Lecture 9

Dimensionality reduction & PCA

We'll start with a simple and fundamental unsupervised learning problem: dimensionality reduction.

Goal: reduce the dimensionality of a dataset so that

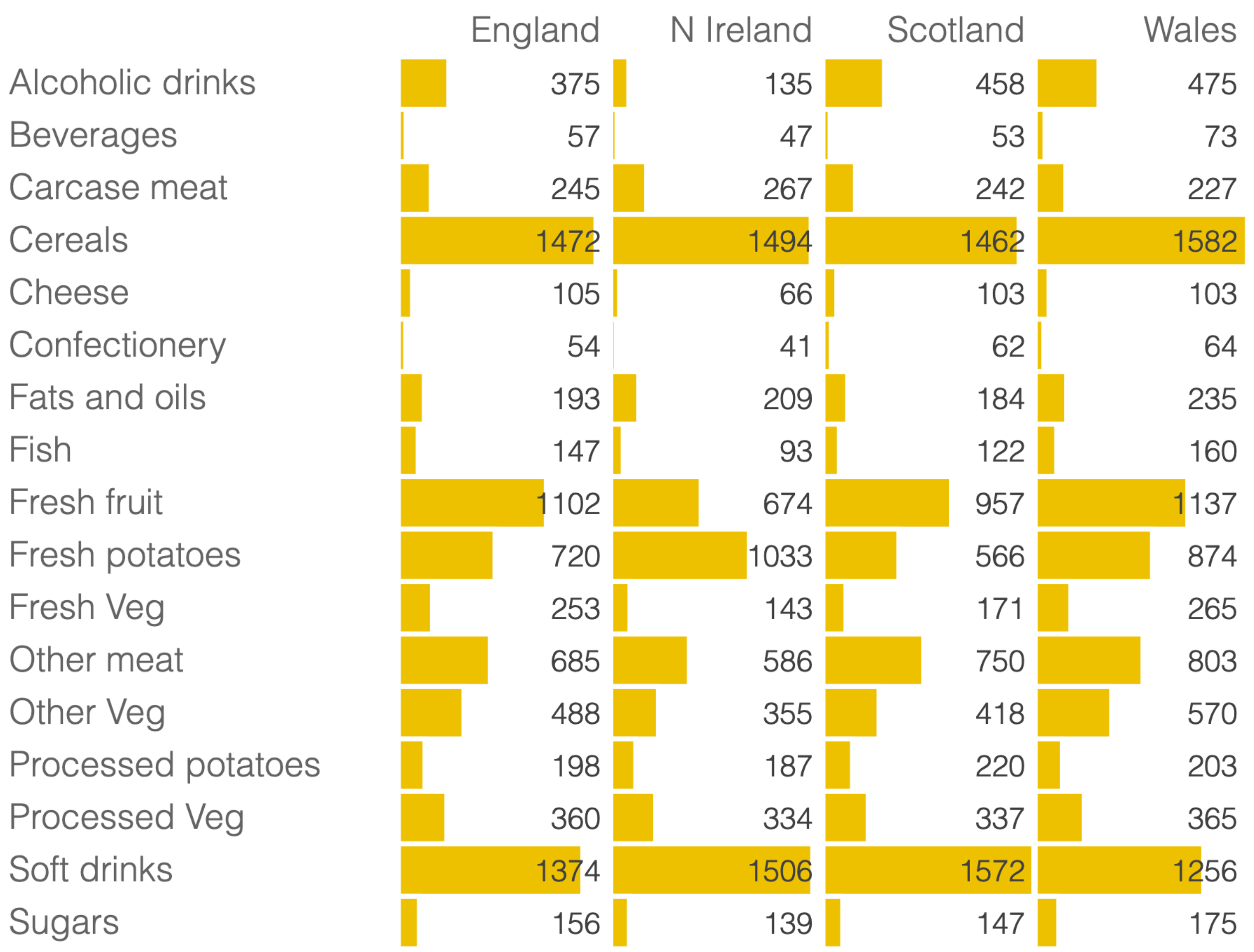
- it is easier to visualize and discover patterns
- it takes less time and space to process for any downstream application (classification, regression, etc)
- noise is reduced

There are many approaches, we focus on a linear method: Principal Component Analysis (PCA).

Consider the following dataset:

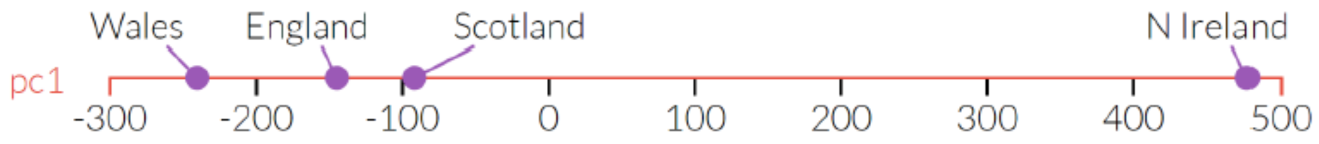
17 features, each represents the average consumption of some food; 4 data points, each represents some country.

What can you tell? Hard to say anything looking at all these 17 features.



