Lab 1 - Statistical Learning Theory

`{bengio,mkeller}@idiap.ch`
`http://www.idiap.ch/~{bengio,mkeller}`

November 25, 2005

1 Some theoretical derivation

1. **Show that the empirical risk is an unbiased estimate of the risk.**

$$
\begin{aligned}
E\left[\hat{R}(f,D)\right] &= E\left[\frac{1}{N}\sum_{i=1}^{N} L(f,Z_i)\right] \\
&= \frac{1}{N}\sum_{i=1}^{N} E\left[L(f,Z_i)\right], \quad Z_i s \text{ are independent} \\
&= \frac{1}{N}\sum_{i=1}^{N} E\left[L(f,Z)\right], \quad Z_i s \text{ are identically distributed} \\
&= \frac{1}{N} N E\left[L(f,Z)\right] \\
&= R(f)
\end{aligned} \tag{1}
$$

2. **Show that $\hat{R}(f^*(D_{train}), D_{test})$ is an unbiased estimate of the risk.**

$$E\left[\hat{R}(f^*(D_{train}), D_{test})\right] = R(f^*(D_{train})), \quad \text{see question 1.}$$

3. **Show the bias-variance-noise decomposition of the risk in a regression problem using mean squared loss function.** Let $Y = f(X) + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, and $f_D(X)$ an estimator of $f(X)$, learned over the training set D.

   The expected prediction error at a particular point $X = x_0$ is:

$$
\begin{aligned}
Err(x_0) &= E\left[(Y - f_D(x_0))^2 | X = x_0\right] \\
&= E\left[(Y - E[f_D(x_0)] + E[f_D(x_0)] - f_D(x_0))^2 | X = x_0\right] \\
&= E\left[(Y - E[f_D(x_0)])^2 | X = x_0\right] + E\left[(E[f_D(x_0)] - f_D(x_0))^2\right] \\
&\quad -2 \cdot E\left[(Y - E[f_D(x_0)]) \cdot (E[f_D(x_0)] - f_D(x_0)) | X = x_0\right]
\end{aligned} \tag{2}
$$

Given that: $E\left[(Y - E[f_D(x_0)]) \cdot (E[f_D(x_0)] - f_D(x_0))|X = x_0\right] = 0$

$$\begin{aligned}
Err(x_0) &= E\left[(Y - E[f_D(x_0)])^2|X = x_0\right] + Var\left[f_D(x_0)\right] \\
&= E\left[(f(x_0) + \epsilon - E[f_D(x_0)])^2\right] + Var\left[f_D(x_0)\right] \\
&= E\left[(f(x_0) - E[f_D(x_0)])^2\right] + E\left[\epsilon^2\right] + Var\left[f_D(x_0)\right] \\
&= \left[Bias\left[f_D(x_0)\right]\right]^2 + \sigma_\epsilon^2 + Var\left[f_D(x_0)\right]
\end{aligned} \tag{3}$$

Since,

$$\begin{aligned}
E\left[Err(x_0)\right] &= E\left[E\left[(Y - f_D(x_0))^2|X = x_0\right]\right] \\
&= E\left[(Y - f_D(X))^2\right] \\
&= R(f_D),
\end{aligned} \tag{4}$$

and $Err(x_i)$, $\quad \forall i$ are independent,

$$\frac{1}{N}\sum_{i=1}^{N} Err(x_i) = \frac{1}{N}\sum_{i=1}^{N}\left[\left[Bias\left[f_D(x_i)\right]\right]^2 + \sigma_\epsilon^2 + Var\left[f_D(x_i)\right]\right] \tag{5}$$

is an unbiased estimator of the risk $R(f_D) = E\left[(Y - f_D(X))^2\right]$.

4. **Show that the capacity of a set of linear discriminants of dimension d is at least $d + 1$.**

   Let $\mathcal{F} = \{f | \forall x \in \mathbb{R}^d, f(x) = \text{sign}(w \cdot x + b), w \in \mathbb{R}^d, b \in \mathbb{R}\}$.

