Bocconi

Mathematical Modelling in Machine Learning - 30554

Luca Raffo
Bocconi University

Contents

1 Overview on Machine Learning **3**

2 KL divergence **4**

3 Linear Algebra intermezzo **8**

4 PCA **9**

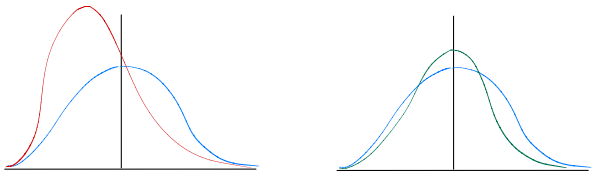
4.1 Maximum Variance Perspective 12

5 **15**

1 Overview on Machine Learning

2 KL divergence

The Kullback-Leibler divergence, also known as relative entropy, is a measure of difference between two probability distributions.



Intuitively, in the previous images, we would say that the green and blue distributions are "closer" than the red and blue distributions.
The KL divergence helps us to formalize this idea.
We could also interpret it as a non-symmetric measure that quantifies the amount of information lost when approximating one distribution with another.

Definition 1.1 *KL divergence.* The mathematical definition of the *KL divergence* between two probability measures p and q is given by:

$$D_{KL}(p||q) = \sum_{x \in \mathscr{X}} p(x) \ln \frac{p(x)}{q(x)}$$

where $\mathscr{X}$ is the sample space and $p(x)$, $q(x)$ are the probability density functions for distributions p and q, respectively.

Properties of KL divergence:

- Non-negativity: KL divergence is always non-negative, meaning that $D_{KL}(p||q) \geq 0$ for any distributions p and q.
- Asymmetry: in general $D_{KL}(p||q) \neq D_{KL}(q||p)$.
- Equality: $D_{KL}(p||q) = 0$ if and only if $p = q$.

Remark: the KL divergence is NOT a distance in the formal mathematical meaning (it doesn't satisfy symmetry), it is rather a measure of "difference" between two probability distributions.

A lower KL divergence between p and q indicates that the two distributions are more similar.

In summary, KL divergence is a useful tool for comparing two probability distributions and provide a measure of the difference between them.
These properties make it a useful tool for evaluating machine learning algorithms that involve approximating one distribution with another.

With the probabilistic formalism:

$$D_{KL}(p||q) = \mathbb{E}_p \left[\ln \frac{p(X)}{q(X)} \right] = \sum_{x \in \mathscr{X}} p(x) \ln \frac{p(x)}{q(x)}$$

or

$$D_{KL}(p||q) = \mathbb{E}_p \left[\ln \frac{p(X)}{q(X)} \right] = \int_{x \in \mathscr{X}} p(x) \ln \frac{p(x)}{q(x)}$$

where we use the following three conventions:

- $0 \ln \frac{0}{0} = 0$
- $0 \ln \frac{0}{q} = 0$
- $p \ln \frac{p}{0} = \infty$

Intuitively, it is often useful to think of p as the first or original form of the data, and q as something that we want to measure the quality of with reference to p.

Now, it is pedagogically appropriate to show the proof of the non-negativity property of the KL divergence, in order to make the reader gain confidence with the topic.

Theorem 1.2 *Non-negativity of the KL divergence.* $D_{KL}(p||q) \geq 0$ holds always for any pair of distributions p and q, and holds with equality if and only if $p = q$.

Proof. We assume that p is discrete, if it was continuous just change sum with integrals. We use Jensen's inequality (which states that if X is a random variable and f is a convex function, then $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$). Let $\mathscr{X}$ be the support of the random variable:

$$-D_{KL}(p||q) = -\sum_{x \in \mathscr{X}} p(x) \ln \frac{p(x)}{q(x)} = \sum_{x \in \mathscr{X}} p(x) \ln \frac{q(x)}{p(x)}$$

here we can apply Jensen's inequality:

$$\sum_{x \in \mathscr{X}} p(x) \ln \frac{q(x)}{p(x)} \leq \ln \sum_{x \in \mathscr{X}} p(x) \frac{q(x)}{p(x)} = \ln \sum_{x \in \mathscr{X}} q(x) = \ln 1 = 0$$

Moreover, as the logarithm is strictly concave, Jensen's also tells us that equality holds if and only if $q(x)/p(x)$ is constant everywhere, the latter implies $p = q$. ■

In practice, in machine learning and in optimization, this measure of difference between distributions is really useful.
Suppose for instance, that $p(x)$ is the empirical distribution. We want to find $q(x)$ that approximates $p(x)$. We can use the KL divergence in the same way as we use canonical distances (e.g. Euclidean distance).
Here, in particular we can use KL divergence as a loss function: we can try fit an approximate to the true posterior, and the process to make sure that $q(x)$ fits $p(x)$ is to minimize the KL divergence between them.

