COMP.4220 - Machine Learning

Homework 6

student name:

---

1. We gave an example of matrix with all positive elements but that has a negative eigenvalue and hence that is not positive definite.
   (a) Find the eigenvalues of the matrix in example by solving the characteristic equation.
   (b) Find an example of the converse property, namely a 2×2 matrix with positive eigenvalues yet that has at least one negative element.

(4 marks)

2. Consider a 2×2 Gram matrix:
   (a) Write down its determinant, $\det(\mathbf{K})$.
   (b) Show that a positive definite kernel $k(x, x')$ satisfies the Cauchy-Schwartz inequality

$$k(x_1, x_2)^2 \leq k(x_1, x_1)k(x_2, x_2)$$

(6 marks)

3. (Programming) In this section, you investigate Gaussian Processes for Regression on a simple data set.

   ⇒**Note:** You are only allowed to use `numpy`, `sklearn`, and `matplotlib` modules for this question.

   ⇒**Note:** For consistency, we use a specific random seed throughout this question. Please do not modify this seed.

   (a) In the `generate_synthetic_data` function, sample 10 points from $\mathcal{U}(0,5)$ distribution and set that to x. Then use x to calculate the target values t from a $\sin(x-2.5)^2$. The call the function and generate the training set (x,t). The generated arrays x and t must have sizes $(10,1)$ and $(10,)$, respectively. Verify this. (1)

   (b) Set up a Radial Basis kernel using the existing RBF kernel in `sklearn` module.

   ```
           from sklearn.gaussian_process.kernels import RBF

   ```

   To do this, you should set up the magnitude, c and the length scale of the kernel, l:

   $$k(x,x') = c\exp(\frac{\|x-x'\|^2}{2l^2})$$

   However, recall that since the hyper-parameters are being learned, these values serve as initial values and will be optimized by the regression algorithm. Also make sure to set a bound for the length scale so the local optimization is bounded. (1)

   (c) Build a Gaussian Process Regressor (GPR) model using the `GaussianProcessRegressor` by importing

   ```
       from sklearn.gaussian_process import
      GaussianProcessRegressor

   ```

   Instantiate a GPR object from this class. For consistency set `random_state`=0. (1)

   (d) To test the regression model, generate x_{test} including 100 points linearly distributed in range $[0,5]$. Make sure this array has size $(100,1)$. (1)

   (e) Use the GPR model to produce the mean $m(x)$, and standard-deviation $\sigma(x)$ (not the covariance) of the prior distribution. Note that this is done before the model has seen any observations. In the top axis of a 2×1 subplot, plot the observations, the m_{prior}, and use the `fill_between` function to add $\pm\sigma_{\text{prior}}$ to the plot. (2)

(f) Then use the GPR model to produce the mean $m(x)$, and standard-deviation $\sigma(x)$ (not the covariance) of the posterior distribution. Note that this is done after the model has seen the observations. In the bottom axis of the same 2×1 subplot, plot the observations, the $m_{\text{posterior}}$, and use the `fill_between` function to add $\pm\sigma_{\text{posterior}}$ to the plot. (2)

(g) For each subplots, generate and plot 5 samples from the learned distributions respectively. Explain how the samples are different between the two subplots. (1)

(h) Print the hyper-parameters of the kernel before and after the optimization. Print the log-likelihood for the hyper-parameters. (1)

(i) So far, we have seen the performance of the GPR on noise-free observations. Generate the noisy data t_{noisy} by adding noise $\mathcal{N}(0, \sigma_{noise} = 0.2)$ to the target values t. (1)

(j) Train another GPR on the noisy data set. Make sure that you set correct parameters for the additional Gaussian measurement noise on the training observations. Use this model to produce the mean $m(x)$, and standard-deviation $\sigma(x)$ (not the covariance) of the posterior distribution. Repeat this part using the following kernels: Radial Basis, Rational Quadratic, Exponential Sine Squared, and Matern. (2)

(k) Plot a figure with 2×2 subplots, each of which should include the noisy observations, the $m_{\text{posterior}}$, and $\pm 1.96\sigma_{\text{posterior}}$ to the plot. Add 5 sample trajectories to each subplot. (1)

(l) For each kernel, print the hyper-parameters before and after training. Which kernel provides a better model. How did you measure that? Explain and show your results. (1)

Tips: refer to the `Scikit-learn` documentation to learn about the functions and their features. (15 marks)