   We will first show that the capacity of the set of linear discriminants of dimension d, $h(\mathcal{F}) \geq d$. For that, it is enough to produce d points $x_1, \ldots, x_d$, such that for any labeling $y_1, \ldots, y_d$ ($y_j \in \{-1, 1\}$), we are able to exhibit a function $f \in \mathcal{F}$ classifying the points in agreement with the labeling.

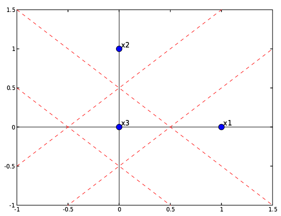
   Choosing,

$$\begin{gathered}
x_1 = (1, 0, 0, \ldots, 0) \\
x_2 = (0, 1, 0, \ldots, 0) \\
\ldots \\
x_d = (0, 0, 0, \ldots, 1),
\end{gathered}$$

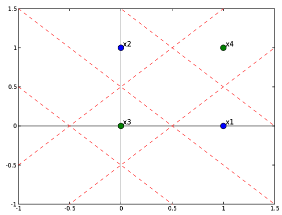
   $b = 0$ and ${}^t w = (y_1, \ldots, y_d)$ does the trick. Indeed $\forall k \in \{1, \ldots, d\}$,

$$\begin{aligned}
f(x_k) &= \text{sign}(w \cdot x_k + b) = \text{sign}\left(\sum_{j=1}^{d} w^j x_k^j\right) \\
&= \text{sign}(w^k) = y_k
\end{aligned} \tag{6}$$

To show that $h(\mathcal{F}) \geq d+1$, we just need to use the same $x_1, \cdots, x_d$ and w, define $x_{d+1} = (0, \cdots, 0)$ and set $b = \frac{y_{d+1}}{2}$.



(a) d=2, 3 points



(b) d=2, 4 points, cannot make the $\{x_1, x_2\}/\{x_3, x_4\}$ classification

2 Some implementations

1. **Getting familiarized with python**

   Download:

   - `train2d`, `train2d_target`, `valid2d` and `valid2d_target` some simple data,
   - `intro.py` a program with simple commands,
   - `bbox.py` some methods related to a black box learner,
   - `decision.py` a set of tools for plotting the decision function.

   Open them with a smart enough editor (*eg* `C:\Program␣Files\Notepad2\Notepad2.exe`), Explore them using `ipython`. Try for example:

   ```
   > cd toyour\download\path
   > run -i intro.py
   > ?bbox.bbox_capacity()
   ```

2. **Make a function which computes the classification error C_{err}, and plot the C_{err} vs the capacity of the black box learner, for the training set and the validation set.**

   Easy...

3. **Estimate the Bias and Variance of a regression function, using generate.py, and show what happens when the capacity of the learner increase.**

Let used as estimators of the bias and variance of a regression function:

$$bias^2(\hat{f}) = \frac{1}{|D_{test}|} \sum_{(x_i, y_i) \in D_{test}} bias^2(\hat{f}(x_i))$$

$$var(\hat{f}) = \frac{1}{|D_{test}|} \sum_{(x_i, y_i) \in D_{test}} var(\hat{f}(x_i))$$

where D_{test} is a test set and

$$bias^2(\hat{f}(x_i)) = \left[y_i - \frac{1}{100} \sum_{k=1}^{100} f_{D^k}(x_i) \right]^2,$$

$$var(\hat{f}(x_i)) = \frac{1}{100} \sum_{k=1}^{100} [f_{D^k}(x_i)]^2 - \left[\frac{1}{100} \sum_{k=1}^{100} f_{D^k}(x_i) \right]^2,$$

$D^k, \forall k \in \{1, \ldots, 100\}$ are training sets sampled from the same distribution as D_{test}.

For an implementation see the file `bias_var.py`.

4. **Implement the leave-one-out cross-validation strategy to estimate the expected risk of a given function which depends on some hyper-parameter.**

   Generate some data (train, valid and test set) with `generate.py` and see files `xv.py` and `xvtest.py`