Machine Learning (Homework 3)

Due date : 2021/1/1 23:55 (Hard Deadline)

1 Gaussian Process for Regression (60%)

In this exercise, please implement Gaussin process (GP) for regression. The file data gp_x.csv and gp_t.csv have input data $\mathbf{x} : \{x_1, x_2, \ldots, x_{100}\}, 0 < x_i < 1$ and the corresponding target data $\mathbf{t} : \{t_1, t_2, \ldots, t_{100}\}$ respectively. Please take the first 50 points as the training set and the rest as the test set. 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 $\beta^{-1} = 1$.

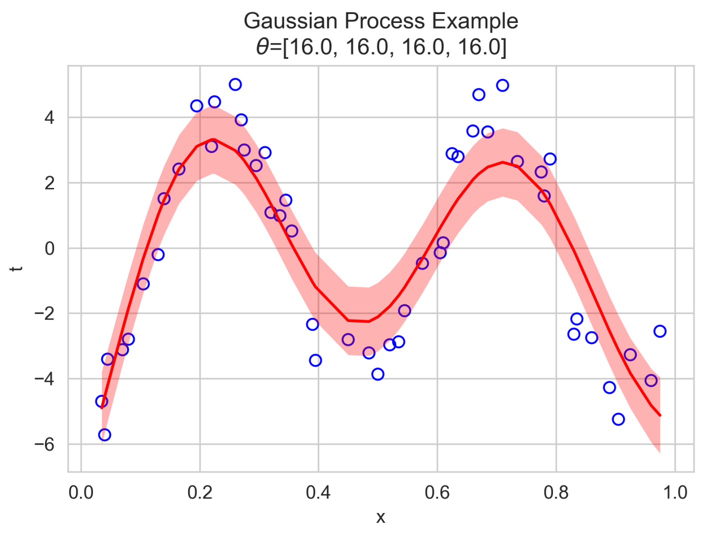
1. Please implement the GP with exponential-quadratic kernel function given by

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

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

   - linear kernel $\boldsymbol{\theta} = \{0, 0, 0, 1\}$
   - squared exponential kernel $\boldsymbol{\theta} = \{1, 16, 0, 0\}$
   - exponential-quadratic kernel $\boldsymbol{\theta} = \{1, 16, 0, 4\}$
   - exponential-quadratic kernel $\boldsymbol{\theta} = \{1, 64, 32, 0\}$

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 the figure should be the value of the hyperparameters used in this model. The red line shows the mean $m(\cdot)$ of the GP predictive distribution. The pink region corresponds to plus and minus one standard deviation. Training data points are shown in blue. An example is provided in below.



3. Show the corresponding root-mean-square errors

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

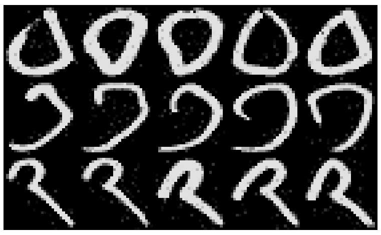
   for both training and test sets with respect to the four kernels.

4. Try to tune the hyperparameters by yourself to find the best combination for the dataset. You can tune the hyperparameters by trial and error or use **automatic relevance determination** (ARD) in Chapter 6.4.4 of textbook. (If you implement the ARD method, you will get the bonus points.)

5. Explain your findings and make some discussion.

2 Support Vector Machine (40%)

Support vector machines (SVM) is known as a popular method for pattern classification. In this exercise, you will implement SVM for classification. Here, the Tibetan-MNIST dataset is given in x_train.csv and t_train.csv. Tibetan-MNIST is a dataset of Tibetan images. The input data including three categories: Tibetan 0, Tibetan 1 and Tibetan 2. Each example is a 28x28 grayscale image, associated with a label.



Data Description

- **x_train** is a 300×784 matrix, where each row is the first two scaled principal values of a training image.
- **t_train** is a 300×1 matrix, which records the classes of the training images. 0, 1, 2 represent Tibetan 0, Tibetan 1 and Tibetan 2, respectively.

