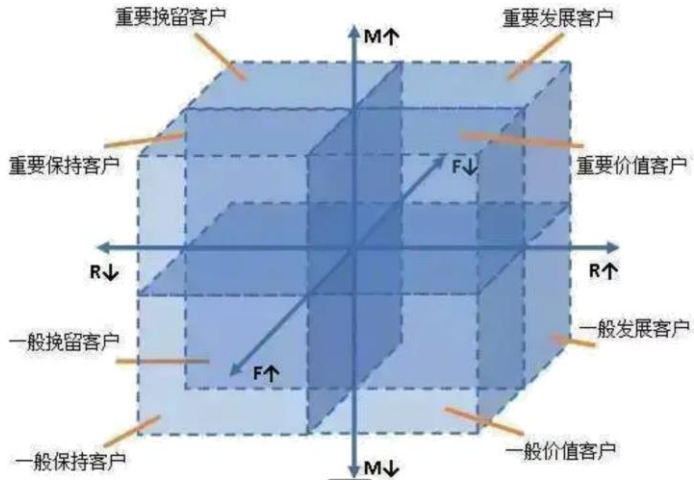
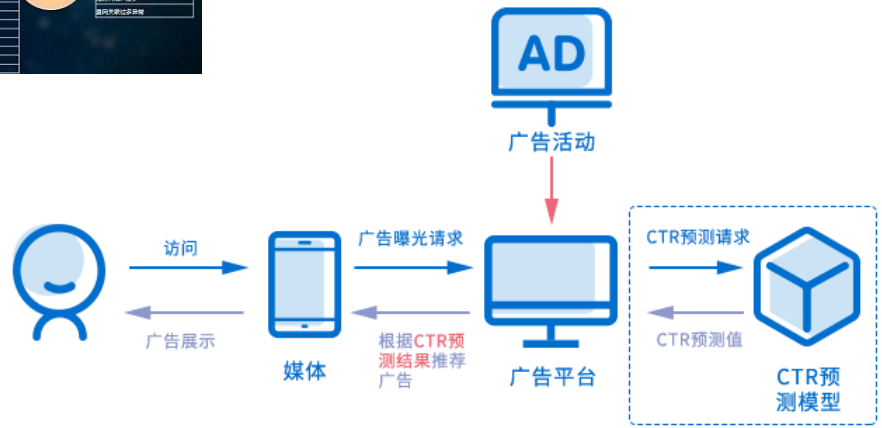
“商务视角下的数据分析”课程所覆盖的专题

1. **简介**
2. **商务思维 (Business thinking)**
   - 所谓的“商务 (BUSINESS)” – 其实就是学会做出获得更多利润的决策 (making decisions to earn more profit)
   - 管理技巧 (Management skills) – 如何落实那些决策
   - 试试创业？ – 可以！但是要慎重！！
3. **数据分析的方法概览 (Data Analytics methods)**
   - 其实，数据分析有着悠久的历史 (HISTORY view about Data Analytics)
   - 理解数据分析方法的 – 一点优化的技巧 (OPTIMIZATION)
   - 来自统计学的数据分析方法 (STATISTICS) – 基于抽样的推断 (一个有趣的视角来梳理而已，不重复)
   - 来自机器学习的数据分析方法 (BASIC + ADVANCED) – 基于数据的知识发现 (KDD)
4. **实用技巧 (Practical skills)**
   - 大商务，需要大数据
   - 大商务的两个挑战: “秒杀” 和 “精准广告/推荐”
5. **课程总结**

机器学习的商务问题 – 客户的深入理解

- **如果说统计学应对的商务问题涉及到商务的方法面面，那么，机器学习/数据挖掘应对的商务问题，可以说主要围绕客户的理解**
  - 关联规则分析 (Association Rule)
    - A priori, ECLAT, …
  - 聚类 (Clustering)
    - 不含 GMM，GMM (EM)
  - 分类 (Classification)
    - 不含 SVM，SVM
  - 神经网络
    - 深度学习(Deep Learning)之前的方法，深度学习
  - 最优多步骤策略 – MDP (马尔科夫决策)
  - 挖掘背后的主题 – Topic Modeling (主题建模)





来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

https://zh.wikipedia.org/wiki/赫尔曼·闵可夫斯基
https://en.wikipedia.org/wiki/Hermann_Minkowski

Euclidean and Minkowski (明可夫斯基) distances

$$dist(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

$$dist(X,Y) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{1/p}$$

L2 norm [L2 范式] $\|X\|_2 = \sqrt{\sum_{i=1}^{M} x_i^2}$

L1 norm [L1范式] – where p=1， $\|X\|_1 = \sum_{i=1}^{M}|x_i|$

L0 norm [L0范式] – Special: least non-zeros

Similarity

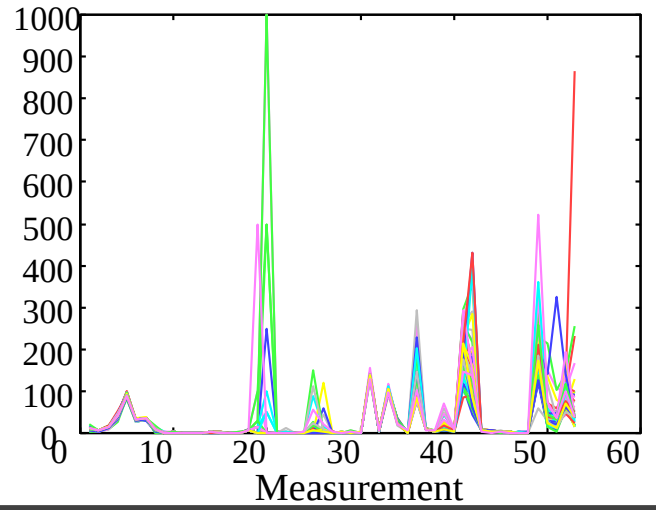
- Given two vectors $\vec{X} = (x_1, \cdots, x_m)$ and $\vec{Y} = (y_1, \cdots, y_m)$, similarity is defined as
  - $Sim(\vec{X}, \vec{Y}) = \frac{\|\vec{X} \cdot \vec{Y}\|}{\|\vec{X}\|\|\vec{Y}\|}$

Data with many properties/variables

- **Example: 53 Blood and urine measurements (wet chemistry) from 65 people (33 alcoholics, 32 non-alcoholics).**

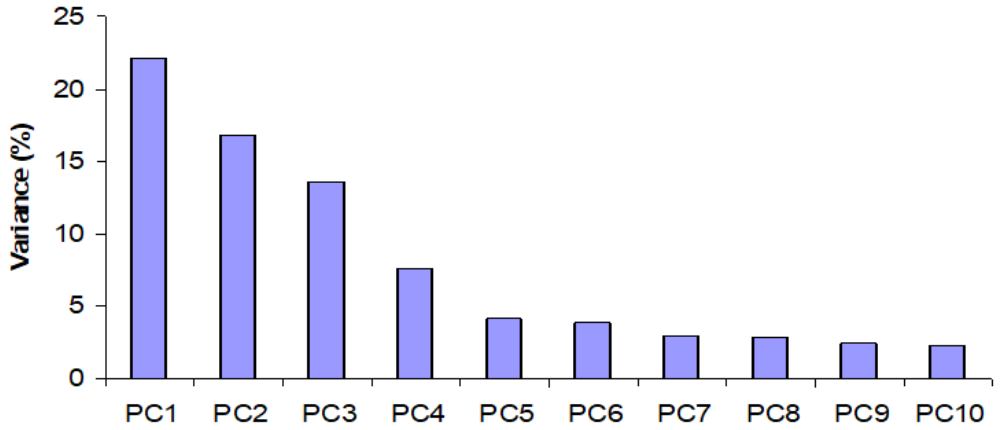
- Do we need a 53 dimension space to view data?
- How to find the 'best' low dimension space that conveys maximum useful information?
- One answer: PCA – Principle Component Analysis

	H-WBC	H-RBC	H-Hgb	H-Hct	H-MCV	H-MCH	H-MCHC
A1	8.0000	4.8200	14.1000	41.0000	85.0000	29.0000	34.0000
A2	7.3000	5.0200	14.7000	43.0000	86.0000	29.0000	34.0000
A3	4.3000	4.4800	14.1000	41.0000	91.0000	32.0000	35.0000
A4	7.5000	4.4700	14.9000	45.0000	101.0000	33.0000	33.0000
A5	7.3000	5.5200	15.4000	46.0000	84.0000	28.0000	33.0000
A6	6.9000	4.8600	16.0000	47.0000	97.0000	33.0000	34.0000
A7	7.8000	4.6800	14.7000	43.0000	92.0000	31.0000	34.0000
A8	8.6000	4.8200	15.8000	42.0000	88.0000	33.0000	37.0000
A9	5.1000	4.7100	14.0000	43.0000	92.0000	30.0000	32.0000



The philosophy is to cope with complex task by digging the major contradiction
[解决复杂问题，要抓住主要矛盾]

- **PCA is "an orthog… …that transfers the data to a ne… …such that**
  - the **greatest variance** by any projection of the data comes to lie on the first coordinate (*first principal component*),
  - the second greatest variance lies on the second coordinate (*second principal component*), and so on."



PCA

1. Given original data set S = {$\mathbf{x}^1$, ..., $\mathbf{x}^k$}, produce new set by **subtracting the mean** of attribute A_i from each x_i.

Data =

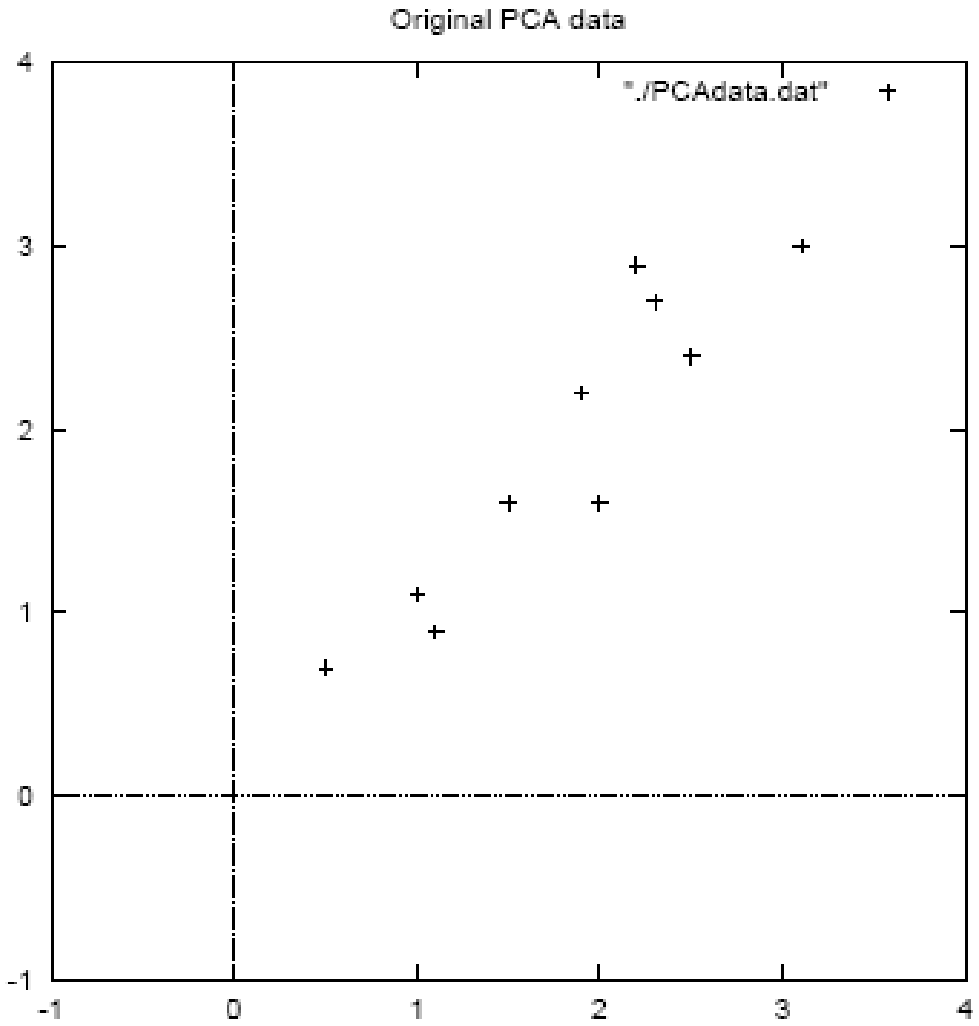
x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

Mean: 1.81 1.91

DataAdjust =

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

Mean: 0 0


Original PCA data
"./PCAdata.dat"
4
3
2
1
0
-1
-1
0
1
2
3
4

Covariance of **two attributes**

$$\mathrm{cov}(A_1, A_2) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)}$$

2. Calculate the covariance matrix (协方差矩阵):

- $\mathrm{cov}(x, y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n}$

$$cov = \begin{array}{c} \\ \mathbf{x} \\ \mathbf{y} \end{array} \begin{array}{c} \begin{array}{cc} \mathbf{x} & \mathbf{y} \end{array} \\ \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix} \end{array}$$

3. Calculate the (unit) eigenvectors (特征向量) and eigenvalues (特征值) of the covariance matrix:

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

- Covariance matrix
  - Suppose we have *n* attributes, A_1, ..., A_n.
  - Covariance matrix:

$$C^{n\times n} = (c_{i,j}), \text{ where } c_{i,j} = \text{cov}(A_i, A_j)$$

$$\text{cov}(\text{x}, \text{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n}$$

  - Example for three attributes (x,y,z):

$$C = \begin{pmatrix} \text{cov}(x,x) & \text{cov}(x,y) & \text{cov}(x,z) \\ \text{cov}(y,x) & \text{cov}(y,y) & \text{cov}(y,z) \\ \text{cov}(z,x) & \text{cov}(z,y) & \text{cov}(z,z) \end{pmatrix}$$

4. Order eigenvectors by eigenvalues – highest to lowest.

$$\mathbf{v}_1 = \begin{pmatrix} -.677873399 \\ -.735178956 \end{pmatrix} \quad \lambda = 1.28402771$$

$$\mathbf{v}_2 = \begin{pmatrix} -.735178956 \\ .677873399 \end{pmatrix} \quad \lambda = .0490833989$$

- In general, you get *n* components. To reduce dimensionality to *p*, ignore *n*–*p* components at the bottom of the list.

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

- Select top eigenvector.
  - New Feature vectors = ($\mathbf{v}_1$, $\mathbf{v}_2$, ...$\mathbf{v}_p$)

$$FeatureVector1 = \begin{pmatrix} -.677873399 & -.735178956 \\ -.735178956 & .677873399 \end{pmatrix}$$

or reduced dimension feature vector:

$$FeatureVector2 = \begin{pmatrix} -.677873399 \\ -.735178956 \end{pmatrix}$$

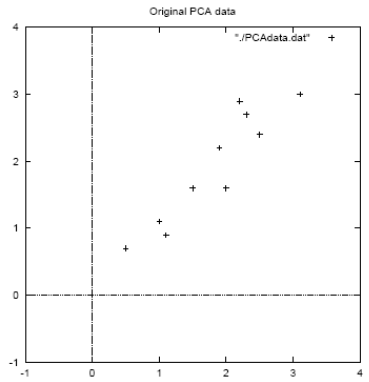
5. Derive the new data set.

- *TransformedData* = *RowFeatureVector* × *RowDataAdjust*

$$\text{RowFeatureVector1} = \begin{pmatrix} -.677873399 & -.735178956 \\ -.735178956 & .677873399 \end{pmatrix}$$

$$\textit{RowDataAdjust}^{\mathsf{T}} = \begin{pmatrix} .69 & -1.31 & .39 & .09 & 1.29 & .49 & .19 & -.81 & -.31 & -.71 \\ .49 & -1.21 & .99 & .29 & 1.09 & .79 & -.31 & -.81 & -.31 & -1.01 \end{pmatrix}$$

$$\text{RowFeatureVector1} = \begin{pmatrix} -.677873399 & -.735178956 \\ -.735178956 & .677873399 \end{pmatrix}$$



PCA as a transformer

Transformed Data=

x	y
-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287

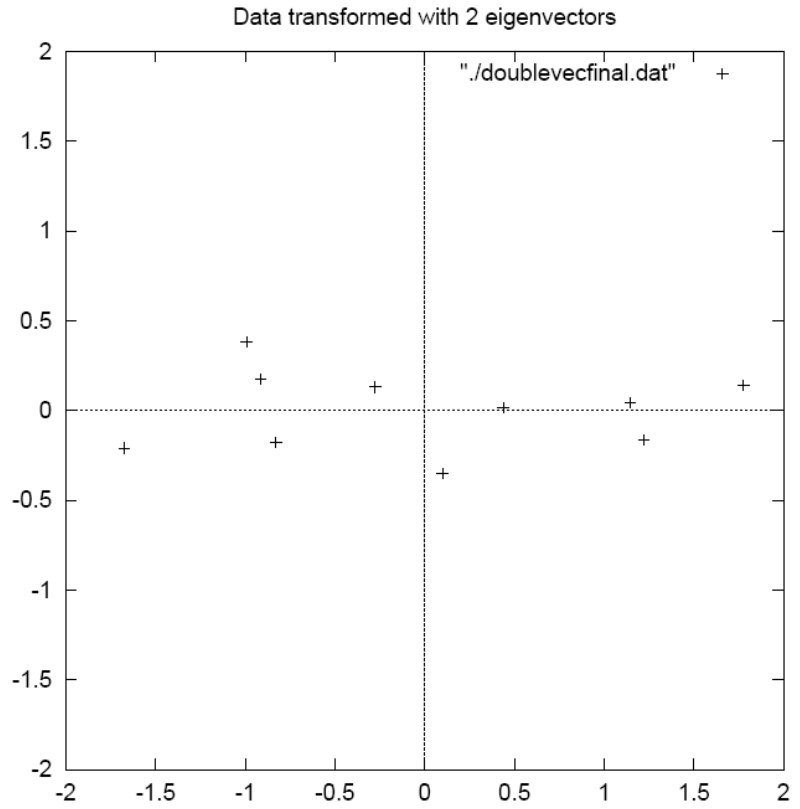


Figure 3.3: The table of data by applying the PCA analysis using both eigenvectors, and a plot of the new data points.