	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

PCA can help us! The projection of the data onto its first principal component:



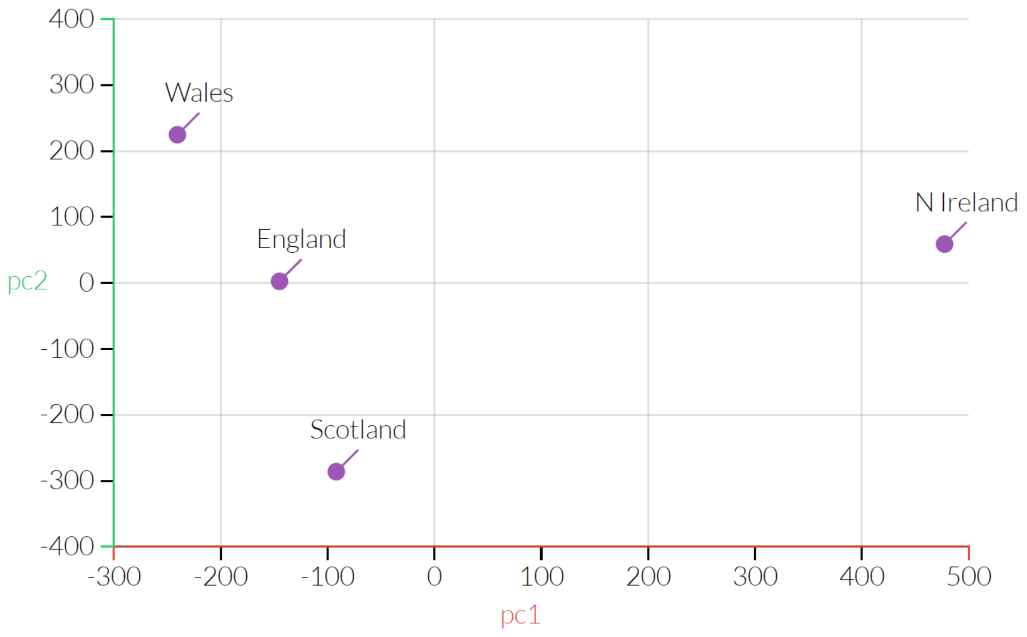
i.e. we reduce the dimensionality from 17 to just 1 .

Now one data point is clearly different from the rest!

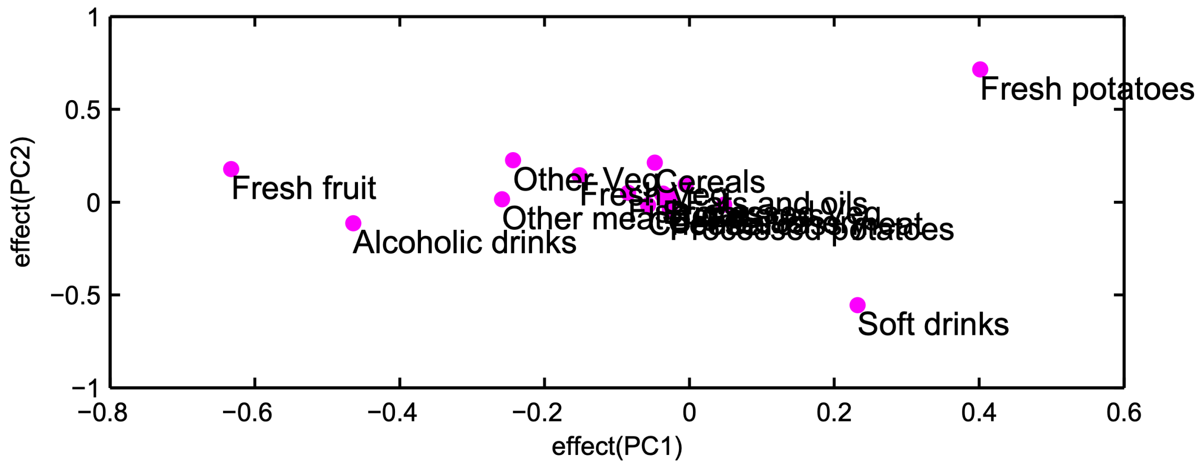
That turns out to be data from Northern Ireland, the only country not on the island of Great Britain out of the 4 samples.

Can also interpret components: $PC1$ tells us that the Northern Irish eat more grams of fresh potatoes and fewer of fresh fruits and alcoholic drinks.

We can find the second (and more) principal components of the data too:



And the components themselves are interpretable too:



High-level goal

Suppose we have a dataset of n datapoints $x_1, \cdots, x_n \in \mathbb{R}^d$. (In previous e.g. , $n = 4, d = 17$)

The high level goal of PCA is to find a set of k principal components (PCs) /principal vectors $v_1, \cdots, v_k \in \mathbb{R}^d$ such that for each x_i

$$x_i \approx \sum_{j=1}^{k} \alpha_{ij} v_j$$

for some coefficients $\alpha_{ij} \in \mathbb{R}$.

x_i is like the food consumption for some countries, v_j is the principal food consumption vectors.

PCA explains the data as different linear combination of some pricinpal components.

- Before we apply PCA, we usually preprocess the data to center it:

Let $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, then set $\tilde{x}_i = x_i - \bar{x}$.

Assume data is centered ($\sum x_i = 0$)

- In many applications, it is also important to scale each coordinate properly. This is especially true if the coordinates are in different units or scales.

