Maschinelles Lernen 1 - Assignment 3

Technische Universität Berlin

Christoph Conrads (315565) Antje Relitz (327289) Benjamin Pietrowicz (332542)

Mitja Richter (324680)

WS 2013/2014

1 Flipping the Coins

a)

$$Pr(D|p) = \prod_{i=1}^{7} Pr(x_i|p) = p^{\#head} * (1-p)^{\#tail} = p^5 * (1-p)^2 = p^5 - p^7$$

b)

The maximum likelihood yields:

$$\nabla Pr(D|p) \stackrel{!}{=} 0 \Leftrightarrow 5p^4 - 7p^6 = 0 \Leftrightarrow p^4(5-7p^2) = 0 \Leftrightarrow p^4 = 0 \vee 5 - 7p^2 = 0$$

From our given sequence of events we know that $p \in]0,1[$

$$\Rightarrow p^2 = \frac{5}{7}$$

Again we know that $p > 0$

$$\Rightarrow p = +\sqrt{\frac{5}{7}}$$

We are now looking for the probability of the given sequence $D_1 = (x_8, x_9) = (head, head)$:

$$Pr(D_1|\sqrt{\frac{5}{7}}) = \prod_{i=8}^{9} Pr(x_i|\sqrt{\frac{5}{7}}) = Pr(\{x_8 = head\}) * Pr(\{x_9 = head\}) = \sqrt{\frac{5}{7}} * \sqrt{\frac{5}{7}} = \frac{5}{7}$$

The probability that the next two tosses are "head" with the given unfair coin is $p = \frac{5}{7}$.

2 Biased Boundaries

b)

Our Maximum Likelihood function for $P(D_1|\mu_1)$:

$$l(\mu_1) = \ln P(D_1|\mu_1) = \sum_{i=1}^{n} \ln P(x_i|\mu_1)$$

Under the Gaussian generative assumtpion we get:

$$P(x_i|\mu_1) = \frac{1}{\sqrt{2\pi\sigma}} \exp^{-\frac{1}{2\sigma^2}(x_i-\mu_1)^2}$$

Applying the logarhitmus for convenience:

$$\ln P(x_i|\mu_1) = -\frac{1}{2}\ln 2\pi\sigma - \frac{1}{2\sigma^2}(x_i - \mu_1)^2$$

Computing the derivate:

$$\frac{d \ln P(x_i|\mu_1)}{d\mu_1} = \frac{1}{\sigma}(x_i - \mu_1)$$

For the dataset D we get:

$$\sum_{i=1}^{n} \frac{1}{\sigma}(x_i - \hat{\mu}_1) \stackrel{!}{=} 0 \rightarrow \hat{\mu}_1 = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \square$$

3 Feature Expansion

a)

There are several general problems with using non-linear feature mapping into higher dimensions in this situation:

- The first problem is the large number of unknown parameters required to learn the gaussian ML-model. With increasing dimension d of the input vector, the estimate of the mean grows linear to a d-dimension vector, but the covariance matrix Σ $(d \times d)$ is growing quadratically.
- The next problem would be the additional computational load of computing $\phi()$, which is especially grave with a large dataset D.
- Furthermore there is the danger of overfitting, since the linear (or quadratic) discriminant is segmenting the training data non-linearly in lower input space. With a noisy dataset this could lead to false classifications.

On the other side this methods produces a linear (or quadratic) discriminant, which makes classification really fast.

b)

With a small dataset, the bias of ML can potentially distort results, since ML is only asymptotically unbiased.