ELEC 378 – Spring 2023
Homework 9

Due: Friday April 7, 5PM

1 Implementing SVM

Download `Train1.mat` and `Train2.mat` from the Resources section of Piazza, two sets of training data each with $n = 100$ and $p = 2$. Using the formulas from the class notes, code a (full, i.e., not stochastic) gradient descent algorithm for learning the decision hyperplane parameters $\mathbf{w}$ and b for the soft margin SVM. Your implementation should work for arbitrary n and p. Turn a documented listing of your code.

(a) Initialize $\mathbf{w}$ and b to random values and plot the resulting (random) hyperplane (a straight line since $p = 2$) on top of the scatter plot of the `Train1.mat` data.

(b) Set up your gradient descent code such that it redraws the decision hyperplane each iteration of training. Run your code to convergence on `Train1.mat`, showing the progression of hyperplanes for $\lambda = 1$. Discuss the convergence, in particular to the ultimate support vectors. Try with a few different initializations for $\mathbf{w}$ and b.

(c) Re-run the experiment from (c) with various values of λ and comment on the results. In particular, what does λ trade off?

(d) Modify your code to take stochastic gradient steps rather than full gradient steps. What changes?

(e) Now use the same implementation to fit an SVM to the `Train2.mat` data with $\lambda = 1$. Plot the learned decision hyperplane on top of the scatter plot of the data and note the misclassification error. Would you call this a good classifier? Why or why not?

(f) Experiment with various nonlinear functions ϕ that map the data points from $\mathbf{x}_i \in \mathbb{R}^{p=2}$ to $\phi(\mathbf{x}_i) \in \mathbb{R}^{P=3}$. Plot the remapped data using a 3D scatter plot to visualize whether your nonlinear function is successful in making the data near linearly separable.

(g) When you have found an appropriate transform ϕ, use the same implementation to fit an SVM to the `Train.mat` data after transforming each data point from $\mathbf{x}_i$ to $\phi(\mathbf{x}_i)$. How does the resulting misclassification error compare to your SVM fit without transforming the data? Optional (but highly recommended): Plot the optimal hyperplane overlaid on the 3D scatter plot.

2 Hard Margin Kernel SVM

The hard margin SVM binary classifier assigns class label $c(\hat{\mathbf{x}}) = \text{sign}(\langle \mathbf{w}^*, \hat{\mathbf{x}} \rangle + b^*)$ to an unlabeled observation $\hat{\mathbf{x}}$, where the weights $\mathbf{w}^*$ and bias b^* are learned from training data $\{(y_i \in \{+1, -1\}, \mathbf{x}_i \in \mathbb{R}^p)\}_{i=1}^n$ as follows:

$$\mathbf{w}^*, b^* = \underset{\mathbf{w},b}{\text{argmin}} \ \|\mathbf{w}\|_2^2 \text{ subject to } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, \ i = 1, \ldots, n \tag{1}$$

Unfortunately, this classifier only works when the data is linearly separable in raw feature space. In this problem, we will expand the capabilities of this classifier to cases where the data is separable in raw feature space in a non-linear fashion.

(a) One can show that there exist $\{\lambda_i\}_{i=1}^n$ such that the following optimization problem is equivalent to (1):

$$\mathbf{w}^*, b^* = \underset{\mathbf{w},b}{\text{argmin}} \ \|\mathbf{w}\|_2^2 + \sum_{i=1}^n \lambda_i \left(1 - y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)\right) \tag{2}$$

In this formulation, find a closed-form expression for the optimal $\mathbf{w}^*$ and show that there exist $\{\alpha_i\}_{i=1}^n$ such that $\mathbf{w}^* = \sum_{i=1}^n \alpha_i \mathbf{x}_i$.

(b) Using the result from the previous part, show that the class label $c(\hat{\mathbf{x}})$ can be written as a function of the inner product of $\hat{\mathbf{x}}$ and the training data points $\{\mathbf{x}_i\}_{i=1}^n$, i.e., that training and testing an SVM only depends on the data through inner products between data points.

