x}_m$$

where the hyperparameters $\boldsymbol{\theta} = \{\theta_0, \theta_1, \theta_2, \theta_3\}$ are fixed. Please use the training set and use four different combinations:

- squared exponential kernel $\boldsymbol{\theta} = \{1, 4, 0, 0\}$
- linear kernel $\boldsymbol{\theta} = \{0, 0, 0, 1\}$
- exponential-quadratic kernel $\boldsymbol{\theta} = \{1, 4, 0, 5\}$
- exponential-quadratic kernel $\boldsymbol{\theta} = \{1, 64, 10, 0\}$

   i. Please plot the prediction result like Figure 6.8 of textbook for training set but one standard deviation instead of two and without the green curve. An example is provided in Figure 1. Title of this figure denotes the value of the hyperparameters used in this model. The red line shows the mean $m(\cdot)$ of the Gaussian process predictive distribution. The pink region corresponds to plus and minus one standard deviation. Training data points are shown in blue.

   ii. Show the corresponding root-mean-square errors

$$E_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\left(m(x_n) - t_n\right)^2}$$

for both training and test sets in a form of

$\{\theta_0, \theta_1, \theta_2, \theta_3\}$	train	test
$\{1, 1, 1, 1\}$	1.6016	1.7649

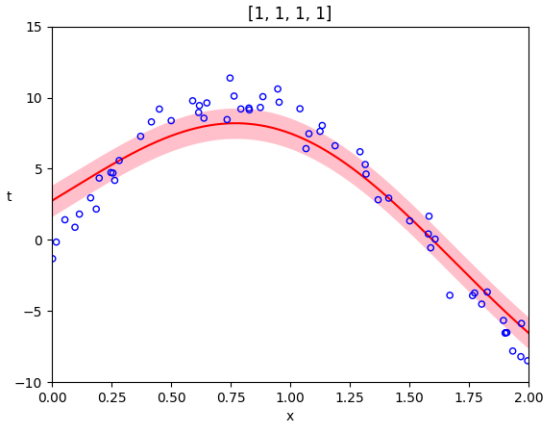


Figure 1: An example of Gaussian process for regression.

iii. Explain your findings.

2. The automatic relevance determination (ARD) framework is developed by considering the following exponential-quadratic kernel

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left\{-\frac{1}{2}\sum_{i=1}^{D}\eta_i(x_{ni} - x_{mi})^2\right\} + \theta_2 + \theta_3 \mathbf{x}_n^\top \mathbf{x}_m. \tag{1}$$

Now we adopt the adjustable hyperparameters $\boldsymbol{\theta} = \{\theta_0, \eta_1, \theta_2, \theta_3\}$ and use the initial values $\{\theta_0, \eta_1, \theta_2, \theta_3\} = \{3, 6, 4, 5\}$. Use the training set to estimate the optimal hyperparameters $\boldsymbol{\theta}$ by applying the gradient decent algorithm through maximizing the log likelihood function

$$\boldsymbol{\theta}^{(\tau+1)} = \boldsymbol{\theta}^{(\tau)} + \eta \frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{t}|\boldsymbol{\theta})$$

where η denotes the learning rate. You can also use the other optimization algorithm like Newton-Raphson or Adam optimization algorithms. Please maximize them until all the absolute values of elements of $\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{t}|\boldsymbol{\theta})$ are smaller than 6.

i. Derive the formula for $\frac{\partial}{\partial \boldsymbol{\theta}} C(\mathbf{x}_n, \mathbf{x}_m)$ using $C(\mathbf{x}_n, \mathbf{x}_m)$ given in Eq. (6.62) of textbook and the kernel function defined in Eq. (1).

ii. Show the values of the hyperparameters as a function of iterations like Figure 6.10 of textbook with normal scale instead of logarithm one and show the final optimal values of the hyperparameters.

iii. Use the optimal hyperparameters to plot the prediction result similar to that in problem 1.

iv. Show the corresponding root-mean-square errors for both training and test set.

v. Compare with the result obtained by using Bayesian linear regression. In this comparison, use the first 60 points for training.

