CS 475 Machine Learning: Homework 2
Supervised Classifiers 2
Solution

1 Analytical (50 points)

The following problems consider a standard binary classification setting: we are given n observations with m features, $x_1, ..., x_n \in \mathbb{R}^m$.

1) Overfitting (10 points) Consider a kernel logistic regression classifier. The classifier has regularization parameter λ, as well as a kernel parameter γ which is used by a non-linear kernel (the exact kernel is unimportant. These parameters can be determined by tuning on held-out development data, or by using cross validation.

(a) Suppose we use cross validation to determine λ and γ and find that the values (λ_1, γ_1) and (λ_2, γ_2) achieve the same cross validation error. What other factors should you consider in determining which are the appropriate parameters to select for the final trained model?

Here are a few possible correct answers.

(I) Since the cross validation errors are the same, we can randomly pick one of them. They are expected to have similar performance on the testing set.

(II) Since we use a logistic regression classifier, we can also calculate the likelihood of the validation data set. Then we choose the combination of the parameters with the larger likelihood.

(III) Since the cross validation error depends on a random partition of the data, we can create another random partition of data for another round of cross validation. Then we choose the combination of the parameters with the smaller classification error.

(IV) We can choose parameters that are more stable, in that small variations to the parameter don't significantly change the held-out accuracy.

$\boxed{\text{There are many possible answers to this question, and we'll accept any correct answer.}}$

(b) Just as with SVMs, we can fit a linear logistic regression classifier using either the primal or dual form. As we know, the primal form has m parameters to learn (number of features), while the dual form has n parameters to learn (number of examples). When $m \gg n$, will the dual form reduce over-fitting since it has fewer parameters? Explain your answer.

No, the dual will not reduce over-fitting. The dual formulation will give you the same solution as the primal.

$\boxed{\text{5 points for each}}$

2) Hinge Loss (12 points) Linear SVMs using a square hinge loss can be formulated as an unconstrained optimization problem:

$$\hat{w} = \arg\min_w \sum_{i=1}^{n} H(y_i(w^T x_i)) + \lambda \|w\|_2^2, \tag{1}$$

where λ is the regularization parameter and $H(a) = \max(1-a, 0)^2$ is the square hinge loss function. The hinge loss function can be viewed as a convex surrogate of the 0/1 loss function $I(a \leq 0)$.

(a) Compared with the standard hinge loss function, what do you think are the advantages and disadvantages of the square hinge loss function?

The square hinge loss is smooth, and therefore easier to optimize than the nonsmooth hinge loss. However, due to its square growth, it tends to be less robust to outliers.

$\boxed{\text{3 points for getting both correct. 2 points for only getting one correct.}}$

(b) Prove that $H(a)$ is a convex function of a.

If we have $\lambda a + (1-\lambda)b \leq 1$, where $\lambda \in [0,1]$. One can verify

$$\begin{aligned}
\max(1 - [\lambda a + (1-\lambda)b], 0)^2 &= (1 - [\lambda a + (1-\lambda)b])^2 \\
&= [\lambda(1-a) + (1-\lambda)(1-b)]^2 \\
&\leq \lambda(1-a)^2 + (1-\lambda)(1-b)^2 \\
&\leq \lambda \max(1-a,0)^2 + (1-\lambda)\max(1-b,0)^2.
\end{aligned}$$

For $a, b \geq 1$, we have $\lambda a + (1-\lambda)b \geq 1$. One can verify

$$\max(1 - [\lambda a + (1-\lambda)b], 0)^2 = 0 \leq \lambda \max(1-a,0)^2 + (1-\lambda)\max(1-b,0)^2..$$

$\boxed{\text{3 points. Partial credit for a correct proof outline with mistakes.}}$

(c) The function $L(a) = \max(-a, 0)^2$ can also approximate the 0/1 loss function. What is the disadvantage of using this function instead?

$L(a)$ does not penalize correctly classified samples. Therefore, it does not push correctly classified samples away from the decision boundary, and fails to create a margin.

$\boxed{\text{3 points.}}$

(d) We can choose a different loss function $H'(a) = \max(0.5 - a, 0)^2$. How will switching to H' from H effect the solution of the objective function? Specifically, the new objective becomes:

$$\hat{w}' = \arg\min \sum_{i=1}^{n} H'(y_i(w^T x_i)) + \lambda' \|w\|_2^2. \tag{2}$$

Explain your answer in terms of the relationship between λ and λ'.

$$\begin{aligned}\hat{w}' &= \arg\min \sum_{i=1}^{n} H'(y_i(w^T x_i)) + \lambda' \|w\|_2^2 \\ &= \arg\min 2 \sum_{i=1}^{n} H'(y_i(w^T x_i)) + 2\lambda' \|w\|_2^2 \\ &= \arg\min \sum_{i=1}^{n} \max(1 - 2y_i w^T x_i, 0) + \frac{\lambda'}{2} \|2w\|_2^2.\end{aligned}$$

Compared with (1), we know that by setting $\lambda' = \frac{\lambda}{2}$, we have $2\hat{w}' = \hat{w}$. Therefore, (2) yields the same SVM as (1). The only difference is the scale, which does not affect the classification.

