Principal Component Analysis

Dr. Fayyaz Minhas

Department of Computer Science
University of Warwick
https://warwick.ac.uk/fac/sci/dcs/teaching/material/cs909/

Accompanying Notebooks

- Please use: https://github.com/foxtrotmike/PCA-Tutorial

Question?

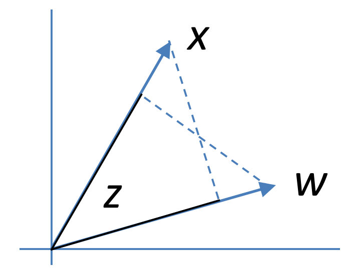
- Consider the vectors
  - $X_1=[1\ 2\ 1\ 4]^T$
  - $X_2=[2\ 4\ 2\ 4]^T$
  - $X_3=[0\ 0\ 0\ 4]^T$
  - $X_4=[3\ 6\ 3\ 4]^T$
  - $X_5=[4\ 8\ 4\ 4]^T$
- To store each vector, how many dimensions (or variables) do we need?

Motivation

- Having large number of related features is not informative
- Visualization of higher dimension data
- Can we reduce the number of features?
  – Dimensionality Reduction

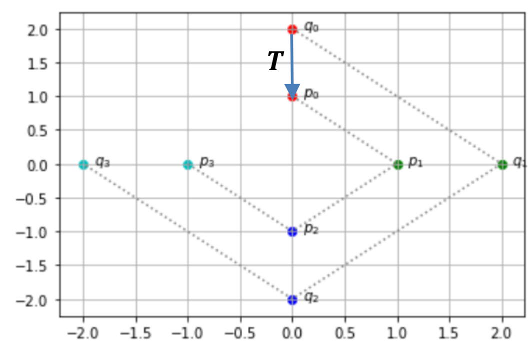
Basics

- Projections
  - Data can be projected onto a vector by taking its dot-product
    - The ith-component of a data point is the projection of the data onto the vector corresponding to the i-th axis
  - $z = w^T x$
    - Projection of x in the direction of w



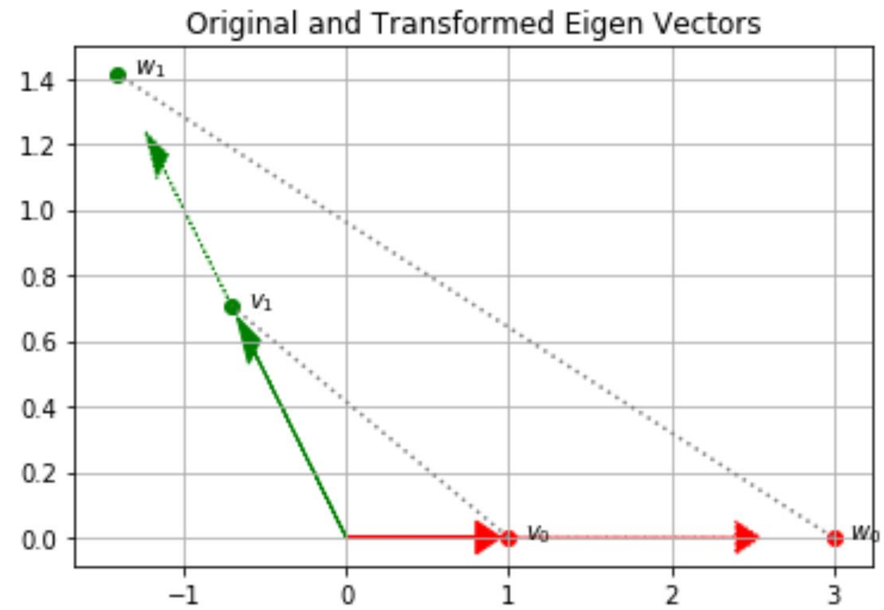
- Transformation
  - Multiplication of a vector with a matrix can be viewed as a geometric transformation of the vector

$$\boldsymbol{T} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \boldsymbol{x} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
$$\boldsymbol{y} = \boldsymbol{Tx} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$



- Eigen Values and Vectors
  - Those points that are characteristic to a given matrix that undergo only a change in scale are called Eigen vectors $\boldsymbol{w} = \boldsymbol{Tv} = \lambda \boldsymbol{v}$
  - How to find them: $(\boldsymbol{T} - \lambda \boldsymbol{I})\boldsymbol{v} = 0$ implies $|\boldsymbol{T} - \lambda \boldsymbol{I}| = 0$
  - See: https://github.com/foxtrotmike/PCA-Tutorial/blob/master/Eigen.ipynb

- $\boldsymbol{T} = \begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}$



- Eigen Vector: $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, Eigen Value: 3

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

  – Note scaling only

- Eigen Vector: $\begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix}$, Eigen Value: 2

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix} = \begin{bmatrix} -1.414 \\ 1.414 \end{bmatrix} = 2 \begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix}$$

Example: Find Eigenvalues and Eigenvectors of a 2x2 Matrix

If

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

then the characteristic equation is

$$|\mathbf{A} - \lambda \cdot \mathbf{I}| = \left| \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right| = 0$$

$$\left| \begin{bmatrix} -\lambda & 1 \\ -2 & -3-\lambda \end{bmatrix} \right| = \lambda^2 + 3\lambda + 2 = 0$$

and the two eigenvalues are

$$\lambda_1=-1,\ \lambda_2=-2$$

All that's left is to find the two eigenvectors. Let's find the eigenvector, $\mathbf{v}_1$, associated with the eigenvalue, $\lambda_1=-1$, first.

$$\mathbf{A} \cdot \mathbf{v}_1 = \lambda_1 \cdot \mathbf{v}_1$$

$$(\mathbf{A} - \lambda_1) \cdot \mathbf{v}_1 = 0$$

$$\begin{bmatrix} -\lambda_1 & 1 \\ -2 & -3-\lambda_1 \end{bmatrix} \cdot \mathbf{v}_1 = 0$$

$$\begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix} \cdot \mathbf{v}_1 = \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix} \cdot \begin{bmatrix} v_{1,1} \\ v_{1,2} \end{bmatrix} = 0$$

so clearly from the top row of the equations we get

$$v_{1,1} + v_{1,2} = 0, \quad \text{so}$$
$$v_{1,1} = -v_{1,2}$$

Note that if we took the second row we would get

$$-2 \cdot v_{1,1} + -2 \cdot v_{1,2} = 0, \quad \text{so again}$$
$$v_{1,1} = -v_{1,2}$$

In either case we find that the first eigenvector is any 2 element column vector in which the two elements have equal magnitude and opposite sign.

$$\mathbf{v}_1 = k_1 \begin{bmatrix} +1 \\ -1 \end{bmatrix}$$

where k_1 is an arbitrary constant. Note that we didn't have to use +1 and -1, we could have used any two quantities of equal magnitude and opposite sign.

