ECE544NA: PCA, python + numpy + TensorFlow tutorial

Raymond Yeh

University of Illinois at Urbana Champaign

yeh17@illinois.edu

September 13, 2016
PCA: Least Squares Formulation

1. Given a dataset of \( \{ \vec{x}_1, \ldots, \vec{x}_N \} \), where \( \vec{x}_i \in \mathbb{R}^d \). We hope to reduce the dimension of the data to \( m < d \), using linear transformations.

2. Let \( \{ \vec{u}_1, \vec{u}_2, \ldots, \vec{u}_d \} \) be a set of orthonormal vectors, then without loss of generality, we can write, each \( \vec{x}_i \) as

\[
\vec{x}_i = \sum_{j=1}^{d} z_{ji} \vec{u}_j
\]  

(1)

3. Recall that the definition of orthonormal vectors,

\[
\vec{u}_j^\top \vec{u}_l = \delta_{jl}
\]  

(2)

where

\[
\delta_{jl} = \begin{cases} 
0, & j \neq l \\
1, & j = l
\end{cases}
\]  

(3)
1. Given a $\vec{x}_i$, we can solve for $z_{ji} = \vec{u}_j^T \vec{x}_i$.

2. We wish to reduce the dimension of the data to $m < d$, meaning we only use $m$ coefficients $z_j$, and the remaining $d - m$ coefficients will be replaced with constants $b_j$, then the reduced dimension vector $\hat{\vec{x}}_i$ can be written as:

$$\hat{\vec{x}}_i = \sum_{j=1}^{M} z_{ji} \vec{u}_j + \sum_{j=m+1}^{d} b_j \vec{u}_j \quad (4)$$

3. Next, we will find the best approximation in the least squares sense. Meaning we minimize the following,

$$L_m = \frac{1}{2} \sum_{i=1}^{N} ||\vec{x}_i - \hat{\vec{x}}_i||^2 \quad (5)$$
PCA: Least Squares Formulation

1. Expand using Equation (4) and orthonormality Equation (2)

\[
L_m = \frac{1}{2} \sum_{i=1}^{N} ||\tilde{x}_i - \hat{x}_i||^2 = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=m+1}^{d} (z_{ji} - b_j)^2
\]  

(6)

2. Set the derivative of \( L_m \) with respect to \( b_i \) to zero, then

\[
b_j = \frac{1}{N} \sum_{i=1}^{N} z_{ji} = \bar{u}_j^\top \left( \frac{1}{N} \sum_{i=1}^{N} \tilde{x}_i \right) = \bar{u}_j^\top \bar{x}
\]  

(7)

3. Substitute \( b_j \) back in to Equation (5)

\[
L_m = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=m+1}^{d} (\bar{u}_j^\top (\tilde{x}_i - \bar{x}))^2
\]  

(8)
Some rewriting:

\[
L_m = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=m+1}^{d} (\vec{u}_j^\top (\vec{x}_i - \bar{\vec{x}}))^2 
\]

(9)

\[
= \frac{1}{2} \sum_{j=m+1}^{d} \sum_{i=1}^{N} (\vec{u}_j^\top (\vec{x}_i - \bar{\vec{x}}))(\vec{u}_j^\top (\vec{x}_i - \bar{\vec{x}}))^\top 
\]

(10)

\[
= \frac{1}{2} \sum_{j=m+1}^{d} \vec{u}_j^\top \left( \sum_{i=1}^{N} ((\vec{x}_i - \bar{\vec{x}}))(\vec{x}_i - \bar{\vec{x}}))^\top \right) \vec{u}_j 
\]

(11)

\[
= \frac{1}{2} \sum_{j=m+1}^{d} \vec{u}_j^\top \Sigma \vec{u}_j 
\]

(12)
Lastly, we will need to find the set of $\tilde{u}_j$ that minimize $L_m$. Minimum occurs when the basis satisfy,

$$\Sigma \tilde{u}_j = \lambda_j \tilde{u}_j$$  \hspace{1cm} (13)

where $\Sigma = \sum_{i=1}^{N}((\tilde{x}_i - \tilde{x}))(\tilde{x}_i - \tilde{x})^\top$ is the covariance matrix of the data set $\{\tilde{x}_i\}$.

Using the Equation (13) and $L_m$ in Equation (12). The minimum is at

$$L_m = \sum_{j=m+1}^{d} \tilde{u}_j^\top \Sigma \tilde{u}_j = \sum_{j=m+1}^{d} \tilde{u}_j^\top \lambda_j \tilde{u}_j = \sum_{j=m+1}^{d} \lambda_j$$  \hspace{1cm} (14)

Therefore, to minimize the $L_m$, we should remove dimension where $\tilde{u}_j$ has the smallest $\lambda_j$. 
PCA: Step by Step

1. Let $X = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_N]$
2. Compute $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} \vec{x}_i$
3. Compute $\Sigma = (X - \bar{x})(X - \bar{x})^\top$
4. Compute Eigenvectors and Eigenvalues of $\Sigma$, keep the $m$ dimensions with the largest Eigenvalues.
5. Lastly, to project examples on to the lower dimensional space, $\vec{z} = U^\top X$, where $U = [\vec{u}_1, \vec{u}_2, ..., \vec{u}_m]$, and $\vec{u}_j$ are the Eigenvectors.
PCA involves computing $XX^T$, $X$ has the dimension of $d \times N$. When $d \gg N$, then compute Eigenvalues and Eigenvectors on $X^TX$ instead to save computation.

Claim: If $\vec{v}$ is Eigenvector of $X^TX$, then $\vec{u} = X\vec{v}$ is the Eigenvector of $XX^T$.

Proof:

\[
X^TX\vec{v} = \lambda\vec{v}
\]  \hspace{1cm} (15)

\[
X(X^TX\vec{v}) = X(\lambda\vec{v})
\]  \hspace{1cm} (16)

\[
(XX^T)(X\vec{v})) = \lambda(X\vec{v})
\]  \hspace{1cm} (17)

\[
(XX^T)(\vec{u})) = \lambda(\vec{u})
\]  \hspace{1cm} (18)
Ipython Notebook will be posted on the website.