Machine Learning (Homework 3)

Due date : 1/4

1 Gaussian Process (60%)

In this exercise, you will implement the Gaussin process (GP) for regression. The data file gp.csv contains the input data $\mathbf{x} : \{x_1, x_2, \ldots, x_{120}\}, 0 \leq x_i \leq 2$ and the corresponding target data $\mathbf{t} : \{t_1, t_2, \ldots, t_{120}\}$. There are $N = 120$ data samples. Please use the first 60 samples as the training data and the remaining samples as the test data. 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 by $\epsilon_n \sim \mathcal{N}(0, \beta^{-1})$ with $\beta^{-1} = 1$

1. Please implement the Gaussian process based on an 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 apply 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 $\boldsymbol{\theta} = \{1, 64, 10, 0\}$

2. 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. The title of figure should be the value of the hyperparameters used in this model. The red line shows the mean $m(\cdot)$ of the Gaussian process predictive distributioin. The pink region corresponds to plus and minus one standard deviation. Training data points are shown in blue.

3. Show the corresponding root-mean-square errors

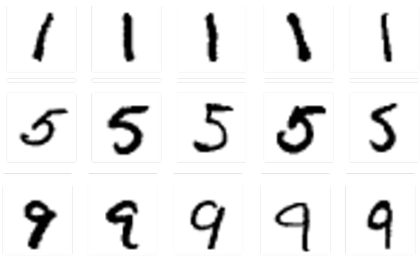
$$E_{\text{RMS}} = \sqrt{\frac{1}{N}(m(x_n) - t_n)^2}$$

   for both training and test sets with respect to four kernels.

4. Explain your findings.

2 Support Vector Machine (40%)

Support vector machines (SVM) is known as a popular method for pattern classification. In this exercise, you will implement the SVM for classification. Here, the MNIST dataset is given in x_train.csv and t_train.csv. The input data is the first two scaled principal components of images of handwritten digits from MNIST dataset which contains three digits: digit 1, digit 4 and digit 9. The dynamic range of values is [0, 1].



Data Description

- **x_train** is a 150×2 matrix where each row is the first two scaled principal values of a training image.
- **t_train** is a 150×1 matrix which records the classes of the training images. 1, 2, 3 represent digit 1, digit 4, and digit 9, respectively.

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 introduce 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 main decision strategies 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:

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

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

Linear kernel:

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

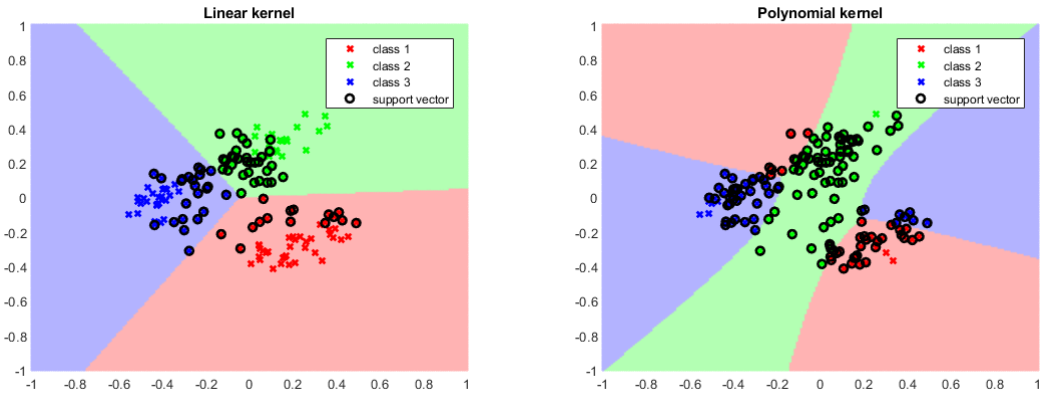
Polynomial (homogeneous) kernel of degree 2:

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

1. Use the dataset to build a SVM with linear kernel to do multi-class classification. Then plot the corresponding decision boundary and support vector.
2. Repeat (1) with polynomial kernel (degree = 2).
3. Please discuss the difference between (1), (2).

Hints

- In this exercise, we strongly recommend using matlab to avoid tedious preprocessing occurred in 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 a Gaussian mixture model (GMM) and apply it for image segmentation. In the initialization, use the K-means algorithm to find K central pixels. Given this initialization, use the expectation-maximization (EM) algorithm (please refer to textbook p.438-p.439) to optimize the parameters of the model. The input data is hw3.jpg. According to the maximum likelihood, you can decide the color μ_k, $k \in [1, \ldots, K]$ of each pixel x_n of ouput 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 the K-means model as means, and calculate the corresponding variances σ_k^2 and mixing coefficient π_k for the initialization of GMM $p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x|\mu_k, \sigma_k^2)$. Optimize the model by maximizing the log likelihood function $\log p(x|\pi, \mu, \sigma^2)$ over all training pixels through EM algorithm. Plot the log likelihood curve of GMM. (Please terminate EM algorithm when the iteration arrives 100)

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

– The input image is with licence free for personal and commercial use. Image from: https://www.pexels.com/photo/white-and-blue-house-under-cumulus-nimbus-clouds-906755/