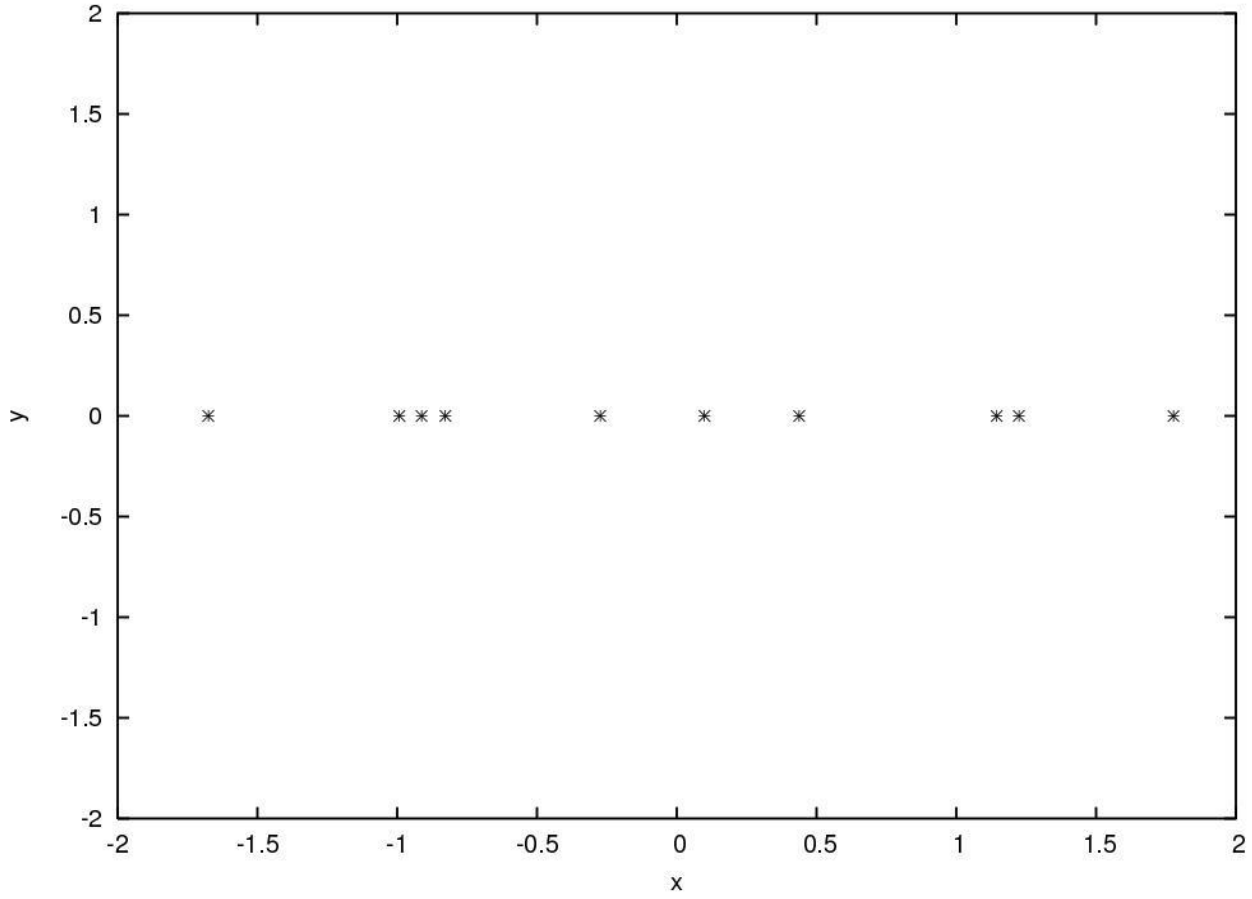
$$\text{RowFeatureVector2} = \begin{pmatrix} -.677873399 & -.735178956 \end{pmatrix}$$

PCA for Dimension Reduction [降维]

$$FeatureVector2 = \begin{pmatrix} -.677873399 \\ -.735178956 \end{pmatrix}$$

Transformed Data (Single eigenvector)

x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056



Example 23

Consider the following data (see Fig. 5.2):

$$\begin{cases} x: & -1.0 & +0.0 & +1.0 & +2.0 & +3.0 & +4.0 \\ y: & +1.1 & +0.7 & +2.3 & +1.4 & +2.2 & +3.7 \end{cases}$$

First, we adjust the data by subtracting their means $\bar{x}$ and $\bar{y}$, respectively, and we have

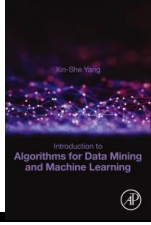
$$\bar{x} = \frac{1}{6}\sum_{i=1}^{6} x_i = 1.5, \quad \bar{y} = \frac{1}{6}\sum_{i=1}^{6} y_i = 1.9.$$

We have $X = x - \bar{x}$ and $Y = y - \bar{y}$:

$$\begin{cases} X: & -2.5 & -1.5 & -0.5 & +0.5 & +1.5 & +2.5 \\ Y: & -0.8 & -1.2 & +0.4 & -0.5 & +0.3 & +1.8 \end{cases}$$

which have zero means. We can then calculate the covariance matrix

$$\boldsymbol{C} = \begin{pmatrix} 3.500 & 1.660 \\ 1.660 & 1.164 \end{pmatrix}.$$



Since $\text{cov}(x, y)$ is positive [so is $\text{cov}(X, Y) = \text{cov}(x, y)$], it is expected that y increases with x or vice versa.

The two eigenvalues of $\boldsymbol{C}$ are

$$\lambda_1 = 4.36, \quad \lambda_2 = 0.302.$$

Their corresponding eigenvectors are

$$\boldsymbol{u}_1 = \begin{pmatrix} 0.887 \\ 0.461 \end{pmatrix}, \quad \boldsymbol{u}_2 = \begin{pmatrix} -0.461 \\ 0.887 \end{pmatrix},$$

and these two vectors are orthogonal, that is, $\boldsymbol{u}_1^T \boldsymbol{u}_2 = \boldsymbol{u}_1 \cdot \boldsymbol{u}_2 = 0$. They span the orthogonal matrix

$$\tilde{\boldsymbol{U}} = \begin{pmatrix} 0.887 & -0.461 \\ 0461 & 0.887 \end{pmatrix}.$$

Using the adjusted data

Though PCA can work well for many applications, its covarianc
sitive to a few large values. Thus, normalization of each dimension
to zero mean and unit variances. In general, PCA works under th
the underlying subspace is linear. For some applications, PCA can
arate data into different classes. In this case, linear discriminant ana

来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等

Euclidean and Minkowski (明可夫斯基) distances

$$dist(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

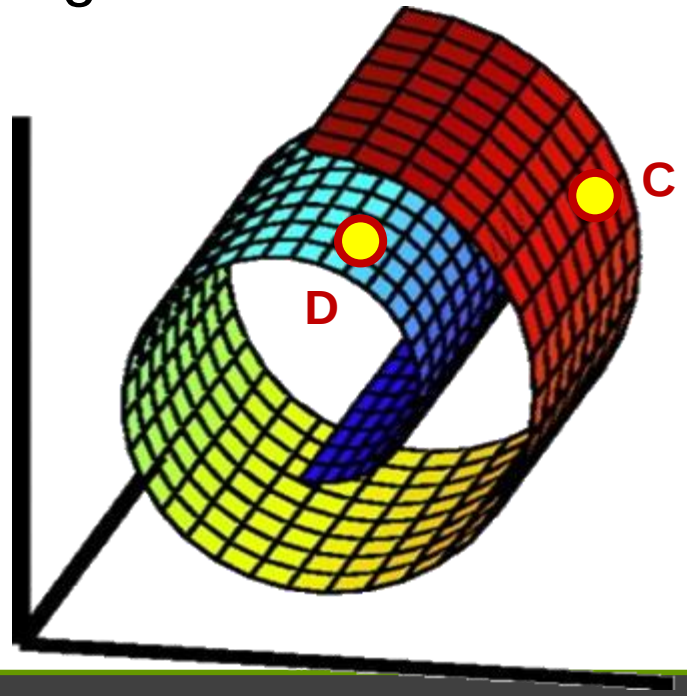
$$dist(X,Y) = (\sum_{i=1}^{n}|x_i - y_i|^p)^{1/p}$$

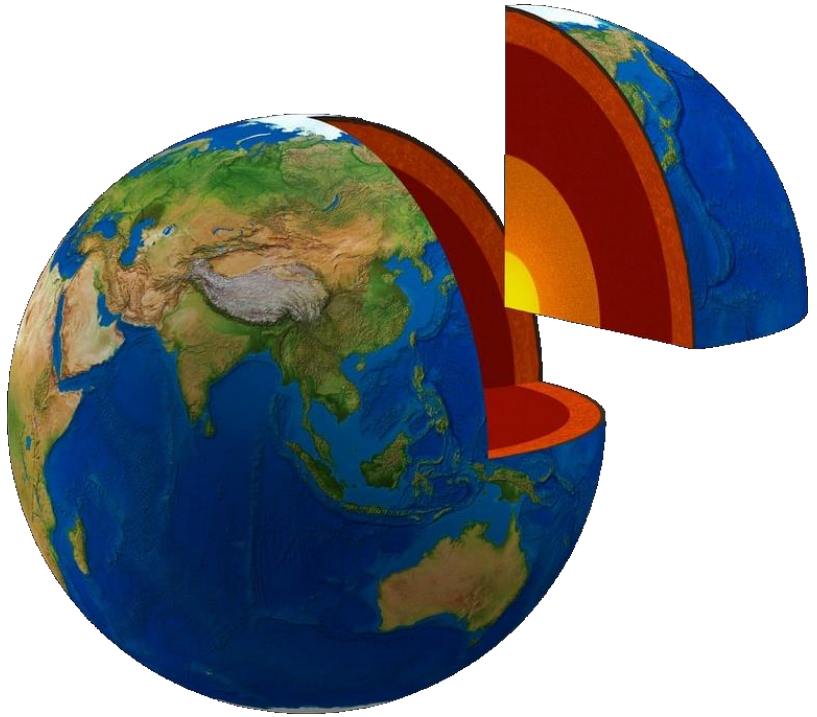
L2 norm [L2 范式] $\|X\|_2 = \sqrt{\sum_{i=1}^{M} x_i^2}$

L1 norm [L1范式] – where p=1，$\|X\|_1 = \sum_{i=1}^{M}|x_i|$

L0 norm [L0范式] – Special: least non-zeros

- **Yes, Euclidean space and distance are intuitive for us – We like them!**
- **But, there are still other spaces and distances not intuitive**
  - Distance between **C** and **D**? – depended on the connection from **C** to **D** along the surface

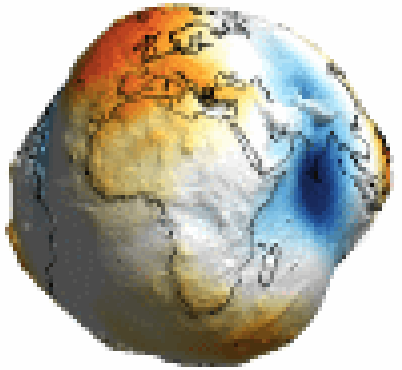




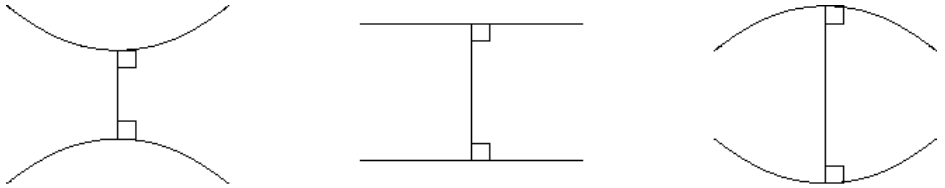
Non-Euclidean Geometries [非欧几何]

We human are 3-D creature

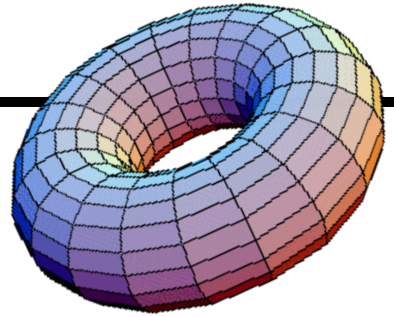
- Since we live on the HUGE earth, we are familiar with Euclidean Geometry first
  - we know Euclidean geometry is the approximation of manifold earth
  - Euclidean distance could be understood as reduced dimension math for 3-D manifold

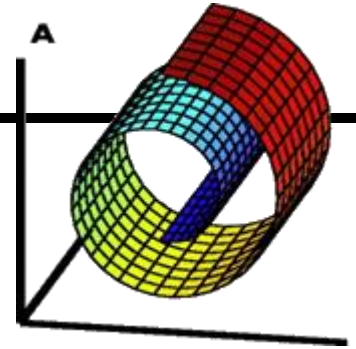


- And we spend many years to know other geometries – Non Euclidean geometries
  - Lobachevsky Geometry
    罗巴切夫斯基几何
  - Riemann Geometry
    黎曼几何

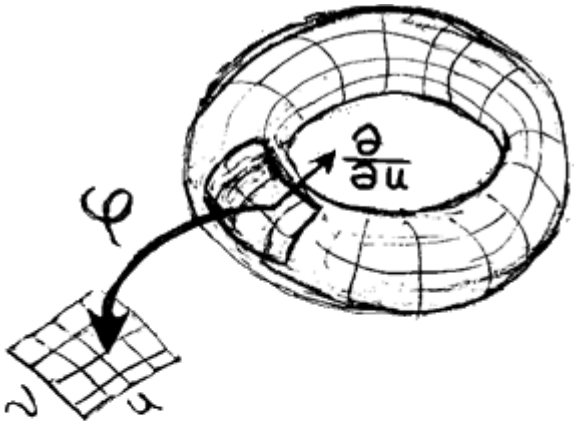


Called as Manifolds

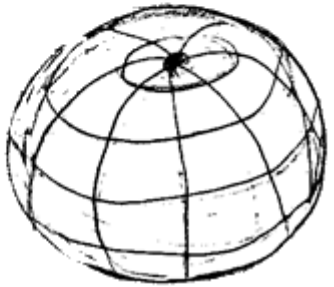




- **Manifolds [流形]?**
  - "*A manifold is a topological space which is* ***locally Euclidean***."
  - In general, any object which is nearly "flat" on small scales is a manifold.



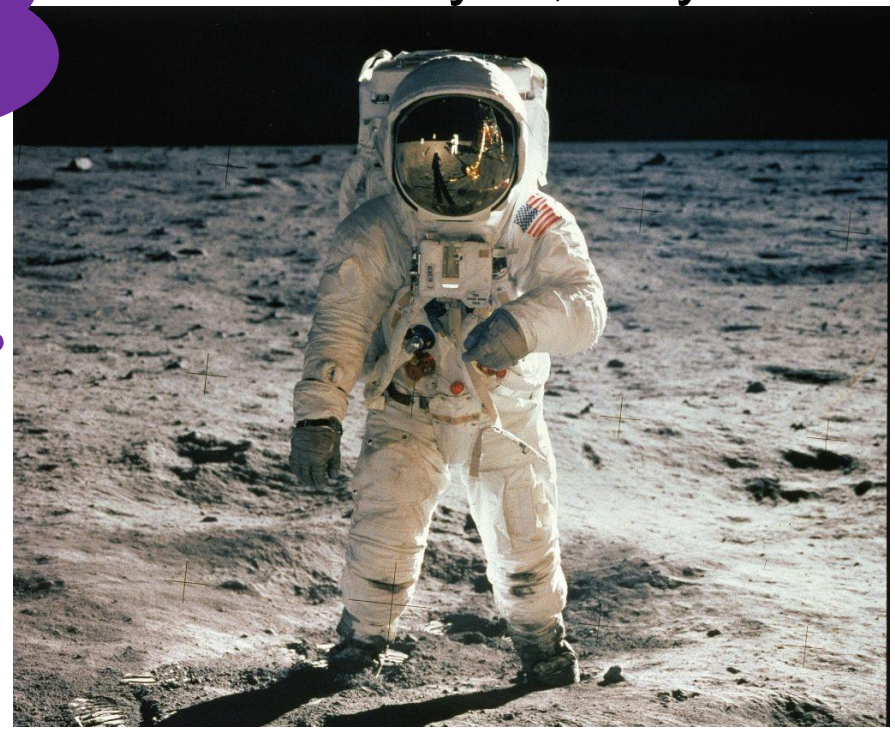
  - Euclidean space is a simplest example of a manifold.



