Alin Tomescu

Lecture 10

Last time we talked about ensembles.

Ensembles were defined as:

$$h_m(x) = \alpha_1 h(x;\theta_1) + \cdots \alpha_m h(x;\theta_m)$$

Every classifier $\alpha_i h(x;\theta_i)$ is applied for every example. So they will all contribute to the classification of the example.

The $h(x;\theta_i)$ is a **weak learner**, easy to estimate individually, and adding them together creates a **much stronger classifier**, which will over-fit. However, *poor optimization will lead to good results*.

See Figure 1:

$$h(\vec{x};\theta) = \text{sign}\left(s\left(x_j - t\right)\right), \theta = \{j, s, t\}$$

How to train

We want to find an ensemble $h_m(x)$ that minimizes the training error:

$$\sum_{i=1}^{n} Loss\left(y^{(i)} h_m\left(x^{(i)}\right)\right) = \sum_{i=1}^{n} e^{-\left(y^{(i)} h_m\left(x^{(i)}\right)\right)}$$

$y^{(i)} h_m\left(x^{(i)}\right)$ is positive when you agree and predict correctly, thus the loss $e^{-y^{(i)} h_m\left(x^{(i)}\right)}$ will be small.

$$\sum_{i=1}^{n} [[y^{(i)} \neq h_m\left(x^{(i)}\right)]] \leq \sum_{i=1}^{n} Loss\left(y^{(i)} h_m\left(x^{(i)}\right)\right)$$

See Figure 3 for graph

Simple way of training (forward fitting)

This corresponds to adding one term at a time.

(0) $h_0(x) = 0$

(1) Fix $\hat{h}_{m-1}(x)$, find $\hat{\alpha}_m, \hat{\theta}_m$ that minimizes $J(\alpha_m, \theta_m) = \sum_{i=1}^{n} Loss\left(y^{(i)} \hat{h}_{m-1}\left(x^{(i)}\right) + y^{(i)} \alpha_m h\left(x^{(i)}; \theta_m\right)\right)$

a. $y^{(i)} \hat{h}_m\left(x^{(i)}\right) = y^{(i)} \hat{h}_{m-1}\left(x^{(i)}\right) + y^{(i)} \alpha_m h\left(x^{(i)}; \theta_m\right)$

Unfortunately this is too hard of a problem.

$$\hat{h}_{m-1} = \begin{bmatrix} \hat{h}_{m-1}\left(x^{(1)}\right) \\ \vdots \\ \hat{h}_{m-1}\left(x^{(n)}\right) \end{bmatrix}, \text{where}$$

$$\hat{h}_{m-1}(\vec{x}) = \sum_{i=1}^{m-1} \alpha_i h(x;\theta_i)$$

Alin Tomescu

$$h_\theta = \begin{bmatrix} h(x^{(1)}, \theta) \\ \vdots \\ h(x^{(n)}, \theta) \end{bmatrix}, \text{where}$$

$$h(\vec{x}; \theta) = \text{sign}\left(s(x_j - t)\right), \theta = \{j, s, t\}$$

$$\|h_\theta\|^2 = n, \text{since the values are} \pm 1$$

How can we choose h_{θ_m} at step m? We want to find one that minimizes:

$$J(\alpha_m, \theta_m) = \sum_{i=1}^{n} y^{(i)} h_{m-1}(x^{(i)}) + y^{(i)} \alpha_m h(x^{(i)}; \theta_m)$$

$$\frac{\partial}{\partial \alpha_m} J(\alpha_m, \theta_m)_{\mid \alpha_m} = 0$$

$$= \sum_{i=1}^{n} \left[\frac{\partial}{\partial z} Loss(z)_{\mid z = y^{(i)} \hat{h}_{m-1}(x^{(i)})}\right] y^{(i)} h(x^{(i)}; \hat{\theta}_m)$$

$$W_{m-1}(i) = e^{-y^{(i)} \hat{h}_{m-1}(x^{(i)})}$$

$$\frac{\partial}{\partial \alpha_m} J(\alpha_m, \theta_m)_{\mid \alpha_m = 0} = \sum_{i=1}^{n} W_{m-1}(i) \left(-y^{(i)} h(x^{(i)}; \hat{\theta}_m)\right) ?? = ?? \sum_{i=1}^{n} W_{m-1}(i)\, z[[y^{(i)} \neq h(x^{(i)}; \theta_m)]] - 1$$

Boosting algorithm

(0) $h_0(x) = 0, w_i = \frac{1}{n}$
(1) Fix $\hat{h}_{m-1}(x)$, find a stump $\hat{\theta}_m$ that minimizes the weighted error: $\sum_{i=1}^{n} w_i \left(-y^{(i)} h(x^{(i)}; \hat{\theta}_m)\right)$
(2) Find how much to rely on that stump $\hat{\alpha}_m$ that minimizes $J(\alpha_m, \hat{\theta}_m)$
(3) Update $w_i = \left[-\frac{\partial}{\partial z} Loss(z) \middle| z = y^{(i)} \hat{h}_{m-1}(x^{(i)}) + y^{(i)} \hat{\alpha}_m h(x^{(i)}; \hat{\theta}_m)\right]$

When using the exponential loss, this is called **AdaBoosting**

See Figure 4

If we select the same stump after step m, it does not add any value, because we've already optimized as much as we can in that direction.

How well can we generalize?

Assume training examples $(x^{(i)}, y^{((i))}) \sim p^*, \mathit{fixed\ unknown\ joint\ distribution}$

The test examples are also drawn at random from p^*

Training error (empirical risk): $R_n(h) = \sum_{i=1}^{n} Loss_{0,1}\left(y^{(i)} h(x^{(i)})\right)$

Test error/risk: $E_{(x,y) \sim p^*}\{Loss_{0,1}(yh(x))\}$

Alin Tomescu

How are these two types of error related?

Hypothesis class (set of classifiers) $\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \cdots \mathcal{H}_k$

We select $\mathcal{H}_k$, then we find $\hat{h}_k = argmin_{h \in \mathcal{H}_k} \{R_n(h)\}$. How well would this generalize?

$$R_n(\hat{h}_k) = \text{random variable}$$

$$\hat{h}_k = \text{random variable}$$

$$R(\hat{h}_k) = \text{gen. error} = \text{random variable}$$

$$R(\hat{h}_k) - R_n(\hat{h}_k) = \varepsilon$$

$$\varepsilon = \varepsilon(n, \mathcal{H}_k, \delta), \delta = \text{confidence ??}$$

We are looking for results with probability at least $1 - \delta$, the generalization error is bounded by the training error plus ε:

$$R(\hat{h}_k) \leq R_n(\hat{h}_k) + \varepsilon$$