Linear Dimensionality Reduction

As before, you only have a training dataset consisting of \( \{x_n\}_{n=1}^N \).
As before, you only have a training dataset consisting of $\langle x_n \rangle_{n=1}^{N}$.

Is there a way to represent each $x_n \in \mathbb{R}^d$ as a lower-dimensional vector?
Linear Dimensionality Reduction

As before, you only have a training dataset consisting of \( \{x_n\}_{n=1}^N \).

Is there a way to represent each \( x_n \in \mathbb{R}^d \) as a lower-dimensional vector?

(Why would we want to do this?)
Dimension of Greatest Variance

Assume that the data are centered, i.e., that $\text{mean} \left( \langle x_n \rangle_{n=1}^{N} \right) = 0$. 
Dimension of Greatest Variance

Assume that the data are *centered*, i.e., that
\[
\text{mean } \left( \langle x_n \rangle_{n=1}^N \right) = 0.
\]
Projection into One Dimension

Let \( u \) be the dimension of greatest variance, and (without loss of generality) let \( \|u\|_2^2 = 1 \).

\( p_n = x_n \cdot u \) is the projection of the \( n \)th example onto \( u \).

(This should remind you a little bit of the perceptron's activation, \( w \cdot x_n + b \).)

Since the mean of the data is 0, the mean of \( \langle p_1, \ldots, p_N \rangle \) is also 0.

This implies that the variance of \( \langle p_1, \ldots, p_N \rangle \) is \( \frac{1}{N} \sum_{n=1}^{N} p_n^2 \).

The \( u \) that gives the greatest variance, then, is:

\[
\text{argmax} \quad u \sum_{n=1}^{N} (x_n \cdot u)^2
\]

(Where did \( N \) go?)
Projection into One Dimension

Let \( \mathbf{u} \) be the dimension of greatest variance, and (without loss of generality) let \( \| \mathbf{u} \|_2^2 = 1 \).

\( p_n = \mathbf{x}_n \cdot \mathbf{u} \) is the projection of the \( n \)th example onto \( \mathbf{u} \).

(This should remind you a little bit of the perceptron’s activation, \( \mathbf{w} \cdot \mathbf{x}_n + b \).)
Projection into One Dimension

Let \( u \) be the dimension of greatest variance, and (without loss of generality) let \( \|u\|_2^2 = 1 \).

\( p_n = \mathbf{x}_n \cdot \mathbf{u} \) is the projection of the \( n \)th example onto \( u \).

(This should remind you a little bit of the perceptron’s activation, \( \mathbf{w} \cdot \mathbf{x}_n + b \).)

Since the mean of the data is 0, the mean of \( \langle p_1, \ldots, p_N \rangle \) is also 0.
Let $\mathbf{u}$ be the dimension of greatest variance, and (without loss of generality) let $\|\mathbf{u}\|_2^2 = 1$.

$p_n = x_n \cdot \mathbf{u}$ is the projection of the $n$th example onto $\mathbf{u}$.

(This should remind you a little bit of the perceptron’s activation, $\mathbf{w} \cdot x_n + b$.)

Since the mean of the data is 0, the mean of $\langle p_1, \ldots, p_N \rangle$ is also 0.

This implies that the variance of $\langle p_1, \ldots, p_N \rangle$ is $\frac{1}{N} \sum_{n=1}^{N} p_n^2$. 


Projection into One Dimension

Let $\mathbf{u}$ be the dimension of greatest variance, and (without loss of generality) let $\|\mathbf{u}\|_2^2 = 1$.

$p_n = \mathbf{x}_n \cdot \mathbf{u}$ is the projection of the $n$th example onto $\mathbf{u}$.
(This should remind you a little bit of the perceptron’s activation, $\mathbf{w} \cdot \mathbf{x}_n + b$.)

Since the mean of the data is $0$, the mean of $\langle p_1, \ldots, p_N \rangle$ is also $0$.

This implies that the variance of $\langle p_1, \ldots, p_N \rangle$ is $\frac{1}{N} \sum_{n=1}^{N} p_n^2$.

The $\mathbf{u}$ that gives the greatest variance, then, is:

$$\text{argmax}_{\mathbf{u}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2$$
Projection into One Dimension

Let \( \mathbf{u} \) be the dimension of greatest variance, and (without loss of generality) let \( \|\mathbf{u}\|_2^2 = 1 \).