In the following theorem we show not only a typical road plan to approximate a set of sample data with a distribution, but also an interesting result that bridges KL divergence and maximum likelihood estimators (MLE).

Theorem 1.3 *Bridge between KL divergence and MLE.* Finding the MLE to an empirical set of sample data is the same as minimizing the KL divergence with respect to the empirical distribution.

Proof. Assume you have a set of sample data: $\mathbf{x}^\mu$ for $\mu = 1, ..., M$.
Consider the empirical distribution: $p(x) = \frac{1}{M}\sum_{\mu=1}^{M} \delta(\mathbf{x}, \mathbf{x}^\mu)$, and an approximating distribution $q_\theta(\mathbf{x})$ parametrized by θ.

We have:

$$KL(p||q_\theta) = \sum_{x\in\mathscr{X}} \frac{1}{M}\sum_{\mu=1}^{M} \delta(\mathbf{x}, \mathbf{x}^\mu) \ln\left(\frac{1}{M}\sum_{\mu=1}^{M}\frac{\delta(\mathbf{x}, \mathbf{x}^\mu)}{q_\theta(x)}\right) =$$

$$\sum_{x\in\mathscr{X}} \frac{1}{M}\sum_{\mu=1}^{M} \delta(\mathbf{x}, \mathbf{x}^\mu)\left[\ln\left(\frac{1}{M}\sum_{\mu=1}^{M}\delta(\mathbf{x}, \mathbf{x}^\mu)\right) - \ln q_\theta(x)\right] =$$

$$= \frac{1}{M}\sum_{\mu=1}^{M}\sum_{x\in\mathscr{X}} \delta(\mathbf{x}, \mathbf{x}^\mu)\ln\frac{1}{M} + \frac{1}{M}\sum_{\mu=1}^{M}\sum_{x\in\mathscr{X}} \delta(\mathbf{x}, \mathbf{x}^\mu)\ln\left(\sum_{\mu=1}^{M}\delta(\mathbf{x}, \mathbf{x}^\mu)\right) - \frac{1}{M}\sum_{\mu=1}^{M}\sum_{x\in\mathscr{X}} \delta(\mathbf{x}, \mathbf{x}^\mu)\ln q_\theta(x) =$$

$$= \ln\frac{1}{M} - \frac{1}{M}\sum_{\mu=1}^{M}\sum_{x\in\mathscr{X}} \delta(\mathbf{x}, \mathbf{x}^\mu)\ln q_\theta(x) = \ln\frac{1}{M} - \frac{1}{M}\sum_{\mu=1}^{M}\ln q_\theta(\mathbf{x}^\mu) = \ln\frac{1}{M} - \frac{1}{M}\ln\left(\prod_{\mu=1}^{M} q_\theta(\mathbf{x}^\mu)\right)$$

Now, by taking the partial derivative in θ of the right hand side, and setting it equal to zero, it is clear that the procedure to find $q_\theta(x)$ in order to minimize the KL divergence is the same procedure we would use to find the MLE. ■

We end this chapter with a pratical example.

Example. You want to approximate the binomial distribution with n numbers of trials and probability p with a gaussian distribution with mean μ and variance σ^2. To find a suitable distribution you investigate the KL divergence in terms of the parameters $\theta = (\mu, \sigma^2)$.

1. Write down the KL divergence.
2. Derive the gradients with respect to θ.
3. Is there an analytic solution for the optimal parameter setting? If yes, derive the corresponding solution. If no, give a short reasoning.
4. Indipendent of the previous exercise, state a numerical procedure to minimize the KL divergence.

Solution.

