EE 559 Homework 6

Posted: March 29, 2023,
Due: March 31, 2023, 11:59 PM PDT

Keith M. Chugg, B. Keith Jenkins
version 5

1 SVM using `sklearn`

In this problem, you will use the `sklearn` implementation of SVM solver, which is based on LIBSVM. You will try different hyperparameters of SVM models on the training and testing datasets 1 (linearly separable) and 3 (not linearly separable) from Homework 1 . You are provided with a routine called `plotSVMBoundaries()` that will allow you to visualize the decision boundaries and regions learned by the SVM.

In all cases below:

- Use class `sklearn.svm.SVC`
- Always report train and test accuracies.
- For linear kernels, also give weight vector $\underline{w}$ including offset $\underline{w_0}$.
- For all plots, show the data (training or testing data), and decision regions using `plotSVMBoundaries()` (separate plots for training data and testing data).
- For some plots, you will be asked to show which training vectors are support vectors, (on the training-data plots only).
- Tip: A definition and description of support vectors is summarized at the end of this problem.

In problems 1(a) - 1(b), you will be using Homework 1 dataset 1.

(a) Use the `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 $C = 0.01$ and $C = 1$. Report the above items and also provide the support vectors in the plots for each value of C. Discuss your results and explain the performance and the differences for the different values of C.

You will provide 4 *plots in total;* 2 *(for train and test sets) using* $C = 0.01$ *and* 2 *(for train and test sets) for* $C = 1$.

(b) Use a `Gaussian (RBF) Kernel` with C parameter set to $C = 0.01$. Set $\gamma = 1, 3, 10, 50$. Report the above items and also show the support vectors in the training-data plots for each value of γ. Explain the linearity or nonlinearity of the decision boundary, and explain the difference in decision regions for the various values of γ. State where (if anywhere) you observe underfitting or overfitting.

You will provide 8 *plots in total.*

In problems 1(c) - 1(e), you will be using Homework 1 dataset 3.

(c) Use the `Linear Kernel` and try different values of slack variable parameter C. Set $C = 1$ and $C = 100$. Report the above items for each value of C. Discuss your results. *You will provide* 4 *plots in total.*

(d) Use a `Gaussian (RBF) Kernel` with C parameter set to $C = 1$. Set $\gamma = 0.1, 10, 200$. Report the above items and also show the support vectors in the training-data plots for each value of γ. Explain the difference in decision regions for the different values of γ. Tip: you might want to try plots at other values of γ to help you understand its effects (no need to include these extra plots in your solution). Do you observe any overfitting or underfitting for any of the given values of γ? *You will provide* 6 *plots in total.*

(e) Use a `Gaussian (RBF) Kernel` and pick the γ parameter from part (d) (from the 3 given values) that results in the minimum test error. Set $C = 0.01, 1, 100$. Report the above items and also provide the support vectors in the plots for each value of C. Explain your observations in the different decision boundaries and the support vectors for the different values of C. *You will provide* 6 *plots in total.*

Support vectors are the data points (from the training set) that determine the position and orientation of the max-margin decision-boundary hyperplane. The support vectors are mathematically defined as the training data points that have Lagrange multiplier value (at the solution) of $\lambda_i \neq 0$. For linearly separable data, the support vectors are the vectors that lie on the margin boundary (for nonlinear kernels, this applies in the expanded dimensional feature space). For datasets that are not linearly separable in the expanded dimensional feature space, the support vectors are the vectors that lie on the margin boundary or on the wrong side of the margin boundary. (You can infer this behavior from the KKT conditions in the SVM solution (c.f. lecture notes).) The `sklearn SVC` model outputs the support vectors with the attribute `support_vectors_`.

2 Backprop Initialization for Multi-class Classification

Let $\underline{a} = \underline{a}^{(L)}$ denote the pre-activation at the final layer of an ANN and let $\underline{\hat{p}} = \underline{v}^{(L)} = \underline{\text{SoftMax}}(\underline{a})$ be the final output when the output layer is a SoftMax layer. Recall that multi-class cross-entropy cost is given as

$$J = -\sum_{i=1}^{C} p_i \ln \hat{p}_i \tag{1}$$

where $\underline{p}$ is the label vector. Finally, recall that delta vector recursion is initialized via

$$\underline{\delta}^{(L)} = \frac{\partial \underline{\hat{p}}}{\partial \underline{a}} \nabla_{\underline{\hat{p}}} J \tag{2}$$

where the denominator convention and right-to-left chain rule for vector differentiation has been used.