\( p_n = \mathbf{x}_n \cdot \mathbf{u} \) is the projection of the \( n \)th example onto \( \mathbf{u} \). (This should remind you a little bit of the perceptron’s activation, \( \mathbf{w} \cdot \mathbf{x}_n + b \).)

Since the mean of the data is \( 0 \), the mean of \( \langle p_1, \ldots, p_N \rangle \) is also \( 0 \).

This implies that the variance of \( \langle p_1, \ldots, p_N \rangle \) is \( \frac{1}{N} \sum_{n=1}^{N} p_n^2 \).

The \( \mathbf{u} \) that gives the greatest variance, then, is:

\[
\argmax_{\mathbf{u}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2
\]

(Where did \( N \) go?)
Finding the Maximum-Variance Direction

$$\arg\max_u \sum_{n=1}^{N} (x_n \cdot u)^2$$

s.t. $\|u\|_2^2 = 1$

(If we didn’t constrain $u$ to have length 1, it could increase the objective arbitrarily in a way that has nothing to do with variance in the data!)
Finding the Maximum-Variance Direction

\[ \text{argmax}_u \sum_{n=1}^{N} (x_n \cdot u)^2 \]
\[ \text{s.t. } \|u\|_2^2 = 1 \]

(If we didn’t constrain \( u \) to have length 1, it could increase the objective arbitrarily in a way that has nothing to do with variance in the data!)

If we let \( X = \begin{bmatrix} x_1^\top \\ x_2^\top \\ \vdots \\ x_N^\top \end{bmatrix} \), then we want: \( \text{argmax}_u \|Xu\|_2^2 \), s.t. \( \|u\|_2^2 = 1 \).
Constrained Optimization

The blue lines represent *isobars*: all points on a blue line have the same objective function value.
Constrained Optimization

The blue lines represent *isobars*: all points on a blue line have the same objective function value. The red circle is all points with a norm of 1. It represents a constraint like the one we have in the maximum-variance projection problem.
Deriving the Solution

Don’t panic.

\[
\begin{align*}
\argmax_u \| Xu \|^2_2, \text{ s.t. } \| u \|^2_2 &= 1
\end{align*}
\]
Deriving the Solution

Don’t panic.

\[
\arg\max_u \|Xu\|_2^2, \text{ s.t. } \|u\|_2^2 = 1
\]

- The Lagrangian encoding of the problem moves the constraint into the objective:

\[
\max_u \min_\lambda \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1) \quad \Rightarrow \quad \min_\lambda \max_u \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1)
\]
Deriving the Solution

Don’t panic.

\[
\arg\max_u \|Xu\|_2^2, \text{ s.t. } \|u\|_2^2 = 1
\]

▶ The Lagrangian encoding of the problem moves the constraint into the objective:

\[
\max_u \min_\lambda \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1) \quad \Rightarrow \quad \min_\lambda \max_u \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1)
\]

▶ Gradient (first derivatives with respect to \(u\)): \(2X^\top Xu - 2\lambda u\)
Deriving the Solution
Don’t panic.

\[
\arg\max_u \|Xu\|_2^2, \text{ s.t. } \|u\|_2^2 = 1
\]

- The Lagrangian encoding of the problem moves the constraint into the objective:

\[
\max_u \min_\lambda \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1) \quad \Rightarrow \quad \min_\lambda \max_u \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1)
\]

- Gradient (first derivatives with respect to \(u\)): \(2X^\top Xu - 2\lambda u\)
- Setting equal to 0 leads to: \(\lambda u = X^\top Xu\)
Deriving the Solution

Don’t panic.

\[
\arg\max_u \|Xu\|_2^2, \text{ s.t. } \|u\|_2^2 = 1
\]

- The Lagrangian encoding of the problem moves the constraint into the objective:

\[
\max_u \min_\lambda \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1) \Rightarrow \min_\lambda \max_u \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1)
\]

- Gradient (first derivatives with respect to \(u\)): \(2X^\top X u - 2\lambda u\)
- Setting equal to 0 leads to: \(\lambda u = X^\top X u\)
- You may recognize this as the definition of an eigenvector \((u)\) and eigenvalue \((\lambda)\) for the matrix \(X^\top X\).
Deriving the Solution

Don’t panic.

\[
\arg\max_u \|Xu\|_2^2, \text{ s.t. } \|u\|_2^2 = 1
\]

- The Lagrangian encoding of the problem moves the constraint into the objective:

\[
\max_u \min_\lambda \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1) \Rightarrow \min_\lambda \max_u \|Xu\|_2^2 - \lambda(\|u\|_2^2 - 1)
\]

- Gradient (first derivatives with respect to \(u\)): \(2X^\top Xu - 2\lambda u\)
- Setting equal to 0 leads to: \(\lambda u = X^\top Xu\)
- You may recognize this as the definition of an eigenvector (\(u\)) and eigenvalue (\(\lambda\)) for the matrix \(X^\top X\).
- We take the first (largest) eigenvalue.
Projecting into Multiple Dimensions

So far, we’ve projected each $x_n$ into one dimension.
Projecting into Multiple Dimensions

So far, we’ve projected each $x_n$ into one dimension.

