EE 559 **Homework 8 (Week 10)** Posted: Sat., 3/28/2020
Jenkins Due: Mon., 4/6/2020, 2:00 PM

In this homework assignment, you will use SVM classifiers with different kernels and datasets. You will use a free SVM software package called LIBSVM provided at http://www.csie.ntu.edu.tw/~cjlin/libsvm/. Please allow time for dealing with any bugs or version issues that appear when you first try to use it.

Notes for Matlab users:

1. Download LIBSVM and read the documentation first to learn how to use it. Check the README file inside the main folder libsvm-3.23 as well as the README file inside the Matlab folder.
2. If you are using Windows, simply add the **windows** (not the Matlab) folder from the LIBSVM folder to your path – no need to run the installation steps.
3. If you are using Unix based systems, navigate to the Matlab folder inside the LIBSVM folder and run make. (I didn't check this method since I work on Windows computers)
4. You can ignore any Matlab warnings regarding the svmtrain function. This warning concerns Matlab's own svmtrain (built-in) and not the function provided by LIBSVM.

Note for Python users:

1. Sklearn's implementation of SVM is based on LIBSVM. Thus, there's no need to download LIBSVM.

We will use 3 different datasets as described below:

Problem No.	Dataset Name	# Features	# Classes	Linear Separable	Has Test Set
1a-c	HW8_1	2	2	Yes	No
1d-e	HW8_2	2	2	No	No
2	wine	13 (first 2 features used)	3	N/A	Yes

1. In this problem, you will use the full dataset D for training, and where requested, the classification accuracy is calculated on the (same) training dataset.

   *For parts (a)-(c) below, load **HW8_1**, which is a linearly separable case.*

   a) Use Linear Kernel and try different values of slack variable parameter *C*. What is the meaning of parameter *C* and how will it impact your classification? Set *C*

= 1, C = 100, report the accuracy of classification and plot the decision boundary using ***plotSVMBoundaries()*** as provided. Explain your observation.

b) For **HW8_1** and C = 100 with Linear Kernel, give the support vectors numerically and circle them (or clearly identify them in some other way) in your plot. Provide the weights w0, w1, w2 and the decision boundary equation.

Hint: for how to obtain the weights in MATLAB, see: https://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html
FAQ: How could I generate the primal variable w of linear SVM?

Hint: python users can pass the support vectors as a parameter to plotSVMBoundaries to get a new plot where they show.

c) **Extra credit**. Continuing from part (b) above, calculate $g(x)$ for each support vector. Are they on the boundary of the margin (corresponding to $g(x) = \pm 1$)? If not, then conjecture as to why not, and test your conjecture computationally.

*For parts (d)-(e) below, load **HW8_2**, which is not linearly separable.*

d) Use Gaussian (RBF) Kernel with default gamma parameter. Set C = 50, 5000. Report the accuracy of classification and plot the decision boundary using ***plotSVMBoundaries()*** as provided. Explain the linearity or nonlinearity of the decision boundary, and explain the difference in decision regions for the two values of C.

e) Use Gaussian Kernel again, but now with default C parameter. Set gamma = 10, 50, 500. Report the accuracy of classification and plot the decision boundary using ***plotSVMBoundaries()*** as provided. Explain the difference in decision regions for the 3 values of *gamma*. Is there any overfitting problem?

2. In this problem, you will use cross-validation for model (parameter) selection, in order to more optimally design an SVM classifier. There is also a separate test set that you can use.

Note that cross-validation is covered in Lecture 18 (Monday 3/30/2020).

In this problem, you are required to code the cross-validation yourself; although you may use MATLAB or Python functions to divide the dataset into subsets.

In all parts that ask for cross-validation, use 5-fold cross-validation on the "training set" (this is the set we called $\mathcal{D}'$ in lecture). Do not use the test set until part (d).

Load **wine**, a 3 class dataset. **In all of the following parts, use only the first two features in training and testing.**

Tip 1: The standard deviation can be estimated by:

$$\hat{\sigma} = \left[\frac{1}{M-1} \sum_{m=1}^{M} \left(p_m - \overline{p} \right)^2 \right]^{\frac{1}{2}} = \left[\frac{1}{M-1} \sum_{m=1}^{M} \left(p_m{}^2 - \overline{p}^2 \right) \right]^{\frac{1}{2}} .$$

or by using available Matlab or Python functions for unbiased sample standard deviation.

Tip 2: For partitioning functions:

- Matlab: use cv = cvpartition(params) (to have a proper partitioning, be sure stratified is set to true), and then idx = training(cv, k).
- Python: look for sklearn's StratifiedKFold

(a) Use Gaussian (RBF) Kernel. Set $\gamma = 1$, C = 1. Report the average cross-validation accuracy.

(b) In this part you will use cross validation to find the best parameter set (model selection).

First, pick some values from the ranges $\gamma \in [10^{-3}, 10^{3}]$ and $\mathrm{C} \in [10^{-3}, 10^{3}]$ (hint: use **logspace()**; at least 50 points for each parameter are recommended)

Next, initialize a 2D array of size (N_γ, N_C) to store average accuracies $\overline{p}$ on the validation set; let's call the array ACC. Also set up another 2D array, DEV, to store the estimated standard deviation of accuracies on the validation set. You will use these two array types, repeatedly, below.

Then, for each pair of $[\gamma, \mathrm{C}]$ perform a 5-fold cross validation and store the average accuracy to the corresponding position in ACC, and similarly the estimated standard deviation of accuracy in DEV. Perform the partitioning randomly but stratified. Also, make sure that each run of the training procedure is initialized (restarted with the new values of all variables) each time.

(i) Visualize ACC. Turn in your visualization (pseudocolor map, plot, or other) with your HW.

Tip for MATLAB: you can use imagesc(), mesh() or surf(), whichever is most illustrative

Tip for Python: you can use imshow() and colorbar() from matplotlib.pyplot

(ii) Is there a clear choice for best values of $[\gamma, \mathrm{C}]$? If so, report the pair of chosen values, as well as its mean cross-validation accuracy and standard deviation. If not, pick one set of values from the candidate pairs (e.g., the pair with the lowest standard deviation), and give its mean cross-validation accuracy and standard deviation.

(c) Repeat the cross validation procedure in (b) *T*=20 times (runs), storing the resulting arrays ACC(t) and DEV(t) each time. No need to visualize every run.

(i) Report on the 20 chosen pairs of $[\gamma, \mathrm{C}]$.

(ii) Accumulate a single overall average ACC array, and a single overall DEV array, based on all T=20 runs; that is, you will have one average accuracy and standard deviation for each pair $[\gamma, \mathrm{C}]$. Is a pick of the best values for $[\gamma, \mathrm{C}]$ over the T=20 runs more well-defined or more reproducible now? Justify your answer. Also, report on the final chosen best values for $[\gamma, \mathrm{C}]$, as well as its mean cross-validation accuracy and standard deviation.

(d) Use the full training set to train the final classifier using your best pair of $[\gamma, \mathrm{C}]$ from (c) (ii) above. Then use the test set to estimate the accuracy of your final classifier on unknowns. Is this estimate within approximately 1 standard deviation of your mean cross-validation accuracy from (c) (ii)? If not, try to explain why.