  For all $j \in [d]$, divide j-th coordinate of each point by $\sqrt{\sum_{i=1}^{n} x_{ij}^2}$

Objective function for PCA

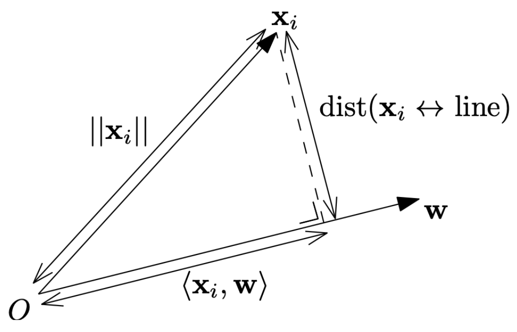
Key difference from supervised learning problems: No labels given, which means no ground-truth to measure the quality of the answer!

However, we can still write an optimization problem based on our high-level goal.

For clarity, we first discuss the special case of $k = 1$.

Optimization problem for finding the 1^{st} principal component v_1 :

$$v_1 = \arg\min_{v:||v||_2=1} \sum_{i=1}^{n} ((distance\ between\ x_i\ and\ line\ spanned\ by\ v)^2)$$



$\mathbf{x}_i$
$||\mathbf{x}_i||$
dist($\mathbf{x}_i \leftrightarrow$ line)
$\mathbf{w}$
$\langle \mathbf{x}_i, \mathbf{w} \rangle$
O

Figure 4: The geometry of the inner product with a unit length vector, $\mathbf{w}$.

$$(dist(x_i \leftrightarrow line\ spanned\ by\ v))^2 + < x_i, v >^2 = ||x_i||_2^2$$

$||x_i||_2^2$ is a constant, independent of choice of v .

$\therefore$ original objective is equivalent to:

$$v_1 = \arg\max_{v:||v||_2=1} \sum_{i=1}^{n} < x_i, v >^2$$

An example:

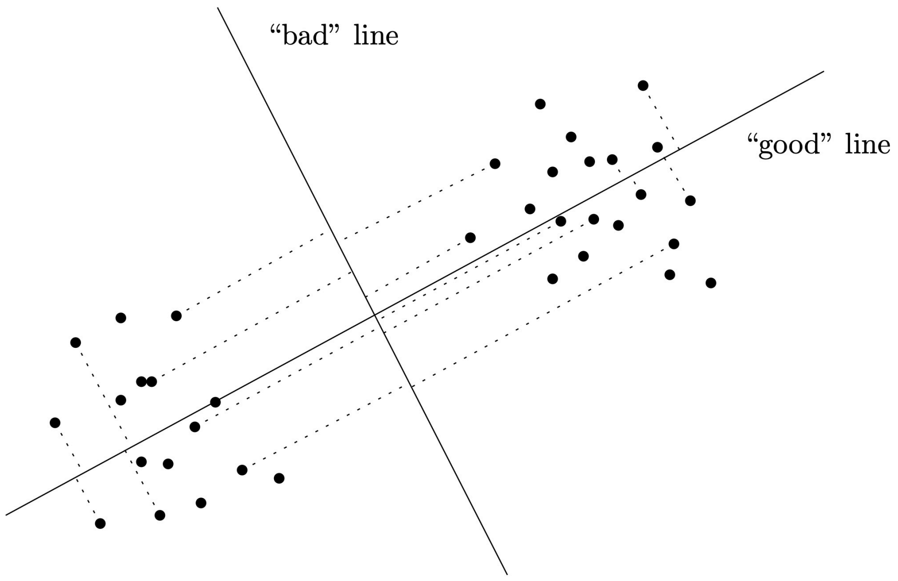


Figure 5: For the good line, the projection of the points onto the line keeps the two clusters separated, while the projection onto the bad line merges the two clusters.

Objective function for larger values of k

The generalization of the original formulation for general k is to find a k-dimensional subspace S such that the points are as close to it as possible:

$$S = \arg \min_{k-dim\ subspaces\ S} \sum_{i=1}^{n} (distance\ between\ x_i\ and\ subspace\ S)^2$$

By the same reasoning as for $k = 1$, this is equivalent to,

$$S = \arg \max_{k-dim\ subspaces\ S} \sum_{i=1}^{n} (length\ of\ x_i's\ projection\ on\ S)^2 \cdots\cdots (1)$$

It is useful to think of the subspace S as the span of k orthonormal vectors $v_1, \cdots, v_k \in \mathbb{R}^d$.

Recall vectors $v_1 \cdots, v_k$ are orthonormal to:

1. $||v_i|| = 1 \ \forall i \in [k]$
2. $< v_i, v_j >= 0 \ \forall i \neq j$

   e.g. : standard basis vectors $e_1 = (1, 0, \cdots, 0), e_2 = (0, 1, \cdots, 0)$

Def: Span of a collection $v_1, \cdots, v_k \in \mathbb{R}^d$ is all the linear combinations $\{\sum_{j=1}^{R} \lambda_j v_j : \lambda_1, \cdots, \lambda_R \in \mathbb{R}\}$.

Example,

- $k = 1$, span is line through the origin.
- $k = 2$, if v_1, v_2 are linearly independent, the span is a plane through the origin, and so on.