To get a second projection $v$, we solve the same problem again, but this time with another constraint:

$$\arg\max_v \|Xv\|_2^2, \text{ s.t. } \|v\|_2^2 = 1 \text{ and } u \cdot v = 0$$

(That is, we want a dimension that’s orthogonal to the $u$ that we found earlier.)
Projecting into Multiple Dimensions

So far, we’ve projected each $x_n$ into one dimension.

To get a second projection $v$, we solve the same problem again, but this time with another constraint:

$$\arg\max_v \|Xv\|_2^2, \text{ s.t. } \|v\|_2^2 = 1 \text{ and } u \cdot v = 0$$

(That is, we want a dimension that’s orthogonal to the $u$ that we found earlier.)

Following the same steps we had for $u$, we can show that the solution will be the second eigenvector.
“Eigenfaces”

Fig. from https://github.com/AlexOuyang/RealTimeFaceRecognition
Principal Components Analysis

**Data:** unlabeled data with mean 0, \( X = [x_1 | x_2 | \cdots | x_N]^\top \), and dimensionality \( K < d \)

**Result:** \( K \)-dimensional projection of \( X \)

let \( \langle \lambda_1, \ldots, \lambda_K \rangle \) be the top \( K \) eigenvalues of \( X^\top X \)

and \( \langle u_1, \ldots, u_K \rangle \) be the corresponding eigenvectors;

let \( U = [u_1 | u_2 | \cdots | u_K] \);

return \( XU \);

**Algorithm 1:** PCA
Principal Components Analysis

**Data:** unlabeled data with mean 0, \( X = [x_1 | x_2 | \cdots | x_N]^\top \), and dimensionality \( K < d \)

**Result:** \( K \)-dimensional projection of \( X \)

let \( \langle \lambda_1, \ldots, \lambda_K \rangle \) be the top \( K \) eigenvalues of \( X^\top X \)

and \( \langle u_1, \ldots, u_K \rangle \) be the corresponding eigenvectors;

let \( U = [u_1 | u_2 | \cdots | u_K] \);

return \( XU \);

**Algorithm 2:** PCA

On your own time, you can read up about many algorithms for finding eigenstuff of a matrix.
Alternate View of PCA

Think of \( p_n = x_n U \) as a new, \( K \)-dimensional representation of \( x_n \).
Alternate View of PCA

Think of \( p_n = x_n U \) as a new, \( K \)-dimensional representation of \( x_n \).

This means that \( p_n U^T \approx x_n \). The closer these vectors are, the lower our reconstruction error, \( \|x_n - p_n U^T\|_2^2 \).
Alternate View of PCA

Think of $p_n = x_n U$ as a new, $K$-dimensional representation of $x_n$.

This means that $p_n U^\top \approx x_n$. The closer these vectors are, the lower our reconstruction error, $\|x_n - p_n U^\top\|^2$.

We could have derived PCA by saying that our goal is to minimize the total reconstruction error on the data:

$$\min_U \|X - XUU^\top\|^2_2$$

s.t. $U^\top U = 1$
To select $K$ for PCA, you can use the same criteria we discussed for $K$-Means (BIC and AIC).
PCA and Clustering

There’s a unified view of both PCA and clustering.

- $K$-Means chooses cluster-means so that squared distances to data are small.
- PCA chooses projections so that reconstruction error of data is small.
PCA and Clustering

There’s a unified view of both PCA and clustering.

- $K$-Means chooses cluster-means so that squared distances to data are small.
- PCA chooses projections so that reconstruction error of data is small.

Both are trying to find a “simple” way to summarize the data; fewer points, or fewer dimensions.
PCA and Clustering

There’s a unified view of both PCA and clustering.

- *K*-Means chooses cluster-means so that squared distances to data are small.
- PCA chooses projections so that reconstruction error of data is small.

Both are trying to find a “simple” way to summarize the data; fewer points, or fewer dimensions.

Both could be used to create new features for supervised learning!