2 Support Vector Machine (SVM) (30%)

Support vector machines (SVM) is known as a popular method for pattern classification. In this exercise, you will implement SVM for classification. You are given the Iris dataset which contains the measurements of 150 specimens from each of three different species of iris—setosa (1), versicolor (2), and virginica (3) —with the following four features: sepal length, sepal width,

petal length and petal width. In the training procedure of SVM, we need to optimize with respect to the Lagrange multiplier $\alpha = \{\alpha_n\}$. Here, we use the sequential minimal optimization to solve the problem. For details, you can refer to the paper [Platt, John. "Sequential minimal optimization: A fast algorithm for training support vector machines." (1998)]. For matlab, we provide smo.m for you to get the multipliers (coefficients). You are required to use this result to implement the rest of the SVM algorithms. As for Python, scikit-learn is a free software machine learning library, and it introduces the sklearn.svm to fit the training data. You are allowed to use the library to get the multipliers (coefficients) instead of using the predict function directly. SVM is binary classifier, but the application here has three classes. To solve this problem, there are two approaches including

- One-versus-the-rest approach constructs a classifier distinguishing between one class and the remaining. Predict the label by finding the corresponding classifier which reports the highest confidence score.
- One-versus-one approach constructs a classifier distinguishing between two classes. Predict the label by voting. All classifiers vote for one class, finally one class which gets the highest number gets predicted.

In this exercise, you will implement two kinds of kernel SVM
SVM :

$$y(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n t_n k(\mathbf{x}, \mathbf{x}_n) = \mathbf{w}^\top \phi(\mathbf{x}) + b$$

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n t_n \phi(\mathbf{x}_n)$$

Linear kernel :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$$

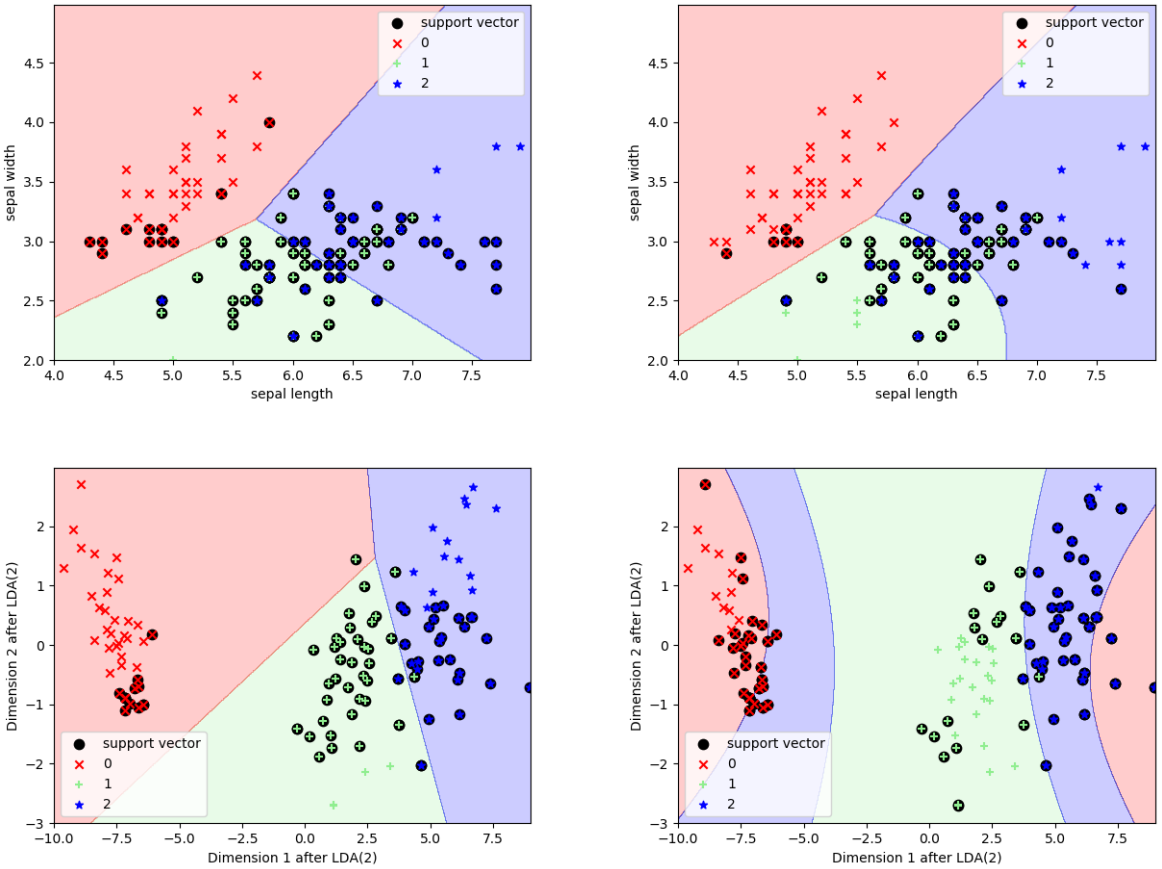
Polynomial (homogeneous) kernel of degree 2:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^2$$
$$\phi(\mathbf{x}) = [x_1^2, \sqrt{2}x_1x_2, x_2^2]^\top$$
$$\mathbf{x} = [x_1, x_2]^\top$$

