Machine Learning
Assignment 2.2

Nikolaj Dybdahl Rathcke (rfq695)

December 14, 2015

1 Principal Component Analysis

1.1 Summarization by the mean

We want to find the b that minimizes the entire sum. This is done by taking the derivative with respect to b, so we want to solve the following:

$$\nabla_b \left(\frac{1}{N} \sum_{i=1}^{N} \|x_i - b\|^2 \right) = 0$$

We can calculate the gradient on the left side after rewriting $\|x_i - b\|^2$ to $(x_i - b)^2$, to get:

$$\begin{aligned}
\nabla_b \left(\frac{1}{N} \sum_{i=1}^{N} (x_i - b)^2 \right) &= \frac{1}{N} \sum_{i=1}^{N} 2(b - x_i) \\
&= \frac{2}{N} \sum_{i=1}^{N} (b - x_i) \\
&= \frac{2}{N} \sum_{i=1}^{N} b - \frac{2}{N} \sum_{i=1}^{N} x_i \\
&= \frac{2Nb}{N} - \frac{2}{N} \sum_{i=1}^{N} x_i \\
&= 2b - \frac{2}{N} \sum_{i=1}^{N} x_i
\end{aligned}$$

We can now solve it for zero and move the sum (and the fraction) to the other side:

$$2b - \frac{2}{N} \sum_{i=1}^{N} x_i = 0 \;\Leftrightarrow\; 2b = \frac{2}{N} \sum_{i=1}^{N} x_i \;\Leftrightarrow\; b = \frac{1}{N} \sum_{i=1}^{N} x_i$$

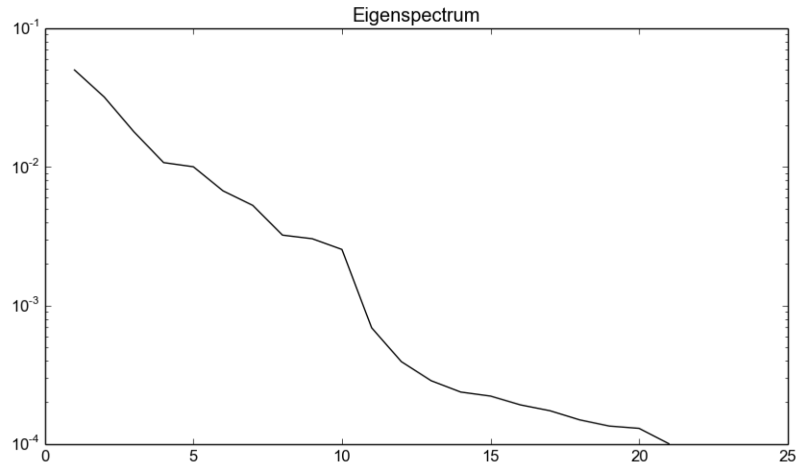
Which is wanted to show. We know it is a minimum as it is a convex second degree polynomial, so it only has one extremum which is of minimum value.

1.2 PCA for high dimensional data and small samples

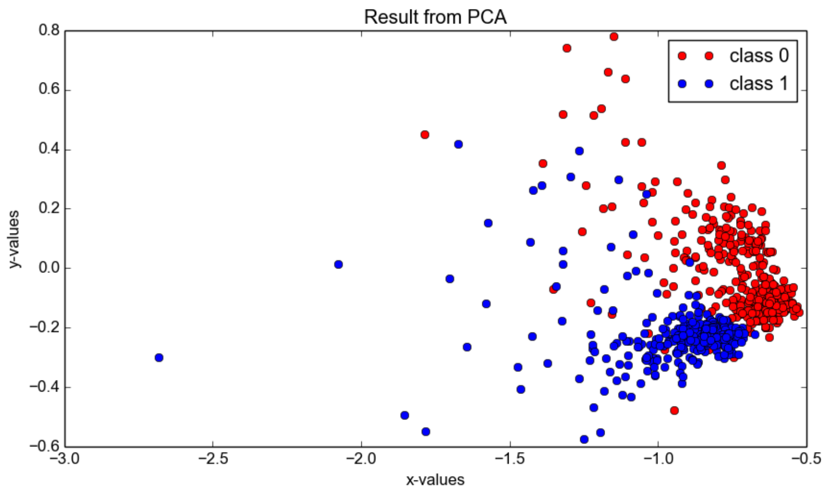
N/A

1.3 Cybercrime Detection

The code that performs the PCA is implemented in `src_pca.py`. Running that file will produce the eigenspectrum and the scatterplot. The eigenspectrum when plotted with a logarithmic y-scale will look like this:



We can see that using more than around 10 or 11 principal components is a bit of a waste as there is a significant drop.
The scatterplot we get will look like this:



where the red dots is the ones with class 0 and the blue dots are those with class 1. As we can see, when the data is projected on the first two principal components they are already quite nicely divided.

2 Occam's Razor

2.1

We use corollary 2.4 from the lecture notes on Occam's Razor bound, with $M = 2^{27^d}$, to bound $L(h) - \hat{L}(h, S)$:

$$\mathbb{P}\left\{\exists h \in \mathcal{H} : L(h) - \hat{L}(h,S) \geq \sqrt{\frac{\ln(2^{27^d}/\delta)}{2n}}\right\} \leq \delta$$

Where we have moved $\hat{L}(h, S)$ to the left side of inequality and $M = 2^{27^d}$ because we have 27^d words of length d, so there must be 2^{27^d} subsets of this, which is the cardinality of the hypotheses space.

2.2

We use theorem 2.5 from the lecture notes, where we do not use it for binary decision trees, but for trees that branch in 27, so

$$\mathbb{P}\left\{\exists h \in \mathcal{H} : L(h) - \hat{L}(h, S) \geq \sqrt{\frac{\ln(2^{27^{d(h)}} 27^{d(h)}/\delta)}{2n}}\right\} \leq \delta$$

where we again moved $\hat{L}(h, S)$ to the left side of the equation and $d(h)$ is the depth function, i.e. just d.