Going through the same procedure for the second eigenvalue:

$$\mathbf{A} \cdot \mathbf{v}_2 = \lambda_2 \cdot \mathbf{v}_2$$

$$(\mathbf{A} - \lambda_2) \cdot \mathbf{v}_2 = \begin{bmatrix} -\lambda_2 & 1 \\ -2 & -3-\lambda_2 \end{bmatrix} \cdot \mathbf{v}_2 = \begin{bmatrix} 2 & 1 \\ -2 & -1 \end{bmatrix} \cdot \begin{bmatrix} v_{2,1} \\ v_{2,2} \end{bmatrix} = 0 \quad \text{so}$$

$$2 \cdot v_{2,1} + 1 \cdot v_{2,2} = 0 \quad (\text{or from bottom line: } -2 \cdot v_{2,1} - 1 \cdot v_{2,2} = 0)$$

$$2 \cdot v_{2,1} = -v_{2,2}$$

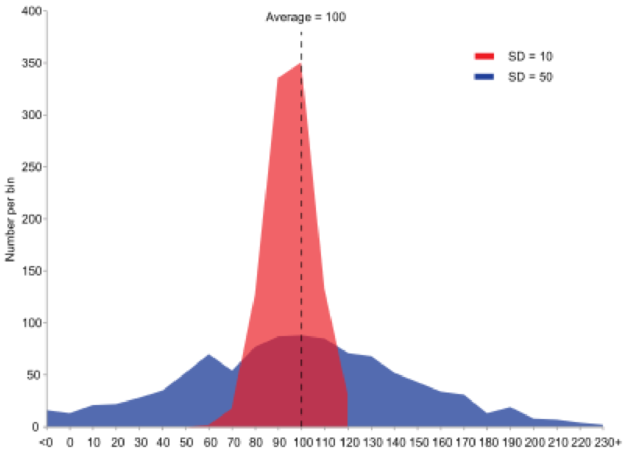
$$\mathbf{v}_2 = k_2 \begin{bmatrix} +1 \\ -2 \end{bmatrix}$$

Again, the choice of +1 and -2 for the eigenvector was arbitrary; only their ratio is important. This is demonstrated in the MatLab code below.

Basics

- Variance
  - Mean of the spread of a variable around its mean
  - $var(z) = \frac{1}{N}\sum_{i=1}^{N}(z_i - \mu_z)^2 = \frac{1}{N}(\boldsymbol{z} - \mu_z)^T(\boldsymbol{z} - \mu_z)$
    - $\boldsymbol{z}$ is an N-dimensional vector composed of the values of all data points in the sample
  - If mean is zero then $var(z) = \frac{1}{N}\boldsymbol{z}^T\boldsymbol{z} = \frac{1}{N}\|\boldsymbol{z}\|^2$
  - $var(z) = E[(z - \mu_z)^2]$

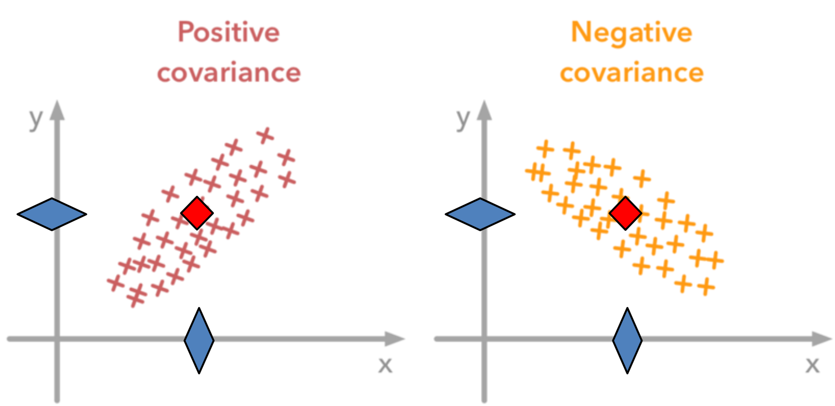
- Variance as an information measure
  - How is variance related to information content?



https://www.khanacademy.org/math/ap-statistics/summarizing-quantitative-data-ap/more-standard-deviation/v/review-and-intuition-why-we-divide-by-n-1-for-the-unbiased-sample-variance

Covariance

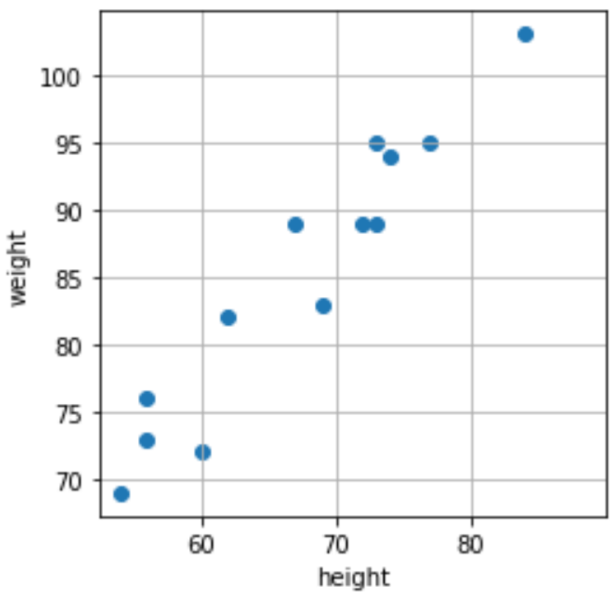
- Co-Variance
  - Given two random variables, to what extent are they linearly related to each other
  - $cov(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) = \frac{1}{N}(\boldsymbol{x} - \mu_x)^T(\boldsymbol{y} - \mu_y)$
    - Covariance is positive if, on average,
      - When one variable is above its mean then the other variable is above its mean too
      - When one variable is below its mean then the other variable is below its mean too
    - Covariance is negative if, on average,
      - When one variable is above the mean, the other is below its mean
  - Assume that the means are zero: $cov(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{N}\boldsymbol{x}^T\boldsymbol{y}$
    - Maximum when the vectors are co-linear or parallel
  - $cov(\boldsymbol{x}, \boldsymbol{y}) = E\left[(y - \mu_y)(x - \mu_x)\right]$
  - Thus, $var(z) = cov(z, z)$



Basics

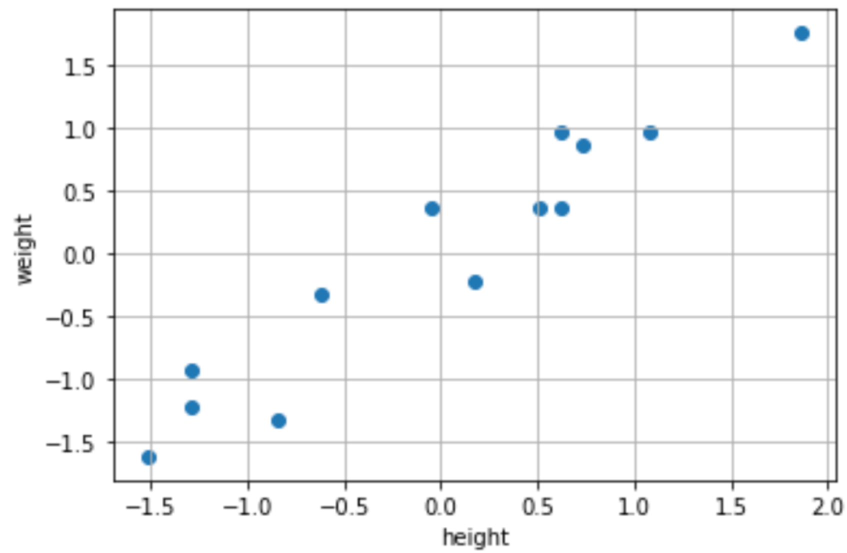
- Why are we interested in covariance
  - If two variables co-vary then they are redundant, or one can be linearly deduced from another
  - Covariance matrix: matrix of all pairwise covariances of all variables
    - $\boldsymbol{C} = \begin{bmatrix} cov(y,y) & cov(z,y) \\ cov(y,z) & cov(z,z) \end{bmatrix}$

Covariance Matrix Example



$$\frac{h-\mu_h}{\sigma_h}$$

$$\frac{w-\mu_w}{\sigma_w}$$




```
The mean is [67.46 85.31]
The standard deviation is: [ 8.86 10.06 ]
The variance is: [ 78.56 101.14]
```