1. Use the first two features of dataset and build a SVM with linear kernel to do multi-class classification. Then plot the corresponding decision boundary and support vectors.
2. Repeat (1) with polynomial kernel (degree = 2).
3. Reduce the dimension of dataset from $\mathbb{R}^4$ to $\mathbb{R}^2$ by linear discriminant analysis (LDA) and build a SVM with linear kernel to do multi-class classification. Then plot the corresponding decision boundary and support vectors.
4. Repeat (3) with polynomial kernel (degree = 2).
5. Please discuss the influence of dimension reduction and the difference between (1), (3), and strategies of choosing the most contributive features in previous homework.

Hints

- In this exercise, we strongly recommend using matlab to avoid tedious preprocessing occurred in using Python.
- If you use other languages, you are allowed to use toolbox only for multipliers (coefficients).
- You need to implement the whole algorithms except for multipliers (coefficients).



3 Gaussian Mixture Model (30%)

In this exercise, you will implement image segmentation using Gaussian Mixture Model (GMM) and Expectation-Maximization (EM) Algorithm (see textbook p.438-p.439). Given an image "hw3_img.jpg" as input data. You should first build and train a GMM model which is initialed by K-means algorithm which finds K central pixels. Second, use EM algorithm to find the optimized parameters for GMM. According to the maximum likelihood, you can decide the color $\mu_k, k \in [1, \ldots, K]$ of each pixel x_n of output image

1. Please build a K-means model by minimizing

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \|x_n - \mu_k\|^2$$

   and show the table of estimated $\{\mu_k\}_{k=1}^{K}$.

2. Use $\{\mu_k\}_{k=1}^{K}$ calculated by K-means model and calculate the corresponding variance σ_k^2, and mixing coefficient π_k. Use these parameters to initialize a GMM model $p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \sigma_k^2)$ and represent the data by maximizing the log likelihood function $\log p(x|\pi, \mu, \sigma^2)$ through EM algorithm. Please plot the log likelihood curve of GMM. (Terminate your EM algorithm when iteration arrives 100)

3. Repeat step (1) and (2) for the cases of K = 2, 3, 5 and 20. Please show the resulting images in your report. Below are some examples.

Hints

- You should normalize 3 channels (R, G, B) of the image. In other words, the original values loaded into the matrix are in range $0 \sim 255$, after normalization, they will be in range $0 \sim 1$
- If an error of exceeding memory requested occurs when you are loading image to an array, we recommend you to resize the input image to 10%.