Supervised learning

Hadrien Montanelli

Context

Setup Consider data $\mathcal{X} \subseteq \mathbb{R}^d$, labels $\mathcal{Y} \subseteq \mathbb{R}$ and a classifier $f : \mathcal{X} \mapsto \mathcal{Y}$. We approximate f by $\hat{f}$ using n labelled data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n) \in \mathcal{X}_0 \times \mathcal{Y}$, $\mathcal{X}_0 \subset \mathcal{X}$.

Bias-variance tradeoff For $y = f(\boldsymbol{x}) + \epsilon$, $\epsilon \sim \mathrm{N}(0, \sigma^2)$,

$$\begin{aligned}
&\mathrm{E}([y - \hat{f}(\boldsymbol{x})]^2) = [\mathrm{Bias}(\hat{f}(\boldsymbol{x}))]^2 + \mathrm{var}(\hat{f}(\boldsymbol{x})) + \sigma^2,\\
&\mathrm{Bias}(\hat{f}(\boldsymbol{x})) = \mathrm{E}(\hat{f}(\boldsymbol{x})) - f(\boldsymbol{x}),\\
&\mathrm{var}(\hat{f}(\boldsymbol{x})) = \mathrm{E}([\hat{f}(\boldsymbol{x}) - \mathrm{E}(\hat{f}(\boldsymbol{x}))]^2).
\end{aligned}$$

1 Naive Bayes $\mathcal{Y} = \{0, 1\}$

$$\hat{f}(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \left(\mathrm{P}(Y = y) \prod_{j=1}^{d} \mathrm{P}(X_j = x_j | Y = y) \right)$$

Training $\mathcal{O}(nd)$

- Compute the class prior $\mathrm{P}(Y = y)$ from $\mathcal{X}_0$.
- Choose a model for each $\mathrm{P}(X_j = x_j | Y = y)$.
- Use the closed-form for the MLE to find the parameters for each $\mathrm{P}(X_j = x_j | Y = y)$.

Classifying $\mathcal{O}(d)$ Evaluate $\hat{f}(\boldsymbol{x})$.

2 k-NNs $\mathcal{Y} = \{0, 1\}$

$$\hat{f}(\boldsymbol{x}) = \text{label} \left(\arg\min_{1 \le i \le n} d_{\boldsymbol{W}}(\boldsymbol{x}, \boldsymbol{x}_i) \right) \quad (k = 1)$$

Training $\mathcal{O}(nd^2 s + n \log(n) d)$

- Find distance $d_{\boldsymbol{W}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{W} (\boldsymbol{x}_i - \boldsymbol{x}_j)$, with s steps of BFGS, that best separates the data.
- Choose k. Find medians to produce k-d trees.

Classifying $\mathcal{O}(\log(n) d)$ Evaluate $\hat{f}(\boldsymbol{x})$.

3 Perceptron $\mathcal{Y} = \{-1, 1\}$

$$\hat{f}(\boldsymbol{x}) = \mathrm{sign}\left(\boldsymbol{w}^T \boldsymbol{x} + w_0 \right)$$

Training $\mathcal{O}(ndP)$, with $P = \max_{1 \le i \le n} \|\boldsymbol{x}_i\|^2 / \gamma^2$ (margin γ)

- Initialize $\boldsymbol{w} = \boldsymbol{0}$ and $w_0 = 0$.
- For $k = 1, 2, \ldots,$
  - if there is $\boldsymbol{x}_i \in \mathcal{X}_0$ such that
    $$\mathrm{sign}\left(\boldsymbol{w}^T \boldsymbol{x}_i + w_0 \right) \ne y_i,$$
    set $\boldsymbol{w} = \boldsymbol{w} + y_i \boldsymbol{x}_i$ and $w_0 = w_0 + y_i$.

Classifying $\mathcal{O}(d)$ Evaluate $\hat{f}(\boldsymbol{x})$.