Fact about orthonormal vectors:

$$length\ of\ x_i's\ projection\ on\ (span(v_1, \cdots . v_e))^2 = \sum_{j=1}^{k} < x_i, v_j >^2 \cdots\cdots (2)$$

Combining $(1), (2)$, we get Formal problem solved by PCA:

Given $x_1, \cdots, x_n \in \mathbb{R}^d$ and a parameter $k \geqslant 1$, compute orthonormal vectors $v_1, \cdots, v_k \in \mathbb{R}^d$ to maximize:

$$\sum_{i=1}^{n} \sum_{j=1}^{k} < x_i, v_j >^2$$

Equivalent view:

- Pick v_1 to be the variance maximizing direction.
- Pick v_2 to be the variance maximizing direction, orthogonal to v_1 .
- Pick v_3 to be the variance maximizing direction, orthogonal to v_1 and v_2 , and so on.

Using PCA for data compression and visualization

Input: n datapoints $x_1, x_2, \cdots, x_n \in \mathbb{R}^d$, components k we want.

Step1: Perform PCA to get top k principal components $v_1, \cdots, v_k \in \mathbb{R}^d$.

Step2: For each datapoint x_i , define its "v_1-coordinate” as "$< x_i, v_1 >$", its "v_2-coordinate" as "$< x_i, v_2 >$". Therefore we associate k coordinates to each datapoint x_i , where the j-th coordinate denotes the extent to which x_i points in the direction of v_j .

Step3: We now have a new "compressed" dataset where each datapoint is k-dimensional. For visualization, we can plot the point x_i in $\mathbb{R}^k$ as the point $(< x_i, v_1 >, < x_i, v_2 >, \cdots, < x_i, v_k >)$.

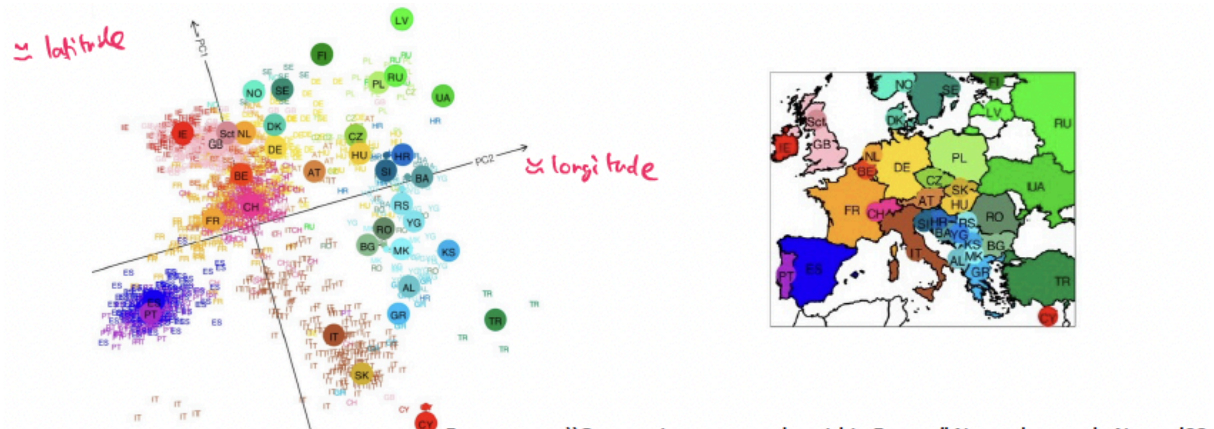
Going back to high-level goal:

$$x_i \approx \sum_{j=1}^{k} < x_i, v_j > v_j$$

Visualization example: Dataset: genomes of 1387 Europeans (each individual’s genotype at 200,000 locations in the genome) , $n = 1387, d \approx 200000$

Project the datapoints onto top 2 PCs.

Plot shown below; looks remarkably like the map of Europe!



From paper: ``Genes mirror geography within Europe'' Novembre et al., Nature'08

Compression example: Dataset: 256×256 (≈ 65 K pixels) dimensional images of about 2500 faces, all framed similarly, $n = 2500, d \approx 65000$.

We can represent each image with high accuracy using only $100 - 150$ principal components!

The principal components (called eigen-faces here) are themselves interpretable too!

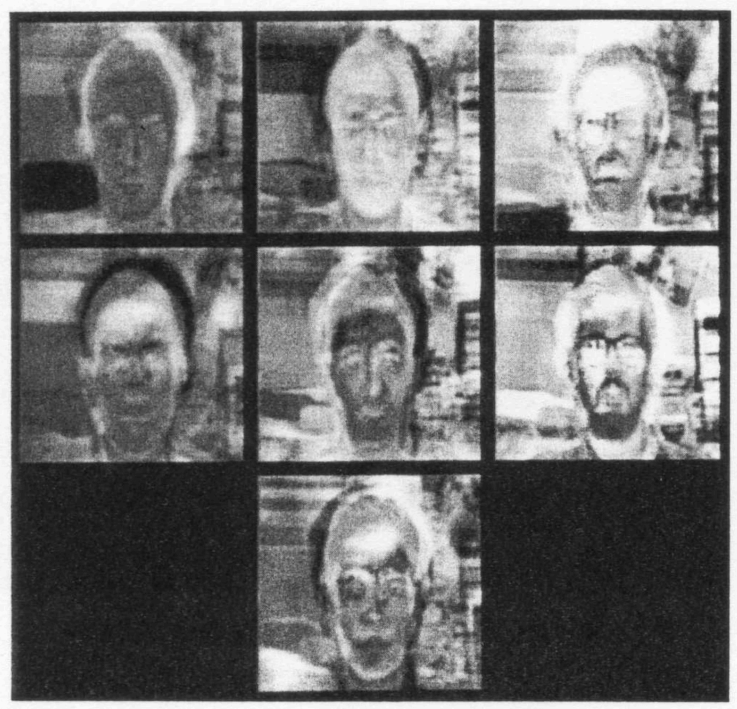


Figure 2. Seven of the eigenfaces calculated from the input images of Figure 1.