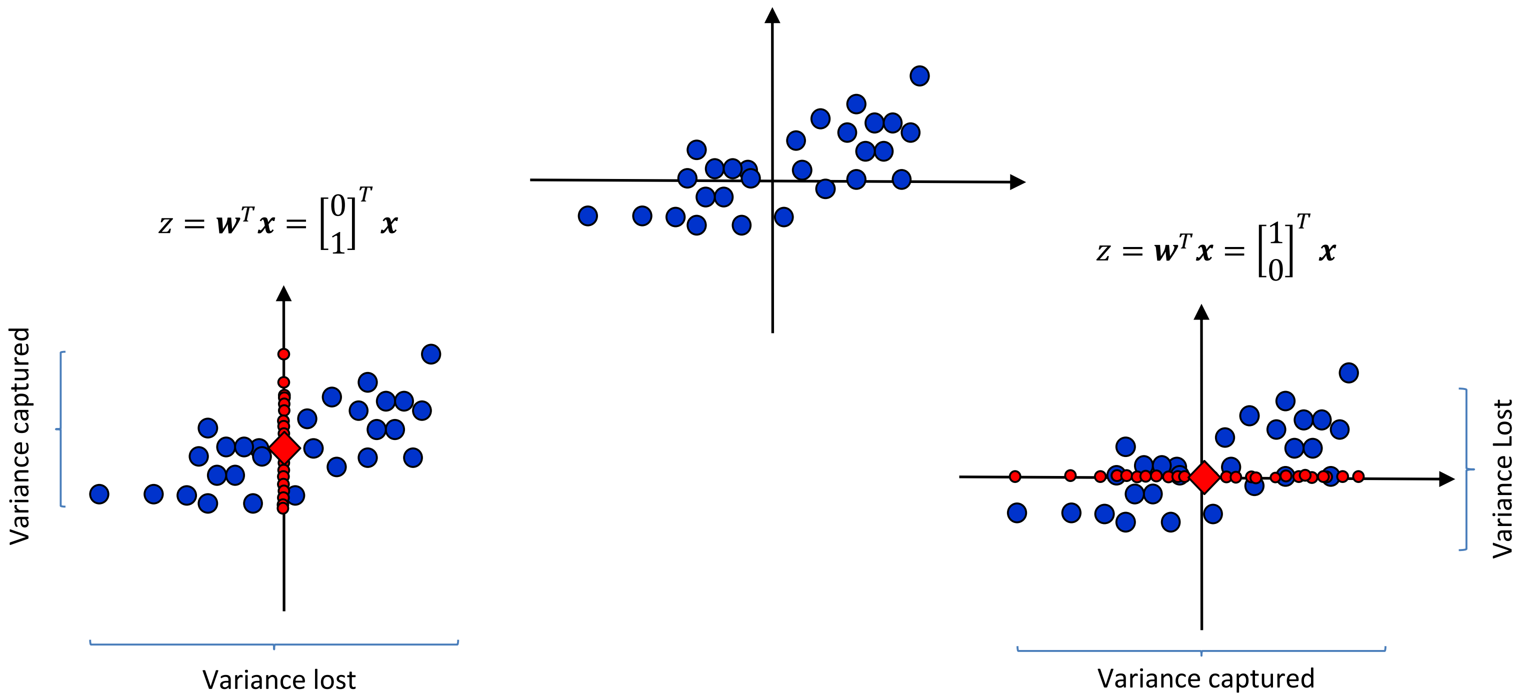
The co-variance matrix is: $\begin{bmatrix} 78.56 & 85.55 \\ 85.55 & 101.14 \end{bmatrix}$

```
The mean is [0 0]
The standard deviation is: [ 1 1 ]
The variance is: [ 1 1]
Total variance: 1+1 = 2.0
```

The co-variance matrix is: $\begin{bmatrix} 1 & 0.96 \\ 0.96 & 1 \end{bmatrix}$

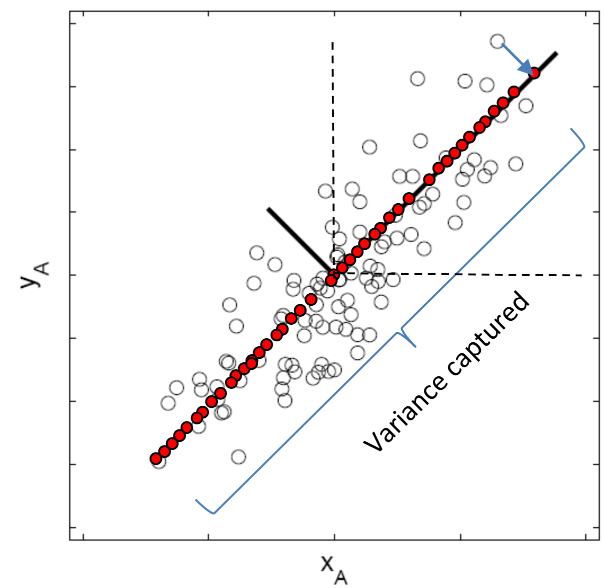
Data Dimensionality Reduction

- How can we reduce dimensions?
  - Drop features?
    - Equivalent to projecting data onto canonical axes
    - Loss in variance



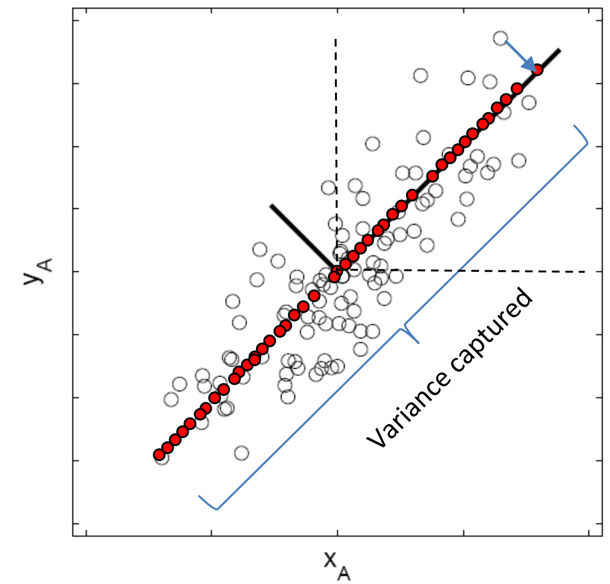
Dimensionality Reduction as Projections

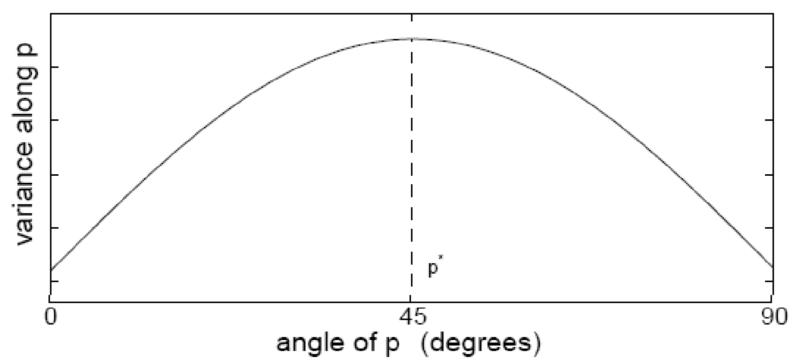
- Projections can be used for reducing dimensions
  - However, projecting data onto a vector loses information
  - We want to reduce the amount of information loss
  - Solution: Find and project along a direction along which information loss is minimum
    - A direction along which most of the variance is captured
  - How to do it?



