Machine Learning, 2024 Spring
Assignment 7

Name: Zhou Shouchen

Student ID: 2021533042

Notice

Plagiarizer will get 0 points.
LaTeXis highly recommended. Otherwise you should write as legibly as possible.

Problem 1 Referring to Figure 4.10, why are both curves increasing with K? Why do they converge to each other with increasing K? (20pt)

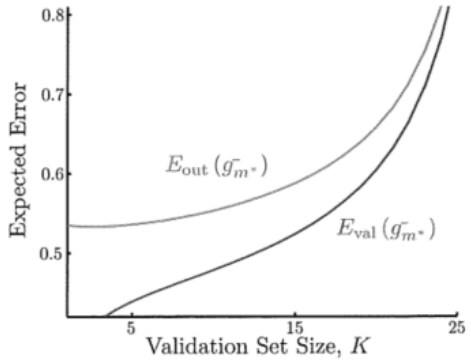

Figure 4.10: Optimistic bias of the validation error when using a validation set for the model selected.

Solution

1. Since the size of the total data is fixed, so as K increases, there are less points used for training. We can assume that are data points are useful data, so less data points used for training will lead to a worse model. And it is reasonable that the performence of the model both on training set and validation set will decrease. i.e. they all have a larger expected error.

So above all, as K increases, the both curves increase.

2. From what we have learned, we can get that

$$E_{\text{out}}(g_{m^*}^-) \leq E_{\text{val}}(g_{m^*}^-) + O\left(\sqrt{\frac{\ln M}{K}}\right)$$

Where $M = \mathcal{H}_{\text{val}}$, K is the size of validation set.

Since M is fixed, so the difference between $E_{\text{out}}(g_{m^*}^-)$ and $E_{\text{val}}(g_{m^*}^-)$ is $O\left(\sqrt{\frac{\ln M}{K}}\right)$, which decreases as K increases.

So above all, as K increases, the difference of the two curves decreases, and they converge to each other.

Problem 2

1. From Figure 4.12, $\mathbb{E}[E_{out}(g_{m^*}^-)]$ is initially decreasing. How can this be, if $\mathbb{E}[E_{out}(g_{m^*}^-)]$ is increasing in K for each m? (10pt)
2. From Figure 4.12 we see that $\mathbb{E}[E_{out}(g_{m^*})]$ is initially decreasing, and then it starts to increase. What are the possible reasons for this? (10pt)
3. When $K = 1$, $\mathbb{E}[E_{out}(g_{m^*}^-)] < \mathbb{E}[E_{out}(g_{m^*})]$. How can this be, if the learning curves for both models are decreasing? (10pt)

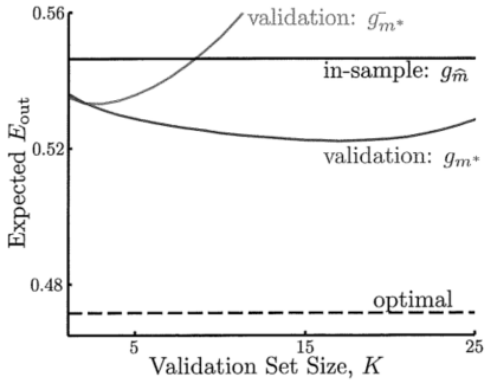

Figure 4.12: Model selection between $\mathcal{H}_2$ and $\mathcal{H}_5$ using a validation set. The solid black line uses E_{in} for model selection, which always selects $\mathcal{H}_5$. The dotted line shows the optimal model selection, if we could select the model based on the true out-of-sample error. This is unachievable, but a useful benchmark. The best performer is clearly the validation set, outputting g_{m^*}. For suitable K, even $g_{m^*}^-$ is better than in-sample selection.

Solution

1.

(a) Initally, the size of validation set K is quite small, so it would not be able to provide a good estimate of the generalization error. As K increases, the estimate of the generalization error becomes more accurate, so $\mathbb{E}[E_{out}(g_{m^*}^-)]$ initially decreases.

(b) Similarly to te Problem 1, the size of the total data is fixed, so as K is large enough, then as K increases, there are less points used for training. We can assume that are data points are useful data, so less data points used for training will lead to a worse model. And it is reasonable that the performence of the model both on training set and validation set will decrease. i.e. $\mathbb{E}[E_{out}(g_{m^*}^-)]$ increases.

So above all, as K increases, the $\mathbb{E}[E_{out}(g_{m^*}^-)]$ is initially decreasing, when K is large enough, $\mathbb{E}[E_{out}(g_{m^*}^-)]$ is increasing in K for each m as K increases.