K-NN focuses on LOCALITY

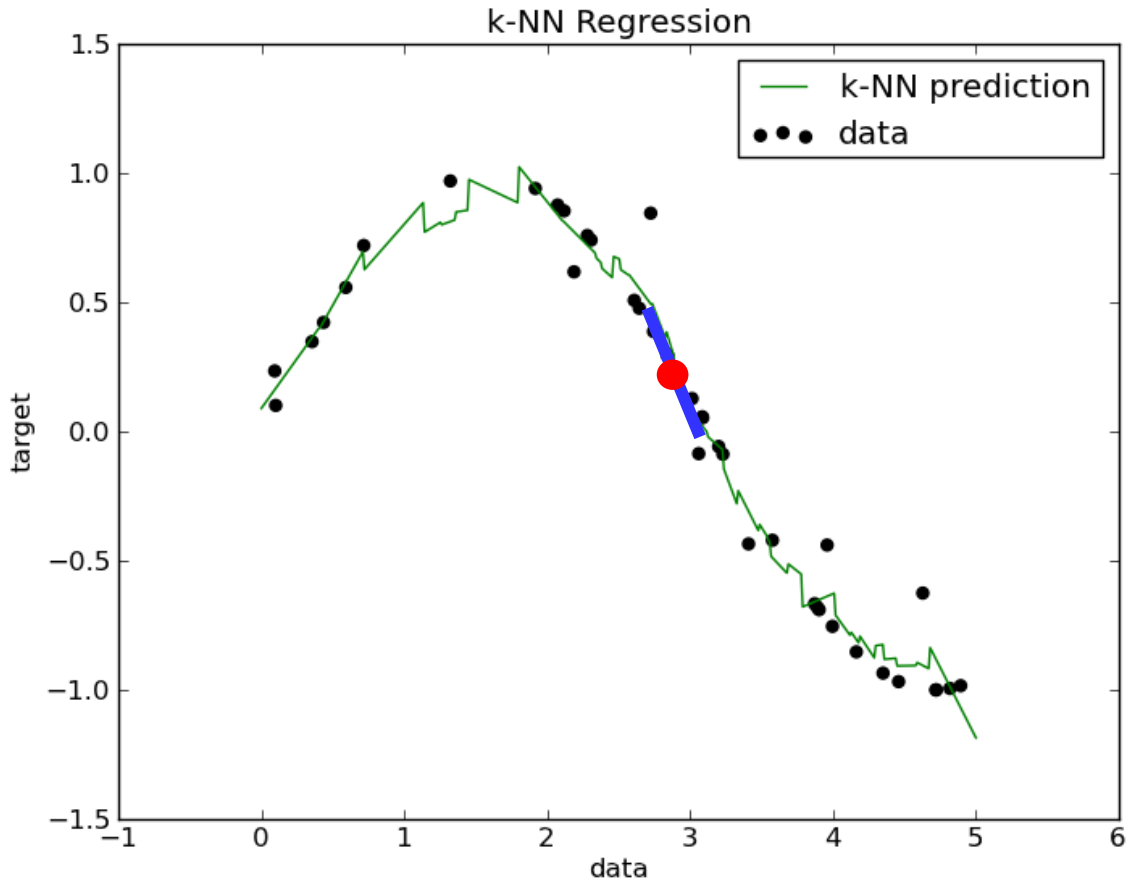
- **K NN (Nearest Neighbor) is a simple idea**
  - “物以类聚人以群分”
    - If you want to know some one, you can learn from TA’s friends
  - You visit sor ome ones like you, maybe from same cou

When a man is on a big ball, his **neighbors** are most important! ☺



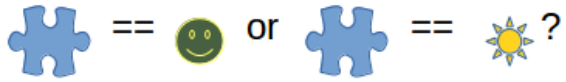
- **Approximation**
  - Based on some distance/similarity measure, top K instances (K nearest neighbors) could be selected to describe or learn the shared properties for the given sample

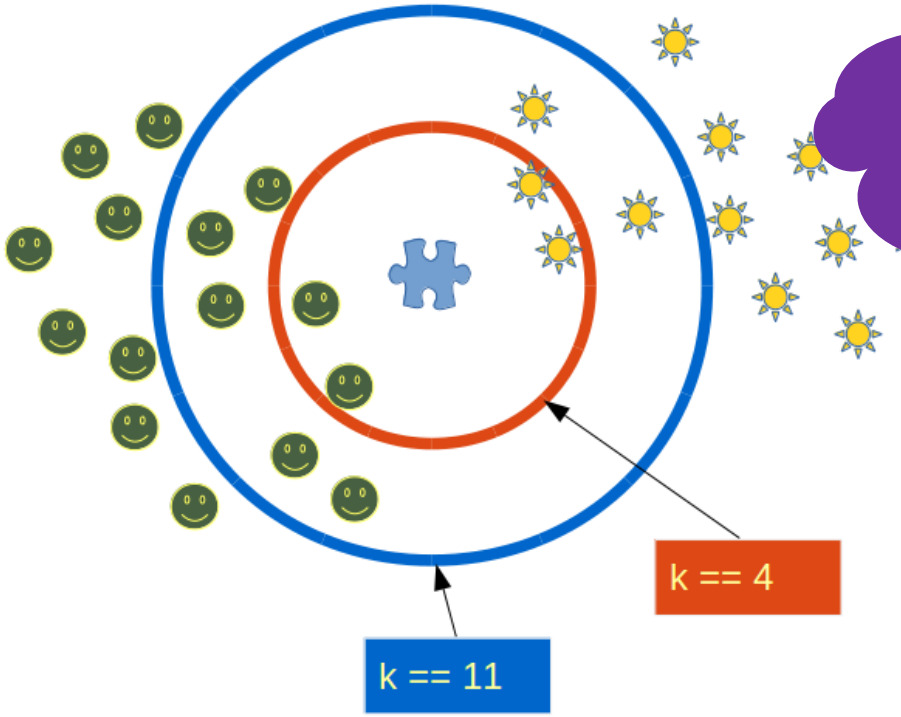
- **KNN as the naïve methodology could be used in many cases**
  - Regression



KNN as the naïve methodology could be used in many cases

- Classification



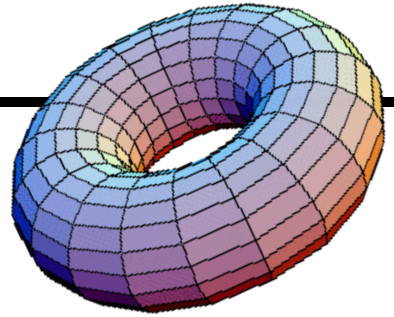


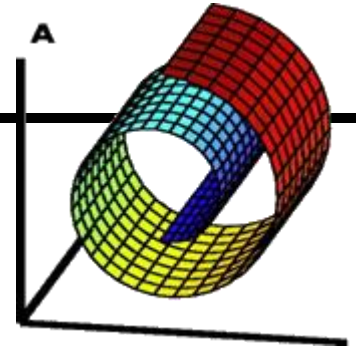
KNN idea is also used later when locating the principle components of the data

来自机器学习的数据分析方法 – 基本概念和思想

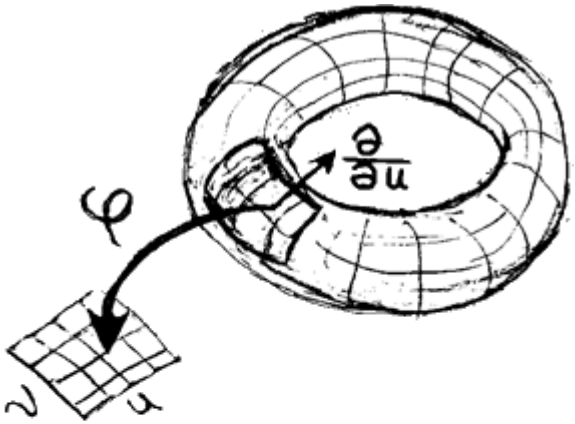
- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

Manifolds

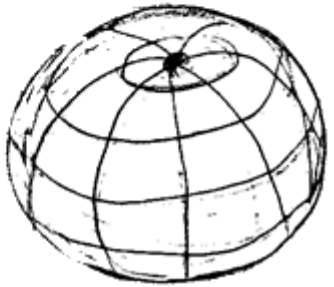




- **Manifolds [流形]?**
  - "*A manifold is a topological space which is locally Euclidean*."
  - In general, any object which is nearly "flat" on small scales is a manifold.

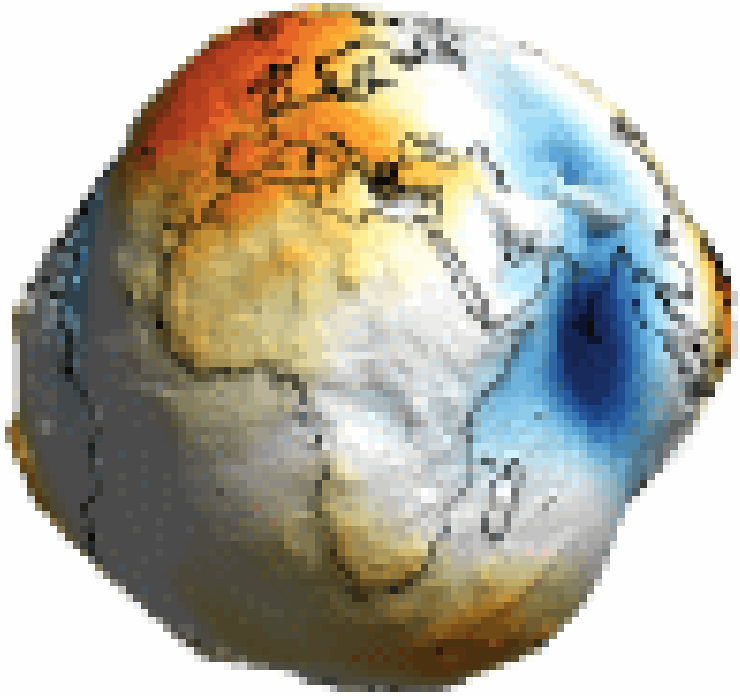


  - Euclidean space is a simplest example of a manifold.



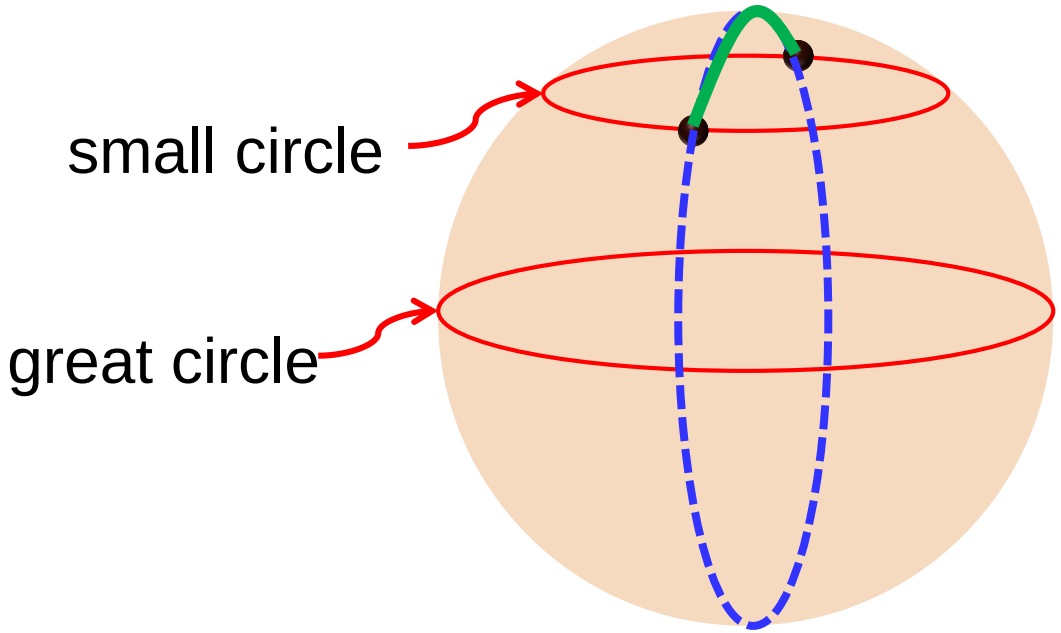
The REAL distance on a big ball?

- **Geodesic[测地线] manifold [流形] distances?**
  - **Manifold** – a method to map higher dimension to lower dimension (of course strict math definition is needed)
    - We are on 3D earth
      - The distance between two locations – like Beijing to New York – is measured by **geodesic** distance: on big circle, not the direct line
    - But still we could map relative locations into 2D map
      - But distance should be **geodesic**

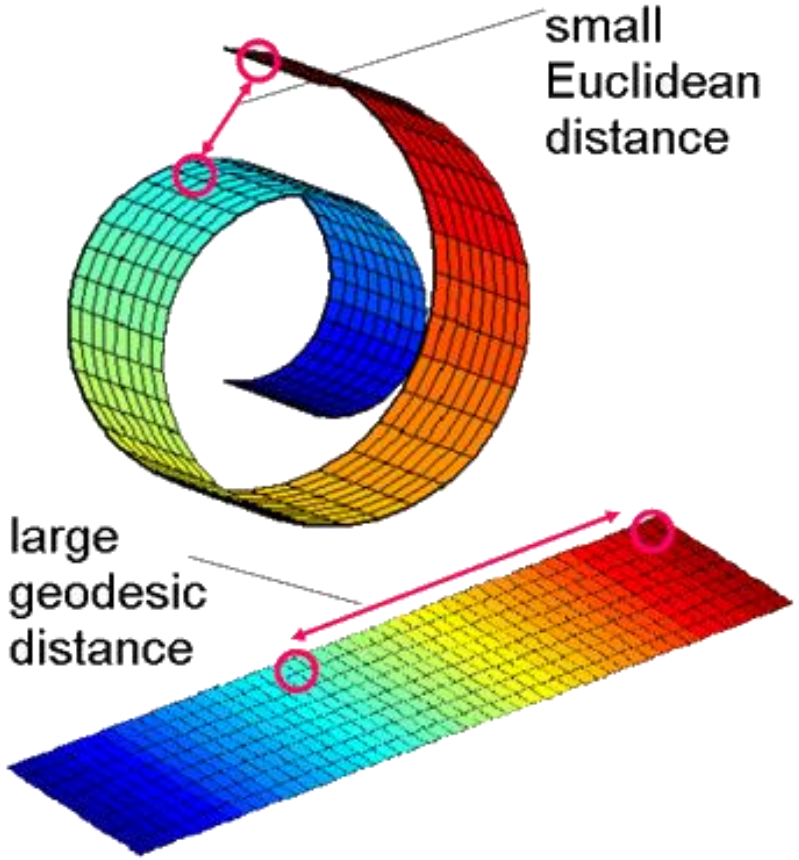


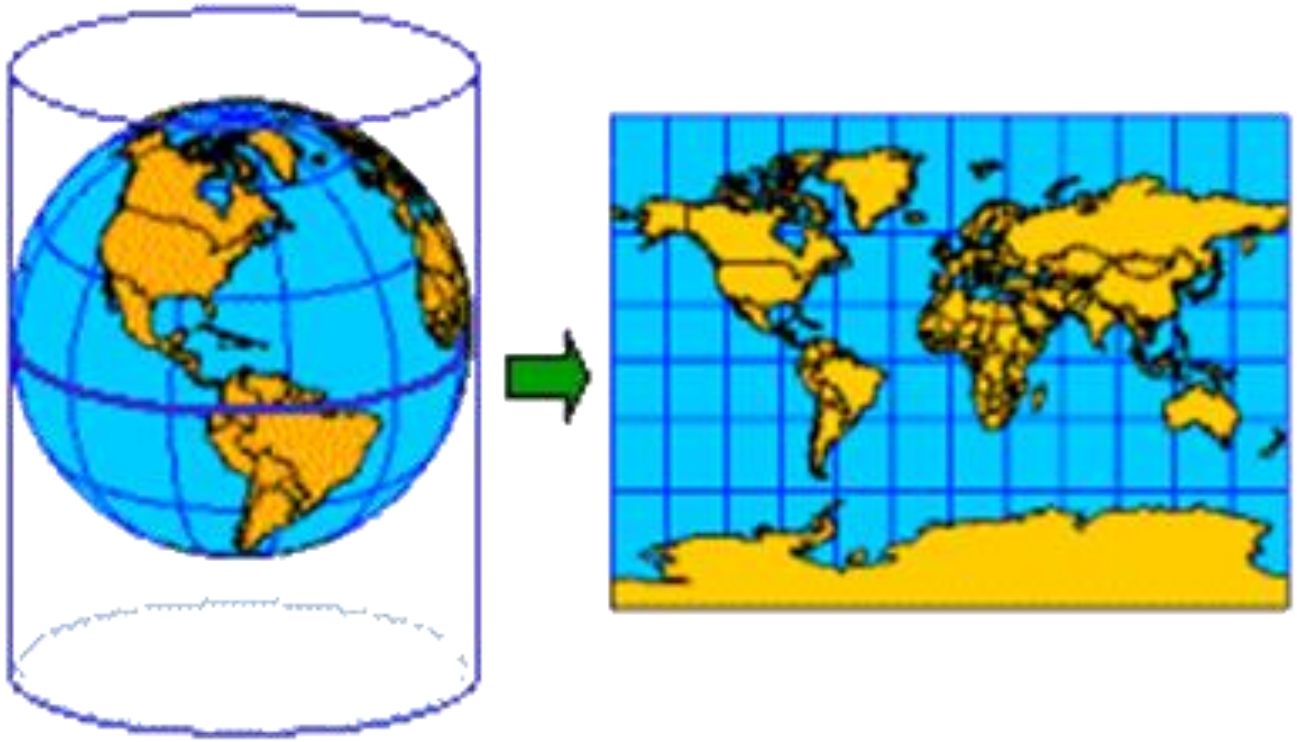
Of course, the REAL distance should be kept, especially the relative position

- **Geodesic [测地线] : the shortest curve on a manifold that connects two points on the manifold**
  - Example: on a sphere, geodesics are great circles
- **Geodesic distance [测地线距离] : length of the geodesic**