How to do it: Naïve Implementation

- Set p = 0
- For p from 0 to π in steps
  - Calculate projection vector
    - $\boldsymbol{w_p} = \begin{bmatrix} \cos(p) \\ \sin(p) \end{bmatrix}$
  - Project your data onto $z_i = \boldsymbol{w}_p^T \boldsymbol{x}_i$
  - Find the variance of the projected data
- Plot the variance across p
- Find the p that gives maximum variance

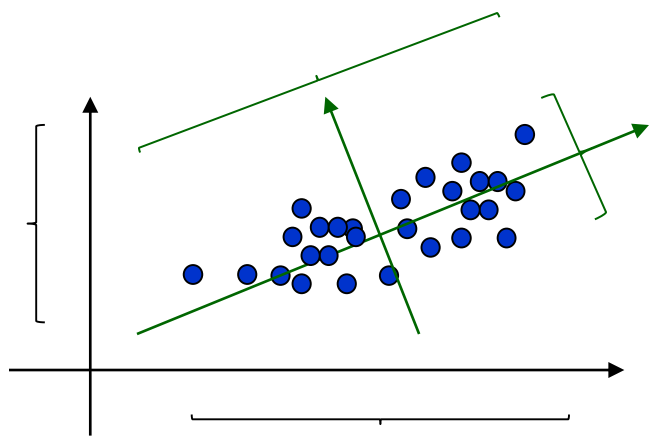




- Issues?

So what is PCA?

- A method for transforming the data
  - Projecting the data onto orthogonal vectors such that the variance of the projected data is maximum
  - Projection of x on the direction of w: $z = w^T x$
  - Find w such that $\mathrm{Var}(z)$ is maximized



Principal Component Analysis

- Relation between variance of projection and covariance matrix

$$\begin{aligned}\mathrm{Var}(z) = \mathrm{Var}(w^T x) &= \mathrm{E}[(w^T x - w^T \mu)^2] \\ &= \mathrm{E}[(w^T x - w^T \mu)(w^T x - w^T \mu)] \\ &= \mathrm{E}[w^T (x - \mu)(x - \mu)^T w] \\ &= w^{\mathrm{T}}\, \mathrm{E}[(x - \mu)(x - \mu)^T] w = w^T\, \mathrm{C}\, w\end{aligned}$$

where $\mathrm{Cov}(x) = \mathrm{E}[(x - \mu)(x - \mu)^T] = \mathrm{C}$

Maximizing variance

- Thus, we want to find w such that the variance is maximized

$$max_{\boldsymbol{w}}\, var(\boldsymbol{w}^T\boldsymbol{x}) \;=\; \boldsymbol{w}^T\boldsymbol{C}\boldsymbol{w}$$

Another view is that we want to project the data such that the loss in variance after projection is minimized. Let's say the total variance is V, then what we want is:

Loss in variance

$$min_{\boldsymbol{w}}\, V - var(\boldsymbol{w}^T\boldsymbol{x}) \;=\; V - \boldsymbol{w}^T\boldsymbol{C}\boldsymbol{w}$$

Total Variance

Variance along **w**

Using Structural Risk Minimization

- We have learned that machine learning models should reduce the structural risk
  - Regularization
  - Empirical Error Minimization

$$argmin_{\boldsymbol{w}} \lambda R(w) + L(X; w)$$

SVM

$$argmin_w \frac{\lambda}{2} w^T w + \sum_{i=1}^{N} \max(0, 1 - y_i x_i)$$

  - In case of dimensionality reduction, the empirical error is the loss in variance due to the projection

$$L(X; w) = V - var(\boldsymbol{w}^T \boldsymbol{x})$$

  - Since V is a constant
    - $L(X; w) = -var(\boldsymbol{w}^T \boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{C} \boldsymbol{w}$

PCA with SRM

- We can write the complete form as:

$$min_{\boldsymbol{w}} \ \alpha \boldsymbol{w}^T \boldsymbol{w} - \boldsymbol{w}^T \boldsymbol{C} \boldsymbol{w}$$

- Taking the derivative of $\alpha \boldsymbol{w}^T \boldsymbol{w} - \boldsymbol{w}^T \boldsymbol{C} \boldsymbol{w}$ with respect to $\boldsymbol{w}$ and substituting it to zero, we get:

$$\frac{\partial}{\partial \boldsymbol{w}} \ \alpha \boldsymbol{w}^T \boldsymbol{w} - \boldsymbol{w}^T \boldsymbol{C} \boldsymbol{w} = \alpha \boldsymbol{w} - \boldsymbol{C} \boldsymbol{w} = \boldsymbol{0}$$

$$\boldsymbol{C} \boldsymbol{w} = \alpha \boldsymbol{w}$$

Connection to Eigen Vectors

- Note that $\boldsymbol{C}\boldsymbol{w} = \alpha\boldsymbol{w}$
- Thus, our solution means that $\boldsymbol{w}$ is, in essence, an eigen vector of $\boldsymbol{C}$ with eigen value α
  - The eigen vectors of the covariance matrix (Called Principal Components) of the given dataset are along the direction of maximum variance of the data!!
  - Typically, the eigen vectors are normalized as unit vectors $\frac{\boldsymbol{w}}{||\boldsymbol{w}||}$

Let's see for our data

The co-variance matrix is: $\begin{bmatrix} 1 & 0.96 \\ 0.96 & 1 \end{bmatrix}$

Eigen vector 1:

$$w_1 = \begin{bmatrix} 0.7071 \\ 0.7071 \end{bmatrix}, \ \alpha_1 = 1.96$$

Variance of data after projecting along w_1: 1.96

Eigen vector 2:

$$w_2 = \begin{bmatrix} -0.7071 \\ 0.7071 \end{bmatrix}, \ \alpha_2 = 0.04$$

Variance of data after projecting along w_1: 0.04

Fraction of variance captured along each PC:

Using PC-1: 1.96/2 = 0.98

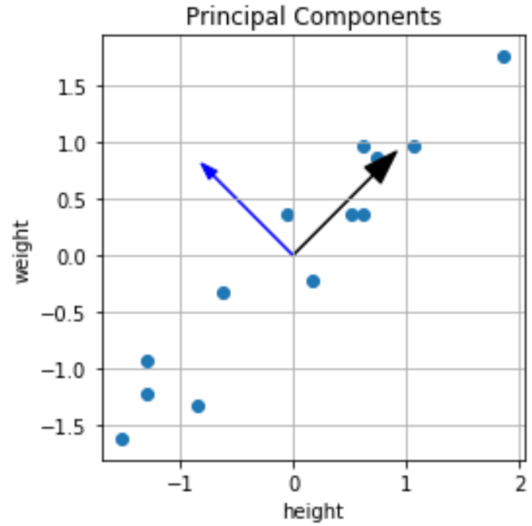
Using PC-1 and PC-2: (1.96+0.04)/2 = 1.0

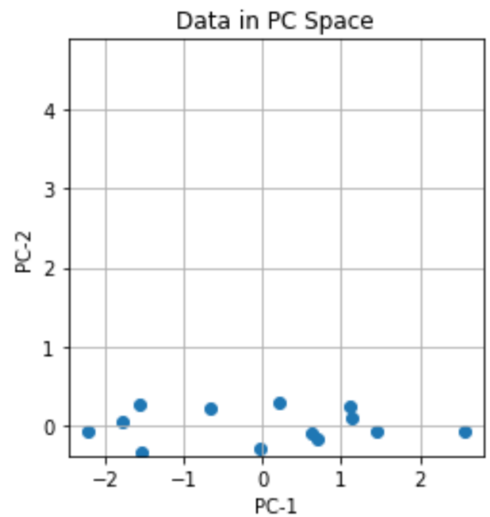
The two PC vectors are orthogonal to each other $w_1^T w_2 = 0$

The PC Matrix is $\mathrm{W} = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$