4 Kernel perceptron $\mathcal{Y} = \{-1, 1\}$

$$\hat{f}(\boldsymbol{x}) = \mathrm{sign} \left(\sum_{i=1}^{n} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) \right)$$

Training $\mathcal{O}(ndP)$, with $P = \max_{1 \le i \le n} \|\boldsymbol{x}_i\|^2 / \gamma^2$ (margin γ)

- Choose K. (Perceptron is $K(\boldsymbol{x}, \boldsymbol{y}) = (1 + \boldsymbol{x}^T \boldsymbol{y})$.)
- Initialize $\boldsymbol{\alpha} = \boldsymbol{0}$.
- For $k = 1, 2, \ldots,$
  - set $\alpha_i = \alpha_i + 1$ if there is $\boldsymbol{x}_i \in \mathcal{X}_0$ such that
    $$\mathrm{sign} \left(\sum_{j=1}^{n} \alpha_j y_j K(\boldsymbol{x}_j, \boldsymbol{x}_i) \right) \ne y_i.$$

Classifying $\mathcal{O}(nd)$ Evaluate $\hat{f}(\boldsymbol{x})$.

5 SVMs $\mathcal{Y} = \{-1, 1\}$

$$\hat{f}(\boldsymbol{x}) = \mathrm{sign}\left(\boldsymbol{w}^T \boldsymbol{x} + w_0 \right)$$

Training Minimize, with s steps of PGD,

$$J(\boldsymbol{w}) = \frac{1}{2} \|\boldsymbol{w}\|^2,$$

such that $y_i(\boldsymbol{w}^T \boldsymbol{x}_i + w_0) \ge 1$, $1 \le i \le n$.

Classifying $\mathcal{O}(d)$ Evaluate $\hat{f}(\boldsymbol{x})$.

6 Kernel SVMs $\mathcal{Y} = \{-1, 1\}$

$$\hat{f}(\boldsymbol{x}) = \mathrm{sign} \left(\sum_{i=1}^{n} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + w_0 \right)$$

Training

- Choose K. (SVMs is $K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{y}$.)
- Maximize, with s steps of SQP,
  $$J(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_j, \boldsymbol{x}_i),$$
  $$\text{such that } \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and } \alpha_i \ge 0,\ 1 \le i \le n.$$
- For i with $\alpha_i \ne 0$, $w_0 = y_i - \sum_{j=1}^{n} \alpha_j y_j K(\boldsymbol{x}_j, \boldsymbol{x}_i)$.

Classifying $\mathcal{O}(nd)$ Evaluate $\hat{f}(\boldsymbol{x})$.

7 Logistic regression $\mathcal{Y} = [0,1]$

$$\hat{f}(\boldsymbol{x}) = \frac{1}{1+e^{-(\boldsymbol{w}^T\boldsymbol{x}+w_0)}}$$

Training $\mathcal{O}(nd^3s)$ MLE, Newton's method (s steps).

Classifying $\mathcal{O}(d)$ Evaluate $\hat{f}(\boldsymbol{x})$.

8 Shallow networks $\mathcal{Y} = [0,1]$

$$\hat{f}(\boldsymbol{x}) = \sigma^{(2)}\left(\boldsymbol{w}^{(2)T}\sigma^{(1)}\left(\boldsymbol{W}^{(1)T}\boldsymbol{x} + \boldsymbol{w}_0^{(1)}\right) + w_0^{(2)}\right)$$

Note that $\boldsymbol{w}_0^{(1)}$ is $N\times 1$, $\boldsymbol{W}^{(1)}$ is $d\times N$ and $\boldsymbol{w}^{(2)}$ is $N\times 1$.

Training $\mathcal{O}(ndNs)$

- Minimize, with s steps of GD,

$$J\left(\boldsymbol{w}_0^{(1)}, \boldsymbol{W}^{(1)}, w_0^{(2)}, \boldsymbol{w}^{(2)}\right) = \frac{1}{n}\sum_{i=1}^{n} L\left(y_i, \hat{f}(\boldsymbol{x}_i)\right).$$

- A popular choice for $L(y,\hat{y})$ is the cross entropy,

$$\begin{aligned} L(y,\hat{y}) &= -\left(y\log\hat{y} + (1-y)\log(1-\hat{y})\right), \\ dL(y,\hat{y}) &= \frac{\hat{y}-y}{\hat{y}(1-\hat{y})}. \end{aligned}$$