Solving the PCA optimization problem

Consider $k = 1$

$$v_1 = \arg \max_{v:||v||_2=1} \sum_{i=1}^{n} < x_i, v >^2$$

$$x = \begin{bmatrix} --x_1^T-- \\ \vdots \\ --x_n^T-- \end{bmatrix} \in \mathbb{R}^{n\times d}$$

$\therefore$ for any $v \in \mathbb{R}^d$,

$$xv = \begin{bmatrix} --x_1^Tv-- \\ \vdots \\ --x_n^Tv-- \end{bmatrix} \in \mathbb{R}^{n}$$

$$\begin{aligned} \therefore \sum_{i=1}^{n} < x_i, v >^2 &= ||xv||_2 \\ &= (xv)^T(xv) \\ &= v^Tx^Txv \end{aligned}$$

$\therefore$ for $A = x^Tx \in \mathbb{R}^{d\times d}$

$$v_1 = \arg \max_{v:||v||_2=1} v^TAv$$

x^Tx : covariance matrix of data (assuming data is centered)

$$A = \begin{bmatrix} | & & | \\ | & & | \\ x_1 & \cdots & x_n \\ | & & | \\ | & & | \end{bmatrix} \begin{bmatrix} --x_1^T-- \\ \vdots \\ --x_n^T-- \end{bmatrix}$$

$A_{11} = \sum_{i=1}^n x_{i,1}^2 \rightarrow$ variance of 1st cordante.

$A_{12} = \sum_{i=1}^n x_{i,1}x_{i,2} \rightarrow$ covariance between 1st and 2nd coordinate.

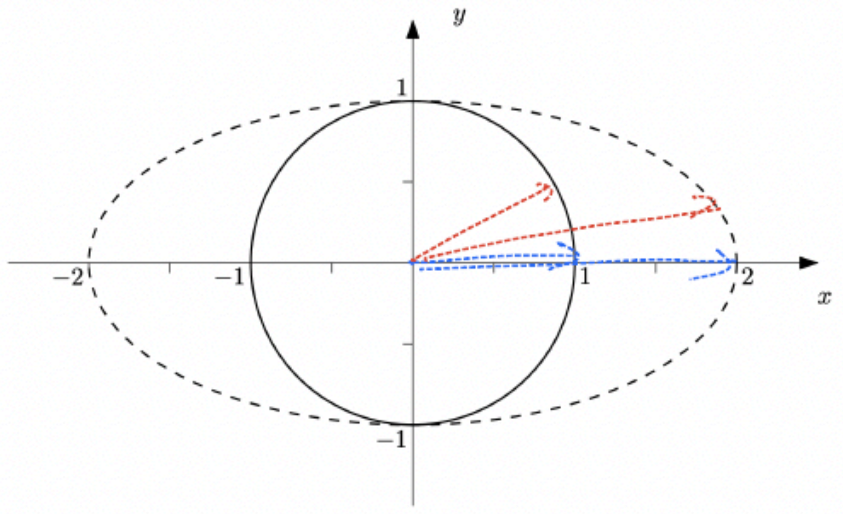
1.The Diagonal Case

Let’s solve $\arg\max_{v:||v||_2=1} v^TAv$ for the special case where $A = x^Tx$ is a diagonal matrix.

$$A = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \dots & \dots & \ddots & \dots \\ 0 & 0 & \cdots & \lambda_d \end{pmatrix} where\ \lambda_1 \geqslant \cdots \geqslant \lambda_d \geqslant 0$$

Any $d \times d$ matrix A can be thought of as a function that maps points in $\mathbb{R}^d$ back to points in $\mathbb{R}^d : v \rightarrow Av$.

The matrix $\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$ maps (x, y) to $(2x, y)$:



Points on circle $\{(x,y) : x^2 + y^2 = 1\}$ are mapped to the ellipse $\{(x,y) : \left(\frac{x}{2}\right)^2 + y^2 = 1\}$.

So what direction v should maximize $v^T A v$ for diagonal A ?

