Machine Learning (CSE 446): Unsupervised Learning: Linear Dimensionality Reduction

Noah Smith

University of Washington
nasmith@cs.washington.edu

October 16, 2017

Administrivia

No office hour for me on Wednesday this week (today is as usual).
No office hour for Kousuke on Tuesday.

Please take the anonymous "quiz" on Canvas to give feedback to Swabha!

Linear Dimensionality Reduction

As before, you only have a training dataset consisting of $\langle \mathbf{x}_n \rangle_{n=1}^{N}$.

Linear Dimensionality Reduction

As before, you only have a training dataset consisting of $\langle \mathbf{x}_n \rangle_{n=1}^N$.

Is there a way to represent each $\mathbf{x}_n \in \mathbb{R}^d$ as a lower-dimensional vector?

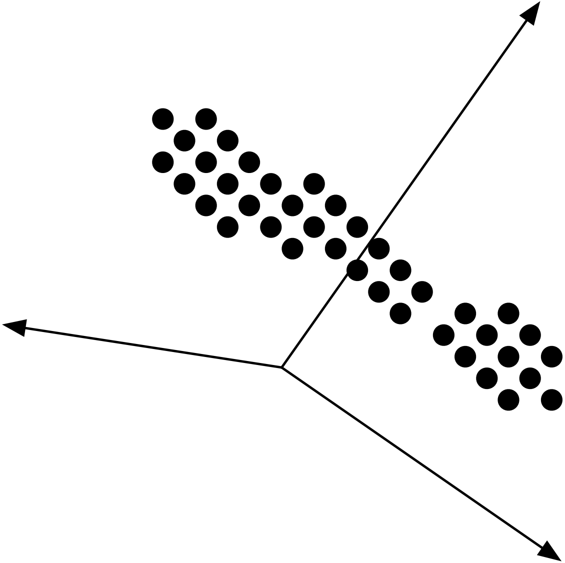
Linear Dimensionality Reduction

As before, you only have a training dataset consisting of $\langle \mathbf{x}_n \rangle_{n=1}^{N}$.

Is there a way to represent each $\mathbf{x}_n \in \mathbb{R}^d$ as a lower-dimensional vector?

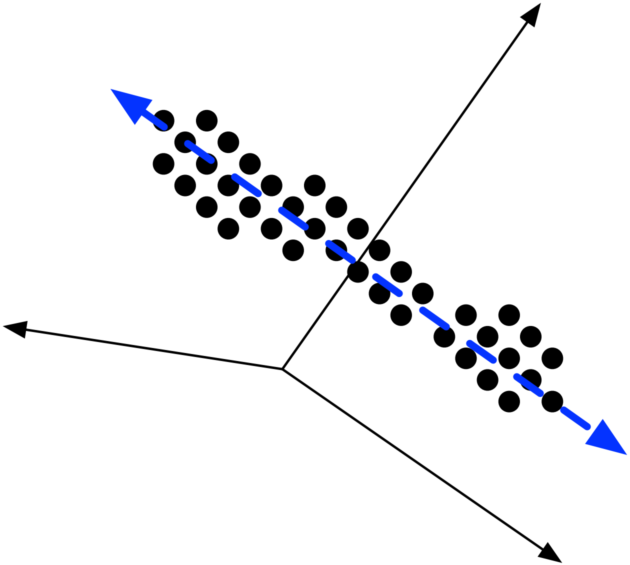
(Why would we want to do this?)

Dimension of Greatest Variance



Assume that the data are *centered*,
i.e., that
mean $\left(\langle \mathbf{x}_n \rangle_{n=1}^{N}\right) = \mathbf{0}$.

Dimension of Greatest Variance



Assume that the data are *centered*,
i.e., that
mean $\left(\langle \mathbf{x}_n \rangle_{n=1}^{N}\right) = \mathbf{0}$.

Projection into One Dimension

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

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 nth 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 $\mathbf{u}$ 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 nth 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 $\mathbf{0}$, the mean of $\langle p_1, \ldots, p_N \rangle$ is also 0.

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 nth 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 $\mathbf{0}$, 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 $\dfrac{1}{N} \displaystyle\sum_{n=1}^{N} p_n^2$.

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 nth 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 $\mathbf{0}$, 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 $\dfrac{1}{N} \displaystyle\sum_{n=1}^{N} p_n^2$.

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

$$\operatorname*{argmax}_{\mathbf{u}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2$$

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 nth 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 $\mathbf{0}$, 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:

$$\underset{\mathbf{u}}{\operatorname{argmax}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2$$

(Where did N go?)

