Boston University
Department of Electrical and Computer Engineering

ENG EC 414 Introduction to Machine Learning

HW 5 Solution

Issued: Fri 9 Oct 2020 **Due:** 10:00am Fri 16 Oct 2020 in Gradescope (non-code) + Blackboard

Important: Before you proceed, please read the documents pertaining to *Homework formatting and submission guidelines* in the Homeworks section of Blackboard. **In particular, for computer assignments you are prohibited from using any online code or built-in MATLAB functions except as indicated in the problem or skeleton code (when provided).**

Note: Problem difficulty = number of coffee cups ☕

Problem 5.1 [13pts] *(SVM by hand)*
Consider again the two-class classification with the following training feature vectors: $\mathbf{x}_P = (2,0)^\top, \mathbf{x}_Q = (0,4)^\top, \mathbf{x}_R = (3,3)^\top, \mathbf{x}_S = (7,5)^\top$ with labels $-1, -1, +1, +1$ respectively.

(a) [6pts] Hand-compute the parameters of the maximum margin linearly separating hyperplane (SVM hyperplane) in *canonical* form.

(b) [3pts] Determine which training points lie on the margin.

(c) [4pts] We said that the SVM hyperplane can always be written as a linear combination of the support vectors: $\boldsymbol{w}_{\text{SVM}} = \sum_{i=1}^{N_{\text{SV}}} \alpha_i z_i$, where z_i are the N_{SV} support vectors. Find the coefficient α_i of this combination for the hyperplane in the point (a).

Solution:

(a) From the figure it is clear that this dataset is linearly separable, e.g., the vertical hyperplane given by $x_1 = 2.5$, i.e., $(1,0)\mathbf{x} - 2.5 = 0$, separates the dataset. Hence, a unique canonical-form SVM that linearly separates this dataset exists.

The convex hull C_- of all the negative samples is the line segment joining $\mathbf{x}_P$ and $\mathbf{x}_Q$ which is contained in the hyperplane of part (f). The convex hull of all the positive samples C_+ is the line segment joining $\mathbf{x}_R$ and $\mathbf{x}_S$. Since $\mathbf{x}_S - \mathbf{x}_- = (6,3)^\top$ is perpendicular to the hyperplane $(\mathbf{w}_f, b_f)$ and is longer than $\mathbf{x}_R - \mathbf{x}_- = (2,1)^\top$, it follows that among all pairs of points, one from C_- and the other from C_+, $(\mathbf{x}_-, \mathbf{x}_R)$ is the unique closest pair.

Thus, $\mathbf{w}_{\text{SVM}} = \gamma(\mathbf{x}_R - \mathbf{x}_-) = \gamma(2,1)^\top$. The midpoint is $(\mathbf{x}_R + \mathbf{x}_-)/2 = (2,2.5)^\top$. The SVM hyperplane passes through the midpoint. Hence, $\mathbf{w}_{\text{SVM}}^\top (2,2.5)^\top + b_{\text{SVM}} = 0 \Rightarrow b_{\text{SVM}} = -\gamma(2,1)(2,2.5)^\top = -6.5\gamma$. Since the hyperplane is in canonical form and $\mathbf{x}_R$ is a point closest to it, $\mathbf{w}_{\text{SVM}}^\top \mathbf{x}_R + b_{\text{SVM}} = +1 \Rightarrow \gamma\left\{(2,1)(3,3)^\top - 6.5\right\} = 1$. Solving for γ we get $\gamma = 2/5$. Therefore, $\mathbf{w}_{\text{SVM}} = (4/5, 2/5)^\top$ and $b_{\text{SVM}} = -13/5$.

(b) Points $\mathbf{x}_P, \mathbf{x}_Q, \mathbf{x}_R$ lie on the marginal hyperplanes. This can be seen from the figure or alternatively by checking that $\mathbf{w}_{\text{SVM}}^\top \mathbf{x} + b_{\text{SVM}}$ equals ± 1 only for these three points.

(c) We have to write the SVM solution as a linear combination of the points $\mathbf{x}_P, \mathbf{x}_Q, \mathbf{x}_R$. It should be easy to see that there are infinite solutions (2 independent equations and 3 unknowns), so we just have to pick anyone of them. One very simple way to do it is to assign the coefficient 0 to $\mathbf{x}_R$ and find the coefficients for the other two points. Given that the other two points have only one coordinate different than 0, this should be immediate:

$$\mathbf{w}_{\text{SVM}} = (4/5, 2/5)^\top = 2/5 \cdot \mathbf{x}_P + 1/10 \cdot \mathbf{x}_Q + 0 \cdot \mathbf{x}_R \, .$$

Problem 5.2 [20pts] *(k-folds cross-validation for Regularized Least Square Regression with polynomials)* In this question, you have to implement linear regression from $\mathbb{R}^d$ to $\mathbb{R}$ using polynomials and choose the best degree of the polynomials using k-folds cross-validation. In other words, **for each coordinate i of the input** x_i, we generate new features $(x_i^2, x_i^3, \ldots, x_i^p)$ and we append them to the original features. Then, we learn a linear predictor in this space. This corresponds to learn a predictor of the form

$$\hat{y} = w_1 x_1 + \cdots + w_d x_d + w_{d+1} x_1^2 + \ldots w_{2d} x_d^2 + \cdots + w_{d(k-1)+1} x_1^p + \cdots + w_{dp} x_d^p + b \, .$$

We also assume that all the coordinates of the input are positive.