- **We still want to understand/simulate them as Euclidean**

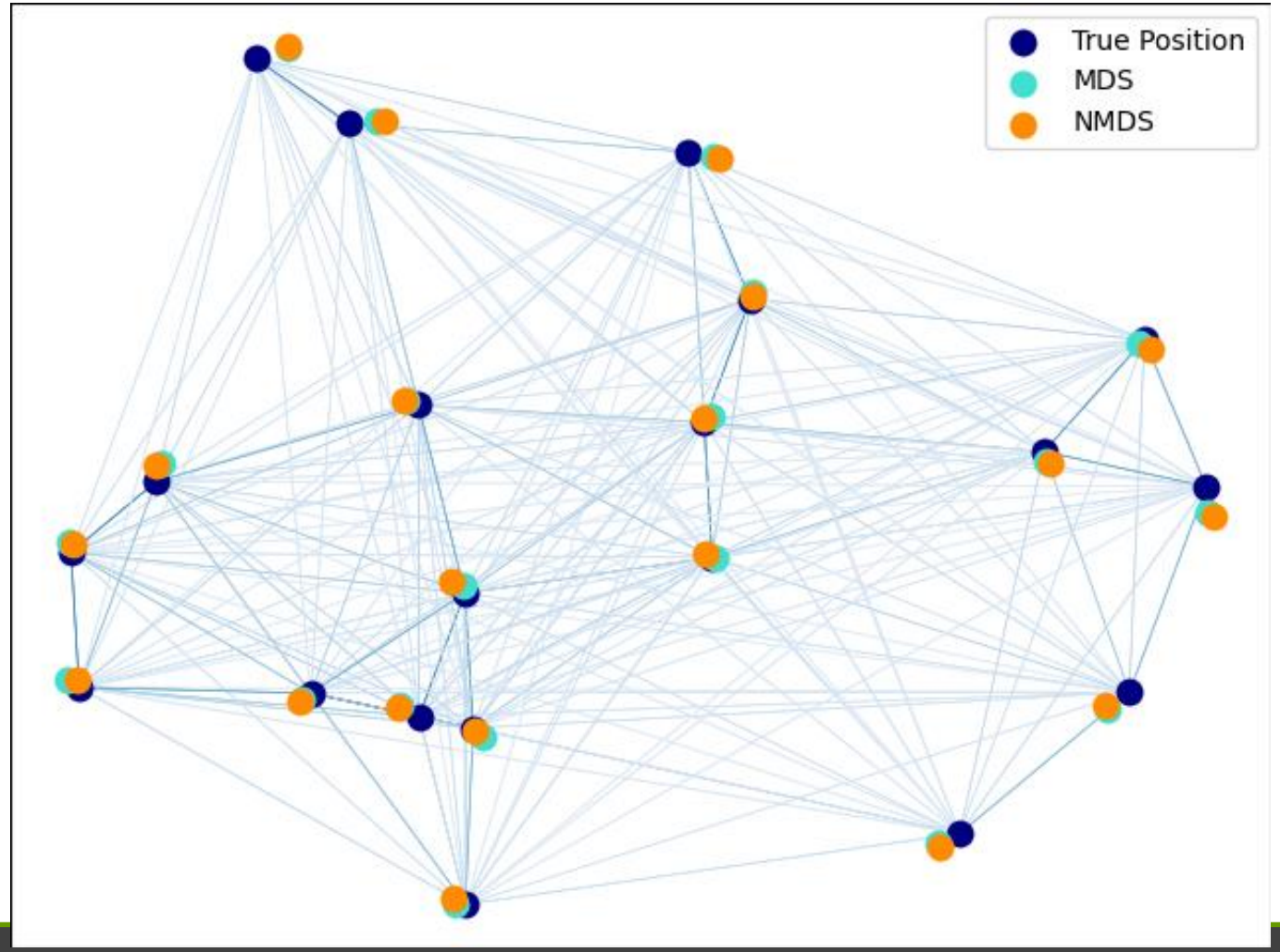




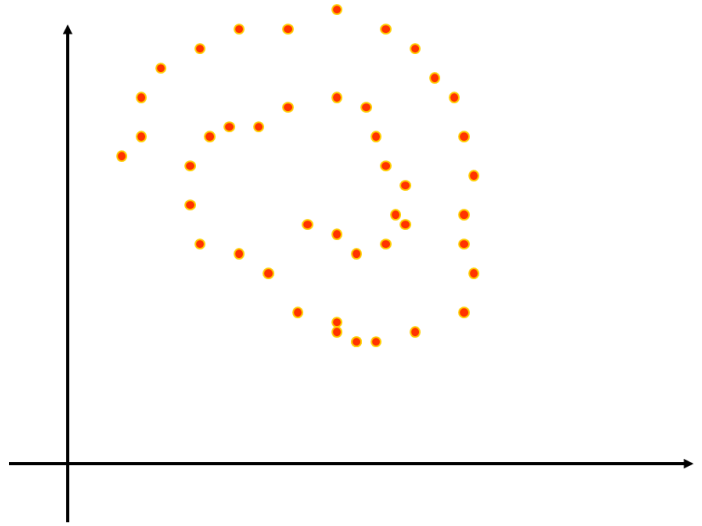
MDS: Multi-Dimensional Scaling

- **MDS as a transformer**

https://scikit-learn.org/stable/auto_examples/manifold/plot_mds.html#sphx-glr-download-auto-examples-manifold-plot-mds-py



- PCA is mainly for linear separable situation



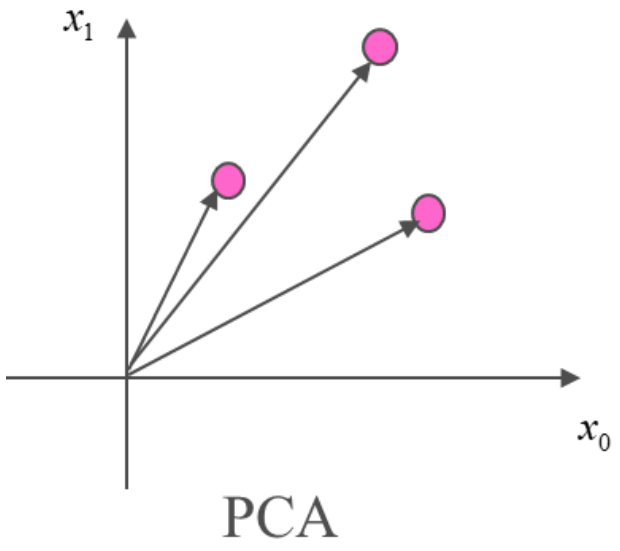
PCA cannot capture NON-LINEAR structure!

- **MDS - Preserves the inter-point distances**, equivalent to PCA when the distances are Euclidean
- MDS is the basis for many other Nonlinear Dimensionality Transforming/Reduction methods
  - IsoMap, LE, LLE, …

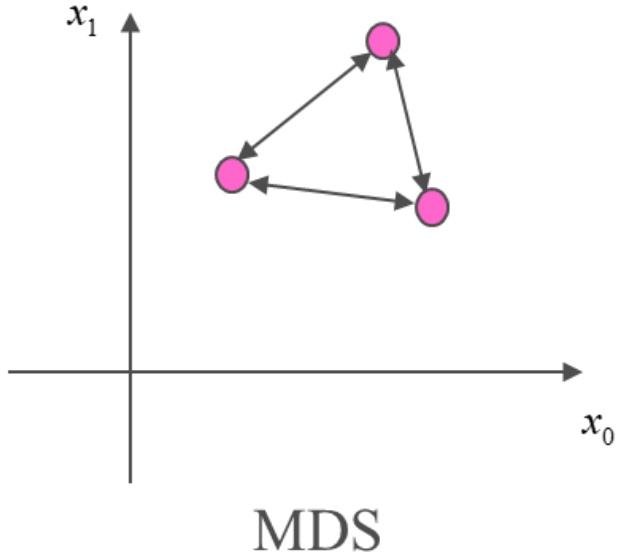
- Multidimensional scaling (MDS) is another classical approach that maps the original high dimensional space to a lower dimensional space.

MDS: Multi-Dimensional Scaling

- **Find an <u>embedding</u> that preserves the inter-point distances, equivalent to PCA when the distances are Euclidean.**



PCA



MDS

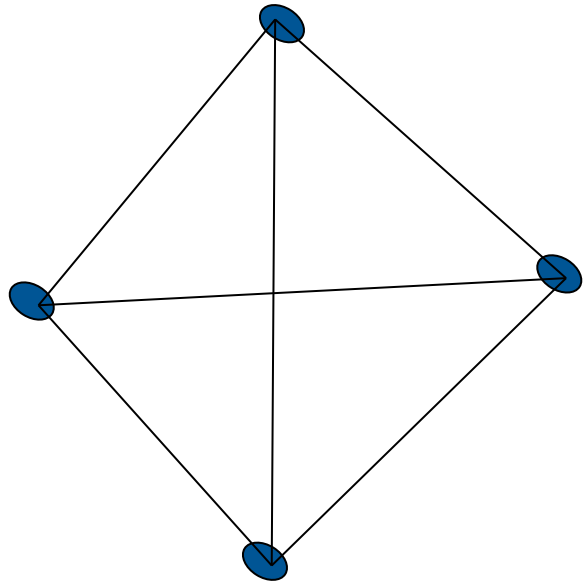
Of course, the **REAL** distance should be kept, especially the **RELATIVE POSITION**

- An embedding is a representation of a **topological object**, **manifold**, **graph**, **field**, etc. in a certain space in such a way that its connectivity or algebraic properties are preserved.

Multi-Dimensional Scaling…

- **Here we are given pairwise distances instead of the actual data points.**
  - First convert **the pairwise distance matrix** into **the dot product m**XX^T
  - **After that** same as PCA.

If we preserve the pairwise distances do we preserve the structure??

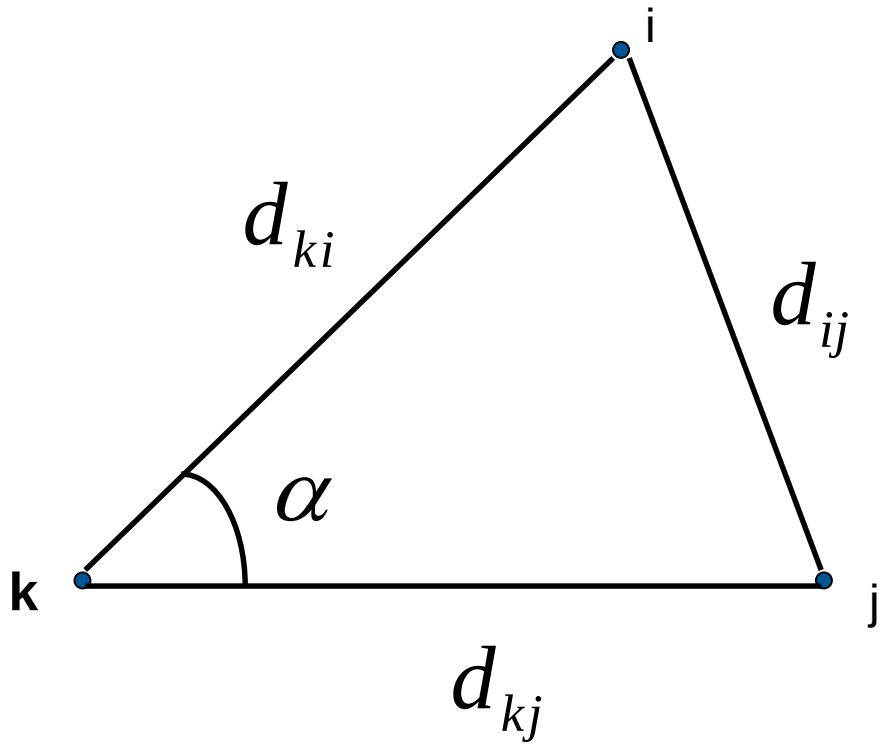


How to get dot product matrix from pairwise distance matrix?

$$d_{ij}^2 = d_{ki}^2 + d_{kj}^2 - 2d_{ki}d_{kj}cos(\alpha)$$

$$b_{ij} = d_{ki}d_{kj}cos(\alpha)$$

$$b_{ij} = \frac{1}{2}(d_{ki}^2 + d_{kj}^2 - d_{ij}^2)$$



- **MDS**
  - distances

d_{ij}

$$d_{ij}^2 = (x_i - x_j)^T (x_i - x_j)$$

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

$$d_{12}^2 = \left(\begin{bmatrix}1\\2\end{bmatrix} - \begin{bmatrix}3\\4\end{bmatrix}\right)^T \left(\begin{bmatrix}1\\2\end{bmatrix} - \begin{bmatrix}3\\4\end{bmatrix}\right)$$

$$= \begin{bmatrix}-2 & -2\end{bmatrix} \begin{bmatrix}-2\\-2\end{bmatrix} = 4 + 4 = 8$$

  - Relation

$$A = -\frac{1}{2} d_{ij}^2$$

$B = HAH^T$, H is the centering matrix

$$H = (I - \frac{1}{N})$$

$$b_{ij} = (x_i - \bar{x})^T (x_j - \bar{x})$$

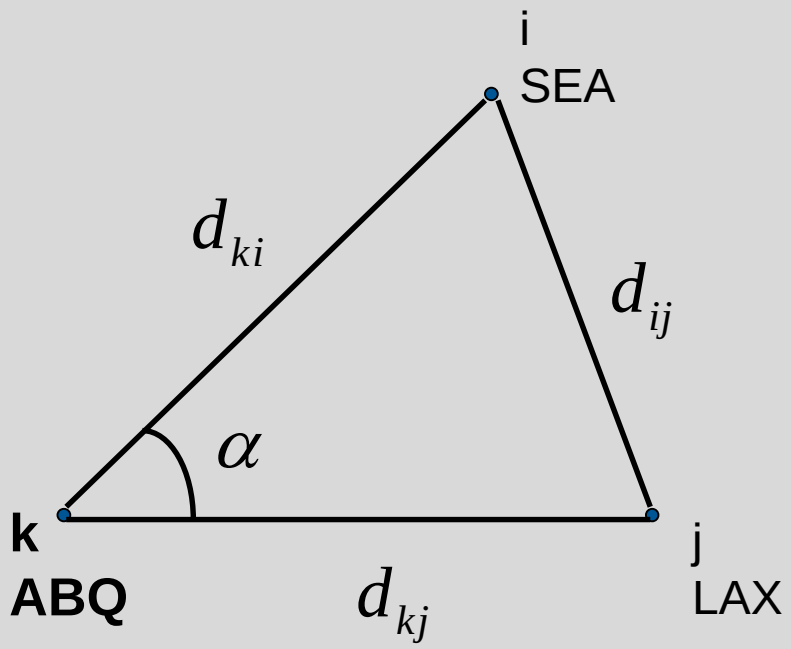
$$then\ B = (HX)(HX)^T = \hat{X}\hat{X}^T$$

$$A = -\frac{1}{2} \begin{bmatrix} 0 & 8 & 32 \\ 8 & 0 & 8 \\ 32 & 8 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 8 & 0 & -8 \\ 0 & 0 & 0 \\ -8 & 0 & 8 \end{bmatrix}$$

After this B, same as PCA.

We'll use the three-point example of the airline distances between ABQ, LAX, and SEA airports. For this example,



$$\mathbf{D} = \begin{pmatrix} 0 & 664 & 1184 \\ 664 & 0 & 959 \\ 1184 & 959 & 0 \end{pmatrix}$$

$$\mathbf{A} = -\begin{pmatrix} 0 & 664^2 & 1184^2 \\ 664^2 & 0 & 959^2 \\ 1184^2 & 959^2 & 0 \end{pmatrix}$$

$$\mathbf{B} = \left(\mathbf{I} - \frac{1}{3}\mathbf{J}\right)\mathbf{A}\left(\mathbf{I} - \frac{1}{3}\mathbf{J}\right)$$

Multidimensional scaling in R

```
> D <- c(0,664,1184,664,0,959,1184,959,0)
> D <- matrix(D,ncol=3)
> D
     [,1] [,2] [,3]
[1,]    0  664 1184
[2,]  664    0  959
[3,] 1184  959    0
> A <- -(1/2)*D^2
> A
         [,1]       [,2]       [,3]
[1,]        0 -220448.0 -700928.0
[2,] -220448       0.0 -459840.5
[3,] -700928 -459840.5       0.0
```

```
> I3 <- diag(3)
> I3
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> J <- matrix(rep(1,9),ncol=3)
> J
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
```

Note that the third eigenvalue is close to 0, suggesting that the data is nearly two-dimensional.

```
> B <- (I3 - (1/3)*J) %*% A %*% (I3 - (1/3)*J)
> B
             [,1]         [,2]       [,3]
[1,]  307313.667    6503.167 -313816.8
[2,]    6503.167  146588.667 -153091.8
[3,] -313816.833 -153091.833  466908.7
> V <- eigen(B)
> V
$values
[1] 7.378113e+05 1.829997e+05 1.205708e-12

$vectors
           [,1]       [,2]       [,3]
[1,] -0.5779379  0.5767620 0.5773503
```

Eigenvectors are in the columns of `V$vectors`.

```
> B %*% V$vectors[,1]
          [,1]
[1,] -426409.1
[2,] -155325.2
[3,]  581734.4
> V$vectors[,1] * V$values[1]
[1] -426409.1 -155325.2  581734.4
```

The three dimensional coordinates are

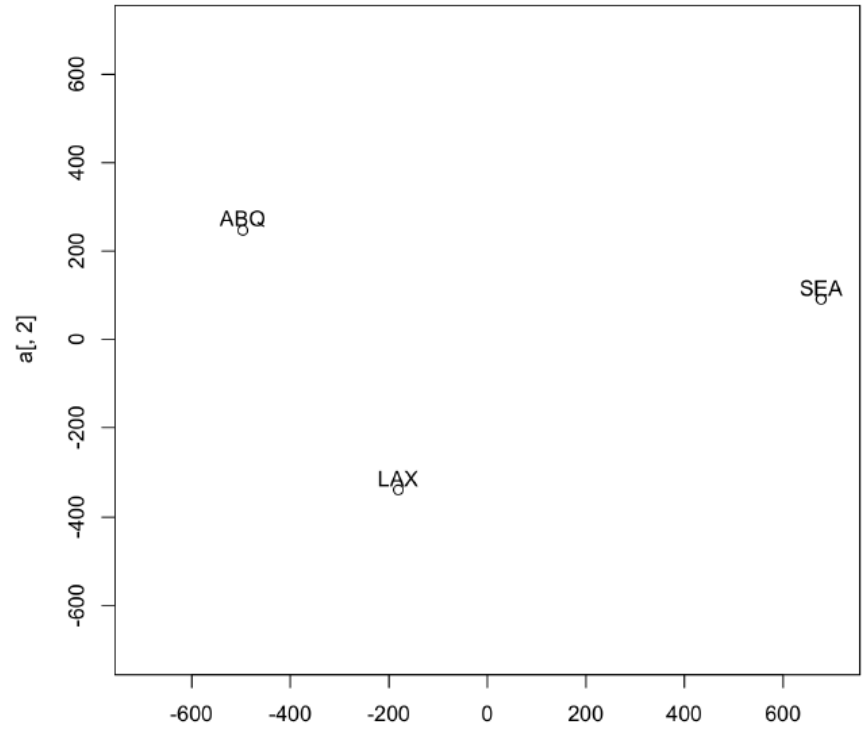
```
> V$vectors %*% diag(sqrt(V$values))
          [,1]       [,2]         [,3]
[1,] -496.425   246.7299 6.33958e-07
[2,] -180.830 -337.4750 6.33958e-07
[3,]  677.255   90.7451 6.33958e-07
```

Since the third coordinate is the same in all cases, it can be ignored, and we get 2-dimensional coordinates for the three cities.

```
> V$vectors %*% diag(sqrt(V$values))
          [,1]       [,2]         [,3]
[1,] -496.425   246.7299 6.33958e-07
[2,] -180.830 -337.4750 6.33958e-07
[3,]  677.255   90.7451 6.33958e-07
> Z <- V$vectors %*% diag(sqrt(V$values))
> plot(a[,1],a[,2])
> sqrt(sum((a[1,]-a[2,])^2)
+ )
[1] 664
> sqrt(sum((a[1,]-a[2,])^2))
[1] 664
> sqrt(sum((a[1,]-a[3,])^2))
[1] 1184
```

```
> plot(a[,1],a[,2],xlim=c(-700,700),ylim=c(-700,700))
> text(a[,1],a[,2]+25,c("ABQ","LAX","SEA"))
> # or try
> plot(-a[,1],a[,2],xlim=c(-700,700),ylim=c(-700,700))
> text(-a[,1],a[,2]+25,c("ABQ","LAX","SEA"))
```

It helps to make the plotting area square.