Claim: It should be the direction of maximum stretch:

Since $\lambda_1 \geqslant \cdots \geqslant \lambda_d$,

$$v = e_1 (where\ e_1 = (1, 0, \cdots, 0))\ is\ 1^{st}\ standard\ basis\ vector$$

Proof:

$$v^T A v = v^T (Av) = (v_1, \cdots, v_d) \begin{pmatrix} \lambda_1 v_1 \\ \vdots \\ \lambda_d v_d \end{pmatrix} = \sum_{i=1}^{d} v_i^2 \lambda_i$$

Since v is a unit vectors, $\sum_{i=1}^{d} v_i^2 = 1$

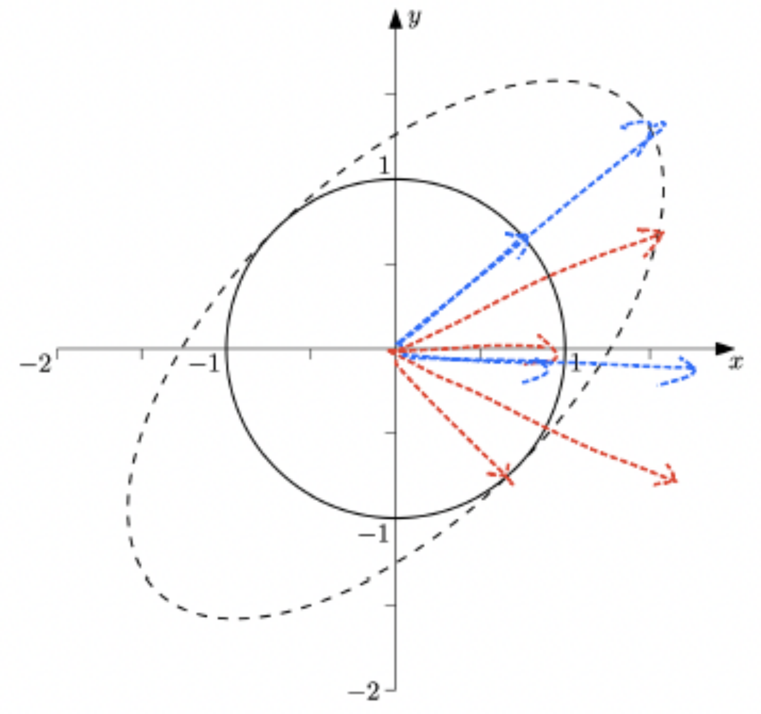
$\therefore$ since λ_1 is largest, to max set $v_1 = 1, v_i = 0\ \forall i > 0 \rightarrow \vec{v} = e_1$

2.Diagonals in Disguise

Consider

$$\begin{aligned} A &= \begin{pmatrix} \frac{3}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{3}{2} \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \\ &= QDQ^T \end{aligned}$$

Q is a rotate matrix, rotate back by 45° , D is a stretch matrix, Q^T is a rotate matrix, rotate clockwise by 45° .



The previous figure, rotated 45 degrees.

How do we formalize the concept of a rotation in high dimensions as a matrix operation?

Answer: Orthogonal matrix (also called orthogonal matrix).

An orthogonal matrix is a matrix Q s.t. for all columns $Q_1, \cdots, Q_d$

$$||Q_i||_2^2 = 1, \ \forall i$$
$$Q_i^T Q_j = 0, \ \forall i \neq j$$

Key properties:

- $Q^T Q = I \quad (Q^{-1} = Q^T)$

$$\begin{pmatrix} - - Q_1^T - - \\ \vdots \\ - - Q_d^T - - \end{pmatrix} \begin{pmatrix} | & & | \\ | & & | \\ Q_1 & \cdots & Q_d \\ | & & | \\ | & & | \end{pmatrix} = \begin{pmatrix} 1 & & \\ & \vdots & \\ & & 1 \end{pmatrix}$$

- $||Qv||_2^2 = ||v||_2^2$

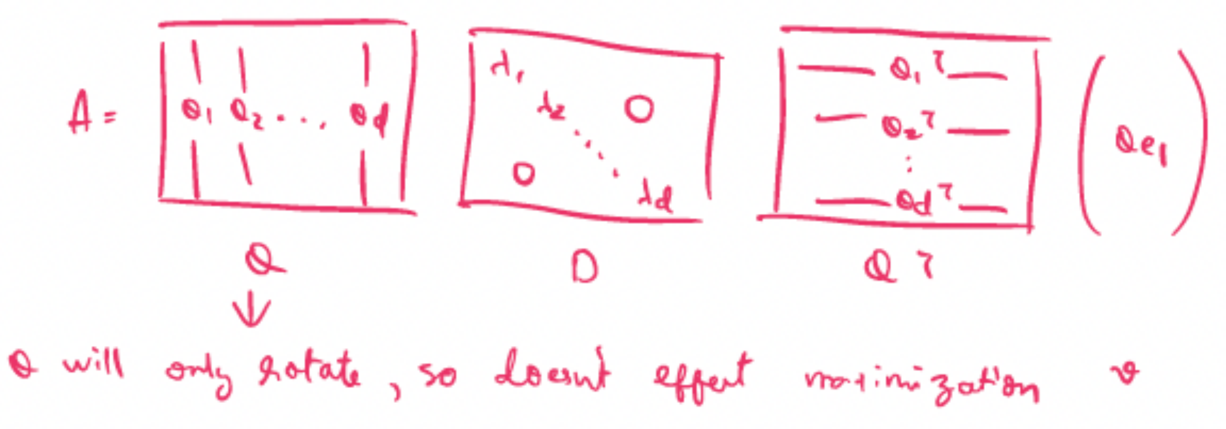
$$||Qv||_2^2 = v^T Q^T Q v = v^T v$$

- If Q is orthogonal, Q^T is also orthogonal.

$$Q^T Q = I \rightarrow Q(Q^T Q) = Q \rightarrow (QQ^T)Q = Q \rightarrow QQ^T = I$$

Recall that we want to find $v_1 = \arg\max_{v:||v||_2=1} v^T A v$.