$\boxed{\text{3 points.}}$

3) Kernel Trick (10 points) The kernel trick extends SVMs to learn nonlinear functions. However, an improper use of a kernel function can cause serious over-fitting. Consider the following kernels.

(a) Inverse Polynomial kernel: given $\|x\|_2 \leq 1$ and $\|x'\|_2 \leq 1$, we define $K(x, x') = 1/(d - x^\top x')$, where $d \geq 2$. Does increasing d make over-fitting more or less likely?

Decreasing d tends to encourage overfitting. To see why this is the case, let us consider an example: We can view kernel function as a similarity function. A more similar pair of samples has a larger value. We choose d to be 1000, which leads to $K(x, x') \approx 1/1000$ and $K(x, x) \approx 1/1000$. This means that all samples are similar. The resulting SVM will treat all samples equally, and cause underfitting.

$\boxed{\text{3 points}}$

(b) Chi squared kernel: Let x_j denote the j-th entry of x. Given $x_j > 0$ and $x'_j > 0$ for all j, we define $K(x, x') = \exp\left(-\sigma \sum_j \frac{(x_j - x'_j)^2}{x_j + x'_j}\right)$, where $\sigma > 0$. Does increasing σ make over-fitting more or less likely?

Increasing σ tends to encourage overfitting. The intuition behind is similar to (a). We choose $\sigma = 10^10$, which leads to $K(x, x') \approx 0$ and $K(x, x) \approx 1$. This means that a sample is only similar to itself, and not similar to others at all. This is essentially an overfitting.

But when we choose 10^{-10}, we have $K(x, x') \approx 1$ and $K(x, x) \approx 1$. This means that all samples are similar. The resulting SVM will treat all samples equally, and cause underfitting. $\boxed{\text{3 points}}$

We say K is a kernel function, if there exists some transformation $\phi : \mathbb{R}^m \rightarrow \mathbb{R}^{m'}$ such that $K(x_i, x_{i'}) = \langle \phi(x_i), \phi(x_{i'}) \rangle$.

(c) Let K_1 and K_2 be two kernel functions. Prove that $K(x_i, x_{i'}) = K_1(x_i, x_{i'}) + K_2(x_i, x_{i'})$ is also a kernel function.

$$\begin{aligned}K_1(x_i, x_{i'}) + K_2(x_i, x_{i'}) &= \langle \phi_1(x_i), \phi_1(x_{i'}) \rangle + \langle \phi_2(x_i), \phi_2(x_{i'}) \rangle \\ &=< [\phi_1(x_i)^T, \phi_2(x_i)^T]^T, \phi_1(x'_i)^T, \phi_2(x'_i)^T]^T > \\ &= K(x_i, x_{i'}).\end{aligned}$$

$\boxed{\text{4 points.}}$

4) Predictions with Kernel (6 points) Let us compare primal linear SVMs and dual linear kernel SVMs in terms of computational complexity at prediction time.

(a) What is the computational complexity of prediction of a primal linear SVM in terms of the numbers of the training samples n and features m? If each sample contains at most q nonzero features, what can we do to accelerate the prediction? What is the resulting computational complexity?

$O(m)$; We can ignore zero entries when calculating vector inner products;$O(q)$

(b) What is the computational complexity of prediction of a dual kernel SVM in terms of the numbers of the training samples n, features m, and support vectors s? If each sample contains at most q nonzero features, what can we do to accelerate the prediction? What is the resulting computational complexity?

$O(sm)$; We can ignore zero entries when calculating vector inner products; $O(sq)$

5) Dual Perceptron (6 points)

(a) You train a Perceptron classifier in the primal form on an infinite stream of data. This stream of data is not-linearly separable. Will the Perceptron have a bounded number of prediction errors?

No. Since the data stream is not linearly separable. The perceptron will keep making classification mistakes.

$\boxed{\text{2 points}}$

(b) Switch the primal Perceptron in the previous step to a dual Perceptron with a linear kernel. After observing T examples in the stream, will the two Perceptrons have learned the same prediction function?

Yes. The primal and dual provide the same solutions.

$\boxed{\text{2 points}}$

(c) What computational issue will you encounter if you continue to run the dual Perceptron and allow T to approach ∞? Will this problem happen with the primal Perceptron? Why or why not?

The dimensionality of the kernel matrix goes to ∞, which make the computation intractable. The primal Perceptron works in $\mathbb{R}^M$. Thus the sample size does not matter.

$\boxed{\text{2 points}}$

6) Robust SVM (6 points) For SVMs, the hinge loss can deal with some examples that are not linearly separable. However in some cases, these examples may be outliers: data points that are so inconsistent with the rest of the data that we suspect they are incorrect. A good classifier should be robust to those outliers. Is it possible for us to modify the hinge loss to achieve our goal? What is the disadvantage? (Hint: is it possible to choose a nonconvex function?)

Yes, we can modify the hinge loss as follows: $H(a) = \min(r, \max(1 - a, 0))$, where $r \geq 1$.

The loss for a misclassified sample is bounded by r, making it robust to outliers. However, this loss function leads to a nonconvex optimization problem. Thus, no algorithm can guarantee a global optimum in polynomial time.

$\boxed{\text{There are many possible answers to this question, and we'll accept any correct answer.}}$