2. The understanding of the notions:

The models are $\mathcal{H}_1, \cdots, \mathcal{H}_M$, we first train with training data $\mathcal{D}_{\text{train}}$ to train all models, and use the cross-validation to get the best model $\mathcal{H}_{m^*}$, with error $g_{m^*}^-$ on the validation set $\mathcal{D}_{\text{val}}$. Then the total dataset $\mathcal{D}$ is used for training the model $\mathcal{H}_{m^*}$, and the error on the test set $\mathcal{D}_{\text{test}}$ is g_{m^*}.

So with the understanding of notions, we can get that:

$\mathbb{E}[E_{out}(g_{m^*})]$ is only relevant to the model we selected. When K is small, as K increases, $\mathcal{D}$ is quite close to $\mathcal{D}_{\text{train}}$, so it is similar to $g_{m^*}^-$, so $\mathbb{E}[E_{out}(g_{m^*})]$ decrease as we can choose the good model with cross validation.

When K is large enough, as K increases, less useful data are in $\mathcal{D}_{\text{train}}$, although we can choose a good trained model among $\mathcal{H}_1, \cdots, \mathcal{H}_M$, but the best trained model is not good enough, so the error $\mathbb{E}[E_{out}(g_{m^*})]$ on the test set $\mathcal{D}_{\text{test}}$ is increasing.

So above all, $\mathbb{E}[E_{out}(g_{m^*})]$ decreases initially, and then it increases as K increases.

3. A possible reason is that $K = 1$ is too small that even though the models are well trained, the g_{m^*} could have a small E_{in}, but could not make sure that E_{out} is small.
But $g_{m^*}^-$ is trained with cross-validation, so it is more likely to have a good E_{out} as it is selecting the best model on $\mathcal{D}_{\text{val}}$. So we have when $K = 1$,

$$\mathbb{E}[E_{out}(g_{m^*}^-)] < \mathbb{E}[E_{out}(g_{m^*})]$$

Problem 3

Definition 1 (leave-one-out cross-validation) *Select each training example in turn as the single example to be held-out, train the classifier on the basis of all the remaining training examples, test the resulting classifier on the held-out example, and count the errors.*

Let the superscript -i denote the parameters we would obtain by finding the SVM classifier f without the ith training example. Define the *leave-one-out CV error* as

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})), \tag{1}$$

where $\mathcal{L}$ is the zero-one loss. Prove that

$$\text{leave-one-out CV error} \leq \frac{\text{number of support vectors}}{\text{n}} \tag{2}$$

(20pt)

Solution

Then 'Support Vectors' mentioned below are the points that are on the margin of SVM trained by all the training data.

Since we are applying Leave-one-out cross-validation, when leaving the i-th point $\mathbf{x}_i$ to the validation set, we have:

1. If $\mathbf{x}_i$ is not the support vector.
   Then the result of the new SVM would not change compared with the original one, so the error is
   $$\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})) = 0$$
2. If $\mathbf{x}_i$ is the support vector.
   - If $\mathbf{x}_i$ is the unique support vector.
     Then the result of the SVM would change, and the margin of the new SVM would increase, which means that $\mathbf{x}_i$ would be misclassified, so the error is
     $$\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})) = 1$$
   - If $\mathbf{x}_i$ is not the unique support vector.
     Then the result of the SVM would not change, so the error is
     $$\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})) = 0$$

   So combine all the cases, we could get that when $\mathbf{x}_i$ is the support vector, the error $\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})) \leq 1$.

So we could get that the leave-one-out CV error is

$$\begin{aligned}
\text{leave-one-out CV error} &= \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})) \\
&\leq \frac{1}{n}\sum_{\mathbf{x}_i \text{is support vector}}\mathcal{L}(y_i, f(\mathbf{x}_i; \mathbf{w}^{-i}, b^{-i})) \\
&\leq \frac{1}{n}\sum_{\mathbf{x}_i \text{is support vector}} 1 \\
&= \frac{\text{number of support vectors}}{n}
\end{aligned}$$

So above all, we have proved that

$$\text{leave-one-out CV error} \leq \frac{\text{number of support vectors}}{\text{n}}$$

Problem 4

The l_1-norm SVM can be formulated as follows

$$\begin{aligned} &\min_{\mathbf{w},b} \|\mathbf{w}\|_1 \\ &\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, i = 1, ..., N. \end{aligned} \tag{3}$$

Please deprive the equivalent linear programming formulation of (3) (10pt), give its dual formulation (10pt). Also, please explain how to determine the support vector SV according to the optimal multiplier. (10pt)

Solution

Suppose there are totally N sample points, and the dimension of the feature space is n. i.e. $\mathbf{w}, \mathbf{x}_i \in \mathbb{R}^n$.

