Matrix factorizations and low rank approximation

Friday, March 11, 2016:

1. Eigenfaces

In this section, we learn intuition and math behind Eigenfaces, a technique to apply matrix SVD factorization to identify people’s faces from the database.

1.1. Image manipulation. Let’s assume we have \( n \) images of people’s faces that are:

- Grey scale
- Same size \((m_1 \times m_2)\)
- Same position/orientation of the face

For the method to work, let’s reshape the matrix into one column vector that contains all of the image’s information:

\[
\text{Face Image} \rightarrow \mathbf{t}_1
\]

\[
\begin{align*}
\text{m}_1 \times \text{m}_2 & \quad \text{m}_1 \times \text{m}_2 \times 1
\end{align*}
\]

After transforming each image into a vector, we can align them together to get the matrix:

\[
\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_n].
\]

Usually, \( m \gg n \).

1.2. EVD - eigenvalue decomposition. In this method, we seek to compress \( \mathbf{T} \) such that:

\[
\mathbf{S} = \mathbf{T} \mathbf{T}^* = \mathbf{U} \mathbf{D} \mathbf{U}^*
\]

Then, let’s pick a tolerance measure to pick \( k \) eigenvalues that reconstruct matrix \( \mathbf{S} \) sufficiently accurately:

\[
\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_n} < 1 - \text{tolerance}
\]

Therefore, we have a matrix:

\[
\mathbf{S} \approx \mathbf{U} \mathbf{D}_{k \times k} \mathbf{U}^*.
\]

Where, vectors \( u_i \) for \( i \in [1 \ldots k] \), are the “eigenfaces”. They form an approximate basis for the columns of \( \mathbf{T} \). Therefore, our reconstruction of original matrix \( \mathbf{T} \) is:

\[
\mathbf{T} \approx \mathbf{U} \mathbf{U}^* \mathbf{T} = \mathbf{U} \mathbf{T}
\]

where,

\[
\mathbf{T} = \mathbf{U}^* \mathbf{T}
\]
Useful applications.

1. **Storage efficiency**: store only matrices $U$ and $\hat{T}$ instead of $T$.

2. **Face recognition**: given a new face image encoded in a vector $S$, we can attempt to find an image $t_j$ in our database that’s closest to $S$.

   Our job is to find $i = \arg\min_{p \leq i \leq n} ||t_p - S||$

   Let:

   $$\hat{S} = U^*S$$

   Check that $||S - UU^*S|| = ||S - U\hat{S}||$ is small. If it’s not, then the given image doesn’t have a match in the database, so we can add it by updating $U$ and $\hat{T}$.

   **Caveat**: $L_2$ distance is not a good measure of closeness between images. A lot of that difference could just be noise, difference in light, shades etc.

1.3. **Problem with Eigenfaces.** $S = TT^*$ is very large.

   Typically $n << m$.

   **Solution 1**: For $S = T^*T$, let’s computer it’s EVD.

   1. Suppose $T^*Tv = \lambda v \rightarrow TT^*Tv = \lambda Tv$

   2. If we set $u = Tv$, we have a familiar system $Su = \lambda u$

   3. Let $v_j$ for $j \in [1 \cdots n]$ be eigenvectors of $T^*t$, normalized so that $||v_j|| = 1$.

   4. Set $u_j = Tv_j$, then $u_i \cdot u_j = u_i^*u_j = \lambda_j^*v_i^*v_j = \begin{cases} \lambda_j, & i = j \\ 0, & i \neq j \end{cases}$

   **Solution 2**: Compute SVD of $T$

   1. Suppose rank of $T$ is $k$. We know that $k \leq \min(m, n)$.

   2. $T = U\sum V^*$

          $TT^* = U\sum^2 U^*$

          $T^*T = V\sum^2 V^*$

   3. The left singular vectors of $T$ are the eigenvectors.

   Observe that $u_j = Tv_j = (\sum_{i=1}^k \sigma_i u_i v_i^*)v_j = \sigma_j u_j$

   4. Rescale to get the left svecs of $T$. In practice, we don’t explicitly form $T^*T$. Instead, $T^*T = \sum_{i=1}^n \sigma_i^2 u_i v_i^*$