Finding the Maximum-Variance Direction

$$\underset{\mathbf{u}}{\operatorname{argmax}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2$$
$$\text{s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

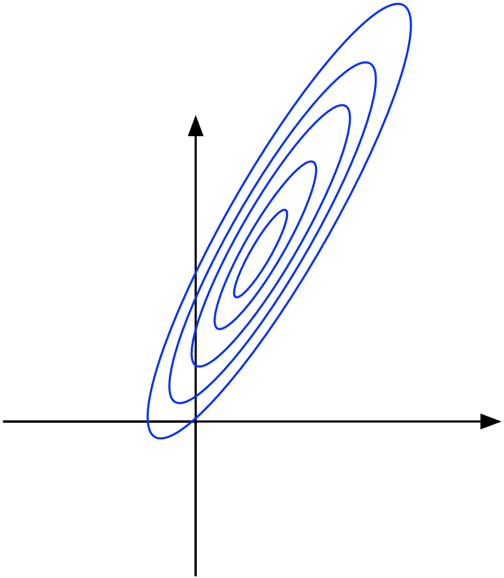
Finding the Maximum-Variance Direction

$$\underset{\mathbf{u}}{\operatorname{argmax}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2$$
$$\text{s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

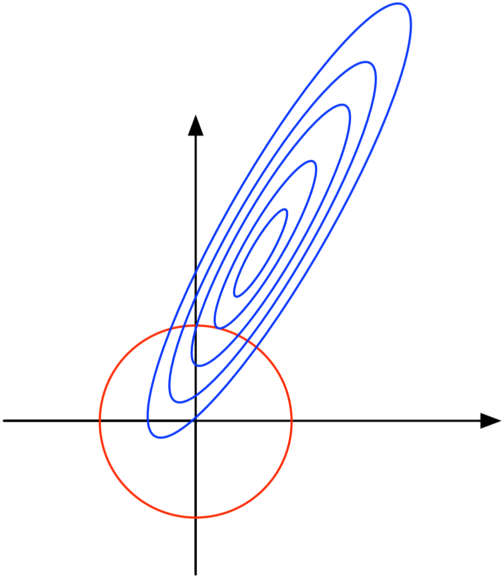
If we let $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{bmatrix}$, then we want: $\underset{\mathbf{u}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{u}\|_2^2$, s.t. $\|\mathbf{u}\|_2^2 = 1$.

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.

$$\underset{\mathbf{u}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{u}\|_2^2, \text{ s.t. } \|\mathbf{u}\|_2^2 = 1$$

Deriving the Solution

Don't panic.

$$\underset{\mathbf{u}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{u}\|_2^2, \text{ s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

$$\max_{\mathbf{u}} \min_{\lambda} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1) \quad \Rightarrow \quad \min_{\lambda} \max_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1)$$

Deriving the Solution

Don't panic.

$$\operatorname*{argmax}_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2, \text{ s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

$$\max_{\mathbf{u}} \min_{\lambda} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1) \quad \Rightarrow \quad \min_{\lambda} \max_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1)$$

- Gradient (first derivatives with respect to $\mathbf{u}$): $2\mathbf{X}^\top\mathbf{X}\mathbf{u} - 2\lambda\mathbf{u}$

Deriving the Solution

Don't panic.

$$\underset{\mathbf{u}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{u}\|_2^2, \text{ s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

$$\max_{\mathbf{u}} \min_{\lambda} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1) \quad \Rightarrow \quad \min_{\lambda} \max_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1)$$

- Gradient (first derivatives with respect to $\mathbf{u}$): $2\mathbf{X}^\top\mathbf{X}\mathbf{u} - 2\lambda\mathbf{u}$
- Setting equal to $\mathbf{0}$ leads to: $\lambda\mathbf{u} = \mathbf{X}^\top\mathbf{X}\mathbf{u}$

Deriving the Solution

Don't panic.

$$\underset{\mathbf{u}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{u}\|_2^2, \text{ s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

$$\max_{\mathbf{u}} \min_{\lambda} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1) \quad \Rightarrow \quad \min_{\lambda} \max_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1)$$

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

Deriving the Solution

Don't panic.

$$\operatorname*{argmax}_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2, \text{ s.t. } \|\mathbf{u}\|_2^2 = 1$$

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

$$\max_{\mathbf{u}} \min_{\lambda} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1) \quad \Rightarrow \quad \min_{\lambda} \max_{\mathbf{u}} \|\mathbf{X}\mathbf{u}\|_2^2 - \lambda(\|\mathbf{u}\|_2^2 - 1)$$

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

Projecting into Multiple Dimensions

So far, we've projected each $\mathbf{x}_n$ into one dimension.

Projecting into Multiple Dimensions

So far, we've projected each $\mathbf{x}_n$ into one dimension.

