Machine Learning Group
Fakulty IV, Technische Universität Berlin
Prof. Dr. Klaus-Robert Müller
Email: klaus-robert.mueller@tu-berlin.de

Problem Set 3: Kernel Ridge Regression, Cross-Validation

Part 1: Implementation

Assignment 1 (20 points)

Implement cross-validation as a general function, which can be used for various methods and objective functions.

```
method = cv(X, y, method, parameters, loss_function = mean_absolute_error,
            nfolds = 10, nrepetitions = 5)
```

The arguments have the following definitions:

- `X` is a $n \times d$ matrix of data.
- `y` is a length n vector, which contains the labels $y_i \in \{\pm 1\}$ or regression targets $y_i \in \mathbb{R}$ for every data point.
- `method` is a `class` which has the following functions:
  - `fit(X, y)` trains the object instantiated with the given parameters with data points `X` and labels `y`.
  - `predict(X)` returns the predictions.
- `parameters` is a dictionary with parameter names as keys. The dictionary values are lists of candidate parameter values. Cross-validation should be carried out for all possible parameter combinations.
- `loss_function` is a function handle to the loss function to be used. It should have the following signature:

  ```
  l = loss_function(y_true, y_pred)
  ```

  where `y_true` are the true targets $\boldsymbol{y}$ and `y_pred` are the predicted targets $\widehat{\boldsymbol{y}}$. This parameter is optional with the standard value `mean_absolute_error`.
- `nfolds` is the number of partitions (m in the guide). This parameter should be optional with a standard value of 10.
- `nrepetitions` is the number of repetitions (r in the guide). This parameter is optional with the standard value 5.

Implement the following functionality:

(a) Write the loss function `mean_absolute_error` which returns the mean absolute error (MAE):

$$\mathcal{L}_{\text{MAE}}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \frac{1}{n} \sum_{i=1}^{n} |\widehat{y}_i - y_i|.$$

This loss should be used as the default loss function, if the optional parameter `loss_function` is not specified.

(b) The function `cv` should return a `method` object, which is an instantiated classifier class that has been trained with the optimal parameter values. In addition, it should have an attribute `method.cvloss` containing the cross-validated loss.

(c) The function should report the progress of the function on the command line and also give an estimate for the remaining run time.

(d) If there is only one parameter combination, the function `cv` should not search for a minimum, instead it should calculate the average loss function output for all repetitions and folds. This will come in handy for the generation of the ROC curves in Assignment 4.

For the iteration over the parameter set, you might want to use `itertools.product`.

Assignment 2 (20 points)

Implement Kernel Ridge Regression as

```
class krr(kernel, kernelparameter, regularization)
```

which has the functions

- `fit(Xtrain, ytrain)` with
  - `Xtrain`: $n \times d$ matrix of training data,

  - `ytrain`: length n array of targets and
- `predict(Xtest)` with
  - `Xtest`: $m \times d$ matrix of test data.

(a) The predict function should return the predicted targets as a length m array.

(b) The following kernels (with the accompanying parameters) should be implemented:

Name	Kernel	Parameter
`linear`	$k(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{x}, \boldsymbol{x}' \rangle$	(none)
`polynomial`	$k(\boldsymbol{x}, \boldsymbol{x}') = (\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + 1)^d$	degree $d \in \{1, 2, 3, \ldots\}$
`gaussian`	$k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 / 2\sigma^2)$	kernel width σ

(c) The Parameter `regularization` is the regularization constant, C in $\widehat{\boldsymbol{\alpha}} = (\mathbf{K} + C\mathbf{I})^{-1}\boldsymbol{y}$. If `regularization` is zero, execute leave-one-out cross-validation on C efficiently (see handbook). Use logarithmically spaced candidates around the mean of the eigenvalues of the kernel matrix $\mathbf{K}$ for C.

Part 2: Applications

Assignment 3 (35 points)

We consider the QM7 data set for predicting atomization energies of molecules from nuclear charges and atomic positions with Kernel Ridge Regression (cf. Rupp et al., 2012). Each of the 7165 molecules in the data set is represented by a Coulomb matrix $\mathbf{M}_i \in \mathbb{R}^{23\times 23}$. The corresponding target atomization energy is given by $y_i \in \mathbb{R}$ in kcal/mol (arrays `X` and `T` respectively in the data file).

The Coulomb matrices cannot be used for regression directly because molecules can have different atomic permutations or can be translated and rotated in space which does not influence their chemical properties. In order to make the data invariant to these transformations, transform each msteolecule into a vector $\boldsymbol{x}_i \in \mathbb{R}^{23}$ of eigenvalues of $\mathbf{M}_i$ in order of decreasing absolute value. We are now equipped with a data matrix $\mathbf{X} \in \mathbb{R}^{7165\times 23}$ and a vector of targets $\boldsymbol{y} \in \mathbb{R}^{7165}$.

(a) Plot the distances $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ against the absolute difference of energies $|y_i - y_j|$ for all pairs of data points.

(b) Shuffle the data randomly and fix a train/test split of size 5000/2165.

(c) Use fivefold cross-validation to estimate on 2500 random training samples:

(1) Width parameter σ of the Gaussian kernel. Candidates are quantiles of pairwise Euclidean distances.

(2) Regularization parameter C. Use logarithmically scaled values between 10^{-7} and 10^{0} as candidates.

Report the best parameters and the mean absolute error on the test set in kcal/mol.

(d) Keep C and σ fixed and plot the MAE on the test set as a function of the number n of training samples with n from 100 to 5000.

(e) Show scatter plots of points $(y_i, \widehat{y}_i)$ with train and test data in two different colors for three models trained on 1000 samples: one that obviously underfits, one that fits well and one that obviously overfits. Explain underfitting and overfitting on basis of these plots.

Assignment 4 (25 points)

Download the classification data sets from ISIS. Apply KRR with efficient leave-one-out cross-validation

(a) Implement the zero-one-loss:

$$\mathcal{L}_{0\text{-}1}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \operatorname{sgn}(\widehat{y}_i))$$

where sgn is the sign function and I the indicator function. Use this loss function throughout this assignment.

(b) Generate a dictionary `results` which contains one dictionary for each of the data sets. These dictionaries have the following fields with the results for the best classifier: the cross-validated loss `cvloss`, the kernel `kernel`, the kernel parameter `kernelparameter`, the regularization strength `regularization` and the predicted labels `y_pred` for the test data.
For example, you could have `results['banana']['kernel'] = 'gaussian'`. Save the dictionary `results` in the file `results.p` using `pickle.dump` and submit it together with your report and implementation code.

(c) Plot the ROC curves for Kernel Ridge Regression resulting from variation of the bias term, i.e. the constant term (independent of the data) in the prediction function $f(\boldsymbol{x})$. Proper cross-validation has to be performed here!
Hint: the easiest way to do this is to define a function `roc_fun` which calculates TPR and FPR for a set of biases. Then supply into `cv` the optimal parameters and `roc_fun` as a loss function.

(d) Note that the efficient cross-validation of the regularization uses a squared loss instead of the 0-1-loss. Is there a difference in performance compared to using `cv` to optimize `regularization`?

References

M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical review letters*, 108(5):058301, 2012. URL `https://arxiv.org/abs/1109.2618`.