The inverse of W is: $\mathrm{W}^{-1} = \begin{bmatrix} 0.7071 & 0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} = W^T$

Thus, $\mathrm{W^T W = I}$

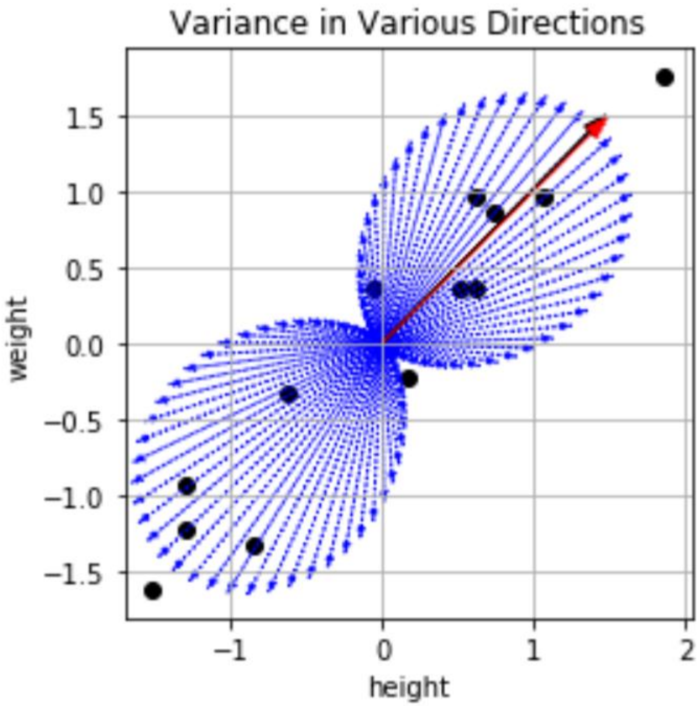


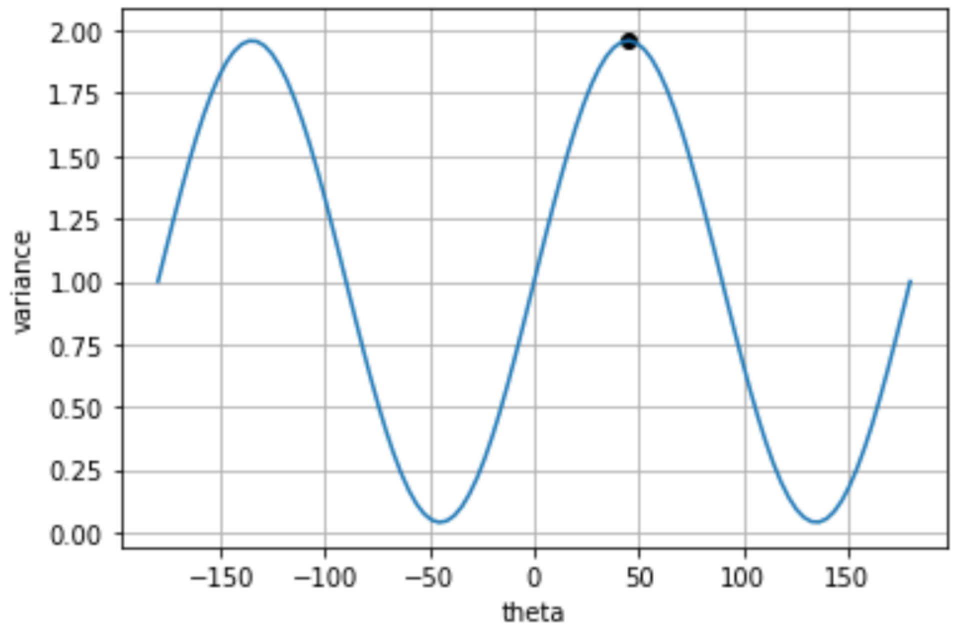


Things to note

- There are two principal components: The one with the largest variance (eigen value) is called the first principal component whereas the other one is called the second principal component.
- The variance along the first principal component is higher in comparison to the second.
- The variance along the first projected direction is higher than the variance along original features which is 1.0 after normalization. Thus, the principal component is a direction that captures more information than any of the original features alone.
- The norm of each of the principal components is 1.0.
- The two principal components are orthogonal to each other.
- The principle component matrix and its transpose are inverses of each other, i.e., $\mathbf{W}^{\mathbf{T}}\mathbf{W} = \mathbf{I}$
- The eigen values correspond to the amount of captured variance: The fraction of variance captured along a direction is exactly equal to the fraction of eigen values. Thus, the first principal component corresponds to the largest eigen value and so on.
- The plot of the fraction of captured variance up to k principal components (called the scree plot) can be used to select how many principal components to retain when reducing dimensionality. For the original data used in this example, upto 98% variance is along the first principal component. Therefore, if the second principal component is dropped, the loss of information will be only ~2%.

Using the naïve implementation






```
Direction of Maximum Variance: [0.70, 0.71]
```

PCA for dimensionality reduction

- An eigen value of the covariance matrix is equal to the variance captured by projecting data along the direction of the corresponding principal component
- Thus, principal components with small eigen values have small contribution to the total variance of the data and these can be discarded without major loss of information
- We can retain 90% variance of the data by storing the largest eigen values and eigen vectors which contribute 90% of the variance and projecting our data on these bases
- Proportion of Variance (PoV) explained

$$PoV(k) = \frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

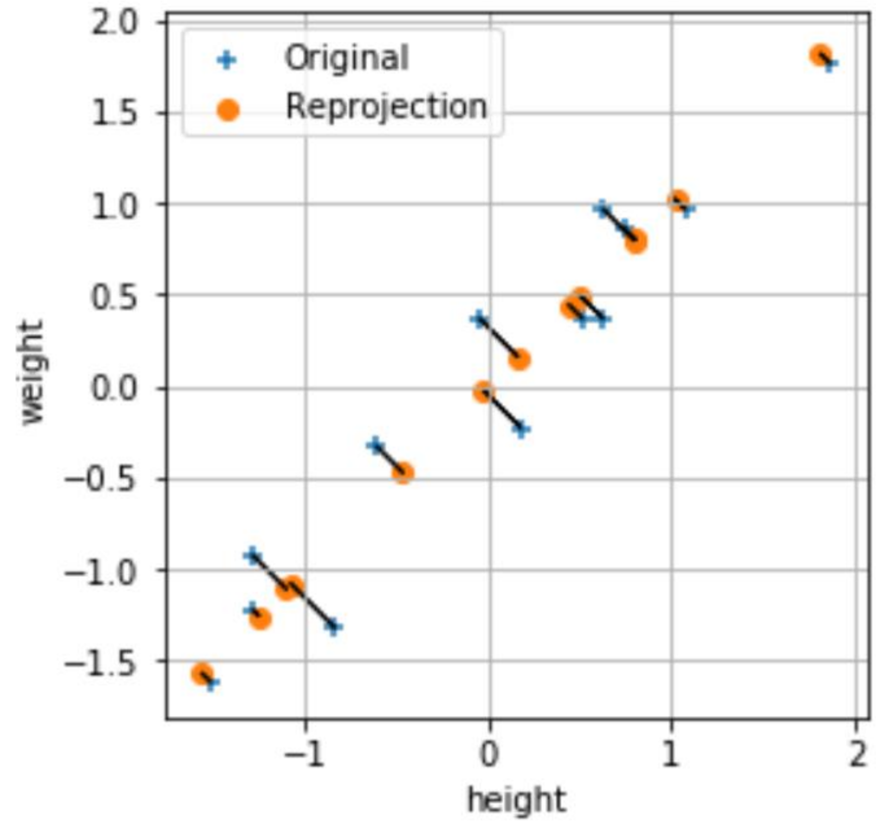
  when λ_i are sorted in descending order
- Typically, stop at PoV > 0.9
- Scree graph plots of PoV vs *k*, stop at "elbow"