Now consider A that can be written as $A = QDQ^T$ for an orthogonal matrix Q and diagonal matrix D with diagonal entries $\lambda_1 \geqslant \cdots \geqslant \lambda_d \geqslant 0$.



Q will only rotate, so doesn't effect maximization.

What is the direction which gets stretched the maximum?

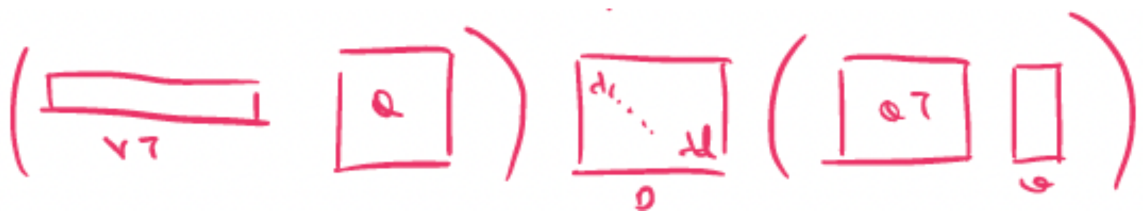
(Informal answer) The maximum possible stretch by D is λ_1 . The direction of maximum stretch under D is e_1. Therefore, direction of maximum stretch under DQ^T is v s.t. $Q^T v = e_1 \Rightarrow v = (Q^T)^{-1} e_1 = Qe_1$.

Claim: for $A = QDQ^T$, $\arg\max_{v:||v||_2=1} v^T A v = Qe_1$

Proof: for $v_1 = Qe_1$

$$\begin{aligned} v_1^T A v_1 = (Qe_1)^T (Qe_1) &= e_1^T Q^T Q D Q^T Q e_1 \\ &= e_1^T D e_1 = \lambda_1 \end{aligned}$$

Now for $v^T A v =$



Q, Q^T preserve length. $\therefore$ $Q^T v (v^T Q)$ as unit vectors.

$\therefore$ $Q^T A Q \leqslant \lambda_1$. $\therefore$ $v_1 = Qe_1$ maximizes $v^T A v$.

3.General Covariance Matrices

Consider any covariance $A = x^T x$

Linear algebra fact: any symmetric matrix A can be written as $A = QDQ^T$ for orthogonal matrix Q and diagonal matrix D .

If $A = x^T x$, A is symmetric and D always has non-negative entries, why?

$$v^T A v = v^T x^T x v = ||xv||_2^2 \geqslant 0$$

If $D_{ii} < 0$, then for $v = Qe_i$, $v^T QDQ^T v < 0$.

When $k = 1$, the solution to $\arg\max_{v:||v||_2=1} v^T A v$ is the first column of Q , where $A = x^T x = QDQ^T$ with Q orthogonal and D diagonal with sorted entries.

General values of k

What is the solution to the PCA objective for general values of k ?

$$\sum_{i=1}^{n} \sum_{j=1}^{k} < x_i, x_j >^2$$

Solution: Pick the first k columns of Q , where the covariance $x^T x = QDQ^T$ with Q orthogonal and D diagonal with sorted entries.

Since Q is orthogonal, the first k columns of Q are orthogonal vectors. These are called the top k principal components (PCs).

Eigenvalues & Eigenvectors

How to compute the top k columns of Q in the decomposition $x^T x = QDQ^T$?

Solution: Eigenvalue decomposition!

Def: An eigenvectors of matrix A is a vector v that is stretched in the same direction as by A , i.e. $Av = \lambda v$ For some $\lambda \in \mathbb{R}$.

λ is the eigenvalue.

- Eigenvectors: axes of stretch in geometric intuition
- Eigenvalues: stretch factors

When we write $A = x^T x$ as $A = QDQ^T$

- Rows of Q^T (columns of Q) are eigenvectors of A .
- Diagonal entries of D are corresponding eigenvalues.

Proof: i-th column of D is given by Qe_i

$$A(Qe_i) = QDQ^TQe_i = QDe_i = Q\lambda_i e_i = \lambda_i(Qe_i)$$

$\therefore$ i-th column of Q is eigenvector of A with eigenvalue λ_i

PCA boils down to computing the k eigenvectors of the covariance matrix x^Tx that have the largest eigenvalues.

Another Approach

for $A = x^Tx \in \mathbb{R}^{d\times d}$:

$$\begin{aligned} v_1 &= \arg\max_{v:||v||_2=1} \sum_{i=1}^{n} < x_i, v >^2 \\ &= \arg\max_{v:||v||_2=1} v^TAv \end{aligned}$$

The Rayleigh quotient of the square A and the non-0 vector v is defined as below:

$$R(A, x) = \frac{v^TAv}{v^Tv}$$

For any non-0 real number k with $R(A, kv) = R(A, v)$, it is shown that the Rayleigh quotient is invariant after vector scaling and there is redundancy. Suppose $\lambda_{\min}, \lambda_{max}$ are the minimum and maximum eigenvalues of the matrix A, then we have：

$$\lambda_{\min} \leqslant R(A, v) \leqslant \lambda_{max}$$

When v are the eigenvectors corresponding to the minimum and maximum eigenvalues, respectively, $R(A, v)$ takes these two values.

