# Dimensionality Reduction

# 1 Motivation I: Data Compression

Dimensionality reduction means transforming from a given amount of features to a smaller amount of them (for example: 2D -> 1D). This is useful when we have redunant features (such as inches and cm) that we can delete and have the same performance but being faster.

# 2 Motivation II: Data Visualization

As we can only visualize data that has three or less dimensions, we can use dimensionality reduction in order to visualize datasets that have >3 dimensions. The hard part is how to select which features represent the data better.

# 3 Principal Component Analysis Problem Formulation

- Needs feature scaling before being applied.
- PCA tries to minimize the projection error.

Problem: Reduce from 2D to 1D: Find a direction onto which to project the data so as to minimize the projection error.

If we have more dimensions than that, we will find n vectors (where n is the amount of dimensions we want to end with) indicating the directions into which to project the data, minimizing the projection error.

Basically, PCA is trying to find a lower dimensional surface onto which to project the data so as to minimize the squared projection error (squared distance between each point and the location in which it gets projected).

## 3.1 PCA is not lineal regression

Linear regression fits a line according to the predictions of the hypothesis (minimizes mse between data and the hyphotesis), whereas PCA minimizes the squared error directly onto the projection vector.

# 4 PCA Algorithm

Remember to preprocess the data normalizing it before applying PCA!

En example of PCA from 2D to 1D can be seen in fig. 1. We need to know how to calculate the vector $u^{(1)}$ and all $z^{(i)}$.

The algorithm for finding out principal components can be found in algorithm 1.

# 5 Choosing the number of principal components

Typically, we choose $k$ to be the smallest value that still retains the 99% of the variance. That is the same as:

$$\frac{\text{average squared projection error}}{\text{total variation in the data}} = \frac{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - x^{(i)}_{\text{approx}}\right\|^2}{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}\right\|^2} \leq 0.01\,(1\%)$$