PCA for dimensionality reduction

- Once we find k, we drop smaller eigen values
- Projections along a vector w is $z = w^T x$
- In matrix form, the projections along k Principal Components can be written as $\mathbf{Z}_{\mathbf{N}\times\mathbf{k}} = \mathbf{X}_{\mathbf{N}\times\mathbf{d}}\mathbf{W}_{\mathbf{d}\times\mathbf{k}}$

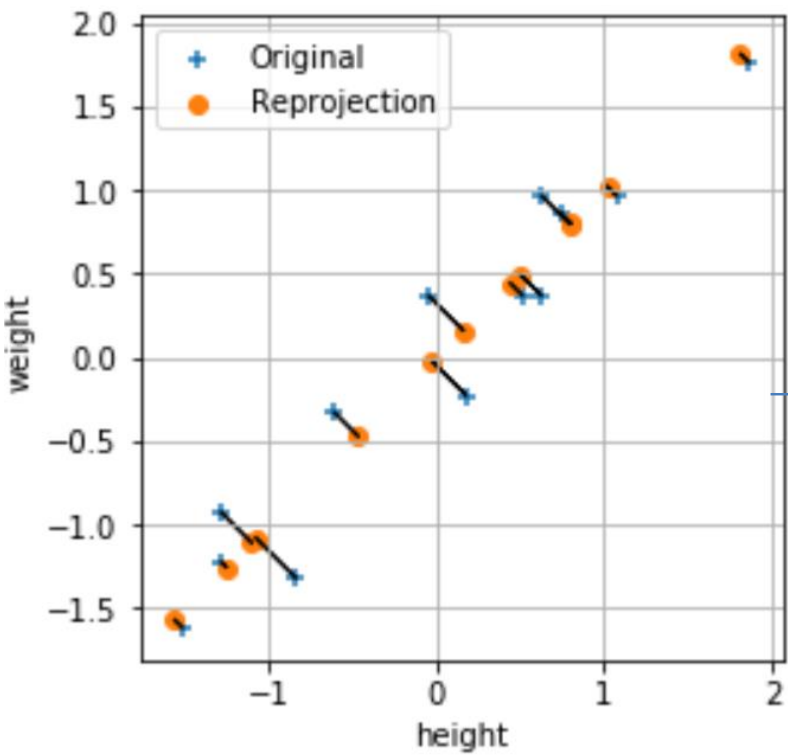
Reconstruction

- It is possible to reconstruct the data from lower dimensional representation
- Since, $\boldsymbol{Z} = \boldsymbol{XW}$
- Thus
- $\boldsymbol{X^r} = \boldsymbol{Z}\mathbf{W}^{-1} = \boldsymbol{Z}\mathbf{W}^{T}$



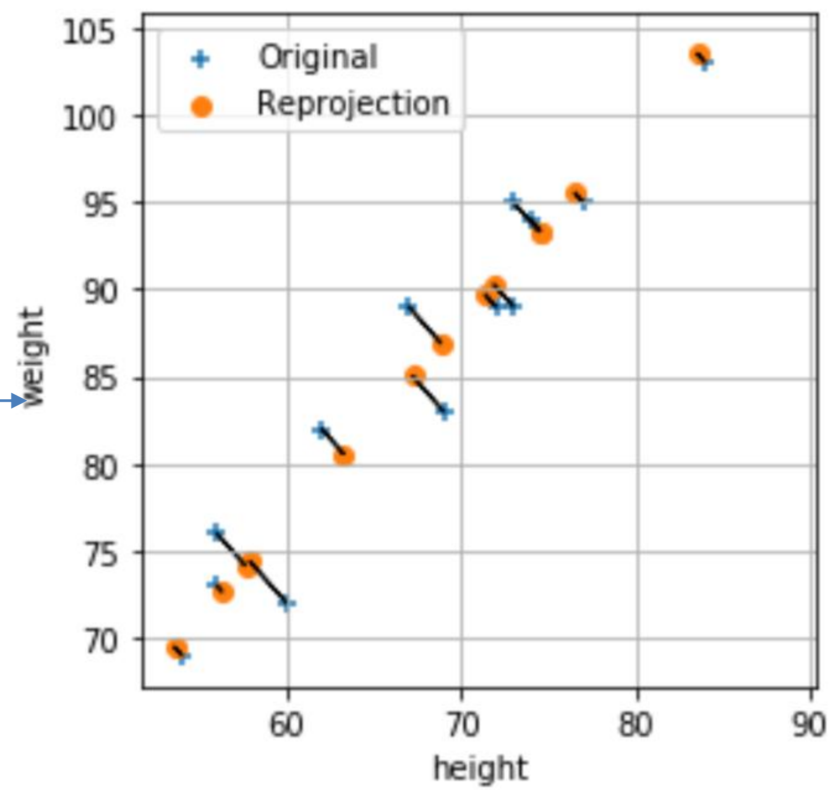
Unnormalization

- We will still need to unnormalize the data
  - Multiply the re-projected data by standard deviation and add the mean



$h'\sigma_h + \mu_h$

$w'\sigma_w + \mu_w$



Reconstruction Error

- The reconstruction error can be written as:
- $E(\boldsymbol{X};\boldsymbol{W}) = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x_i} - \boldsymbol{x_i^r}\|^2 = \frac{1}{N}\|\boldsymbol{X} - \boldsymbol{X^r}\|_F^2 = \frac{1}{N}\left\|\boldsymbol{X} - \boldsymbol{Z}\mathbf{W}^{\boldsymbol{T}}\right\|_F^2 = \frac{1}{N}\left\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}\mathbf{W}^{\boldsymbol{T}}\right\|_F^2$
- For our case:
- The average reconstruction error is: 0.402
- Which is exactly equal to the Eigen value of the dimension which was dropped

PCA Interpretation

- PCA
  - Finds directions of maximum variance and projects the data along those direction
  - Finds the directions that minimize reconstruction error

$$\min_{\mathbf{W}} \frac{1}{N} \left\| \mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^{\mathbf{T}} \right\|_{\mathrm{F}}^{2}$$

Such that

$$\mathbf{W}^{\mathbf{T}}\mathbf{W} = \mathbf{I}$$

An algorithmic view of how PCA Works

- **Input**: $\boldsymbol{X}_{N \times d}$
- **Output**: A transformation matrix **W** which can be used for dimensionality reduction
- **Parameters**: Selection of principal components
  - Proportion of variance
  - Number of principal components (k)
  - Which principal components to retain

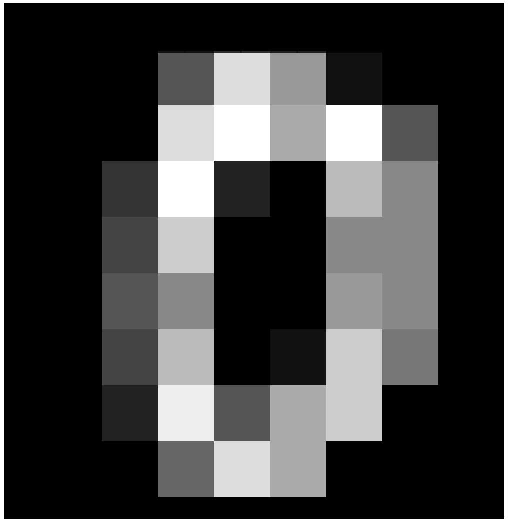
- **Internal Working**
  - Normalize data
    - Calculate feature wise mean and standard deviation and normalize data to zero mean and unit standard deviation
  - Find Covariance Matrix
  - Find Principal Components (Eigen Value Problem)
  - Select Principal Components
    - Using Scree Graph
    - Intuition
  - Reduce dimensionality by Projection along selected components