1. What is the dimension of each of the following quantities: $\frac{\partial \underline{\hat{p}}}{\partial \underline{a}}$, $\nabla_{\underline{\hat{p}}} J$, and $\underline{\delta}^{(L)}$?

2. Find $\frac{\partial \underline{\hat{p}}}{\partial \underline{a}}$.

3. Find $\nabla_{\underline{\hat{p}}} J$.

4. Show that $\underline{\delta}^{(L)} = \underline{\hat{p}} - \underline{p}$

3 Backprop by Hand

An MLP has three input nodes, two hidden layers, and three outputs. The activation for the hidden layers is ReLu. The output layer is softmax. The weights and biases for this MLP are:

$$\underline{\underline{W}}^{(1)} = \begin{bmatrix} 1 & -2 & 1 \\ 3 & 4 & -2 \end{bmatrix}, \quad \underline{b}^{(1)} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

$$\underline{\underline{W}}^{(2)} = \begin{bmatrix} 1 & -2 \\ 3 & 4 \end{bmatrix}, \quad \underline{b}^{(2)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\underline{\underline{W}}^{(3)} = \begin{bmatrix} 2 & 2 \\ 3 & -3 \\ 2 & 1 \end{bmatrix}, \quad \underline{b}^{(3)} = \begin{bmatrix} 0 \\ -4 \\ -2 \end{bmatrix}$$

1. **Feedforward Computation:** In this part you will perform the feedforward computation for the input vector $\underline{x} = [\ +1\ \ -1\ \ +1\ ]^T$. The standard notation used in the slides and handouts is used. Specifically, $\underline{a}^{(l)}$ is the linear activation – *i.e.,* $\underline{v}^{(l)} = \mathbf{h}(\underline{a}^{(l)})$ – and $\dot{\underline{v}}^{(l)} = \dot{\mathbf{h}}(\underline{a}^{(l)})$. Find $\underline{a}^{(l)}$ and $\underline{v}^{(l)}$ for $l = 1, 2, 3$ and $\dot{\underline{v}}^{(l)}$ for $l = 1, 2$. In the process, if you need the value of the derivitive of ReLU(a) at $a = 0$, use $1/2$.

2. **Back-propagation Computation:** Run the standard SGD back-propagation for this input assuming a multi-category cross-entropy loss function and the one-hot labeled target: $\underline{p} = [\ 0\ \ 0\ \ 1\ ]^T$. Again, our standard notation is used so that $\underline{\delta}^{(l)} = \nabla_{\underline{a}^{(l)}} J$. Determine $\underline{\delta}^{(l)}$ for $l = 3, 2, 1$ and provide the updated weights and biases assuming a learning rate of $\eta = 0.5$ – *i.e.,* find $\underline{\underline{W}}^{(l)}(i+1)$ and $\underline{b}^{(l)}(i+1)$ for $l = 1, 2, 3$ assuming that weights and biases provided are the values at iteration i.

3. For this fixed input $\underline{x}$, show that the mapping from $\underline{x}$ to to $\underline{a}^{(3)}$, as performed by the network, can be written as $\underline{a}^{(3)} = \underline{\underline{W}}_{\text{eff}}\,\underline{x} + \underline{b}_{\text{eff}}$, where $\underline{\underline{W}}_{\text{eff}}$ and bias $\underline{b}_{\text{eff}}$ are the "effective" weight matrix and bias vector used by the network to map the input to the final linear-activation. Are these effective weights and biases a function of $\underline{x}$? Explain how this allows one to interpret an ANN with ReLu activations and discuss how this compares to a linear model that maps each input $\underline{x}$ to output $\underline{\underline{W}}\underline{x} + \underline{b}$ for fixed $\underline{\underline{W}}$ and $\underline{b}$. **Hint:** when the pre-activation to a ReLu is negative, the node activation is zero, so it is equivalent to removing that node from the network for that specific input.