In the training procedure of SVM, you 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 [John Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines." (1998)]. Scikit-learn is a free software machine learning library based on Python. This library provides the sklearn.svm. You are allowed to use the library to calculate the multipliers (coefficients) rather than using the **prediction function** directly.

In this exercise, you will implement SVM based on two kinds of kernel functions

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

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

- **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:**

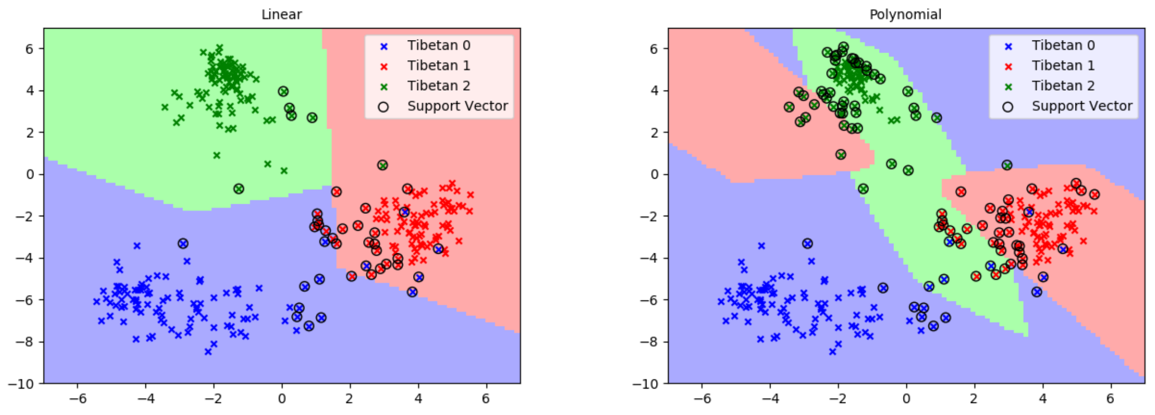
$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^2$$
$$\phi(\mathbf{x}) = [x_1^2, \sqrt{2}x_1x_2, x_2^2]$$
$$\mathbf{x} = [x_1, x_2]$$

SVM is a binary classifier, but the application here has three classes. To handle this problem, there are two decision approaches, one is the one-versus-the-rest', and another is the 'one-versus-one'

1. Analyze the difference between two decision approaches (one-versus-the-rest and one-versus-one). Decide which one you want to use and explain why you choose this approach.
2. Use the dataset to build a SVM with linear kernel to do multi-class classification. Then plot the corresponding decision boundary and support vectors.
3. Repeat (2) with polynomial kernel (degree = 2).
4. Discuss the difference between (2) and (3).

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 in image segmentation. First, use a K-means algorithm to find K central pixels. Second, use Expectation maximization (EM) algorithm (please refer to textbook p.438-p.439) to optimize the parameters of the model. The input data is hw3_3.jpeg. 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

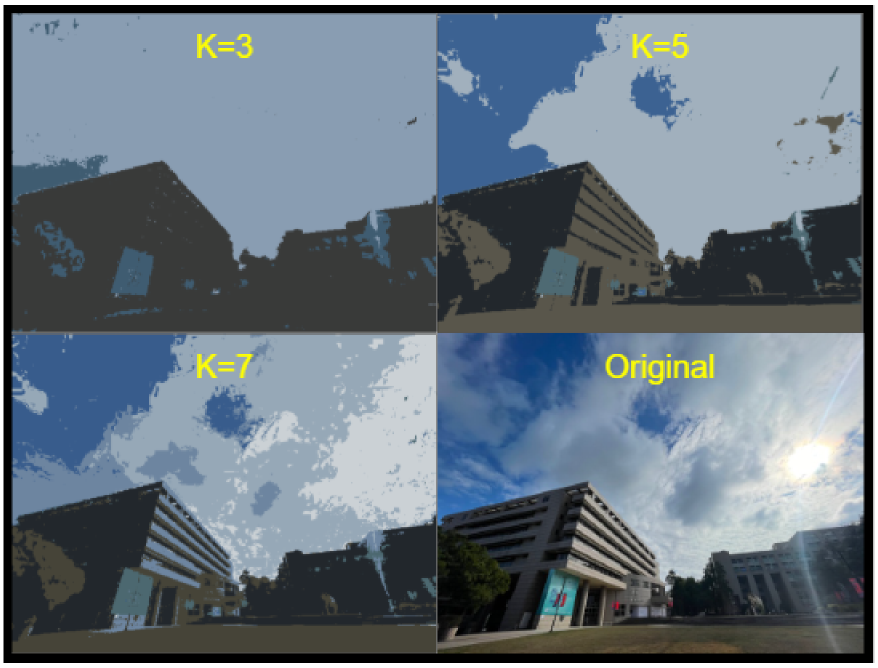
1. Please build a K-means model by minimizing