To choose the dimension k

It helps to make the plotting area square.

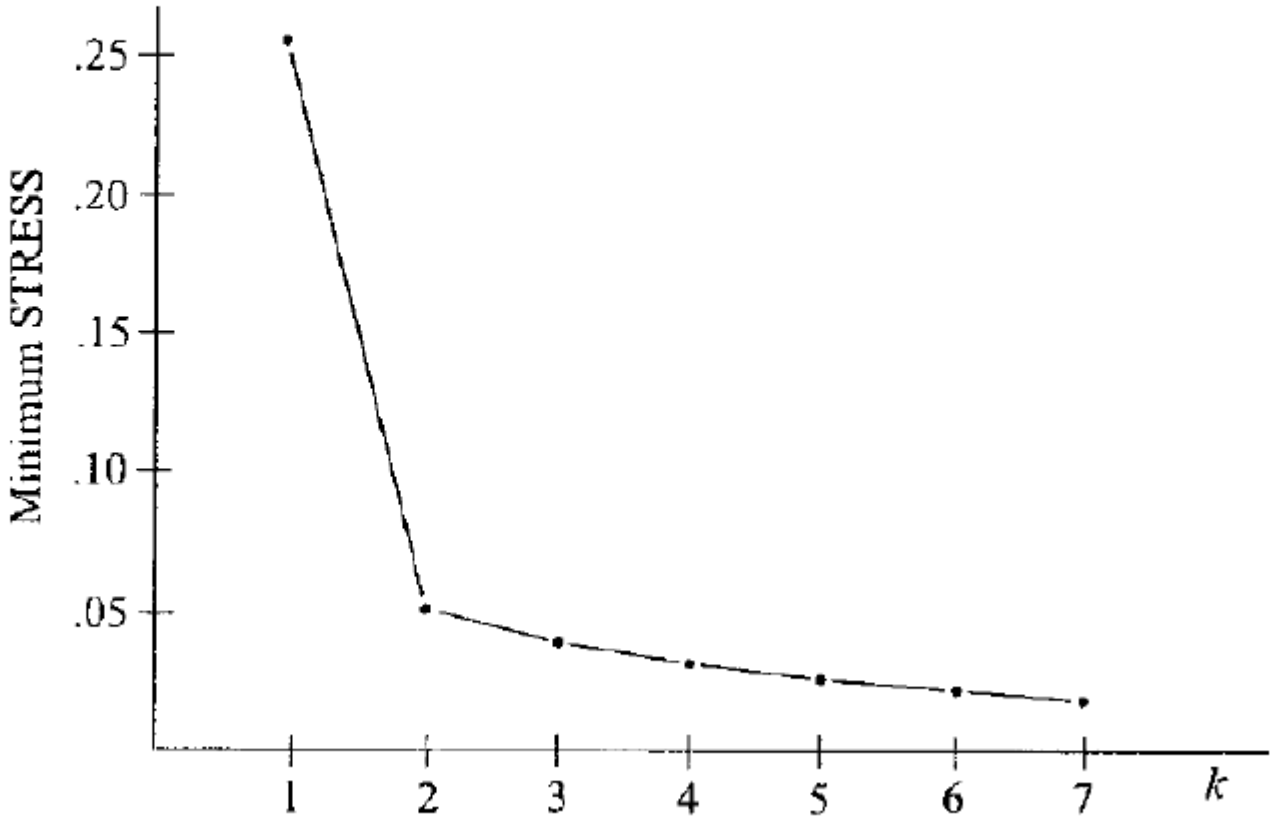


Figure 15.4. Ideal plot of minimum STRESS versus k.

MDS is built into R.

```
> MDS
$points
          [,1]       [,2]
[1,]   496.425 -246.7299
[2,]   180.830  337.4750
[3,] -677.255   -90.7451

$eig
[1] 7.37811e+05 1.83000e+05 4.36557e-11
```

This essentially agrees with what we got before but the new points are rotated compared with what we had. The `cmdscale` command also requires $k < n$, so the function returns an error for $k = 3$ even though the theory works out.

来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

- **Linear methods**
  - PCA (Principal Component Analysis) *1901*
  - MDS (Multidimensional Scaling) *1952*
- **Nonlinear methods**
  - ISOMAP(等距映射) *2000*
    - J.B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. Science, vol. 290, pp. 2319--2323, 2000
  - LLE (Locally Linear Embedding:局部线性嵌入) *2000*
    - S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. Science, vol. 290, pp. 2323--2326, 2000
  - LE (Laplacian Eigenmap:拉普拉斯特征映射) *2003*
    - M. Belkin, P. Niyogi, Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. Neural Computation, Vol. 15, Issue 6, pp. 1373 –1396, 2003

ISOMAP, LLE and Laplacian Eigenmap

- **The graph-based algorithms have 3 basic steps.**
  1. Find K nearest neighbors.
  2. Estimate **local properties of manifold by looking at neighborhoods** found in Step 1.
  3. Find a **global embedding** that preserves the properties found in Step 2.

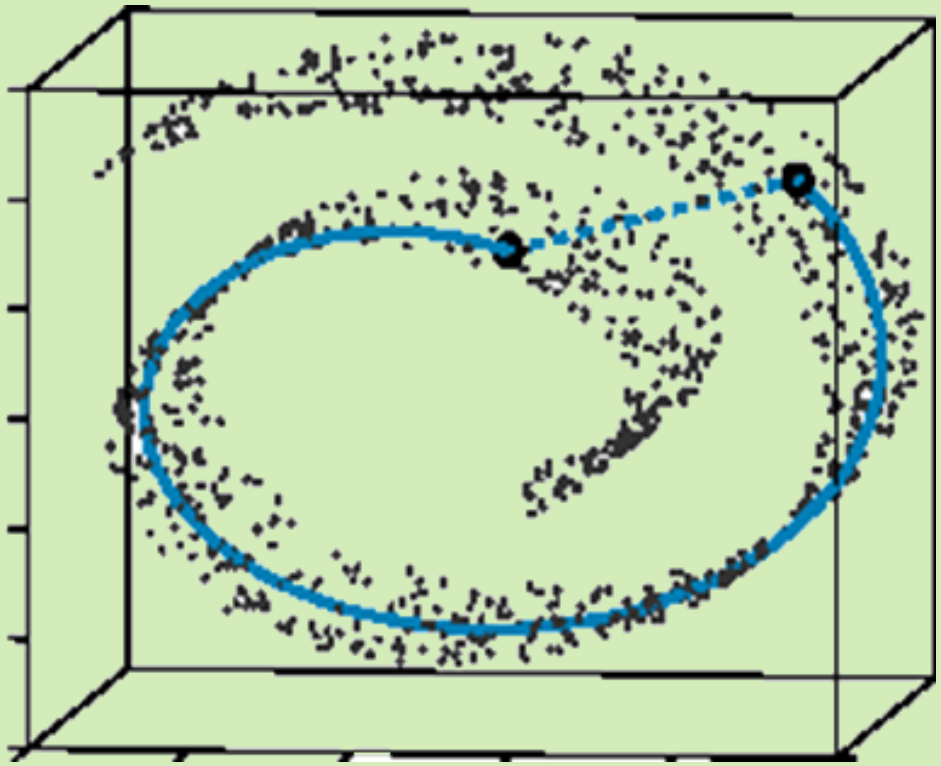
ISOMAP (algorithm description)

KNN here

- Step 1
  - Determining **neighboring points** within a fixed radius based on the input space distance $d_x(i,j)$ ← **Do you remember KNN**?
  - These neighborhood relations are represented as a **weighted graph G** over the data points.
    - ✓ Distance matrix: $d_{ij} = Neighbor\ distance$ if i and j are neighbors; else $d_{ij} = \infty$.
- Step 2
  - Estimating the geodesic distances $d_G(i,j)$ between all pairs of points on the manifold by computing their **shortest path distances (Floyd or Dijkstra)** in the graph **G**.
- Step 3
  - Constructing an **global embedding** of the data in d-dimensional (**Top d – PCA/MDS**) Euclidean space Y that best preserves the manifold's geometry.

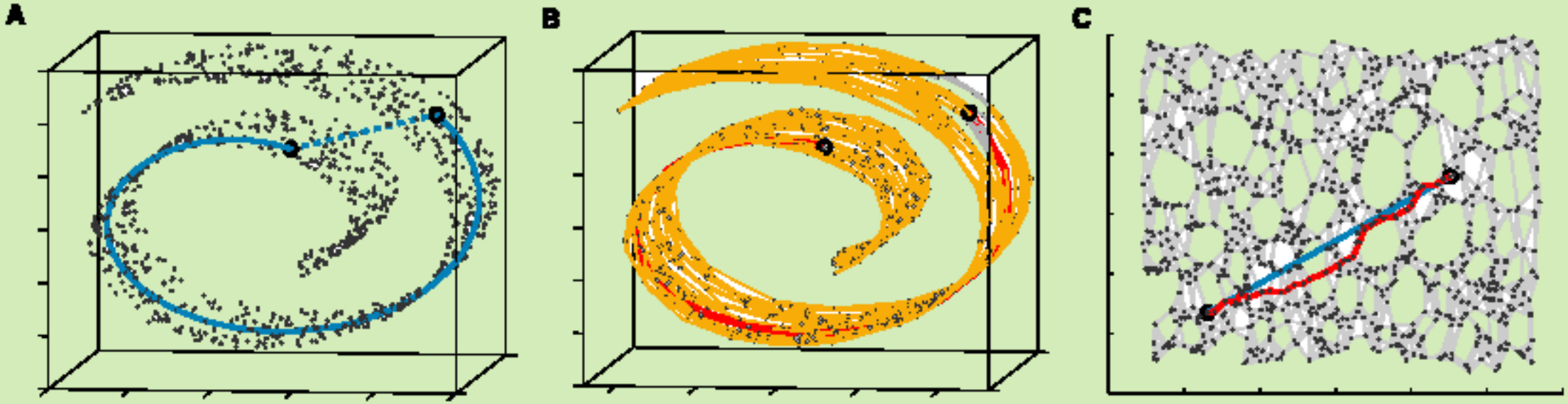
Sample points with Swiss Roll

- Altogether there are 20,000 points in the “Swiss roll” data set. We sample 1000 out of 20,000.



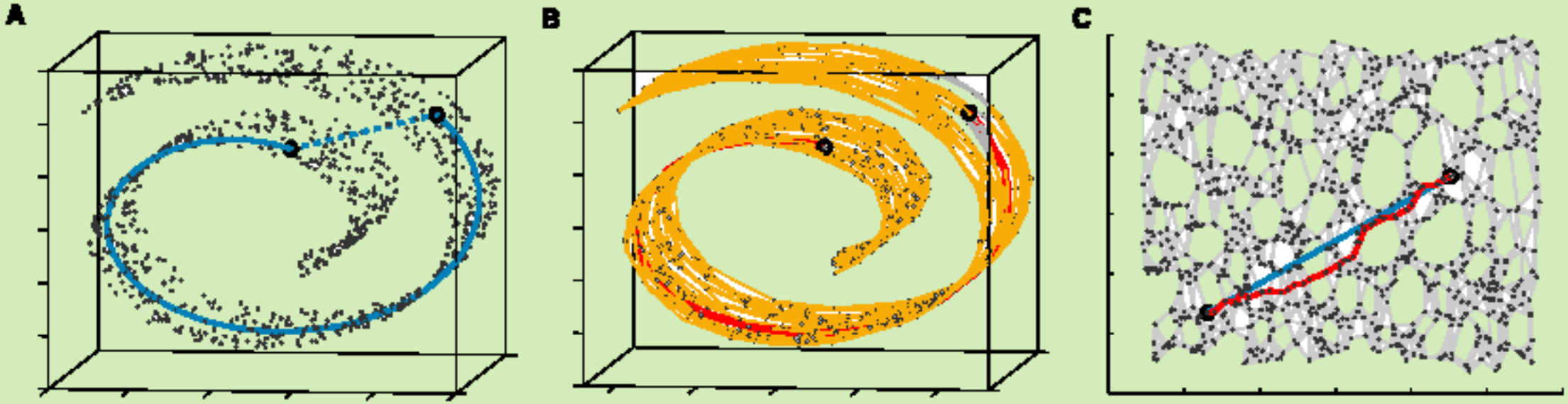
Construct neighborhood graph G

- K- nearest neighborhood (K=7)
- D_G is 1000 by 1000 (Euclidean) distance matrix of two neighbors (figure A)



Compute all-points shortest path in G

- Now D_G is 1000 by 1000 geodesic distance matrix of two arbitrary points along the manifold(figure B)



Use MDS/PCA to embed graph in R^d

- Find a d-dimensional Euclidean space Y (Figure c) to **minimize** the cost function:

$$E = \|\tau(D_G) - \tau(D_Y)\|_{L^2}$$

$$where\ D_Y(i,j) = \|y_i - y_j\|$$

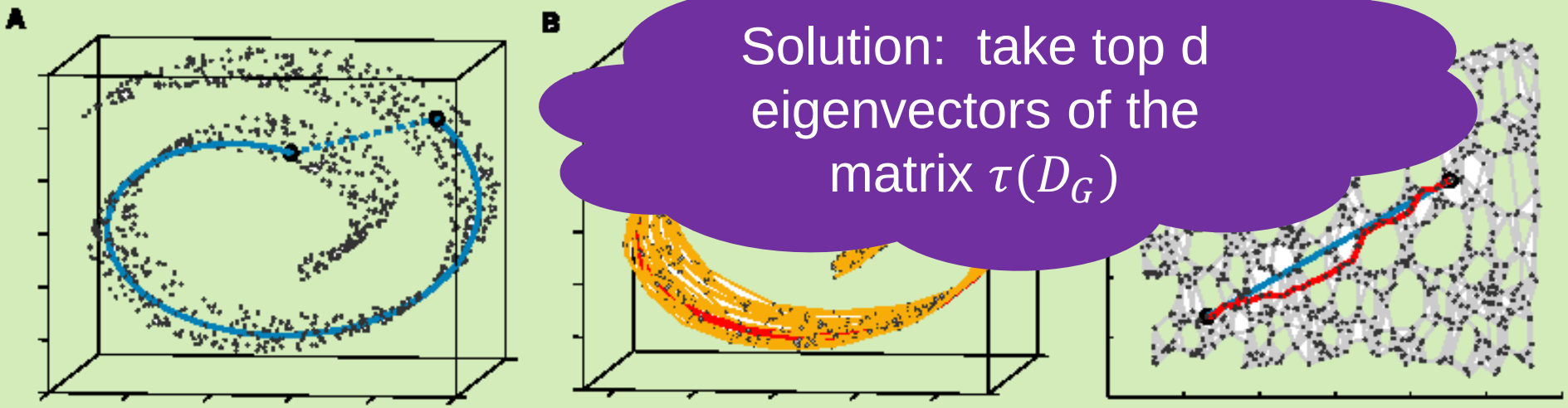
$$D_G(i,j) = d_G(i,j)$$

$$and$$

$$\tau(D) = -\frac{1}{2}(I - \frac{1}{N})D^{.2}(I - \frac{1}{N})$$

- best preserves the manifold's geometr

Solution: take top d eigenvectors of the matrix $\tau(D_G)$



LLE (Locally Linear Embedding)

- Neighborhood preserving embeddings
  - Recovering global nonlinear structure from [illegible] linear fits.
- Each data point and it's **neighbors** is expected to lie on or close to a **locally linear patch**.
  - Each data point is constructed by it's neighbors:

KNN here

$$\hat{\vec{X}}_i = \sum_j W_{ij} \vec{X}_j$$

$$W_{ij} = 0 \text{ if } \vec{X}_j \text{ is not a neighbor of } \vec{X}_i$$

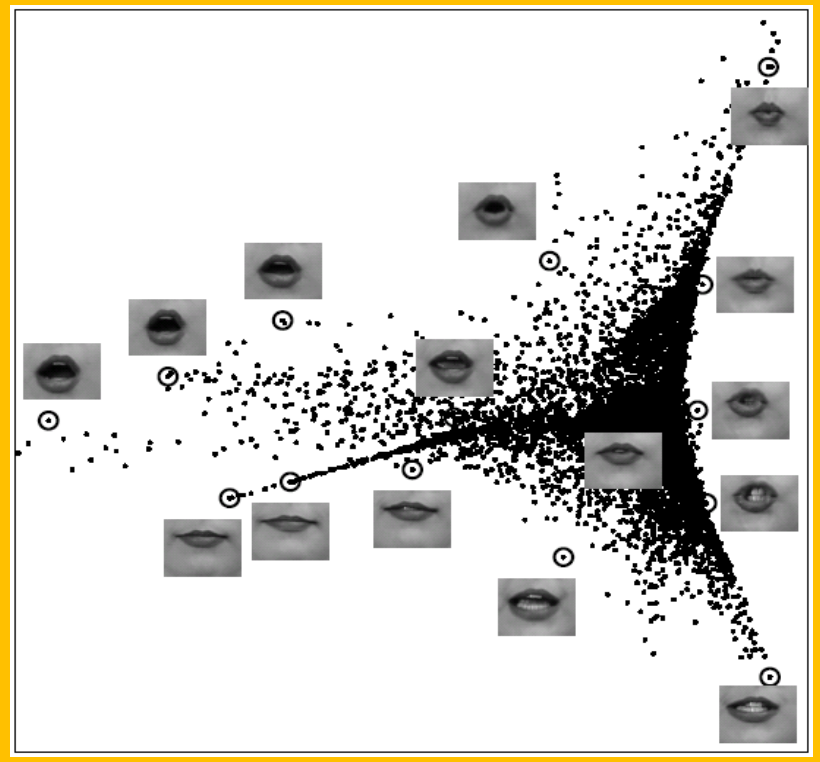
- Where W_{ij} summarize the contribution of j^{th} data point to the i^{th} data reconstruction and is what we will estimated by optimizing the error.
- Reconstructed from only its neighbors.