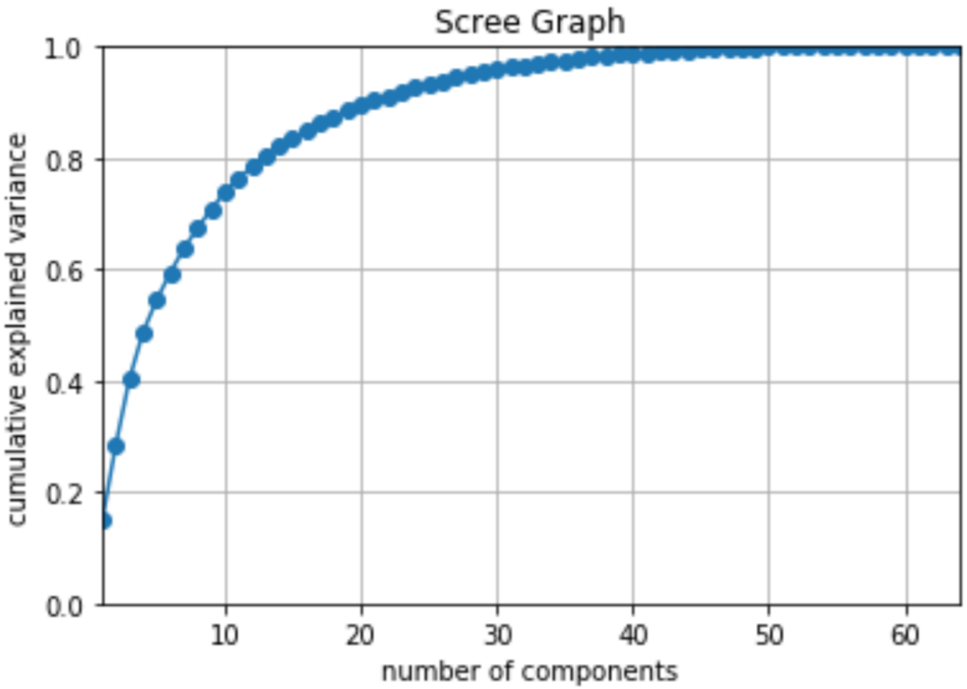
How to code?

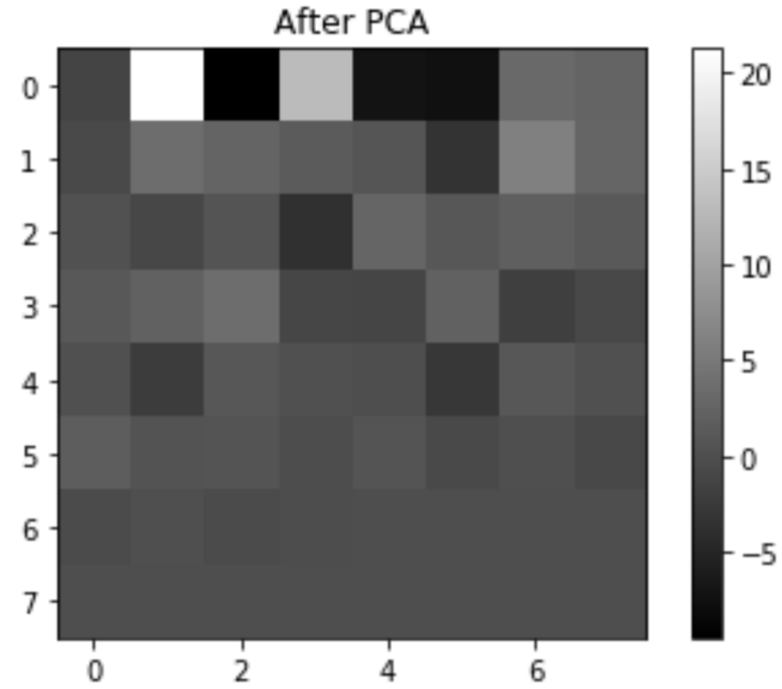
- **Fitting PCA to training data**
- `from sklearn.decomposition import PCA`
- `pca = PCA(n_components=4)`
- `pca.fit(X)`
- **Projection**
- `Z = pca.transform(X)`
- **Visualization**
- **Screen Graph**
- `plt.plot(np.cumsum(pca.explained_variance_ratio_),'o-’)`
- **Reconstruction**
- `Xr = pca.inverse_transform(Z)`