(1) To convert the l_1-norm SVM into a linear programming problem, for each w_i, we can split it into w_i^+ and w_i^- to avoid the absolute value, which is nonlinear. And w_i^+ and w_i^- are both non-negative. i.e.

$$\begin{aligned} w_i^+ &= \max(w_i, 0) \\ w_i^- &= \max(-w_i, 0) \\ w_i &= w_i^+ - w_i^- \\ |w_i| &= w_i^+ + w_i^- \end{aligned}$$

Define $\mathbf{w}^+ = [w_1^+, w_2^+, ..., w_n^+]^T$, $\mathbf{w}^- = [w_1^-, w_2^-, ..., w_n^-]^T$, then we have $\mathbf{w} = \mathbf{w}+ - \mathbf{w}-$.
So the l_1-norm SVM can be reformulated as:

$$\begin{aligned} \min_{\mathbf{w}^+, \mathbf{w}^- \in \mathbb{R}^n, b \in \mathbb{R}} \quad & \sum_{i=1}^{n} (w_i^+ + w_i^-) \\ \text{s.t. } \quad & y_i \left[(\mathbf{w}^+ - \mathbf{w}^-)^T \mathbf{x}_i + b \right] \geq 1, i = 1, ..., N \\ & \mathbf{w}^+ \succeq 0 \\ & \mathbf{w}^- \succeq 0 \end{aligned}$$

(2) Define $\mathbf{1}_n = [1, 1, \cdots, 1]^T$(1 repeat n times).
Then the Lagrangian of the above linear programming problem is:

$$\begin{aligned} L(\mathbf{w}^+, \mathbf{w}^-, b, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}) &= \sum_{i=1}^{n} (w_i^+ + w_i^-) - \sum_{i=1}^{N} \lambda_i \left[y_i \left[(\mathbf{w}^+ - \mathbf{w}^-)^T \mathbf{x}_i + b \right] - 1 \right] - \boldsymbol{\mu}^T \mathbf{w}^+ - \boldsymbol{\nu}^T \mathbf{w}^- \\ &= \mathbf{1}_n^T \mathbf{w}^+ + \mathbf{1}_n^T \mathbf{w}^- - \sum_{i=1}^{N} \lambda_i \left[y_i \mathbf{x}_i^T \mathbf{w}^+ - y_i \mathbf{x}_i^T \mathbf{w}^- + y_i b - 1 \right] - \boldsymbol{\mu}^T \mathbf{w}^+ - \boldsymbol{\nu}^T \mathbf{w}^- \\ &= (\mathbf{1}_n - \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i - \boldsymbol{\mu})^T \mathbf{w}^+ + (\mathbf{1}_n + \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i - \boldsymbol{\nu})^T \mathbf{w}^- - \sum_{i=1}^{N} \lambda_i y_i b + \sum_{i=1}^{N} \lambda_i \end{aligned}$$

Where $\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu} \succeq 0$.
The dual objective function is:

$$g(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \min_{\mathbf{w}^+, \mathbf{w}^-, b} L(\mathbf{w}^+, \mathbf{w}^-, b, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$$

To get the $g(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$, we need to take the gradient of the primal variables and set them to zero. i.e.

$$\begin{cases} \dfrac{\partial L}{\partial \mathbf{w}^+} = \mathbf{1}_n - \sum\limits_{i=1}^{N} \lambda_i y_i \mathbf{x}_i - \boldsymbol{\mu} = 0 \\ \dfrac{\partial L}{\partial \mathbf{w}^-} = \mathbf{1}_n + \sum\limits_{i=1}^{N} \lambda_i y_i \mathbf{x}_i - \boldsymbol{\nu} = 0 \\ \dfrac{\partial L}{\partial b} = - \sum\limits_{i=1}^{N} \lambda_i y_i = 0 \end{cases}$$

So put these equations into the Lagrangian, we can get the dual objective function:

$$g(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \sum_{i=1}^{N} \lambda_i$$

So above all, the dual formulation of the l_1-norm SVM is:

$$\begin{aligned} \max_{\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\nu}} \ & \sum_{i=1}^{N} \lambda_i \\ \text{s.t.} \ & \boldsymbol{\lambda} \succeq 0 \\ & \boldsymbol{\mu} \succeq 0 \\ & \boldsymbol{\nu} \succeq 0 \\ & \sum_{i=1}^{N} \lambda_i y_i = 0 \end{aligned}$$

(3) To determine the support vector SV according to the optimal multiplier, we need to check the KKT conditions.
From the complementary condition of KKT, we have:

$$\lambda_i \left[y_i \left[(\mathbf{w}^+ - \mathbf{w}^-)^T \mathbf{x}_i + b \right] - 1 \right] = 0$$

For the support vectors $\mathbf{x}_i$, we know that they have

$$y_i \left[(\mathbf{w}^+ - \mathbf{w}^-)^T \mathbf{x}_i + b \right] = 1$$

And since $\lambda_i \geq 0$, so we have if $\lambda_i > 0$, then $\mathbf{x}_i$ is a support vector.

So above all, we check the values of the optimal multipliers $\boldsymbol{\lambda}$, and if $\lambda_i > 0$, then $\mathbf{x}_i$ is a support vector.