Experimental Results (LLE)

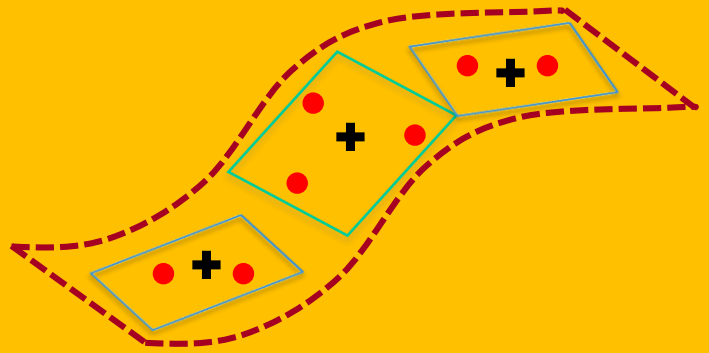
- **Lips**

 # PCA

 # LLE



Two approaches to local sparse coding

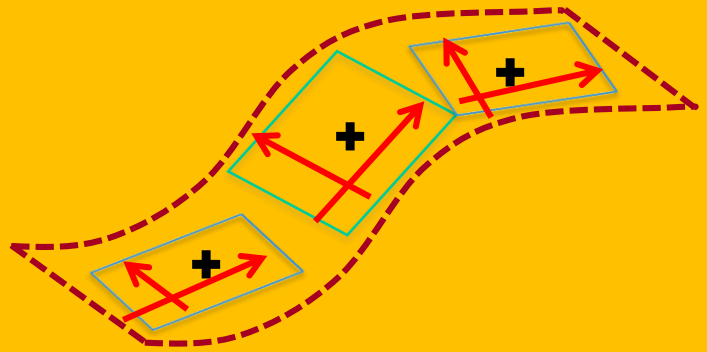


Approach 1
Coding via local anchor points

Local coordinate coding

Learning locality-constrained linear coding for image classification, Jingjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang. In **CVPR 2010**.

Nonlinear learning using local coordinate coding, Kai Yu, Tong Zhang, and Yihong Gong. In **NIPS 2009**.



Approach 2
Coding via local subspaces

Super-vector coding

Image Classification using Super-Vector Coding of Local Image Descriptors, Xi Zhou, Kai Yu, Tong Zhang, and Thomas Huang. In **ECCV 2010**.

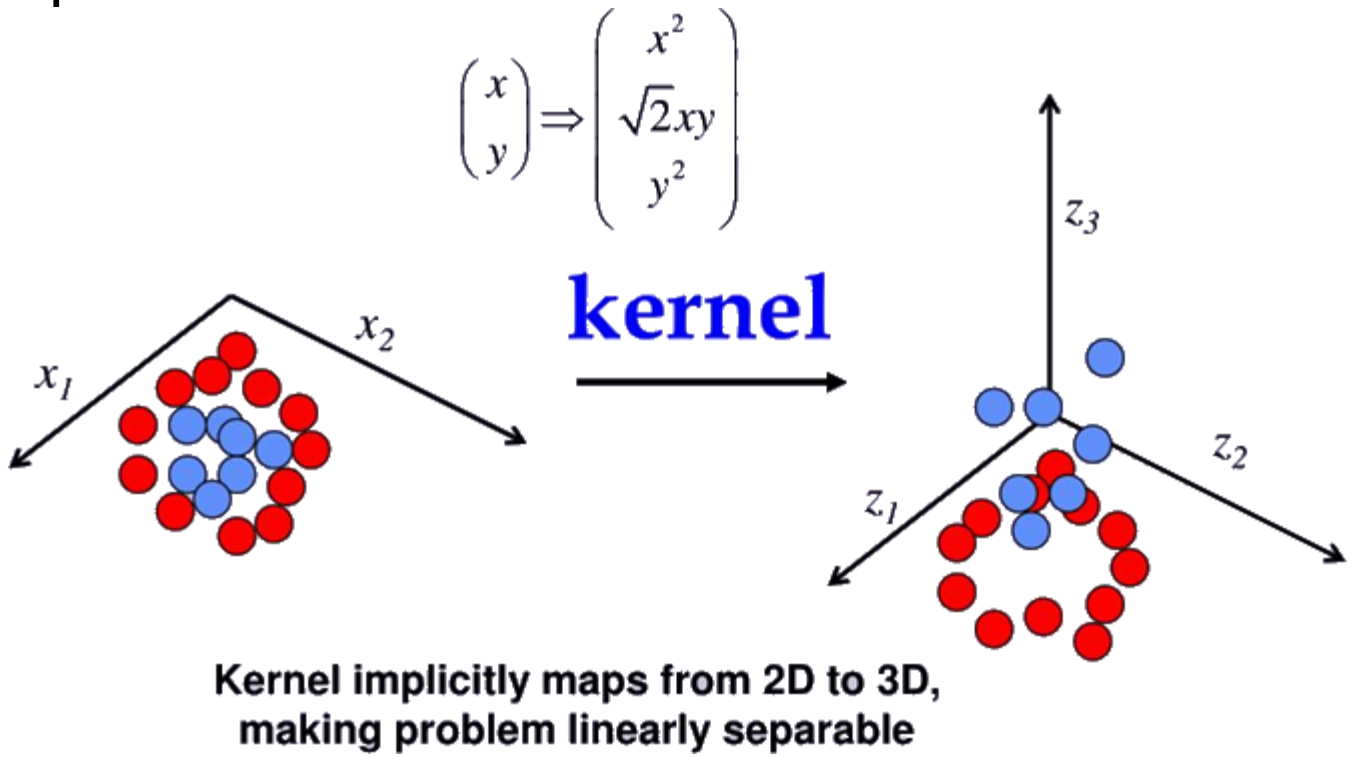
Large-scale Image Classification: Fast Feature Extraction and SVM Training, Yuanqing Lin, Fengjun Lv, Shenghuo Zhu, Ming Yang, Timothee Cour, Kai Yu, LiangLiang Cao, Thomas Huang. In **CVPR 2011**

- And it’s always one hot topic in Scientific Fictions to imagine how the more than 3D looks like

来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

- **Kernel is a skill to indirectly compare the closeness of two data records**
  - Points that are not linearly separable in 2 dimension, might be linearly separable in 3.



Kernel implicitly maps from 2D to 3D, making problem linearly separable

令 $x=[x_1,x_2,x_3]^T, y=[y_1,y_2,y_3]^T$，我们定义

$\phi(x)=[x_1x_1,x_1x_2,x_1x_3,x_2x_1,x_2x_2,x_2x_3,x_3x_1,x_3x_2,x_3x_3]$ 将原始数据从三维空间映射到九维空间中，让我们来计算 $\phi(1,2,3)\cdot\phi(4,5,6)$：

$$

$$\begin{aligned}
\phi(1,2,3) &= [1,2,3,2,4,6,3,6,9]^T \\
\phi(4,5,6) &= [16,20,24,20,25,30,24,30,36]^T \\
\phi(1,2,3)\cdot\phi(4,5,6) &= 1\times16+2\times20+3\times24+2\times20+4\times25+6\times30+3\times24+6\times30+9\times36 \\
&= 16+40+72+40+100+180+72+180+324 \\
&= 1024
\end{aligned}$$

可以看出计算相当繁琐，嗯，我们来尝试找找对应的核函数：

$$\begin{aligned}
\phi(x)\cdot\phi(y) &= [x_1x_1,x_1x_2,x_1x_3,x_2x_1,x_2x_2,x_2x_3,x_3x_1,x_3x_2,x_3x_3]^T\cdot[y_1y_1,y_1y_2,y_1y_3,y_2y_1,y_2y_2,y_2y_3,y_3y_1,y_3y_2,y_3y_3] \\
&= x_1y_1x_1y_1+x_1y_1x_2y_2+x_1y_1x_3y_3+x_2y_2x_1y_1+x_2y_2x_2y_2+x_2y_2x_3y_3 \\
&+ x_3y_3x_1y_1+x_3y_3x_2y_2+x_3y_3x_3y_3 \\
&= (x_1y_1+x_2y_2+x_3y_3)^2 \\
&= (x^Ty)^2 \\
&= K(x,y)
\end{aligned}$$

通过上面的推导，我们发现虽然维度转化的过程较为繁琐复杂，但是矢量点积的结果确实相当简洁，这一次我们直接用核函数计算：

$$K(x,y)=K((1,2,3),(4,5,6))=(1\times4+2\times5+3\times6)^2=(32)^2=1024$$

相比于从低维映射到高维空间再进行矢量积运算，核函数大大简化了计算的过程，使得向更高维转

相比于从低维映射到高维空间再进行矢量积运算，核函数大大简化了计算的过程，使得向更高维转

huanlan.zhihu.com/p/24291579

相比于从低维映射到高维空间再进行矢量积运算，核函数大大简化了计算的过程，使得向更高维转化变为了可能，我们不需要知道低维空间的数据是怎样映射到高维空间的，我们只需要知道结果是怎么计算出来的。

- **In general we need not know the form of ϕ.**
- **Just specifying the kernel function is sufficient.**

$$K(x_i, x_j)$$

- **A good kernel:**
  - Computing $K(x_i,x_j)$ is cheaper than $\phi(x_i)$
- **Valid Kernels:**
  - Symmetric $$K(\vec{x}, \vec{z}) = K(\vec{z}, \vec{x})$$
  - Must be decomposable into ϕ functions
    - **Gram matrix [格拉姆矩阵]** is positive semi-definite (**PSD**).

$$K_{ij} = \phi(x_i)^T \phi(x_j)$$

    - Diagonal entries are larger than the sum of the abs. values of the off diagonal entries in each row

$$\vec{x}^T K_{ij} \vec{x} \geq 0$$

- **线性核函数**

$$\kappa(x, x_i) = x \cdot x_i$$

线性核，主要用于线性可分的情况，我们可以看到特征空间到输入空间的维度是一样的，其参数少速度快，对于线性可分数据，其分类效果很理想，因此我们通常首先尝试用线性核函数来做分类，看看效果如何，如果不行再换别的

- **多项式核函数**

$$\kappa(x, x_i) = ((x \cdot x_i) + 1)^d$$

多项式核函数可以实现将低维的输入空间映射到高纬的特征空间，但是多项式核函数的参数多，当多项式的阶数比较高的时候，核矩阵的元素值将趋于无穷大或者无穷小，计算复杂度会大到无法计算。

- **高斯（RBF）核函数**

$$\kappa(x, x_i) = exp(-\frac{||x - x_i||^2}{\delta^2})$$

高斯径向基函数是一种局部性强的核函数，其可以将一个样本映射到一个更高维的空间内，该核函数是应用最广的一个，无论大样本还是小样本都有比较好的性能，而且其相对于多项式核函数参数要少，因此大多数情况下在不知道用什么核函数的时候，优先使用高斯核函数。

- **sigmoid核函数**

$$\kappa(x, x_i) = tanh(\eta < x, x_i > +\theta)$$

采用sigmoid核函数，支持向量机实现的就是一种多层神经网络。

- **在选用核函数的时候，**
  - 如果我们对我们的数据有一定的先验知识，就利用先验来选择符合数据分布的核函数；
  - 如果不知道的话，通常使用交叉验证的方法，来试用不同的核函数，误差最下的即为效果最好的核函数，或者也可以将多个核函数结合起来，形成混合核函数。

- **在吴恩达的课上，也曾经给出过一系列的选择核函数的方法：**
  - 如果特征的数量大到和样本数量差不多，则选用LR或者线性核的SVM；
  - 如果特征的数量小，样本的数量正常，则选用SVM+高斯核函数；
  - 如果特征的数量小，而样本的数量很大，则需要手工添加一些特征从而变成第一种情况。

3. 核技巧在支持向量机中的应用

我们注意到在线性支持向量机的对偶问题中，无论是目标函数还是决策函数（分离超平面）都只涉及输入实例与实例之间的内积．在对偶问题的目标函数(7.37)中的内积 $x_i \cdot x_j$ 可以用核函数 $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$ 来代替．此时对偶问题的目标函数成为

$$W(\alpha) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{N}\alpha_i \tag{7.67}$$

同样，分类决策函数中的内积也可以用核函数代替，而分类决策函数式成为

$$f(x) = \text{sign}\left(\sum_{i=1}^{N_s} a_i^* y_i \phi(x_i) \cdot \phi(x) + b^*\right) = \text{sign}\left(\sum_{i=1}^{N_s} a_i^* y_i K(x_i, x) + b^*\right) \tag{7.68}$$

这等价于经过映射函数 ϕ 将原来的输入空间变换到一个
入空间中的内积 $x_i \cdot x_j$ 变换为特征空间中的内积 $\phi(x_i) \cdot \phi($
从训练样本中学习线性支持向量机．当映射函数是非线性函数
核函数的支持向量机是非线性分类模型．

One use in **SVM** for **Non-linear data** classifier

Some Methods Can Be "Kernelized"

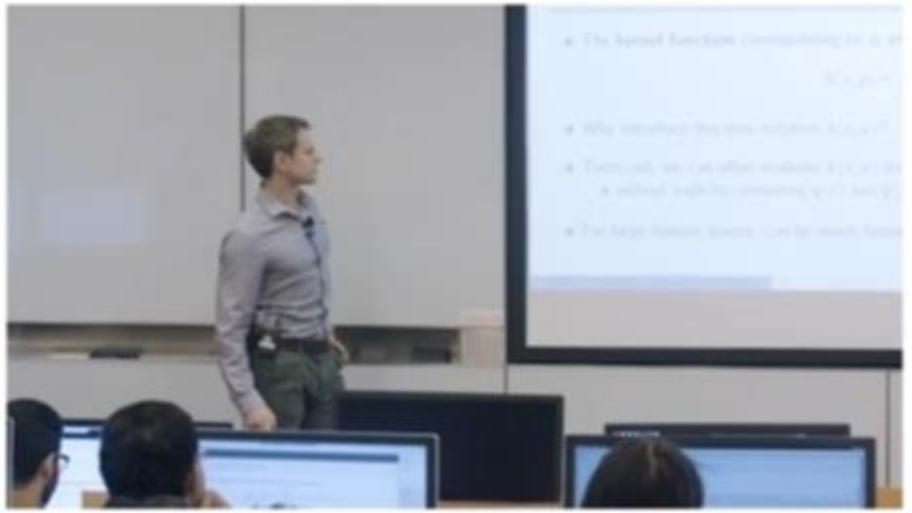
Definition

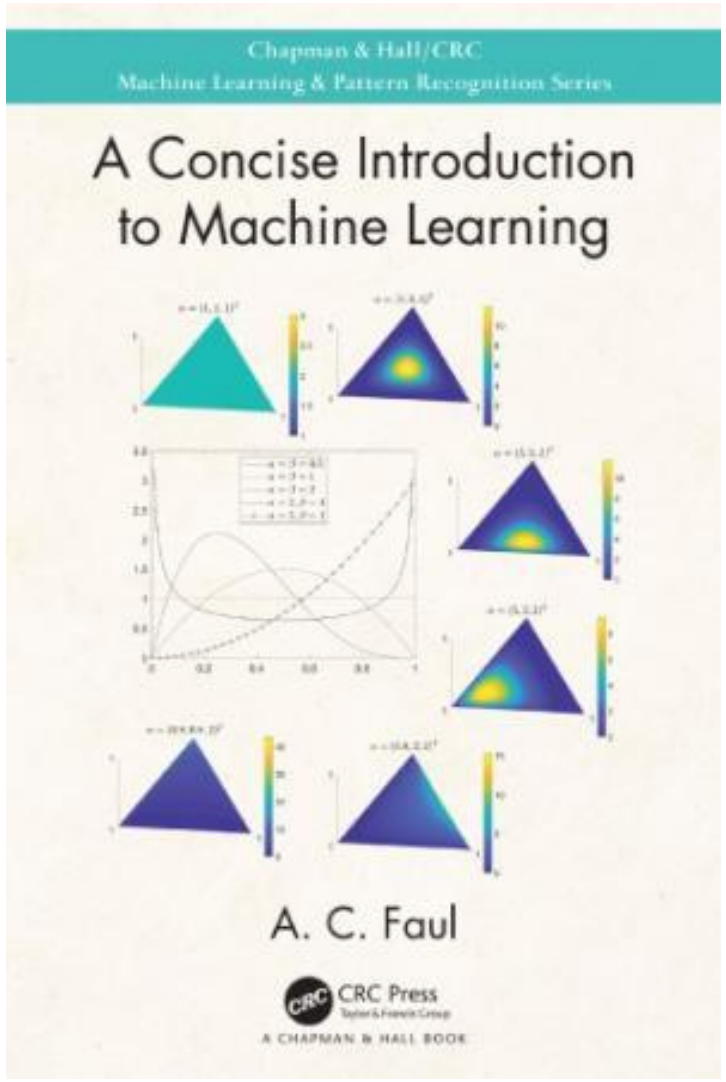
A method is **kernelized** if inputs only appear inside inner products: $\langle \psi(x), \psi(y) \rangle$ for $x, y \in \mathcal{X}$.