Example

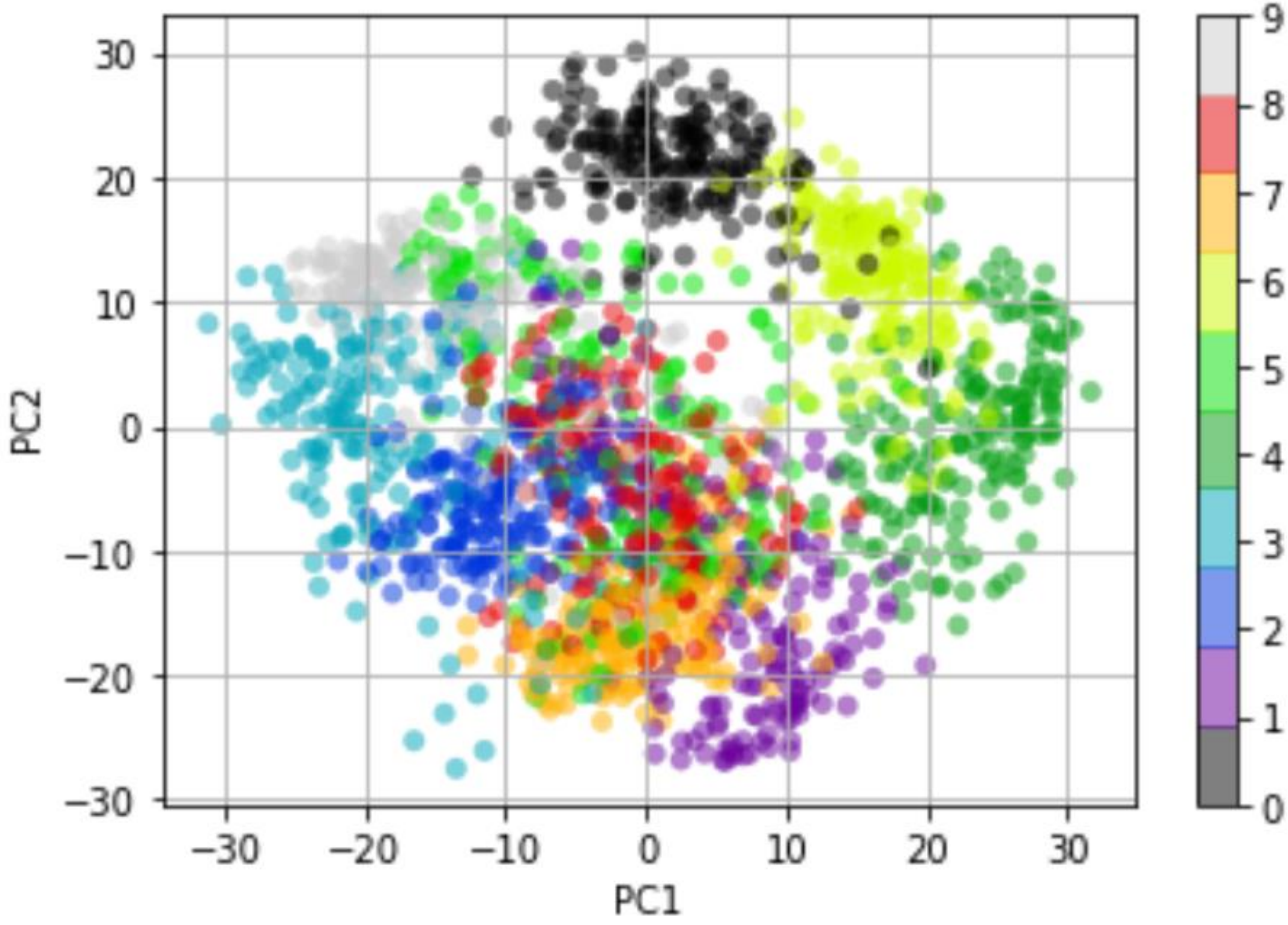
- MNIST visualization
- X: 1797x64







Visualization



Algorithm for PCA: Classical Method

- Each of the N samples is stored as a d-dimensional vector $x^i = \begin{bmatrix} x_1^i & \cdots & x_d^i \end{bmatrix}^T$
- The data matrix is formed as $X = \begin{bmatrix} x^1 & \cdots & x^N \end{bmatrix}$
- Compute the mean m=[m_1 … m_d]^T from *X* using $m_i = \frac{1}{N}\sum_{j=1}^{P} x_i^j$
- Centralize each sample in the data as $\bar{x}^i = x^i - m$ $\bar{X} = \begin{bmatrix} \bar{x}^1 & \cdots & \bar{x}^N \end{bmatrix}$
- Compute Covariance Matrix $S = \frac{\bar{X}\bar{X}^T}{N-1}$
- Find the Eigen Values $\lambda_1, \lambda_2 \ldots \lambda_d$ & *d*-dimensional Eigen Vectors $w_1, w_2 \ldots w_d$, of *S* using $S\lambda = w\lambda$ and sort the eigen values in decreasing magnitudes. Normalize the eigen vectors.
- Calculate the required dimension *k* based on proportion of variance based approach explained earlier for a given threshold
- Form $W=[\, w_1\, w_2 \ldots w_k\,]_{(d \times k)}$
- A vector *x* can be projected using $z = W^T(x - m)$

Algorithm for PCA: Snapshot Method

- If the input dimension (d) is large then the size of the covariance matrix is also large making its calculations computationally demanding
- It is known that for a d x N matrix the maximum number of non-zero eigenvectors is min(d-1,N-1)
- If N < d, then we can compute the eigen vectors w_i' of

$$S'_{(N\times N)} = \frac{\bar{X}^T X}{N-1}$$

instead of S. The eigen values for both S and S' are same and the eigen vectors of S can be obtained from those of S' using

$$w_{i_{(d\times 1)}} = \bar{X}_{(d\times N)} w'_{i_{(N\times 1)}}$$

Algorithm for PCA: Snapshot Method

- Each of the N samples is stored as a d-dimensional vector $x^i = \begin{bmatrix} x_1^i & \cdots & x_d^i \end{bmatrix}^T$
- The data matrix is formed as $X = \begin{bmatrix} x^1 & \cdots & x^N \end{bmatrix}$
- Compute the mean m=[m_1 … m_d]^T from *X* using $m_i = \frac{1}{N}\sum_{j=1}^{P} x_i^j$
- Centralize each sample in the data as $\bar{x}^i = x^i - m$ $\bar{X} = \begin{bmatrix} \bar{x}^1 & \cdots & \bar{x}^N \end{bmatrix}$
- Compute Covariance Matrix $S^{'} = \frac{\bar{X}^T \bar{X}}{N-1}$
- Find the Eigen Values (sorted in decreasing values) $\lambda'_1, \lambda'_2 \ldots \lambda'_d$ & *d*-dimensional Eigen Vectors $w'_1, w'_2 \ldots w'_d$, of *S* using $S\lambda'=w'\lambda'$.
- Calculate the required dimension *k* based on proportion of variance based approach explained earlier for a given threshold
- Form $W=[\, w_1\, w_2 \ldots w_k\,]_{(d \times k)}$ $w_i{*} = \bar{X}w_i^{'},$ $w_i = \frac{w_i{*}}{|w_i{*}|}$
- A vector *x* can be projected using $z = W^T(x - m)$

Principal Component Analysis: Another view

- Maximize Var(z_1) subject to $\|\mathbf{w}_1\| = 1$

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \boldsymbol{C} \mathbf{w}_1 - \alpha(\mathbf{w}_1^T \mathbf{w}_1 - 1)$$

$Cw_1 = \alpha w_1$

Differentiating w.r.t $\mathbf{w}_1$

  - w_1 is an eigenvector of $\sum$
  - Choose the one with the largest eigenvalue for Var(z_1) to be max
- Second principal component: Max Var(z_2), s.t., $\|\mathbf{w}_2\|$ =1 and orthogonal to $\mathbf{w}_1$

$$\max_{\mathbf{w}_2} \mathbf{w}_2^T \boldsymbol{C} \mathbf{w}_2 - \alpha(\mathbf{w}_2^T \mathbf{w}_2 - 1) - \beta(\mathbf{w}_2^T \mathbf{w}_1 - 0)$$

$Cw_2 = \alpha\, w_2$

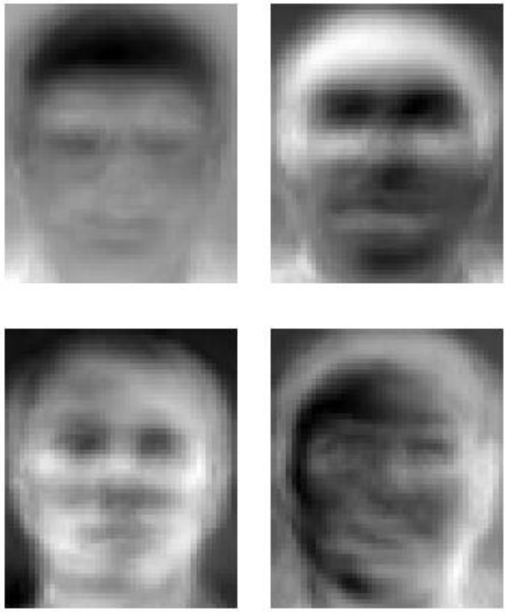
  - w_2 is another eigenvector of $\sum$

$$\Rightarrow \quad 2\Sigma \mathbf{w}_2 - 2\alpha \mathbf{w}_2 - \beta \mathbf{w}_1 = 0$$
$$\Rightarrow \quad 2\mathbf{w}_1^T \Sigma \mathbf{w}_2 - 2\alpha \mathbf{w}_1^T \mathbf{w}_2 - \beta \mathbf{w}_1^T \mathbf{w}_1 = 0$$
$$\Rightarrow \quad 2\mathbf{w}_2^T \Sigma \mathbf{w}_1 - 2\alpha(0) - \beta(1) = 0$$
$$\Rightarrow \quad 2\lambda_1 \mathbf{w}_2^T \mathbf{w}_1 - \beta = 0 \qquad \Rightarrow \beta = 0$$

- And so on. The Eigen values are sorted in decreasing order and the eigen vectors with positive eigen values are kept

Practical Use

- Face Recognition
  - Eigen Faces
    - Turk and Pentland
      - https://en.wikipedia.org/wiki/Eigenface



How to use it in your work

- Dimensionality reduction
- Visualization
- Selecting which dimensions are more informative for classification

Issues

- Slow
- Iterative Implementations
- Robustness
- Does not consider label information: Unsupervised technique
- Linear Projections only
- Other techniques
  - tSNE
  - UMAP