$$\begin{aligned}
\min_{\boldsymbol{w},b,\boldsymbol{\xi}^+,\boldsymbol{\xi}^-} \quad & \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{C}{n}\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-) \\
\text{s.t.} \quad & y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b \leq \epsilon + \xi_i^+ \quad \forall i \\
& \boldsymbol{w}^T\boldsymbol{x}_i + b - y_i \leq \epsilon + \xi_i^- \quad \forall i \\
& \xi_i^+ \geq 0 \quad \forall i \\
& \xi_i^- \geq 0 \quad \forall i
\end{aligned}$$

where $\boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, $\boldsymbol{\xi}^+ = (\xi_i^+, \ldots, \xi_n^+)^T$, and $\boldsymbol{\xi}^- = (\xi_i^-, \ldots, \xi_n^-)^T$.

Here $C > 0$, $\epsilon > 0$ are fixed, and $\|\cdot\|_2$ is the Euclidean norm.

Show that for an appropriate choice of λ, SVR solves

$$\min_{\boldsymbol{w},b} \quad \frac{1}{n}\sum_{i=1}^{n}\ell_\epsilon(y_i, \boldsymbol{w}^T\boldsymbol{x}_i + b) + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2$$

where $\ell_\epsilon(y,t) = \max\{0, |y-t| - \epsilon\}$ is the so-called ϵ-*insensitive loss*, which does not penalize prediction errors below a level of ϵ.

Remark: The above method can be kernelized in a manner analogous to the SVM.

4. Model Selection

In this problem, you will be applying kernel ridge regression and support vector regression to a large-scale real world data set about air pollution, and using cross-validation to improve the performance of your algorithms. You will be using sklearn's built-in implementations for KRR and SVR, but you must implement the model selection aspects of the problem yourself.

Motivation

For many large cities, air pollution is a severe problem. We will be examining machine learning algorithms that can accurately predict air quality concentrations in a city. Air quality data are essential to determine which areas contain high concentrations of pollutants, and this can help governments and the general public make informed decisions.

Data

The dataset you will be looking at contains several air quality variables, namely SO_2, NO_2, O_3, CO, and $PM2.5$ (Particulate Matter with a diameter of 2.5 micrometers or less). Specifically, we will be predicting PM2.5 concentrations using the concentrations of the other pollutants. All units are in $\mu g/m^3$ (microgram/cubic meter). The air pollution data come from Beijing, China. The data includes concentrations for all of SO_2, NO_2, O_3, CO, and $PM2.5$. The dataset has been preprocessed for you and is included in the homework folder along with skeleton code.

- air-quality-train.csv contains data for SO_2, NO_2, O_3, CO, and $PM2.5$. You will use the data in this file to train your models.
- air-quality-test.csv contains data for SO_2, NO_2, O_3, CO, and $PM2.5$. You will use the data in this file to test your models.
- air-quality-code is the skeleton code that is provided for you that will load in the data into standard training and test dataset

Problem Statement

The goal is to predict the $PM2.5$ concentration from SO_2, NO_2, O_3, and CO concentrations. The evaluation metric that will primarily be used is Root Mean Squared Error (RMSE) between the predicted PM2.5 concentrations and the ground-truth PM2.5 concentrations. Here RMSE is simply the square root of the mean squared error. We have included all the functions from `sklearn` needed in air-quality-code, and you are not allowed to use any other more advanced functions from `sklearn` nor other external libraries not loaded in the skeleton code for this problem. For simplicity, we will only be working with the Gaussian kernel, and the only two hyperparameters we will be tuning are the regularization parameter and the parameter for Gaussian kernel.

Note: sklearn parametrizes the Gaussian kernel as $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma\|\boldsymbol{x} - \boldsymbol{y}\|^2)$, and refers to it by the more general name "radial basis function" (rbf). See also `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.rbf_kernel.html`.

Questions

(a) Implement support vector regression using machine learning library `sklearn`. See `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html` for more detailed documentation. We have provided very thorough skeleton code so all you really need to do is call each function with specified parameters (use rbf kernel, set regularization term 1, kernel parameter as 0.1 and rest as default which include $\epsilon = 0.1$) and evaluate the learned model. Please **report RMSE on the test data**.

Note: Unlike our formulation in problem 3, sklearn's regularization parameter C is not normalized by $1/n$. See `https://scikit-learn.org/stable/modules/svm.html#svm-classification` for more details.

(b) Implement kernel ridge regression using machine learning library `sklearn`. See `scikit-learn.org/stable/modules/generated/sklearn.kernel_ridge.KernelRidge.html` for more detailed documentation. We have provided very thorough skeleton code so all you really need to do is call each function with specified parameters (use rbf kernel, set regularization term 0.5, kernel parameter as 0.1 and rest as default) and evaluate the learned model. Please **report RMSE on the test data**.

Note:

1. sklearn implements ridge regression based on the same objective function stated in homework 3:
$$PRSS = \sum_{j=1}^{n} (\boldsymbol{w}^T \boldsymbol{x}_j + w_0 - y_j)^2 + \alpha\|\boldsymbol{w}\|^2$$
with one notational change: α replaces λ. See `https://scikit-learn.org/stable/modules/linear_model.html#ridge-regression` for more details.
2. α and C are related by $\alpha = 1/(2C)$. In other words, if KRR's α and SVR's C are related by $\alpha = 1/(2C)$, then the two methods differ only by the loss: squared error vs. ϵ-insensitive.

(c) Fill in the code to perform a grid search over the parameter space using K-fold cross validation with $K = 5$. Your selection criteria should be based on the K-fold cross-validation error estimate. As specified in the skeleton code, we will use rbf kernel. The regularization term's search space (for C) will be from 10^{-1} to 10^1 on logspace with base 10 (i.e $0.1, 1, 10$), and the kernel parameter's search space will be from 10^{-2} to 1 on

logspace with base 10 (i.e $0.01, 0.1, 1$). Please **report the optimal tuning parameters and their RMSE on the test data** for both support vector regression and kernel ridge regression.

Notes:

1. Helpful Python commands and libraries: numpy.arange, numpy.random.shuffle and numpy.array_split.
2. As stated in part (a) and (b), sklearn uses different parametrizations of the penalty terms for SVR and KRR. The specified search space is for C, and you can compute the corresponding α by $\alpha = 1/(2C)$.
3. For SVR report the optimal regularization in terms of C, while for KRR repor the optimal regularizaton in terms of α.

Submissions

- Make sure everything we ask for (**bold** items) is included in your write-up submitted to HW4 Gradescope
- Submit a single Python file including your implementation of part (a), (b), and (c) on Canvas with each part separated by comments.
- Submit printed code to part (a), (b), and (c) as part of your HW 4 Gradescope submission. This will be used to asses part (a)-(c).