1. This is just a matter of computation:

$$D_{KL}(f||q) = \mathbb{E}_f\left[\ln \frac{f(X)}{q_\theta(X)}\right] = \mathbb{E}_f[\ln f(X)] - \mathbb{E}_f[\ln q_\theta(X)]$$

2. For the gradients, we must derive the partial derivatives of the second part of the KL divergence. The involved log-density is:

$$\ln q_\theta(X) = const - 0.5 \ln \sigma^2 - \frac{1}{2\sigma^2}(X-\mu)^2$$

$$\frac{\partial D_{KL}(f||q)}{\partial \mu} = -\mathbb{E}_f \ln q_\theta(x) = \mathbb{E}_f \frac{1}{\sigma^2}(X-\mu) \tag{2.1}$$

$$\frac{\partial D_{KL}(f||q)}{\partial \sigma^2} = -\mathbb{E}_f \ln q_\theta(x) = \mathbb{E}_f\left[\frac{1}{2\sigma^2} + \frac{-1}{2\sigma^2}(X-\mu)^2\right] \tag{2.2}$$

3. Yes, there is. We can first set (1.1) to zero and get: $\mu = \mathbb{E}_f(X) \iff \mu = np$. We now use this solutions for (1.2), which we also set to zero:

   $$(1.2) = 0 \iff \mathbb{E}_f[(X-\mu)^2] = Var_f(X) + (\mathbb{E}_f[X-\mu])^2 = np(1-p) + (\mathbb{E}_f[X-\mu])^2$$

   Using $\mu = np$, the second term vanishes and we get the optimal $\sigma^2 = np(1-p) = Var_f(X)$. Note that we would have to prove that the second derivative is < 0 to be sure that we found a minimum.

4. We could, alternatively, use the gradients and do gradient descent to find the optimal θ.

3 Linear Algebra intermezzo

In order to deeply understand the next chapters, it is necessary a review of some fundamental linear algebra.

Definition 2.1 *Diagonalizable matrix.* A matrix $A \in \mathbb{R}^{nxn}$ is *diagonalizable* if it is similar to a diagonal matrix, i.e., if there exists an invertible matrix $P \in \mathbb{R}^{nxn}$ such that $D = P^{-1}AP$, where $D \in \mathbb{R}^{nxn}$ is diagonal.

Theorem 2.2 *Eigendecomposition.* A square matrix $A \in \mathbb{R}^{nxn}$ can be factored into

$$A = PDP^{-1}$$

where $P \in \mathbb{R}^{nxn}$ and D is a diagonal matrix whose diagonal entries are the eigenvalues of A, if and only if the eigenvectors of A form a basis of $\mathbb{R}^n$.

Theorem 2.3 *Consequence of the spectral theorem.* A symmetric matrix $S \in \mathbb{R}^{nxn}$ can always be diagonalized. The proof is omitted, for those who are interested in a further understanding it is recommended to study the spectral theorem on some linear algebra book.

Definition 2.4 *positive definite matrix.* A square matrix is called *positive* if it is symmetric and all it eigenvalues λ are positive, that is $\lambda > 0$.

Theorem 2.5 If A is positive definite, the it is invertible and $det(A) > 0$.

Proof. If $A \in \mathbb{R}^{nxn}$ and the eigenvalues (with multiplicity) are $\lambda_1, \lambda_2, ...\lambda_n$, then $det(A) = \lambda_1\lambda_2...\lambda_n > 0$. ■

Theorem 2.6 A symmetric matrix A is positive definite if and only if $\mathbf{v} \cdot A\mathbf{v} > 0$ for any $\mathbf{v} \in \mathbb{R}^n$

Proof. A is symmetric so let $P^TAP = D = diag(\lambda_1, \lambda_2, ..., \lambda_n)$ where $P^{-1} = P^T$ and the λ_i are the eigenvalues of A. Given a vector $\mathbf{x} \in \mathbb{R}^n$, write $\mathbf{y} = P^T\mathbf{x} = [y_1, y_2, ..., y_n]^T$. Then

$$\mathbf{x}^TA\mathbf{x} = \mathbf{x}^T(PDP^T)\mathbf{x} = \mathbf{y}^TD\mathbf{y} = \lambda_1y_1^2 + \lambda_2y_2^2 + ... + \lambda_ny_n^2 \tag{3.1}$$

If A is positive definite and $\mathbf{x} \neq \mathbf{0}$, then $\mathbf{x}^TA\mathbf{x} > 0$ because some $y_j \neq 0$ and every $\lambda_i > 0$. Conversely, if $\mathbf{x}^TA\mathbf{x} > 0$ whenever $\mathbf{x} \neq \mathbf{0}$, let $\mathbf{x} = P\mathbf{e}_j \neq \mathbf{0}$ where $\mathbf{e}_j$ is column j of Id_n. Then $\mathbf{y} = \mathbf{e}_j$, so (2.1) reads $\lambda_j = \mathbf{x}^TA\mathbf{x} > \mathbf{0}$. ■

Example. If U is any invertible matrix with $U \in \mathbb{R}^{nxn}$, show that $A = U^TU$ is positive definite.

