Problem Set 2: Kernels, SVMs, and Theory

Eitan Joseph Caroline Wang

June 30, 2020

1 Problem 1

Kernel ridge regression

In contrast to ordinary least squares which has a cost function

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(\theta^T x^{(i)} - y^{(i)}\right)^2 \tag{1}$$

we can also add a term that penalizes large weights in θ. In *ridge regression*, our least squares cost is regularized by adding a term $\lambda||\theta||^2$, where $\lambda > 0$ is a fixed (known) constant (regularization will be discussed at greater length in an upcoming course lecutre). The ridge regression cost function is then

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(\theta^T x^{(i)} - y^{(i)}\right)^2 + \frac{\lambda}{2}||\theta||^2$$

(a) Use the vector notation described in class to find a closed-form expreesion for the value of θ which minimizes the ridge regression cost function.
answer:

We know that the cost function without the penalty term can be written in closed form as

$$\frac{1}{2}\left(X\theta - \vec{y}\right)^T\left(X\theta - \vec{y}\right)$$

We can then rewrite the penalty term to be

$$\frac{\lambda}{2}\theta^T\theta$$

Therefore, together with the penalty term, the $J(\theta)$ will be as follows:

$$J(\theta) = \frac{1}{2}\left(X\theta - \vec{y}\right)^T\left(X\theta - \vec{y}\right) + \frac{\lambda}{2}\theta^T\theta$$

So we can now evaluate the gradient respect to $J(\theta)$:

$$\nabla_\theta J(\theta) = X^T X\theta - X^T\vec{y} + \lambda\theta$$

Using the knowledge from lecture 2 that

$$\nabla_\theta \frac{1}{2}\left(X\theta - \vec{y}\right)^T\left(X\theta - \vec{y}\right) = X^T X\theta - X^T\vec{y}$$

To find the optimization point, we want to set the gradient vector to zero

$$\vec{0} = X^T X\theta - X^T \vec{y} + \lambda\theta$$

And then solve for θ to get

$$\theta = \left(X^T X + \lambda I\right)^{-1} X^T \vec{y}$$

(b) Suppose that we want to use kernels to implicitly represent our feature vectors in a high-dimensional (possibly infinite dimensional) space. Using a feature mapping ϕ, the ridge regression cost function becomes

$$J(\theta) = \frac{1}{2}\left(X\theta - \vec{y}\right)^T \left(X\theta - \vec{y}\right) + \frac{\lambda}{2}\theta^T\theta$$

Making a prediction on a new input x_{new} would now be done by computing $\theta^T\phi(x_{new})$. Show how we can use the "kernel trick" to obtain a closed form for the prediction on the new input without ever explicitly computing $\phi(x_{new})$. You may assume that the parameter vector θ can be expressed as a linear combination of the input feature vectors; i.e.$\sum_{i=1}^m \alpha_i\phi(x^{(i)})$ for some set of parameters α_i.

Answer:
From part a, we know that the optimized θ can be written as:

$$\theta = \left(X^T X + \lambda I\right)^{-1} X^T \vec{y}$$

By using the identity $\left(\lambda I + BA\right)^{-1} B = B\left(\lambda I + AB\right)^{-1}$ we can rewrite θ as:

$$\theta = X^T\left(XX^T + \lambda I\right)^{-1}\vec{y}$$

The kernel algorithm maps the feature matrix (training data set matrix) to a higher dimension before applying the linear classification algorithm. Therefore, we will denote the new feature matrix Φ with columns $\phi(x^{(i)})$.

Then we can use our new feature matrix to redefine θ as follows:

$$\theta = \Phi^T\left(\Phi\Phi^T + \lambda I\right)^{-1}\vec{y}$$

We know that the kernel matrix is the covariance matrix of Φ with itself: $K = \Phi\Phi^T$. Therefore:

$$\theta = \Phi^T\left(K + \lambda I\right)^{-1}\vec{y}$$

Given that the prediction on x_{new} denoted $y_{new} = \theta^T\phi(x_{new})$, we can substitute for θ and solve

$$y_{new} = y^T\left(K + \lambda I\right)^{-1}\Phi\phi(x_{new})$$

To use the assumption that that the parameter vector θ can be expressed as a linear combination of the input feature vectors, we rewrite θ in the form $\sum_{i=1}^m \alpha_i\phi(x^{(i)})$ by defining some feature set α.

