Spectral Methods for Machine Learning

Daniel Khashabi

Summer 2015
Last Update: November 28, 2015

1 Introduction

Although spectral techniques have long been existed in linear algebra and graph theory, these techniques have seen two relatively recent surge of attention from the machine learning community. The first was mostly with regards to clustering techniques (i.e. spectral clustering). The more recent one extends previous findings to general framework of latent variable models.

This document summarizes basic examples from the existing literature.

2 PCA: Spectral Learning on Covariance Matrix

The goal in PCA is finding a subspace (lower dimensional basis) P which best expresses a set of points $x_i \in \mathbb{R}^d$. One can model PCA with quadratic objective function:

$$J = \min_P \sum_i \|x_i - Px_i\|^2 \tag{1}$$

This can be thought of minimizing the sum of distances between points and their projections on P. The project matrix $P \in \mathbb{R}^{d\times d}$ has rank k, which is less than d (or can approximated with such rank). One rank-k realization of this matrix is $P = U_k U_k^\top$, where $U_k \in \mathbb{R}^{d\times k}$. The question is what U_k realizes the objective function of 1. Let's simplify the objective function:

$$\begin{aligned}
\|x_i - Px_i\|^2 &= (x_i - Px_i)^\top (x_i - Px_i) \\
&= \left(x_i^\top - x_i^\top P^\top\right)(x_i - Px_i) \\
&= x_i^\top x_i - x_i^\top \left(P + P^\top\right) x_i + x_i^\top P^\top P x_i
\end{aligned}$$

Now with the low-rank assumption $P = U_k U_k^\top$, and assuming that $U_k^\top U_k = 1$ (so as to normalize the norm of the basis functions), simplify the objective:

$$\|x_i - Px_i\|^2 = x_i^\top x_i - x_i^\top \left(P + P^\top\right) x_i + x_i^\top P^\top P x_i = x_i^\top x_i - x_i^\top U_k U_k^\top x_i$$

Therefore the objective simplifies to:

$$J = \min_{\substack{U_k \in \mathbb{R}^{d\times k} \\ U_k^\top U_k = I_k}} \sum_i x_i^\top x_i - x_i^\top U_k U_k^\top x_i = \max_{\substack{U_k \in \mathbb{R}^{d\times k} \\ U_k^\top U_k = I_k}} \sum_i x_i^\top U_k U_k^\top x_i$$

Writing in matrix form:

$$J = \max_{\substack{U_k \in \mathbb{R}^{d\times k} \\ U_k^\top U_k = I_k}} \operatorname{tr}\left(X^\top U_k U_k^\top X\right) = \max_{\substack{U_k \in \mathbb{R}^{d\times k} \\ U_k^\top U_k = I_k}} \operatorname{tr}\left(U_k^\top X X^\top U_k\right)$$

where $X = [x_1, \ldots, x_n] \in \mathbb{R}^{d\times n}$. For simplicity of future notation let's define $S = XX^\top$ (in the special case when X has zero mean, S is the covariance matrix).

Now let's deviate a little to a useful tool from linear algebra.

Rayleigh quotient: For a matrix $M \in \mathbb{R}^{d\times d}$ and vector $x \in \mathbb{R}^d$, the Rayleigh quotient is defined as $R(M, x) = \frac{x^\top M x}{x^\top x}$. One important property about this function is that, its maximum is $\lambda_{\max}$, the biggest eigenvalue of M and it is reached at $x = v_{\max}$, the eigenvector corresponding to the biggest eigenvalue. Similar the lower bound on Rayleigh quotient is $\lambda_{\min}$ and is achieved at $x = v_{\min}$.

Suppose we have an objective functions

$$\max_{\substack{u \in \mathbb{R}^d \\ \|u\|=1}} u^\top S u.$$

The optimal u is the (normalized) eigenvector corresponding to the biggest eigenvalue of S. Instead suppose we have:

$$\max_{\substack{\{u_i\}_{i=1}^k \in \mathbb{R}^d \\ \|u_i\|=1}} \sum_{i=1}^k u_i^\top S u_i,$$

when each of the u_i vectors are linearly independent of each other. The answer can be found by choosing k eigenvectors corresponding to the top k eigenvalues. The maximum value achieved by this objective function is $\lambda_1 + \ldots + \lambda_k$. To keep things more general we can rewrite the above objective function in the following form:

$$\max_{\substack{U_k \in \mathbb{R}^{d\times k} \\ U_k^\top U_k = I_k}} \operatorname{tr}\left(U_k^\top S U_k\right),$$

which is the same as the PCA objective function, which can be simplified as the following simple steps:

- Calculate $S = XX^\top$.
- Find the top k eigenvalues and corresponding eigenvectors, as principal components.

You can easily try these steps in MATLAB:

```
% consider an artificial data set of 50 variables (e.g., genes) and 20 samples
X = rand(50,20);

% make sure all datapoints have mean zero
X = X - repmat(mean(X,2),1,size(X,2));

% calculate eigenvectors (loadings) V, and eigenvalues of the covariance matrix
[V, lambdaMatrix] = eig(cov(data'));
lambdas = diag(lambdaMatrix);

% order by largest eigenvalue
lambdas = lambdas(end:-1:1);
U = V(:,end:-1:1);

% generate PCA component space (PCA scores)
pc = U' * X;

% plot PCA space of the first two PCs: PC1 and PC2
plot(pc(1,:),pc(2,:),'.')
```