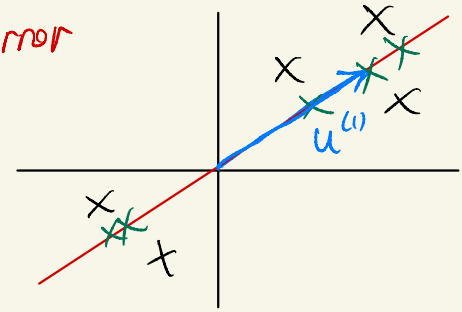
May 30, 2021

Problem Formulation:

e.g. Task: Reduce 2D data to 1D

PCA: Find a direction (i.e. a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the square projection error



Principal Component Analysis reduces data from n-dimensions to k-dimensions ($k \leq n$) by finding k vectors $u^{(1)}, u^{(2)} \dots u^{(k)}$ onto which to project the data, so as to minimize the projection error.

Data Preprocessing:

Mean Normalization:

$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}$, replace each $x_j^{(i)}$ w/ $x_j - \mu_j$

Feature Scaling:

Scale features to have a comparable range of values.

$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j}$

So how do we find $u^{(1)}, u^{(2)} \dots u^{(k)}$?

PCA Algorithm:

※ These are the Principal Components

To reduce from n-dimensions to k-dimensions,

1. Compute the covariance matrix

$$\Sigma = \frac{1}{m} \sum_{i=1}^{n} \underbrace{(x^{(i)})}_{n \times 1} \underbrace{(x^{(i)})^T}_{1 \times n}, \quad \Sigma \in {}^{n}\mathbb{R}^{n}$$

2. Compute the eigenvectors of matrix Σ

$$[U, S, V] = \text{svd}(\text{Sigma});$$

↳ singular value decomposition

$$U = \underbrace{\begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(m)} \\ | & | & & | \end{bmatrix}}_{k} \in {}^{n}\mathbb{R}^{n}$$

※ $m = n$ for PCA

$$x \in {}^{n}\mathbb{R}^{1} \rightarrow z \in {}^{k}\mathbb{R}^{1}$$

$$\hookrightarrow U_{reduce} = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(k)} \\ | & | & & | \end{bmatrix}^T \in {}^{n}\mathbb{R}^{k}$$

$$z = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(k)} \\ | & | & & | \end{bmatrix}^T x$$

$$= \begin{bmatrix} — & u^{(1)T} & — \\ — & u^{(2)T} & — \\ & \vdots & \\ — & u^{(k)T} & — \end{bmatrix} x$$

$k \times n$ $\quad$ $n \times 1$

$\therefore z \in {}^k\mathbb{R}^1$

Summary:

Mean normalization, feature scaling.

$$Sigma = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)})(x^{(i)})^T$$

```
[U, S, V] = svd(Sigma);
Ureduce = U(:, 1:k);
z = Ureduce' * x;
```

Reconstruction from Compressed Data:

$z^{(i)} = U_{reduce}^T x^{(i)} \rightarrow$ PCA

$x_{approx}^{(i)} = U_{reduce} z^{(i)} \rightarrow$ inverse operation

$n \times 1 \quad n \times k \quad k \times 1$

Choosing k (# of Principal Components):

Avg Squared Projection Error: $\frac{1}{m} \sum_{i=1}^{m} \| x^{(i)} - x_{approx}^{(i)} \|^2$

Total Variation in Data: $\frac{1}{m} \sum_{i=1}^{m} \| x^{(i)} \|^2$

Typically, we choose as small a "k" as possible s.t.

$$\frac{\frac{1}{m} \sum_{i=1}^{m} \| x^{(i)} - x_{approx}^{(i)} \|^2}{\frac{1}{m} \sum_{i=1}^{m} \| x^{(i)} \|^2} \leq 0.01 \quad (1\%)$$

i.e. 99% of variance is retained.

↳ The compressed data isn't too far off from the original.

Algorithm:

Try PCA w/ $k=1$, $k=2$ ...

1. Compute $U_{reduce}, z^{(1)} \dots z^{(m)}, x_{approx}^{(1)} \dots x_{approx}^{(m)}$

2. Check if $\dfrac{\frac{1}{m}\sum_{i=1}^{m} \|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m}\sum_{i=1}^{m} \|x^{(i)}\|^2} \leq 0.01$

↳ $[U, S, V] = svd(Sigma)$

$$S = \begin{bmatrix} S_{11} & & 0 \\ & S_{22} & \\ & & \ddots \\ 0 & & S_{nn} \end{bmatrix}$$

For a given k,

$$1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \leq 0.01 \ ?$$

↳ $$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \geq 0.99$$

Easy way to compute 2.

* Pick smallest k for which this is satisfied.

Supervised Learning Speedup:

e.g. $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}) \dots (x^{(m)}, y^{(m)})$

$x^{(i)} \in {}^{10000}\mathbb{R}^1$ for 100×100 image data

$x^{(1)}, x^{(2)} \dots x^{(m)} \in {}^{10000}\mathbb{R}^1$

$\downarrow$ PCA

$z^{(1)}, z^{(2)} \dots z^{(m)} \in {}^{1000}\mathbb{R}^1$

New Training Set:

$(z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}) \dots (z^{(m)}, y^{(m)})$

For new predictions, run them through the same mapping found by PCA to get z, then use

$$h_\theta(z) = \frac{1}{1+e^{-\theta^T z}}.$$

↳ i.e. For $x_{cv}^{(i)}$, $x_{test}^{(i)}$.

⊗ Only run PCA on training set to get mapping, then use mapping for validation and testing sets.

Main Applications of PCA!

1. Compression
   a. Reduce memory/disk needed to store data
   b. Speed up learning algorithm
2. Visualization
   ↳ $k = 2, 3$

⛤ Don't use PCA for overfitting; use regularization instead!