We can define the set of parameters α to be: $\left(K + \lambda I\right)^{-1} y$. Thus,

$$y_{new} = \sum_{i=1}^{m} \alpha_i\phi(x^{(i)})^T\phi(x_{new})$$

Finally, we can use the fact that the kernel function is defined as $K(x^{(i)}, x_{new}) = \phi(x^{(i)})^T \phi(x_{new})$ to rewrite the equation as

$$y_{new} = \sum_{i=1}^{m} \alpha_i K(x^{(i)}, x_{new})$$

2 Problem 2

l_2 norm soft margin SV

In class, we saw that if our data is not linearly separable, then we need to modify our support vector machine algorithm by introducing an error margin that must be minimized. Specifically, the formulation we have looked at is known as the l_1 norm soft margin SVM. In this problem we will consider an alternative method, known as the l_2 norm soft margin SVM. This new algorithm is given by the following optimization problem (notice that the slack penalties are now squared:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}||w||^2 + \frac{C}{2}\sum_{i=1}^{m} x_i^2 \\ \text{s.t.} \quad & y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, i = 1, \dots m \end{aligned}$$

(a) Notice that we have dropped the $\xi_i \geq 0$ constraint in the l_2 problem. Show that these non-negativity constraints can be removed. That is, show that the optimal value of the objective will be the same whether or not these constraints are present.

answer:
For any $\xi_i < 0$ that satisfies the convex constraint, the value $\xi_i = 0$ also satisfies the constraint and minimizes the objective function.

(b) What is the Lagrangian of the l_2 soft margin SVM optimization problem?

$$\mathcal{L}(w, b, \alpha, \xi) = \frac{1}{2} w^T w + \frac{C}{2}\sum_{i=1}^{m} \xi_i^2 - \sum_{i=1}^{m} \alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1 + \xi_i]$$

(c) Minimize the Lagrangian with respect to w, b, and ξ by taking the following gradients: $\nabla_w \mathcal{L}$, $\frac{\partial}{\partial b}\mathcal{L}$, and $\nabla_\xi \mathcal{L}$, and then setting them equal to 0.

answer:

$$\nabla_w \mathcal{L} = w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \overset{set}{=} 0$$

By using the fact that we set the gradient to 0 we can solve for w

$$w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}$$

And solve for b by taking the partial derivative

$$\frac{\partial}{\partial b}\mathcal{L} = \sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

And finally solve for ξ by taking the gradient

$$\begin{aligned}\nabla_\xi \mathcal{L} =& C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i = 0\\ C\sum_{i=1}^{m}\xi_i =& \sum_{i=1}^{m}\alpha_i\end{aligned}$$

(d) What is the dual of the l_2 soft margin SVM optimization problem?

answer: To compute the dual problem, we need to reparameterize the Lagrangian as a function of w on α

$$\begin{aligned}W(\alpha) =& \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}(\alpha_i y^{(i)}x^{(i)})^T(\alpha_j y^{(j)}x^{(j)}) + \frac{1}{2}\sum_{i=1}^{m}\frac{\alpha_i}{\xi_i}\xi_i^2\\ & - \sum_{i=1}^{m}\alpha_i\left[y^{(i)}\left(\left(\sum_{j=1}^{m}(\alpha_j y^{(j)}x^{(j)})\right)^T x^{(i)} + b\right) - 1 + \xi_i\right]\\ =& \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^T x^{(j)} + \frac{1}{2}\sum_{i=1}^{m}\alpha_i\xi_i - \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^T x^{(j)}\\ & - b\sum_{i=1}^{m}a_i y^{(i)} + \sum_{i=1}^{m}a_i - \sum_{i=1}^{m}\alpha_i\xi_i\\ =& \sum_{i=1}^{m}a_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^T x^{(j)} - \frac{1}{2}\sum_{i=1}^{m}\alpha_i\xi_i\\ =& \sum_{i=1}^{m}a_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^T x^{(j)} - \frac{1}{2}\sum_{i=1}^{m}\frac{\alpha_i^2}{C}\end{aligned}$$

Now to formulate the dual optimization problem, we simply add the constraints derived in part c to our objective function

$$\begin{aligned}\max_{\alpha}\quad & \sum_{i=1}^{m}a_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^T x^{(j)} - \frac{1}{2}\sum_{i=1}^{m}\frac{\alpha_i^2}{C}\\ s.t.\quad & \sum_{i=1}^{m}a_i y^{(i)} = 0\end{aligned}$$

3 Problem 3

SVM with Gaussian kernel

Consider the task of training a support vector machine using the Gaussian kernel $K(x,z) = \exp(-||x-z||^2/\tau^2)$. We will show that as long as there are no two identical points in the training set, we can always find a value for the bandwidth parameter τ such that the SVM achieves zero training error.