- The **kernel function** corresponding to ψ and inner product $\langle \cdot, \cdot \rangle$ is

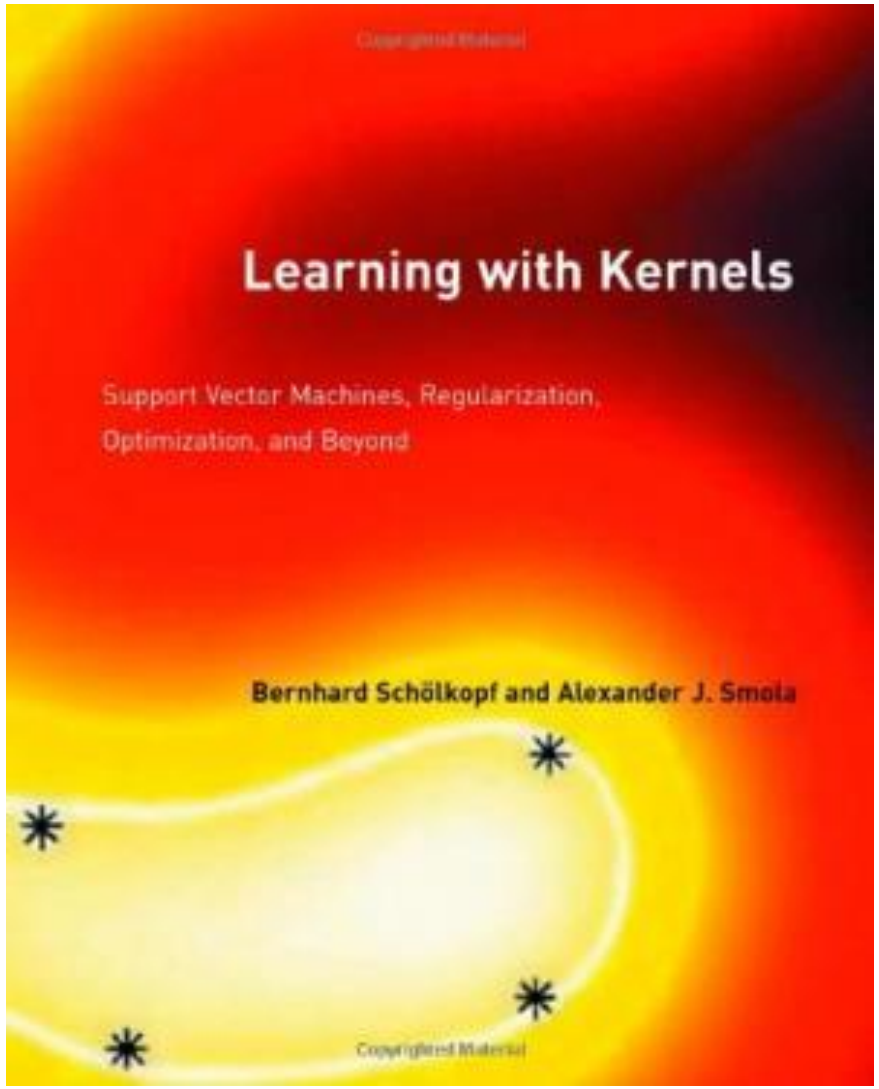
$$k(x, y) = \langle \psi(x), \psi(y) \rangle .$$

- Why introduce this new notation $k(x, y)$?
- Turns out, we can often evaluate $k(x, y)$ directly,
  - without explicilty computing $\psi(x)$ and $\psi(y)$.
- For large feature spaces, can be much faster.

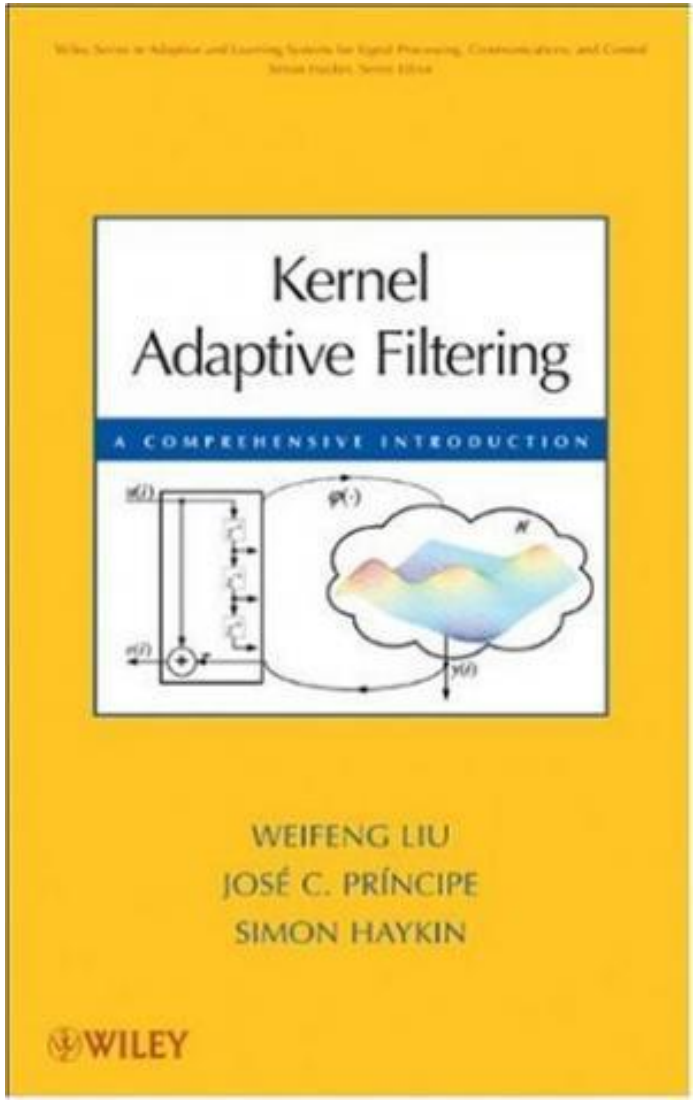




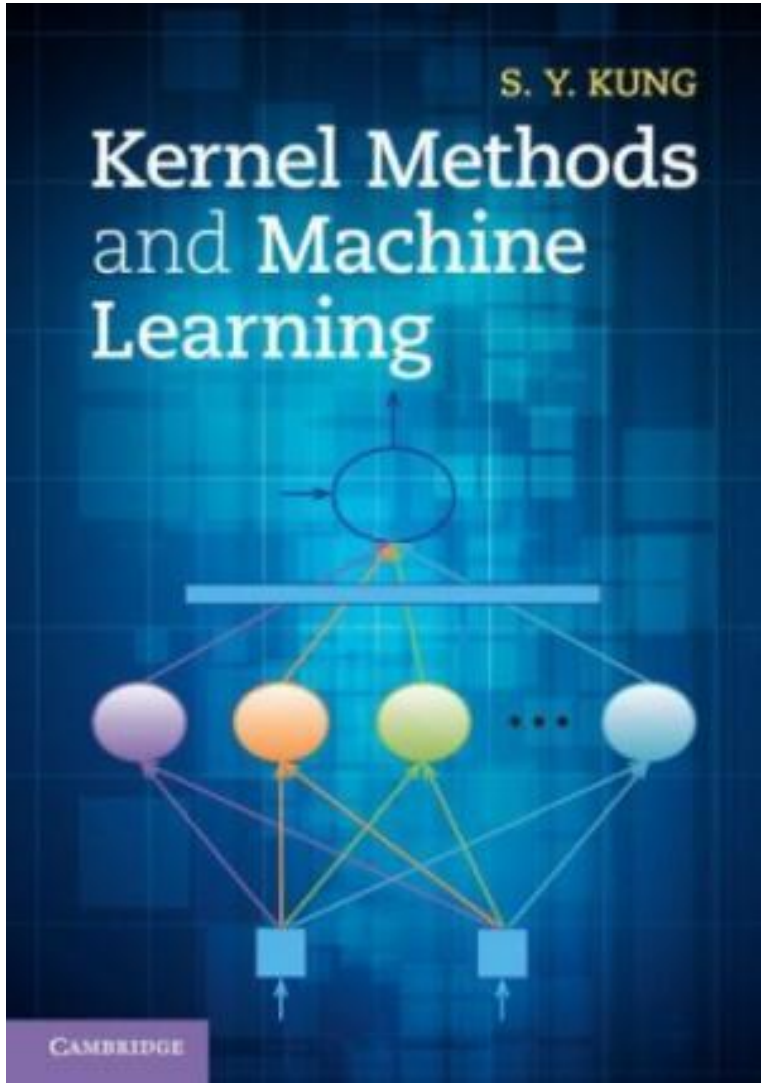
- A Concise Introduction to Machine Learning
- *Anita C. Faul*
- **2020**



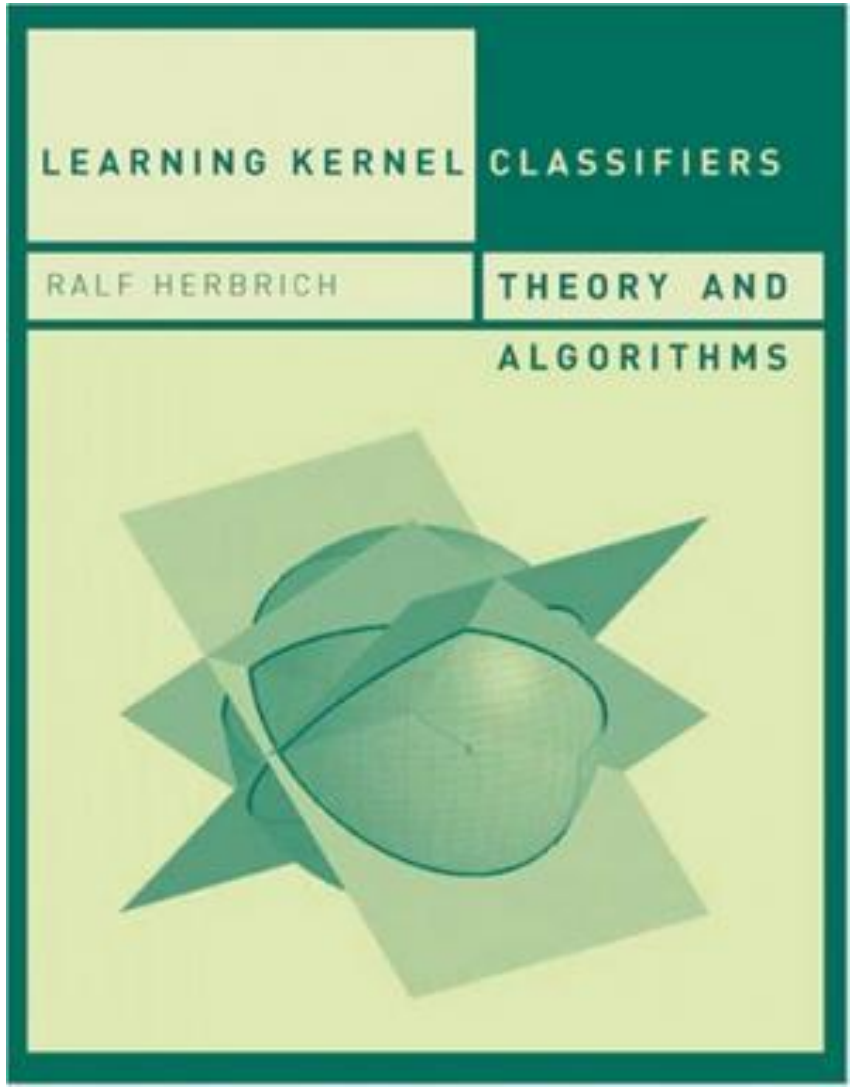
- Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)
- *Bernhard Schölkopf*, *Alexander J. Smola*
- 2001



- Kernel Adaptive Filtering: A Comprehensive Introduction (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)
- *Weifeng Liu*, *Jose C. Principe*, *Simon Haykin*
- **2010**



- Kernel Methods and Machine Learning
- *Kung S.Y.*
- **2014**



- Learning Kernel Classifiers: Theory and Algorithms
- *Ralf Herbrich*
- **2001**

来自机器学习的数据分析方法 – 基本概念和思想

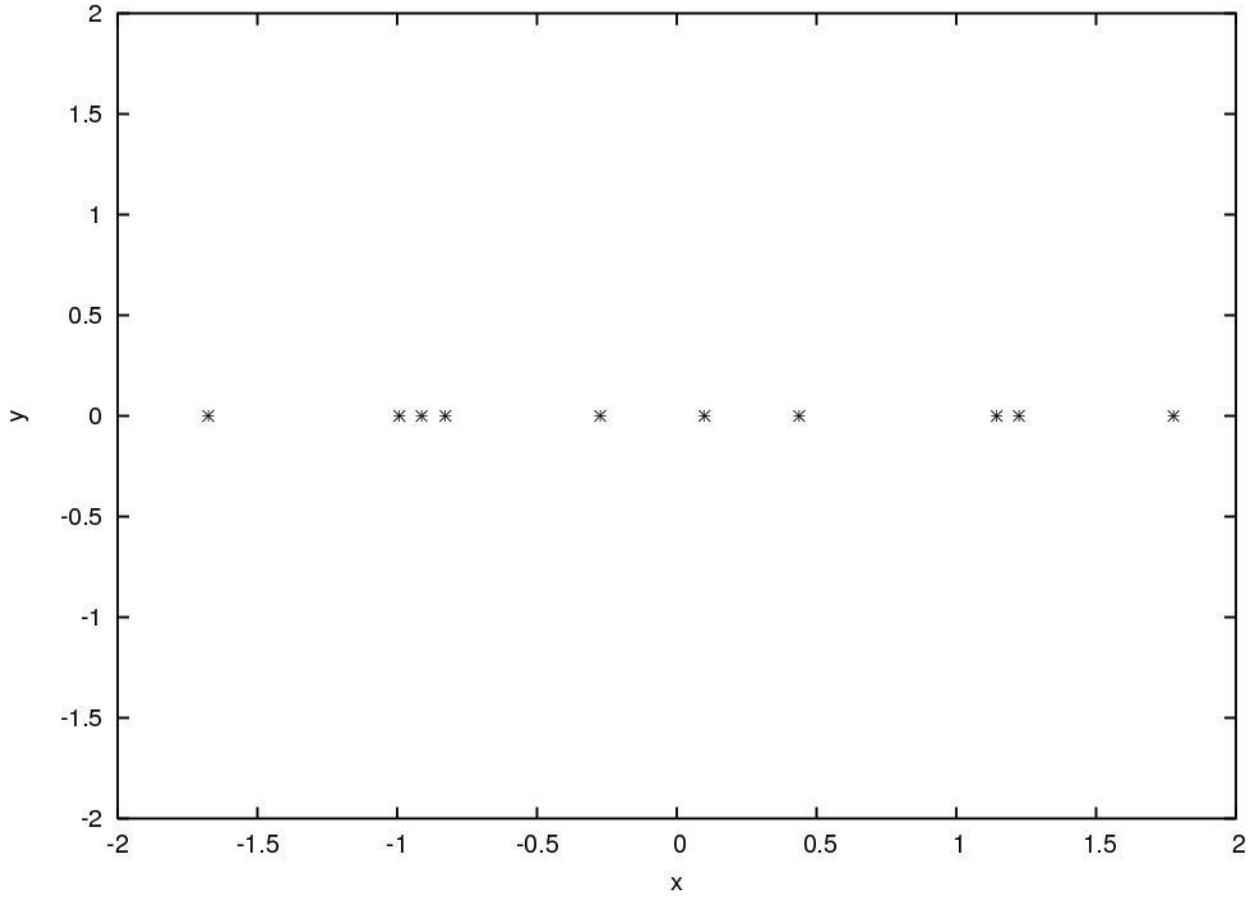
- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

$$\text{RowFeatureVector2} = \begin{pmatrix} -.677873399 & -.735178956 \end{pmatrix}$$

PCA for Dimension Reduction [降维]

Transformed Data (Single eigenvector)

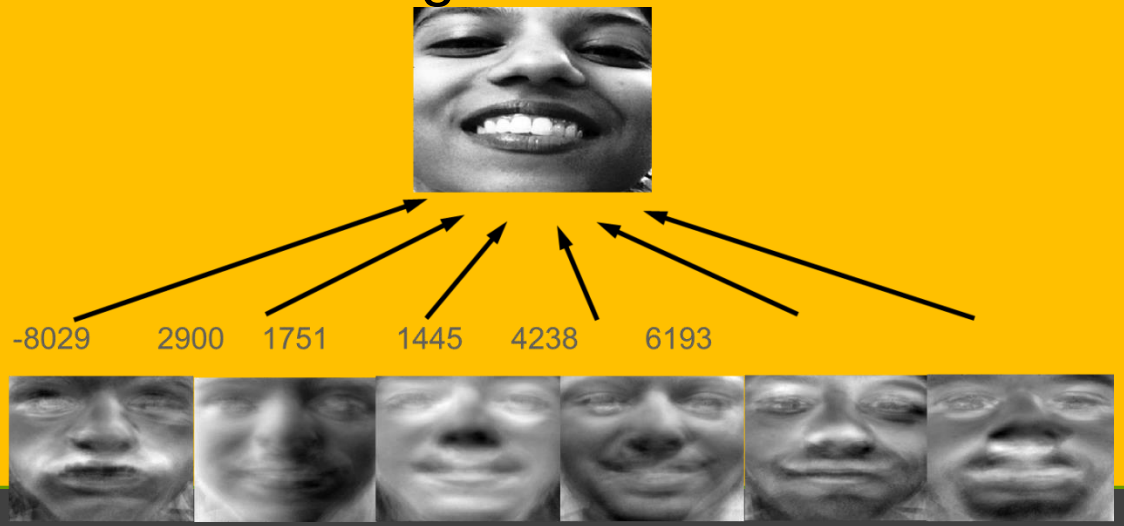
x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056