- Group the training data into a $n\times d$ matrix $\boldsymbol{X}$ and the labels into a $1\times n$ vector $\boldsymbol{y}$.
- Forward:

$$\begin{aligned} \underset{N\times n}{\boldsymbol{Z}^{(1)}} &= \underset{N\times d}{\boldsymbol{W}^{(1)T}}\underset{d\times n}{\boldsymbol{X}^T} + \text{repmat}\big(\underset{N\times 1}{\boldsymbol{w}_0^{(1)}}\big), \\ \underset{N\times n}{\boldsymbol{A}^{(1)}} &= \sigma^{(1)}\big(\underset{N\times n}{\boldsymbol{Z}^{(1)}}\big), \\ \underset{1\times n}{\boldsymbol{Z}^{(2)}} &= \underset{1\times N}{\boldsymbol{w}^{(2)T}}\underset{N\times n}{\boldsymbol{A}^{(1)}} + \text{repmat}\big(\underset{1\times 1}{w_0^{(2)}}\big), \\ \underset{1\times n}{\boldsymbol{A}^{(2)}} &= \hat{\boldsymbol{y}} = \sigma^{(2)}\big(\underset{1\times n}{\boldsymbol{Z}^{(2)}}\big). \end{aligned}$$

- Backward:

$$\begin{aligned} \underset{1\times n}{\boldsymbol{dA}^{(2)}} &= dL\big(\boldsymbol{y}, \boldsymbol{A}^{(2)}\big), \\ \underset{1\times n}{\boldsymbol{dZ}^{(2)}} &= \underset{1\times n}{\boldsymbol{dA}^{(2)}} * d\sigma^{(2)}\big(\underset{1\times n}{\boldsymbol{Z}^{(2)}}\big), \\ \underset{N\times 1}{\boldsymbol{dW}^{(2)}} &= \underset{N\times n}{\boldsymbol{A}^{(1)}}\underset{n\times 1}{\boldsymbol{dZ}^{(2)T}}, \quad \underset{1\times 1}{dw_0^{(2)}} = \text{sum}\big(\underset{1\times n}{\boldsymbol{dZ}^{(2)}}\big), \\ \underset{N\times n}{\boldsymbol{dA}^{(1)}} &= \underset{N\times 1}{\boldsymbol{W}^{(2)}}\underset{1\times n}{\boldsymbol{dZ}^{(2)}}, \\ \underset{N\times n}{\boldsymbol{dZ}^{(1)}} &= \underset{N\times n}{\boldsymbol{dA}^{(1)}} * d\sigma^{(1)}\big(\underset{N\times n}{\boldsymbol{Z}^{(1)}}\big), \\ \underset{d\times N}{\boldsymbol{dW}^{(1)}} &= \underset{d\times n}{\boldsymbol{X}^T}\underset{n\times N}{\boldsymbol{dZ}^{(1)T}}, \quad \underset{N\times 1}{\boldsymbol{dw}_0^{(1)}} = \text{sum}\big(\underset{N\times n}{\boldsymbol{dZ}^{(1)}}\big). \end{aligned}$$

Classifying $\mathcal{O}(dN)$ Evaluate $\hat{f}(\boldsymbol{x})$.

9 Deep networks $\mathcal{Y} = [0,1]$

$$\hat{f}(\boldsymbol{x}) = \sigma^{(L+1)}\left(\boldsymbol{W}^{(L+1)T}\sigma^{(L)}\Big(\dots\Big) + \boldsymbol{W}_0^{(L+1)}\right)$$

Note that $\boldsymbol{W}_0^{(\ell)}$ is $N^{(\ell)}\times 1$ and $\boldsymbol{W}^{(\ell)}$ is $N^{(\ell-1)}\times N^{(\ell)}$ with $N^{(0)} = d$ and $N^{(L+1)} = 1$.

Training $\mathcal{O}(nN^2Ls)$ Cost for $N^{(\ell)} = N \approx d$, $1\leq\ell\leq L$.