(a) Recall from class that the decision function learned by the support vector machine can be written

as

$$f(x) = \sum_{i=1}^{m} \alpha_i y^{(i)} K(x^{(i)}, x) + b$$

Assume that the training data $f\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$ consists of points which are separated by at least a distance of ϵ; that is, $||x^{(j)} - x^{(i)}|| \geq \epsilon$ for any $i \neq j$. Find values for the set of parameters $\{\alpha_1, \ldots, \alpha_m, b\}$ and Gaussian kernel width τ such that $x^{(i)}$ is correctly classified, for all $i = 1, \ldots, m$. [Hint: Let $\alpha_i = 1$ for all i and $b = 0$. Now notice that for $y \in \{-1, 1\}$ the prediction on $x^{(i)}$ will be correct if $|f(x^{(i)}) - y^{(i)}| < 1$, so find a value of τ that satisfies this inequality for all i.

answer:

We will first let $\alpha_i = 1 \forall i$ and we will let $b = 0$. We notice here that since $y \in \{-1, +1\}$, the prediction on $x^{(i)}$ will be correct if $\left| f\left(x^{(i)}\right) - y^{(i)} \right| < 1$.

Now we can substitute in our $f\left(x^{(i)}\right)$) to get the equation

$$\begin{aligned} \left| f\left(x^{(i)}\right) - y^{(i)} \right| &= \left| \left(\sum_{k=1}^{m} \alpha_k y^{(k)} K(x^{(k)}, x^{())} + b \right) - y^{(i)} \right| < 1 \\ &\Longrightarrow \left| \left(\sum_{k=1}^{m} y^{(k)} K(x^{(k)}, x^{())} \right) - y^{(i)} \right| < 1 \end{aligned}$$

Substituting the Gaussian Kernel for K gives

$$\left| \left(\sum_{k=1}^{m} y^{(k)} \exp\left(-\frac{||x^{(k)} - x^{(i)}||^2}{\tau^2} \right) \right) - y^{(i)} \right| < 1$$

By pulling out $y^{(i)}$ from the sum we get

$$\left| y^{(i)} + \left(\sum_{\substack{k=1 \\ k \neq i}}^{m} y^{(k)} \exp\left(-\frac{||x^{(k)} - x^{(i)}||^2}{\tau^2} \right) \right) - y^{(i)} \right| < 1$$

$$\Longrightarrow \left| \sum_{\substack{k=1 \\ k \neq i}}^{m} y^{(k)} \exp\left(-\frac{||x^{(k)} - x^{(i)}||^2}{\tau^2} \right) \right| < 1$$

Since each $y^{(k)}$ is either -1 or $+1$ in the worse case for the inequality all $y^{(k)}$s are the same. Essentially, $\sum_{k=1}^{m} y^{(i)} \leq |\sum_{k=1}^{m} y^{(i)}|$, and since $||x^{(j)} - x^{(i)}|| \geq \epsilon$ for any $i \neq j$, we can write

$$\left| \sum_{\substack{k=1 \\ k \neq i}}^{m} y^{(k)} \right| \exp\left(-\frac{\epsilon^2}{\tau^2} \right) < 1$$

$$\Longrightarrow (m-1) \exp\left(-\frac{\epsilon^2}{\tau^2} \right) < 1$$

Then we can solve for τ:

$$\tau < \frac{\epsilon}{\sqrt{\log(m-1)}}$$

For simplification, we can choose:

$$\tau = \frac{\epsilon}{\log m}$$

(b) Suppose we run a SVM with slack variables using the parameter τ you found in part (a). Will the resulting classifier necessarily obtain zero training error? Why or why not? A short explanation (without proof) will suffice.

answer:

The classifier will obtain zero training error. We can verify this by observing that the SVM algorithm WITHOUT any slack variable will achieve zero training error if it can satisfy the convex constraint. If we can find a solution that satisfies the convex constraint without a slack variable, then we have proved that the classifier will still obtain zero training error. This is because in order to minimize the objective function, the algorithm will necessarily choose the slack variables to be equal to zero if possible.