Thus, by using the Lagrange multiplier method, we construct the Lagrange multiplier function, λ is a constant:

$$L(v, \lambda) = v^TAv + \lambda(v^Tv - 1)$$

To maximize the function, we find the gradient of v and let it be 0： $2Av + 2\lambda v = 0 \rightarrow Av = \lambda v$. This means that all extreme values of the Rayleigh quotient are obtained at the eigenvalues and eigenvectors of the matrix A . Plug λ_i, x_i in Rayleigh quotient, we can prove that:

$$R(A, v_i) = \frac{v_i^T(Av_i)}{v_i^Tv_i} = \frac{v_i^T(\lambda_i v_i)}{v_i^Tv_i} = \lambda_i$$

Therefore, the Rayleigh quotient has a maximum value at the largest eigenvalue and a minimum value at the smallest eigenvalue.

According to the derivation of Rayleigh's quotient theorem above, when v_1 satisfies:

$$Av_1 = \lambda_{\max}(A)v_1$$

$\arg\max_{v:||v||_2=1} v^T Av$ takes the maximum value, i.e., for v_1 , 洋the Rayleigh quotient:

$$\frac{v_1^T A v_1}{v_1^T v_1} = \lambda_{\max}(A)$$

Thus, if $k > 1$, v can be solved by $Av = \lambda v$, where λ_i is the corresponding eigenvalue of PCs v_i .

Solving for the eigenvalues and eigenvectors of A can be done by using the SVD singular value decomposition.

Conclusion

How many PCs to use?

For visualization, we usually choose k to be small and just pick the first few principal components.

In other applications such as compression, it is a good idea to plot the eigenvalues and see. A lot of data is close to being low rank, so the eigenvalues may decay and become small.

We can also choose the threshold based on how much variance we want to capture. Suppose we want to capture 90% of the variance in the data. Then we can pick k such that i.e.

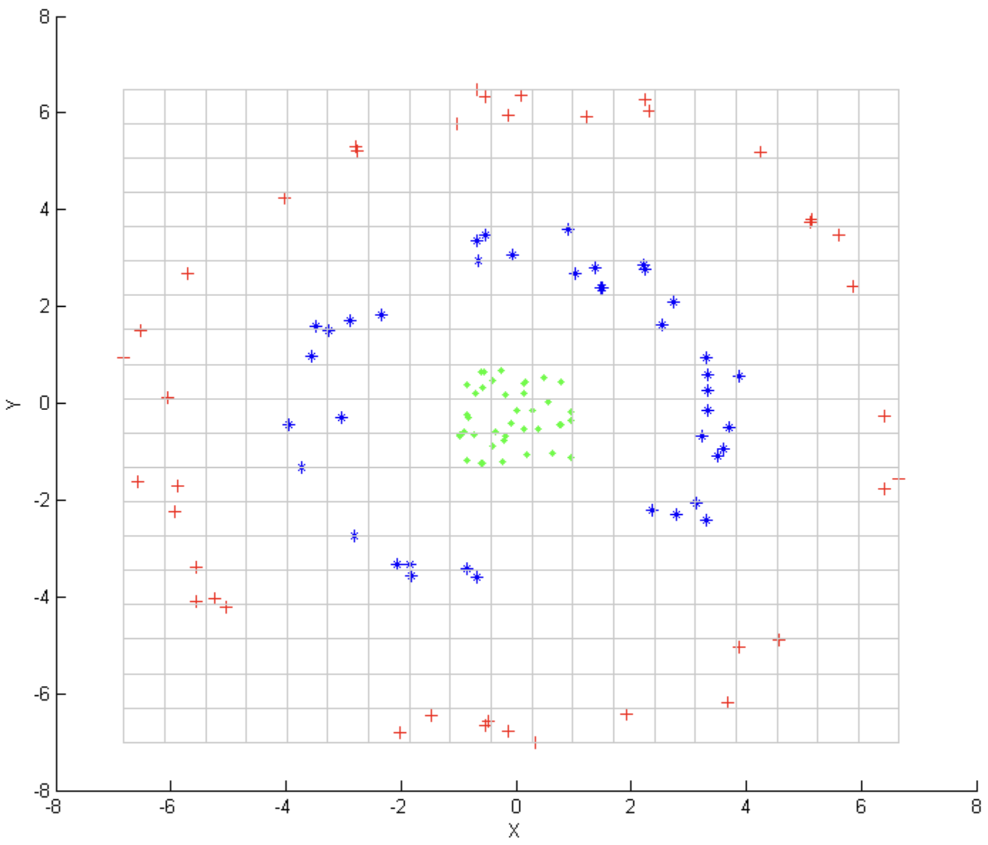
$$\frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{d} \lambda_j} \geqslant 90\%$$

Where $\lambda_1 \geqslant \cdots \geqslant \lambda_d$ are sorted eigenvalues.

Note: $\sum_{j=1}^{d} \lambda_j = trace(x^T x)$, so no need to actually find all eigenvalues.

When and why does PCA fail?

1. Data is not properly scaled/normalized.
2. Non-orthogonal structure in data: PCs are forced to be orthogonal, and there may not be too many orthogonal components in the data which are all interpretable.
3. Non-linear structure in data.


8
6
4
2
0
-2
-4
-6
-8
Y
-8
-6
-4
-2
0
2
4
6
8
X