- Minimize, with s steps of GD,

$$J\left(\boldsymbol{W}_0^{(\ell)}, \boldsymbol{W}^{(\ell)}\right) = \frac{1}{n}\sum_{i=1}^{n} L\left(y_i, \hat{f}(\boldsymbol{x}_i)\right).$$

- Group the training data into a $n\times d$ matrix $\boldsymbol{X}$ and the labels into a $1\times n$ vector $\boldsymbol{y}$.
- Forward ($1\leq\ell\leq L+1$):

$$\begin{aligned} \underset{N^{(\ell)}\times n}{\boldsymbol{Z}^{(\ell)}} &= \underset{N^{(\ell)}\times N^{(\ell-1)}}{\boldsymbol{W}^{(\ell)T}}\underset{N^{(\ell-1)}\times n}{\boldsymbol{A}^{(\ell-1)}} + \text{repmat}\big(\underset{N^{(\ell)}\times 1}{\boldsymbol{W}_0^{(\ell)}}\big), \\ \underset{N^{(\ell)}\times n}{\boldsymbol{A}^{(\ell)}} &= \sigma^{(\ell)}\big(\underset{N^{(\ell)}\times n}{\boldsymbol{Z}^{(\ell)}}\big). \end{aligned}$$

  Note that $\boldsymbol{A}^{(0)} = \boldsymbol{X}^T$ and $\boldsymbol{A}^{(L+1)} = \hat{\boldsymbol{y}}$.
- Backward ($1\leq\ell\leq L+1$):

$$\begin{aligned} \underset{N^{(\ell)}\times n}{\boldsymbol{dA}^{(\ell)}} &= \underset{N^{(\ell)}\times N^{(\ell+1)}}{\boldsymbol{W}^{(\ell+1)}}\underset{N^{(\ell+1)}\times n}{\boldsymbol{dZ}^{(\ell+1)}}, \\ \underset{N^{(\ell)}\times n}{\boldsymbol{dZ}^{(\ell)}} &= \underset{N^{(\ell)}\times n}{\boldsymbol{dA}^{(\ell)}} * d\sigma^{(\ell)}\big(\underset{N^{(\ell)}\times n}{\boldsymbol{Z}^{(\ell)}}\big), \\ \underset{N^{(\ell-1)}\times N^{(\ell)}}{\boldsymbol{dW}^{(\ell)}} &= \underset{N^{(\ell-1)}\times n}{\boldsymbol{A}^{(\ell-1)}}\underset{n\times N^{(\ell)}}{\boldsymbol{dZ}^{(\ell)T}}, \quad \underset{N^{(\ell)}\times 1}{\boldsymbol{dW}_0^{(\ell)}} = \text{sum}\big(\underset{N^{(\ell)}\times n}{\boldsymbol{dZ}^{(\ell)}}\big), \end{aligned}$$

except for $\ell = L+1$ where $\boldsymbol{dA}^{(L+1)} = dL(\boldsymbol{y}, \boldsymbol{A}^{L+1})$.

Classifying $\mathcal{O}(N^2L)$ Evaluate $\hat{f}(\boldsymbol{x})$.

10 Decision trees

$$\hat{f}(\boldsymbol{x}) = \text{label}\left(C(\boldsymbol{x})\right)$$

Training Find feature and threshold that maximize

$$u(C) - (p_L u(C_L) + p_R u(C_R)), \; u(C) = -\sum_{y\in\mathcal{Y}} p_y\log p_y.$$

Classifying $\mathcal{O}(\log(n)d)$ Evaluate $\hat{f}(\boldsymbol{x})$.

11 Random forests

$$\hat{f}(\boldsymbol{x}) = \frac{1}{T}\sum_{t=1}^{T}\text{label}\left(C_t(\boldsymbol{x})\right)$$

Training Sample T times, with replacement, $m < n$ training examples from $\mathcal{X}_0$ to construct T decision trees.

Classifying $\mathcal{O}(\log(n)dT)$ Evaluate $\hat{f}(\boldsymbol{x})$.