Solution. If $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{x} \neq \mathbf{0}$, then

$$\mathbf{x}^TA\mathbf{x} = \mathbf{x}^T(U^TU)\mathbf{x} = (U\mathbf{x})^T(U\mathbf{x}) = \|U\mathbf{x}\|^2 > 0$$

4 PCA

Analysing high-dimensional data, such as images, gene expressions, neurons spiking rates,... comes with some difficulties:

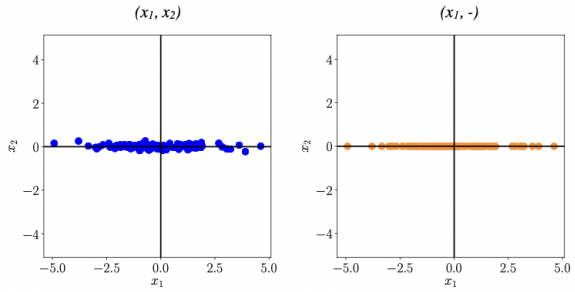
- It is hard to analyze.
- Interpretation is difficut.
- Visualization is nearly impossible.
- Storage of the data vectors can be expensive.

We can exploit some properties to make the study more efficient:

- High-dimensional data is often overcomplete, i.e., many dimensions are redundant and can be explained by a combination of other dimensions.
- Dimensions in high-dimensional data are often correlated (intrinsic lower-dimensional structure).

Dimensionality reduction exploits structure and correlation and allows us to work with a more compact representation of the data, ideally without losing information.
We can think of dimensionality reduction as a compression technique, similar to *jpeg* or *mp*3, which are compression algorithms for images and music.

PCA, proposed by Pearson (1901) and Hotelling (1933), is one of the most important techniques used for data compression and data visualization. It is also used for the identification of simple patterns, latent factors, and structures of high dimensional data.
Dimensionality reduction generally exploits a property of high-dimensional data (e.g., images) that it often lies on a low-dimensional subspace.
For instance, in the following two pictures, we see on the left the plot of some 2-dimensional data points, it is clear that they spread out a lot along the horizontal axis, while they don't spread much along the vertical axis: this means that the horizontal axis is much more informative! In fact, on the right, we visualize only the horizontal axis of the same data points, and the pictures seem quite the same. The generalization of this intuition leads us to PCA.

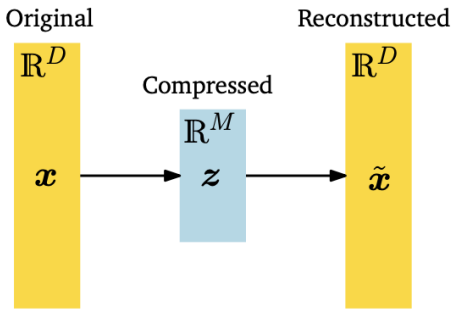


General setting: We are interested in finding projections of data points $\mathbf{x}_n \in \mathbb{R}^D$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.
What does this mean, mathematically?

Data: We consider an i.i.d. dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n\}, \mathbf{x}_n \in \mathbb{R}^D$. To ease notation, without loss of generality, we assume the data to be centered, i.e. its mean is $\mathbf{0}$.
We would like to encode in a lower dimensional space while still being able to decode without loosing information.
We therefore seek for a low dimensional representation of $\mathbf{x}_n$, *(a code)*.

$$\mathbf{z}_n = B^T \mathbf{x}_n \in \mathbb{R}^M, \qquad M < D$$

where $B := [\mathbf{b}_1, ..., \mathbf{b}_M] \in \mathbb{R}^{DxM}$.

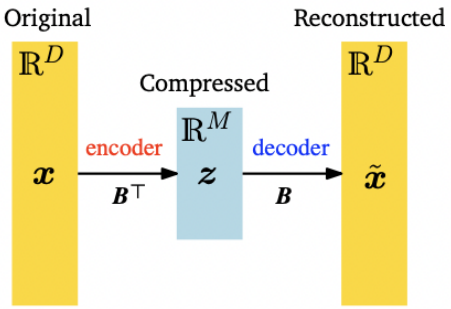


We assume that:

- The columns of B are orthonormal so that $\mathbf{b}_i^T \mathbf{b}_j = 0$ if and only if $i \neq j$, and $\mathbf{b}_i^T \mathbf{b}_i = 1$.
- We seek an M-dimensional subspace $U \subseteq \mathbb{R}^D$, $dim(U) = M < D$ onto which we project the data.
- We denote the projected data by $\tilde{\mathbf{x}}_n \in U$, and their coordinates (with basis vectors $\mathbf{b}_1, ..., \mathbf{b}_M$ of U) by $\mathbf{z}_n$.