To do this we first observe that our convex constraint is

$$y^{(i)}\left(w^T x^{(i)} + b\right) = y^{(i)} f(x^{(i)}) > 1$$

By using our τ from part (a) we can ensure that $y^{(i)} f(x \sup i) > 0 \ \forall i$ (meaning that every classification of $x^{(i)}$ is correct), and therefore we can take each α_i to be large enough to satisfy the inequality. Since we can satisfy this constraint, it is clear that we can satisfy the same constraint with a slack variable.

(c) Suppose we run the SMO algorithm to train an SVM with slack variables, under the conditions stated above, using the value of τ you picked in the previous part, and using some arbitrary value of C (which you do not know beforehand). Will this necessarily result in a classifier that achieve zero training error? Why or why not? Again, a short explanation is sufficient.

answer:

The classifier will not be able to obtain zero training error because there exists a constant C in front of $(C \sum_{i=1}^{m} \xi_i)$ for which we have no information on. If the constant C happened to be ≤ 0, then the minimization of the objective function could be achieved with non-zero slack variables.

4 Problem 5

Uniform convergence
In class we proved that for any finite set of hypotheses $\mathcal{H} = \{h_1, \ldots, h_k\}$, if we pick the hypothesis $\hat{h}$ that minimizes the training error on a set of m examples, then with probability at least $1 - \delta$,

$$\epsilon(\hat{h}) \leq \left(\min_i \epsilon(h_i)\right) + 2\sqrt{\frac{1}{2m} \log \frac{2k}{\delta}}$$

where $\epsilon(h_i)$ is the generalization error of hypothesis h_i. Now consider a special case (often called the *realizable* case) where we know, a priori, that there is some hypothesis in our class $\mathcal{H}$ that achieves zero error on the distribution from which the data is drawn. Then we could obviously just use the above bound with $\min_i \epsilon(h_i) = 0$; however, we can prove a better bound than this.

(a) Consider a learning algorithm which, after looking at m training examples, chooses some hypothesis $\hat{h} \in \mathcal{H}$ that makes zero mistakes on this training data. (By our assumption, there is at least one such hypothesis, possibly more.) Show that with probability $1 - \delta$

$$\epsilon(\hat{h}) \leq \frac{1}{m} \log \frac{k}{\delta}$$

Notice that since we do not have a square root here, this bound is much tighter. [Hint: Consider the probability that a hypothesis with generalization error greater than γ makes no mistakes on the training data. Instead of the Hoeffding bound, you might also find the following inequality useful: $(1 - \gamma)m \leq e^{-\gamma m}$].

answer:

The problem states that the probability that h incorrectly predicts some training example $(x^{(i)}, y^{(i)})$ from the distribution D is γ. Therefore we have that

$$P(h \in \mathcal{H} \text{ that h predicts correctly}) = 1 - \gamma$$

For h to predict correctly m times, once for each training example, we have

$$P(h \in \mathcal{H} \text{ that h predicts correctly m times}) = (1 - \gamma)^m \leq e^{-\gamma m}$$

Since there are total k hypothesis in $\mathcal{H}$,

$$P(\forall h \in \mathcal{H} \text{ that h predicts correctly m times}) = k(1 - \gamma)^m \leq ke^{-\gamma m}$$

After setting this probability to δ and solving for γ we obtain the following:

$$ke^{-\gamma m} = \delta$$
$$\implies \gamma = \frac{1}{m} \log \frac{k}{\delta}$$

From the lecture we know that $|\hat{\epsilon}(\hat{h}) - \epsilon(\hat{h})| > \gamma$ with probability $1 - \delta$, therefore since $\hat{\epsilon}(\hat{h}) = 0$ we obtain the inequality

$$\epsilon(\hat{h}) \leq \frac{1}{m} \log \frac{k}{\delta}$$

(b) Rewrite the above bound as a sample complexity bound, i.e., in the form: for fixed δ and γ, for $\epsilon(\hat{h}) \leq \gamma$ to hold with probability at least $(1 - \delta)$, it suffices that $m \leq f(k, \gamma, \delta)$. *answer:*

From part (a) we have that $\gamma \leq \frac{1}{m} \log \frac{k}{\delta}$ We now simply need to rewrite this as an inequality on m as follows

$$m \leq \frac{1}{\gamma} \log \frac{k}{\delta}$$