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

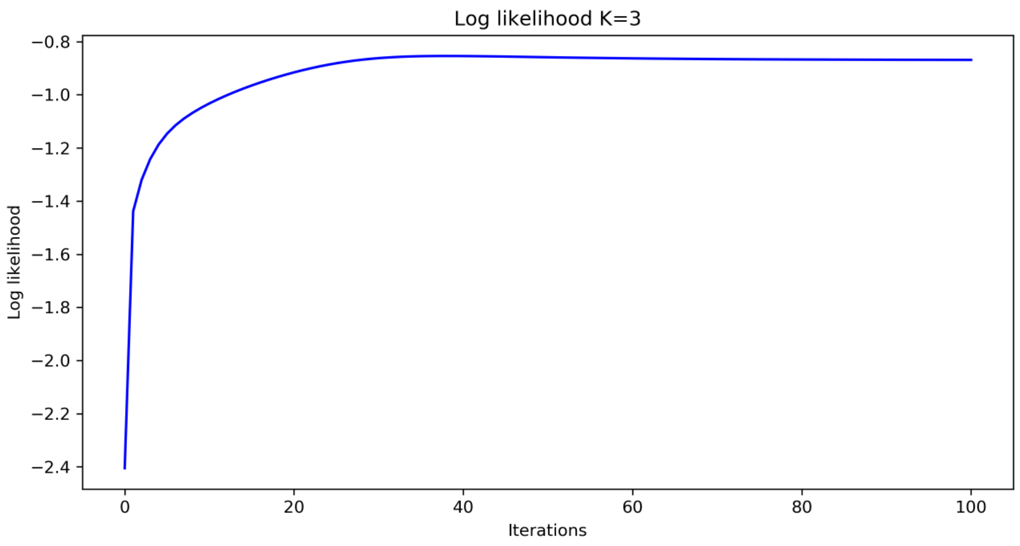
and show the table of estimated $\{\mu_k\}_{k=1}^{K}$.

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 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 $\log p(x|\pi, \mu, \sigma^2)$ 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 $K = 3, 5, 7,$ and 10. Please show the resulting images in your report. Below are some examples.



4. Please show the graph of Log likelihood at different iterations for $K = 3, 5, 7, 10$ respectively. Example are shown below. (This graph is only for your reference.)



5. You can make some discussion about what is crucial factor to affect the output image and explain the reason?

6. The image shown below is your input image taken by TA, and it is only allowed to be used for homework3.

4 Rule

- In your submission, you need to submit three files.
  Note : Only the following three files are accepted, so the code of each exercise should be written in one **.py** file.
  - **hw3_StudentID.ipynb** file which contains all the results and codes for this homework.
  - **hw3_StudentID.py** the content of this file must be consistent with .ipynb file.
  - **hw3_StudentID.pdf** file which is the report that contains your description for this homework.
    (e.g. hw3_0123456.ipynb; hw3_0123456.py; hw3_0123456.pdf)
- Implementation will be graded by
  - Completeness
  - Algorithm Correctness
  - Description of model design
  - Discussion and analysis
- Only Python implementation is acceptable.
- Numpy, Pandas and Matplotlib library are recommended for the implementation.
- Don't use high level toolbox/module functions (e.g. sklearn, polyfit).
- DO NOT PLAGIARISM. (We will check program similarity score.)