Our aim is to find codes $\mathbf{z}_n$ and the basis vectors $\mathbf{b}_1, ..., \mathbf{b}_M$ so that the projections $\tilde{\mathbf{x}}_n \in \mathbb{R}^D$ are as similar to the original data $\mathbf{x}_n$: e.g. $\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$ as small as possible.

In PCA, we consider a linear relationship between the orginal data $\mathbf{x}$ and its low-dimensional code $\mathbf{z}$, so that $\mathbf{z} = B^T\mathbf{x}$ and $\tilde{\mathbf{x}} = B\mathbf{z}$ for a suitable matrix B.



- **Decoder**: the linear mapping represented by B can be thought of a *decoder*, which maps the low-dimensional code $\mathbf{z} \in \mathbb{R}^M$ back into the original data space $\mathbb{R}^D$.
- **Encoder**: similarly, B^T can be thought of an *encoder*, which encodes the orignal data $\mathbf{x}$ as a low-dimensional (compressed) code $\mathbf{z}$.

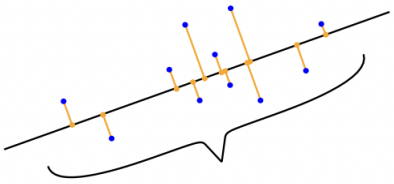
So, we seek for the matrix B that retains as much information as possible: this can be achieved in with two different approaches that lead to the same result.
In the *Maximum variance perspective* section, we will find low-dimensional representations that retain as much information as possible and minimize the compression loss.
In the *Projection perspective* section, we will be looking at minimizing the squared reconstruction error $\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$ between the orginal data $\mathbf{x}_n$ and its projection $\tilde{\mathbf{x}}_n$.

4.1 Maximum Variance Perspective

In the first picture in the **PCA** section, we gave an example of how a two-dimensional dataset can be represented using a single coordinate: in the right hand picture we chose to ignore the x_2-coordinate of the data because it didn't add too much information, so that the compressed data is similar to the original data in the left hand picture. We could have chosen to ignore the x_1-coordinate, but then the compressed data had been very dissimilar from the original data, and much information in the data would have been lost.

If we interpret information content in the data as how "space filling" the dataset is, then we can describe the information contained in the data by looking at the spread of the data, and we can derive **PCA** as a dimensionality reduction algorithm that maximizes the variance in the low-dimensional representation of the data to retain as much information as possible.

The following picture illustrates this:



So, our goal is to find a matrix B that retains as much information as possible when compressing data by projectin it onto the subspace spanned by the columns $\mathbf{b}_1, ..., \mathbf{b}_M$ of B. Retaining most information after data compression is equivalent to capturing the largest amount of variance in the low-dimensional code (this comes from a theorem proved by Hotelling in 1933).

Remark: *Data covariance matrix*. Given the dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$, with mean $\mathbf{0}$, the *data covariance matrix* is:

$$S := \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^T$$

Remark: *Centered data*. For the data covariance matrix we assumed *centered data*. We can make this assumption without loss of generality.
Let us assume that $\boldsymbol{\mu}$ is the mean of the data. Using properties of the variance, we get:

$$\mathbb{V}_{\mathbf{z}}[\mathbf{z}] = \mathbb{V}_{\mathbf{x}}[B^T(\mathbf{x} - \boldsymbol{\mu})] = \mathbb{V}_{\mathbf{x}}[B^T\mathbf{x} - B^T\boldsymbol{\mu}] = \mathbb{V}_{\mathbf{x}}[B^T\mathbf{x}]$$

i.e., the variance of the low-dimensional code does not depend on the mean of the data. Therefore, we assume without loss of generality that the data has mean $\mathbf{0}$ for the remainder of this section. With this assumption the mean of the low-dimensional code is also $\mathbf{0}$ since $\mathbb{E}_{\mathbf{z}}[\mathbf{z}] = \mathbb{E}_{\mathbf{x}}[B^T\mathbf{x}] = B^T\mathbb{E}_{\mathbf{x}}[\mathbf{x}] = \mathbf{0}$.