(c) How does the computational cost of assigning $c(\hat{\mathbf{x}})$ scale with p?

(d) Let $\phi : \mathbb{R}^p \to \mathbb{R}^P$ be a non-linear transformation defined as

$$\mathbf{x} = \begin{bmatrix} x[1] \\ \vdots \\ x[p] \end{bmatrix} \mapsto \begin{bmatrix} x[1]^2 \\ \vdots \\ x[p]^2 \\ \sqrt{2}x[1]x[2] \\ \sqrt{2}x[1]x[3] \\ \vdots \\ \sqrt{2}x[1]x[p-1] \\ \sqrt{2}x[1]x[p] \\ \sqrt{2}x[2]x[3] \\ \vdots \\ \sqrt{2}x[2]x[p] \\ \vdots \\ \vdots \\ \sqrt{2}x[p-1]x[p] \end{bmatrix} = \begin{bmatrix} x_\phi[1] \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ x_\phi[P] \end{bmatrix} = \phi(\mathbf{x}).$$

Find P as a function of p.

(e) Suppose that the training data is not linearly separable, but the transformed training data $\{(y_i, \phi(\mathbf{x}_i) \in \mathbb{R}^P)\}_{i=1}^n$ is. We can now learn a hard margin SVM

$$\tilde{\mathbf{w}}^*, \tilde{b}^* = \underset{\mathbf{w},b}{\operatorname{argmin}} \ \|\mathbf{w}\|_2^2 \text{ subject to } y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b) \geq 1, \ i = 1, \ldots, n$$

and classify unlabeled observations as $c(\hat{\mathbf{x}}) = c_\phi(\hat{\mathbf{x}}) = \text{sign}(\langle \tilde{\mathbf{w}}^*, \phi(\hat{\mathbf{x}}) \rangle + \tilde{b}^*)$. Applying the result in part (b), we can write $c_\phi(\hat{x})$ as a function of the inner product of $\phi(\hat{\mathbf{x}})$ and the transformed training data. Using the result from the previous part, how does the computational cost of assigning $c(\hat{\mathbf{x}}) = c_\phi(\hat{\mathbf{x}})$ scale with p? How much worse is it than what you found in part (c), where the data is not transformed?

(f) Using the multinomial theorem, it can be shown that, for $\mathbf{u}, \mathbf{v} \in \mathbb{R}^p$,

$$\begin{aligned}
\langle \phi(\mathbf{u}), \phi(\mathbf{v}) \rangle &= \sum_{i=1}^{P} \phi(u[i])\phi(v[i]) \\
&= \sum_{i=1}^{p} (u[i]^2)(v[i]^2) + \sum_{i=2}^{p} \sum_{j=1}^{i-1} (\sqrt{2}u[i]u[j])(\sqrt{2}v[i]v[j]) \\
&= \left(\sum_{i=1}^{p} u[i]v[i] \right)^2 \\
&= (\langle \mathbf{u}, \mathbf{v} \rangle)^2.
\end{aligned}$$

Using this fact and the results in the previous parts, describe how one can fit an SVM to nonlinearly separable data without computing any transformation of the data points. Provide an expression for $c_\phi(\hat{\mathbf{x}})$ that classifies an unlabeled observation in P-dimensional space without computing $\phi(\mathbf{x})$ directly, and compare the computational complexity of this method to that of part (e), where the transform ϕ is computed for each data point.

3 SVM Classification of Speech Emotion

Using `sklearn`, train an SVM on the final project data with any combination of input features and hyperparameters you see fit and submit its predictions for the test data on Kaggle.

Submission Instructions

Every student must submit their work in PDF format, providing intermediate and final results as well as any necessary code. Submit your homework on Gradescope.

Collaboration Policy

Collaboration both inside and outside class is encouraged. You may talk to other students for general ideas and concepts, but individual write-ups must be done independently.

Plagiarism

Plagiarism of any form will not be tolerated. You are expected to credit all sources explicitly.