(a) [5pts] First, complete the skeleton code of `generate_poly_features.m` to implement a function that generates a matrix of input samples that contains the polynomials of each feature from the linear term to the polynomial of degree `p`, with prototype

```
[X_poly] = generate_poly_features(X,p)
```

Using our notation, `X` is $m \times d$, where m is the number of training samples and d is the dimension. `X_poly` will have the same number of rows and columns equal to $d \times p$. Do not include the term of order 0, that is, the column of 1s.

(b) [6pts] Complete the skeleton code of the function `cross_validation_rls.m` to implement k-folds cross-validation for regularized least square. The prototype is

```
[validation_loss] = cross_validation_rls(X,y,lambda,k)
```

As we have seen in class, in `k`-folds cross-validation, we divide the training data contained in `X` and `y` in k disjoint sets. We assume the training data to be shuffled, so it does not matter how you create the folds. Then, we use one of the k folds as the validation fold and we use the remaining $k-1$ to train our RLS predictor with $\lambda =$`lambda`. Evaluate the loss of the trained predictor on the validation fold, repeat the above k times, and return the averages of the mean losses on the validations folds in `validation_loss`. Note that `validation_loss` is a scalar. The function to run RLS is also provided in `train_rls.m`.

(c) [4pts] In the zip file there is also the "cadata" training/test data in the file "cadata_train_test.mat". It is a random train/test split of the Housing dataset from the UCI repository. The task is to predict the median house value from features describing a town. I normalized the features for you, to be in $[0, 1]$. Complete the skeleton code in `problem_5_2c.m` to use `cross_validation_rls` and `generate_poly_features` to try polynomials up to degree 10 with 8-folds cross-validation and $\lambda = 0.001$. The code should record the cross-validation loss for each choice of the degree of the polynomial from 1 to 10 in a vector of dimension 10.

(d) [2pts] Plot the 8-folds validation loss for each degree of the polynomial using the code in the previous point and report the degree of the polynomial that gives the best 8-folds cross-validation loss. Discuss the results: is this what you expected? Does it make sense? (No code to submit here.)

(e) [3pts] Re-train a RLS predictor with the best degree found in the previous point and report its mean square loss error on the test set. (No code to submit here.)

Problem 5.3 [9pts] *(Questions)*
Clearly explain your answers.

(a) [2pts] Consider least square regression with polynomials. Does the bias decrease with the degree of the polynomials?

Solution: The bias is how different is the best predictor in our hypothesis set from the best possible predictor. Given that increasing the degree of the polynomials we are able to express a richer and richer class of functions, the bias will decrease (or to be more precise non-increase) with the degree of the polynomials.

(b) [2pts] Can I alleviate underfitting increasing the number of training samples?

Solution: We have underfitting when the hypothesis class is too "simple" for our problem. We can detect underfitting when the training error is too high. In other words, in underfitting we have too much bias. To alleviate underfitting we have to choose a richer hypothesis set; increasing the number of training samples will not help.

(c) [2pts] Overfitting is due to too much bias or too much variance?

Solution: We defined informally overfitting when the test error and training error are too far one from the other. It is not a matter of "learning the noise in the data" or "memorizing the training set", rather it is due to high variance, that is we don't have enough training samples for the complexity of our hypothesis class.

(d) [3pts] ☕ We want to use a hard-margin SVM with polynomials features. For a fixed training set, does the margin of the trained SVM increase or decrease with the degree of the polynomial?

Solution: The margin of a linearly separable training set is defined as the maximum over all possible separating hyperplanes of the minimal distance between the training points and the hyperplane itself. If you increase the degree of the polynomial, we are essentially adding more dimensions. This implies that the margin will increase (or to be more precise non-decrease) because the set of linearly separable hyperplanes will increase. Indeed, maximizing any function over a larger set always gives a bigger or equal maximum.

Code-submission via Blackboard: You must prepare 3 files: `generate_poly_features.m` for Problem 5.2(a), `cross_validation_rls.m` for Problem 5.2(b), and `problem_5_2c.m` for Problem 5.2(c). Place them in a **single** directory which should be zipped and uploaded into Blackboard. Your directory must be named as follows: `<yourBUemailID>_hwX` where `X` is the homework number. For example, if your BU email address is `charles500@bu.edu` then for homework number 5 you would submit a single directory named: `charles500_hw5.zip` which contains all the MATLAB code (and only the code).

Three corresponding skeleton code files are provided for your reference. Reach-out to the TAs via Piazza and their office/discussion hours for questions related to coding.