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

$$\underset{\mathbf{v}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{v}\|_2^2, \text{ s.t. } \|\mathbf{v}\|_2^2 = 1 \text{ and } \boxed{\mathbf{u} \cdot \mathbf{v} = 0}$$

(That is, we want a dimension that's orthogonal to the $\mathbf{u}$ that we found earlier.)

Projecting into Multiple Dimensions

So far, we've projected each $\mathbf{x}_n$ into one dimension.

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

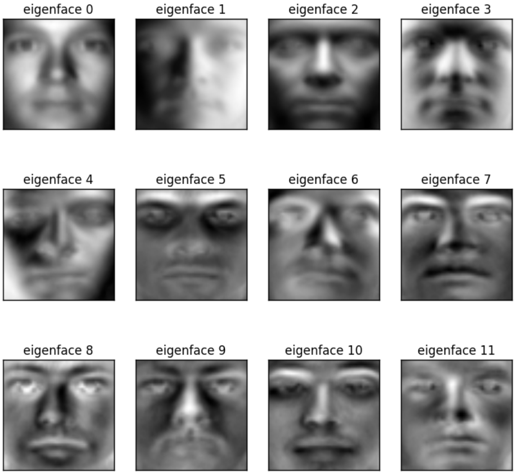
$$\operatorname*{argmax}_{\mathbf{v}} \|\mathbf{X}\mathbf{v}\|_2^2, \text{ s.t. } \|\mathbf{v}\|_2^2 = 1 \text{ and } \boxed{\mathbf{u} \cdot \mathbf{v} = 0}$$

(That is, we want a dimension that's orthogonal to the $\mathbf{u}$ that we found earlier.)

Following the same steps we had for $\mathbf{u}$, 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 $\mathbf{0}$, $\mathbf{X} = [\mathbf{x}_1|\mathbf{x}_2|\cdots|\mathbf{x}_N]^\top$, and dimensionality $K < d$
Result: K-dimensional projection of $\mathbf{X}$
let $\langle \lambda_1, \ldots, \lambda_K \rangle$ be the top K eigenvalues of $\mathbf{X}^\top \mathbf{X}$
and $\langle \mathbf{u}_1, \ldots, \mathbf{u}_K \rangle$ be the corresponding eigenvectors;
let $\mathbf{U} = [\mathbf{u}_1|\mathbf{u}_2|\cdots|\mathbf{u}_K]$;
return $\mathbf{XU}$;

Algorithm 1: PCA

Principal Components Analysis

Data: unlabeled data with mean $\mathbf{0}$, $\mathbf{X} = [\mathbf{x}_1 | \mathbf{x}_2 | \cdots | \mathbf{x}_N]^\top$, and dimensionality $K < d$
Result: K-dimensional projection of $\mathbf{X}$
let $\langle \lambda_1, \ldots, \lambda_K \rangle$ be the top K eigenvalues of $\mathbf{X}^\top \mathbf{X}$
  and $\langle \mathbf{u}_1, \ldots, \mathbf{u}_K \rangle$ be the corresponding eigenvectors;
let $\mathbf{U} = [\mathbf{u}_1 | \mathbf{u}_2 | \cdots | \mathbf{u}_K]$;
return $\mathbf{XU}$;

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 $\mathbf{p}_n = \mathbf{x}_n \mathbf{U}$ as a new, K-dimensional representation of $\mathbf{x}_n$.

Alternate View of PCA

Think of $\mathbf{p}_n = \mathbf{x}_n \mathbf{U}$ as a new, K-dimensional representation of $\mathbf{x}_n$.

This means that $\mathbf{p}_n \mathbf{U}^\top \approx \mathbf{x}_n$. The closer these vectors are, the lower our reconstruction error, $\|\mathbf{x}_n - \mathbf{p}_n \mathbf{U}^\top\|_2^2$.

Alternate View of PCA

Think of $\mathbf{p}_n = \mathbf{x}_n \mathbf{U}$ as a new, K-dimensional representation of $\mathbf{x}_n$.

This means that $\mathbf{p}_n \mathbf{U}^\top \approx \mathbf{x}_n$. The closer these vectors are, the lower our reconstruction error, $\|\mathbf{x}_n - \mathbf{p}_n \mathbf{U}^\top\|_2^2$.

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

$$\min_{\mathbf{U}} \left\| \mathbf{X} - \mathbf{X}\mathbf{U}\mathbf{U}^\top \right\|_2^2$$
$$\text{s.t. } \mathbf{U}^\top \mathbf{U} = \mathbf{1}$$

Choosing K (Hyperparameter Tuning)

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!

References I

Hal Daume. *A Course in Machine Learning (v0.9)*. Self-published at http://ciml.info/, 2017.