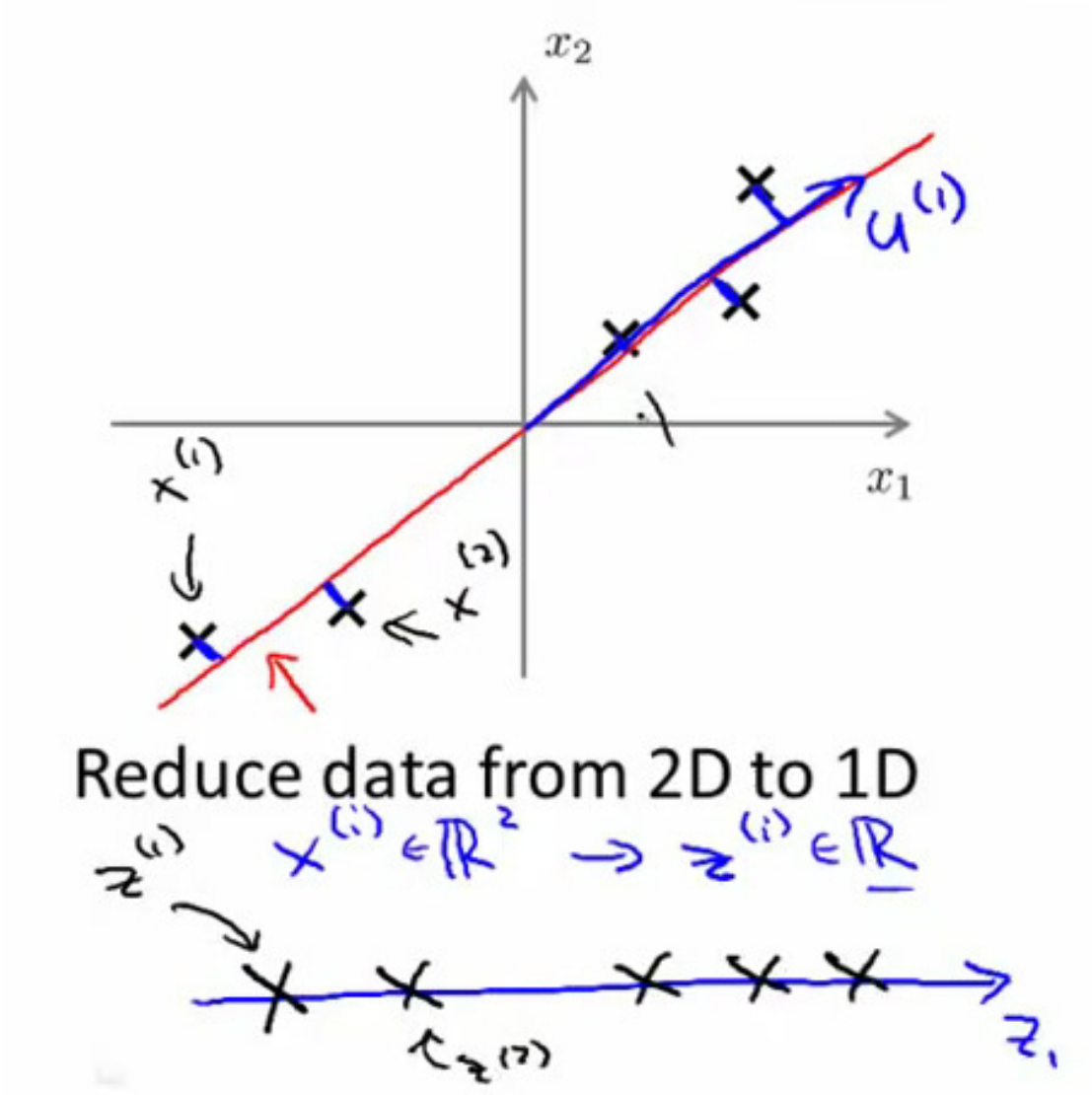


Figure 1: Example of PCA from 2D to 1D

**Algorithm 1** PCA algorithm

**function** PCA($x, k$) ▷ x = unlabeled dataset
    Reduces data from $n$-dimensions to $k$-dimensions.
    Compute covariance matrix:
        $\Sigma = \frac{1}{m} \sum_{(i=1)}^{n} (x^{(i)})(x^{(i)})^T$
    Compute "eigenvectors" of matrix $\Sigma$:
        $[U, S, V]$ = svd(Sigma); ▷ Singular Value Decomposition
        We only need $U$. $k$-first $U$ vectors are the ones that we will use to describe the data.
        $U_{\text{reduce}} \leftarrow$ First $k$ vectors from $U$ (in columns)
    $z \leftarrow U_{\text{reduce}}^T x$
    **return** $z$
**end function**

Then, we can select $k$ iteratively, starting from one and trying every single value until we retain 99% of the variance, at which point we can select the $k$ that satisfies that restriction.

Above solution is very slow though. We can take advantage that we are computing the "svd". We can use the $S$ matrix returned by it to calculate the retained variance in a much more easy way:

$$\text{For a given } k : \frac{\text{average squared projection error}}{\text{total variation in the data}} = 1 - \frac{\sum_{(i=1)}^{k} Sii}{\sum_{(i=1)}^{n} Sii}$$

Using above formula, we can simply calculate the svd once and iterate with the $S$ matrix until we find the correct $k$.

# 6 Deconstruction from Compressed Representation

We're going to see how to reconstruct the data after having applied PCA to it. It's a lossy method, so we're not going to get the same exact data, just an approximation of it.

$$X_{\text{approx}} = U_{\text{reduce}} z$$

# 7 Advice for Appylying PCA

We can use PCA in a supervised machine learning problem (using only $X$) to speed up the algorithm. Just remember to also use the PCA with the predict data!

The PCA mapping should be defined using only the training set, as we could bias it running it with the cv/test dataset as well.