We maximize the variance of the low-dimensional code using a sequential approach. We

start by seeking a single vector $\mathbf{b}_1 \in \mathbb{R}^D$ that maximizes the variance of the projected data, i.e., we aim to maximize the variance of the first coordinate z_1 of $\mathbf{z} \in \mathbb{R}^M$ so that:

$$V_1 := \mathbb{V}[z_1] = \frac{1}{N}\sum_{n=1}^{N} z_{1n}^2 \tag{4.1}$$

is maximized, where we exploited the i.i.d. assumption of the data and defined z_{1n} as the first coordinate of the low-dimensional representation $\mathbf{z}_n \in \mathbb{R}^M$ of $\mathbf{x}_n \in \mathbb{R}^D$.
Note that first component of $\mathbf{z}_n$ is given by:

$$z_{1n} = \mathbf{b}_1^T \mathbf{x}_n \tag{4.2}$$

i.e., it is the coordinate of the orthogonal projection of $\mathbf{x}_n$ onto the one-dimensional subspace spanned by $\mathbf{b}_1$.
We substitute (3.2) into (3.1), which yelds:

$$V_1 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{b}_1^T \mathbf{x}_n)^2 = \frac{1}{N}\sum_{n=1}^{N} \mathbf{b}_1^T \mathbf{x}_n \mathbf{x}_n^T \mathbf{b}_1 = \tag{4.3}$$

$$= \mathbf{b}_1^T \left(\frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^T\right)\mathbf{b}_1 = \mathbf{b}_1^T S \mathbf{b}_1 \tag{4.4}$$

where S is the data covariance matrix defined previously.
In (3.3) we have used the fact that the dot product of two vectors is symmetric with respect to its arguments, that is, $\mathbf{b}_1 \mathbf{x}_n = \mathbf{x}_n^T \mathbf{b}_1$.
Notice that arbitrarily increasing the magniture of the vector $\mathbf{b}_1$ increases V_1, that is, a vector $\mathbf{b}_1$ that is two times larger can result in V_1 that is potentially four times larger. Therefore, we restrict all solutinos to $\|\mathbf{b}_1\|^2 = 1$, which results in a constraine optimization problem in which we seek the direction along which the data varies most.
With the restriction of the solution space to unit vectors, the vector $\mathbf{b}_1$ that points in the direction of maximum variance can be found by the constrained optimization problem:

$$\max_{\mathbf{b}_1} \ \mathbf{b}_1^T S \mathbf{b}_1$$

$$\text{subject to } \|\mathbf{b}_1\|^2 = 1$$

This is equivalent to maximize the Lagrangian:

$$\mathcal{L}(\mathbf{b}_1, \lambda) = \mathbf{b}_1^T S \mathbf{b}_1 + \lambda(1 - \mathbf{b}_1^T \mathbf{b}_1).$$

The partial derivatives of $\mathcal{L}$ with respect to $\mathbf{b}_1$ and λ are:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}_1} = 2\mathbf{b}_1^t S - 2\lambda \mathbf{b}_1^T, \qquad \frac{\partial \mathcal{L}}{\partial \lambda} = 1 - \mathbf{b}_1^T \mathbf{b}_1$$

respectively. Setting these partial derivatives to $\mathbf{0}$ gives us the relations:

$$S\mathbf{b}_1 = \lambda \mathbf{b}_1,$$

$$\mathbf{b}_1^T \mathbf{b}_1 = 1.$$

This is the eigenvector problem!
We see that $\mathbf{b}_1$ is an eigenvector of the data covariance matrix S, and the Lagrange multiplier λ plays the role of the corresponding eigenvalue. So we can rewrite the variance objective as:

$$V_1 = \mathbf{b}_1^T S \mathbf{b}_1 = \lambda \mathbf{b}_1^T \mathbf{b}_1 = \lambda$$

i.e., the variance of the data projected onto a one-dimensional subspace equals the eigenvalue that is associated with the largest eigenvalue of the data covariance matrix. This eigenvector is called the *first principal component*. We can determine the effect/contribution of the principal component $\mathbf{b}_1$ in the original data space by mapping the coordinate z_{1n} back into data space, which gives us the projected data point:

$$\tilde{\mathbf{x}}_n = \mathbf{b}_1 z_{1n} = \mathbf{b}_1 \mathbf{b}_1^T \mathbf{x}_n \in \mathbb{R}^D$$

in the original space.
Remark. Although $\tilde{\mathbf{x}}_n$ is a D-dimensional vector, it only requires a single coordinate z_{1n} to represent it with respect to the basis vector $\mathbf{b}_1 \in \mathbb{R}^D$.

...

5