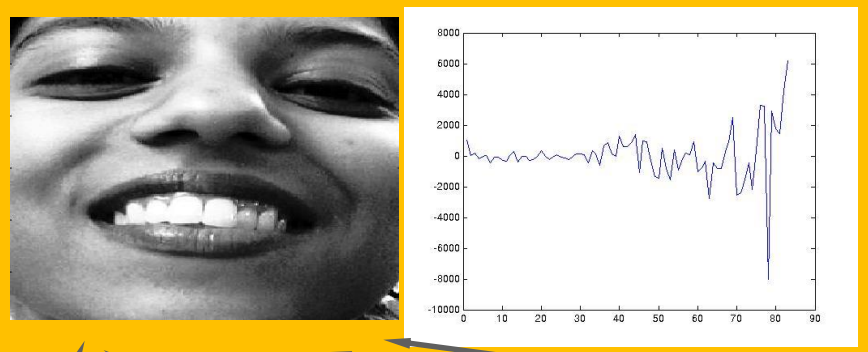
Eigenfaces

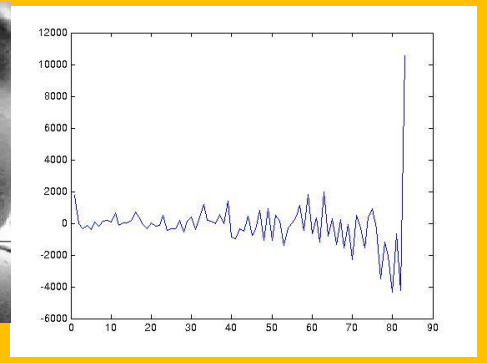
My work is a little related with this ☺

- **Developed in 1991 by M.Turk [PDF]**
  - Based on Principal Component Analysis (PCA)
  - Relatively simple, Fast, Robust
- **Idea:**
  - Think of a face as being a weighted combination of some “component” or “basis” faces
  - These basis faces are called eigenfaces



- These basis faces can be differently weighted to represent any face
- So we can use different vectors of weights to represent different faces

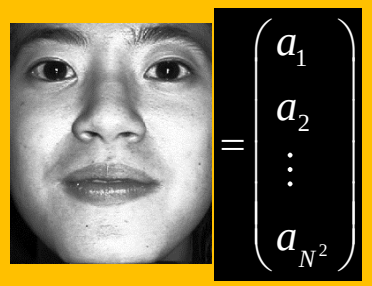




-8029 -1183 2900 -2088 1751 -4336 1445 -669 4238 -4221 6193 10549

Eigenfaces, the algorithm

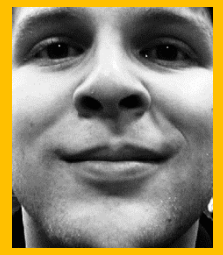
- The database



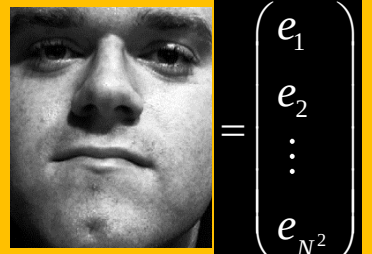
$$= \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N^2} \end{pmatrix}$$

$$= \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{N^2} \end{pmatrix}$$

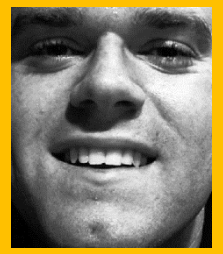
$$= \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N^2} \end{pmatrix}$$



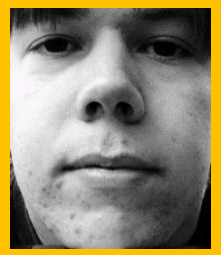
$$= \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{N^2} \end{pmatrix}$$



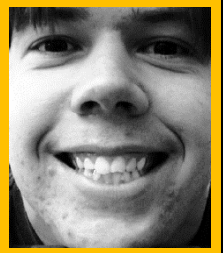
$$= \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_{N^2} \end{pmatrix}$$



$$= \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N^2} \end{pmatrix}$$



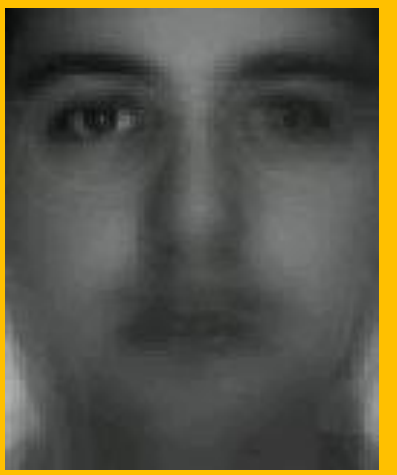
$$= \begin{pmatrix} g_1 \\ g_2 \\ \vdots \\ g_{N^2} \end{pmatrix}$$



$$= \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_{N^2} \end{pmatrix}$$

- We compute the average face

$$\vec{m} = \frac{1}{M}\begin{pmatrix} a_1 & +b_1 & +\cdots+h_1 \\ a_2 & +b_2 & +\cdots+h_2 \\ \vdots & \vdots & \vdots \\ a_{N^2} & +b_{N^2} & +\cdots+h_{N^2} \end{pmatrix}, \quad where\ M = 8$$



- Then subtract it from the training faces

$$\vec{a}_m = \begin{pmatrix} a_1 - m_1 \\ a_2 - m_2 \\ \vdots \\ a_{N^2} - m_{N^2} \end{pmatrix}, \quad \vec{b}_m = \begin{pmatrix} b_1 - m_1 \\ b_2 - m_2 \\ \vdots \\ b_{N^2} - m_{N^2} \end{pmatrix}, \quad \vec{c}_m = \begin{pmatrix} c_1 - m_1 \\ c_2 - m_2 \\ \vdots \\ c_{N^2} - m_{N^2} \end{pmatrix}, \quad \vec{d}_m = \begin{pmatrix} d_1 - m_1 \\ d_2 - m_2 \\ \vdots \\ d_{N^2} - m_{N^2} \end{pmatrix},$$

$$\vec{e}_m = \begin{pmatrix} e_1 - m_1 \\ e_2 - m_2 \\ \vdots \\ e_{N^2} - m_{N^2} \end{pmatrix}, \quad \vec{f}_m = \begin{pmatrix} f_1 - m_1 \\ f_2 - m_2 \\ \vdots \\ f_{N^2} - m_{N^2} \end{pmatrix}, \quad \vec{g}_m = \begin{pmatrix} g_1 - m_1 \\ g_2 - m_2 \\ \vdots \\ g_{N^2} - m_{N^2} \end{pmatrix}, \quad \vec{h}_m = \begin{pmatrix} h_1 - m_1 \\ h_2 - m_2 \\ \vdots \\ h_{N^2} - m_{N^2} \end{pmatrix}$$

- Now we build the matrix which is N^2 by M

$$A = \left[\vec{a}_m \ \vec{b}_m \ \vec{c}_m \ \vec{d}_m \ \vec{e}_m \ \vec{f}_m \ \vec{g}_m \ \vec{h}_m \right]$$

- The covariance matrix which is N^2 by N^2

$$Cov = AA^{\mathrm{T}}$$

- Find eigenvalues of the covariance matrix
  - The matrix is very large
  - The computational effort is very big
- We are interested in at most K eigenvalues
  - We can reduce the dimension of the matrix
- Compute another matrix which is M by M

$$L = A^{\mathrm{T}} A$$

- Find the K eigenvalues and eigenvectors
  - Eigenvectors of Cov and L are equivalent
- Build matrix V from the eigenvectors of L
- Eigenvectors of Cov are linear combination of image space with the eigenvectors of L

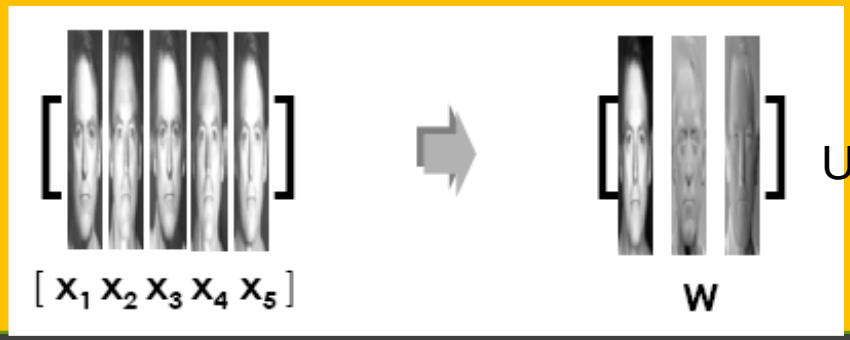
$$U = AV$$

V is Matrix of eigenvectors

$$A = \left[\vec{a}_m \ \vec{b}_m \ \vec{c}_m \ \vec{d}_m \ \vec{e}_m \ \vec{f}_m \ \vec{g}_m \ \vec{h}_m \right]$$

- Eigenvectors represent the variation in the faces ← **PCA**

A: collection of the training faces



U: Face Space / Eigen Space

- Eigenface of original faces

- Compute for each face its **projection onto** the face space

$$\Omega_1 = U^{\mathrm{T}}(\vec{a}_m), \quad \Omega_2 = U^{\mathrm{T}}(\vec{b}_m), \quad \Omega_3 = U^{\mathrm{T}}(\vec{c}_m), \quad \Omega_4 = U^{\mathrm{T}}(\vec{d}_m),$$

$$\Omega_5 = U^{\mathrm{T}}(\vec{e}_m), \quad \Omega_6 = U^{\mathrm{T}}(\vec{f}_m), \quad \Omega_7 = U^{\mathrm{T}}(\vec{g}_m), \quad \Omega_8 = U^{\mathrm{T}}(\vec{h}_m)$$

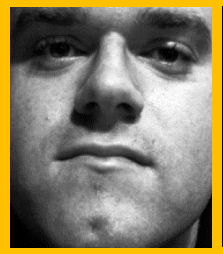
Details to recognize a face?

This threshold implies to recognize if a new face is corresponding to a recorded one or not

1. Compute the threshold

$$\theta = \frac{1}{2}\max\left\{\left\|\Omega_i - \Omega_j\right\|\right\} \text{ for } i, j = 1..M$$

2. For the new face



$$= \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_{N^2} \end{pmatrix}$$

   a. Subtract the average face from it

$$\vec{r}_m = \begin{pmatrix} r_1 - m_1 \\ r_2 - m_2 \\ \vdots \quad \vdots \\ r_{N^2} - m_{N^2} \end{pmatrix}$$

b. Compute its projection onto the face space U

$$\Omega = U^{\mathrm{T}}\left(\vec{r}_m\right)$$

c. Compute the distance in the face space between the face and all known faces

$$\varepsilon_i^2 = \left\|\Omega - \Omega_i\right\|^2 \quad for\ i = 1..M$$

d. Reconstruct the face from eigenfaces

$$\vec{s} = U\Omega$$

e. Compute the distance between the face and its reconstruction

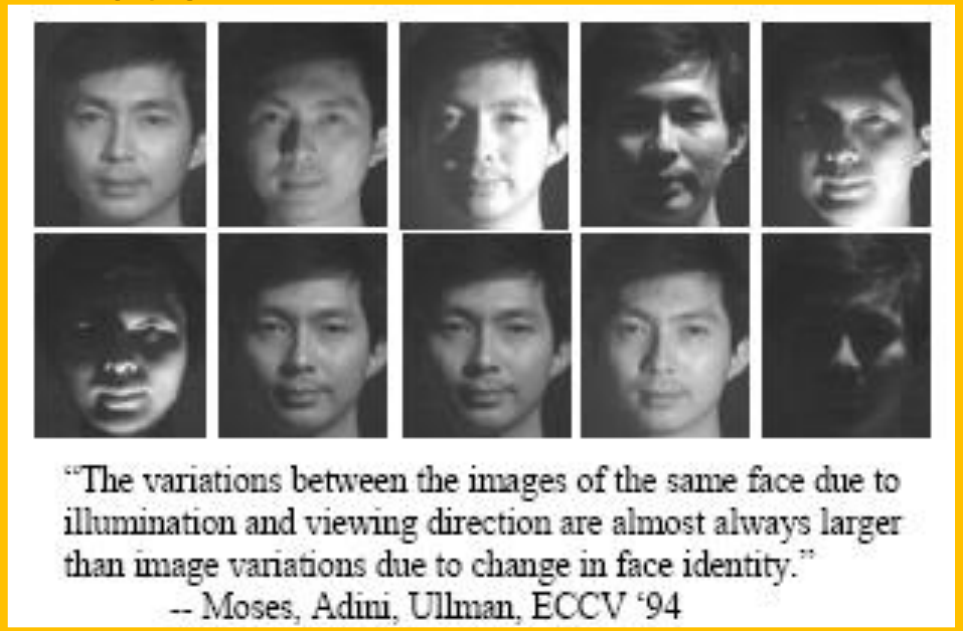
$$\xi^2 = \left\|\vec{r}_m - \vec{s}\right\|^2$$

f. Distinguish between

- If $\xi \geq \theta$ then it's not a face; the distance between the face and its reconstruction is larger than threshold
- If $\xi < \theta \; and \; \min\{\varepsilon_i\} < \theta$ then it's a new face
- If $\xi < \theta \; and \; \varepsilon_i \geq \theta, (i = 1..M)$ then it's a known face because the distance in the face space between the face and all known faces is larger than threshold

Problems with eigenfaces

- Different illumination



"The variations between the images of the same face due to illumination and viewing direction are almost always larger than image variations due to change in face identity."
-- Moses, Adini, Ullman, ECCV '94

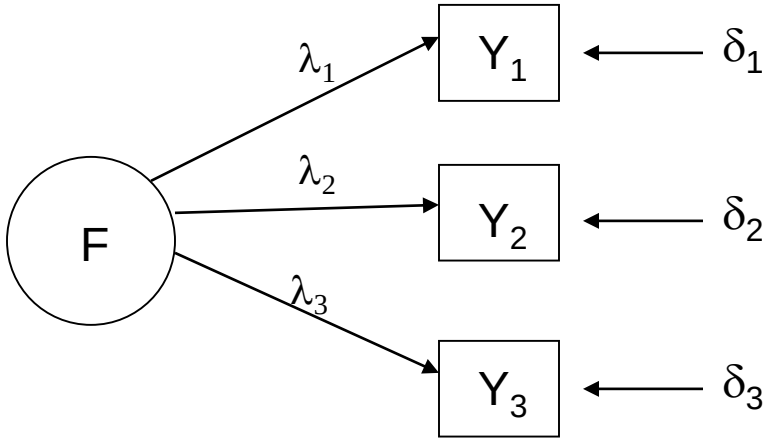
- Different head pose
- Different alignment
- Different facial expression
- . . .

FA: Factor Analysis

- **Your responsibility ☺**

Well-used latent variable models

Latent variable scale	Observed variable scale	
	Continuous	Discrete
Continuous	**Factor analysis** LISREL	Discrete FA IRT (item response)
Discrete	Latent profile Growth mixture	Latent class analysis, regression



$$Y_1 = \lambda_1 F + \delta_1$$
$$Y_2 = \lambda_2 F + \delta_2$$
$$Y_3 = \lambda_3 F + \delta_3$$

- The factor F is <u>not observed</u>; only Y_1, Y_2, Y_3 are observed
- δ_i represent variability in the Y_i NOT explained by F
- Y_i is a <u>linear</u> function of F and δ_i

with n variables and m factors

$$\mathbf{Y_{nx1} = \Lambda_{nxm} F_{mx1} + \delta_{nx1}}$$

$$\begin{bmatrix} Y_1 \\ \vdots \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} \lambda_{11} & \cdots & \cdots & \lambda_{1m} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \lambda_{n1} & \cdots & \cdots & \lambda_{nm} \end{bmatrix}_{n \times m} \begin{bmatrix} F_1 \\ \vdots \\ F_m \end{bmatrix}_{m \times 1} + \begin{bmatrix} \delta_1 \\ \vdots \\ \vdots \\ \delta_n \end{bmatrix}_{n \times}$$

假设某一社会经济系统问题，其主要特性可用4个指标表示，它们分别是生产、技术、交通和环境。其相关矩阵为：

$$R=\begin{bmatrix}1 & 0.64 & 0.29 & 0.1\\ 0.64 & 1 & 0.7 & 0.3\\ 0.29 & 0.7 & 1 & 0.84\\ 0.1 & 0.3 & 0.84 & 1\end{bmatrix}$$

相应的特征值、占总体百分比和累计百分比如下表：

特征值、占总体百分比和累计百分比表

序　号	特 征 值	占总体百分比×100	累计百分比×100
1	2.49	62.25	62.25
2	1.13	28.25	90.50
3	0.35	8.74	99.24
4	0.031	0.76	100.00

对应特征值的特征向量矩阵为：

$$U=\begin{bmatrix}0.38 & 0.67 & 0.62 & -0.14\\ 0.54 & 0.36 & -0.61 & 0.46\\ 0.59 & -0.3 & -0.2 & -0.72\\ 0.47 & -0.58 & 0.45 & 0.50\end{bmatrix}$$

假如要求所取特征值反映的信息量占总体信息量的90%以上，则从累计特征值所占百分比看，只需取前两项即可。也就是说，只需取两个主要因子。对应于前两列特征值的特征向量，可求的其因子载荷矩阵A为：

$$A=\begin{bmatrix}0.60 & 0.71\\ 0.85 & 0.38\\ 0.93 & -0.32\\ 0.74 & -0.40\end{bmatrix}$$

于是，该问题的因子模型为：

$$X_1 = 0.60f_1 + 0.71f_2$$

$$X_2 = 0.85f_1 + 0.38f_2$$

$$X_3 = 0.93f_1 - 0.32f_2$$

$$X_4 = 0.74f_1 - 0.40f_2$$

由以上可以看出，两个因子中，f_1是全面反映生产、技术、交通和环境的因子，而f_2却不同，它反映了对生产和技术这两项增长有利，而对交通和环境增长不利的因子。也就是说，按照原有统计资料得出的相关矩阵分析的结果是如果生产和技术都随f_2增长了，将有可能出现交通紧张和环境恶化的问题，f_2反映了这两方面的相互制约状况。

CCA – Canonical Correlation Analysis

CCA seeks to identify and quantify the associations between two sets of variables.

Given two random vectors $\mathbf{X} \in \mathbb{R}^r$ and $\mathbf{Y} \in \mathbb{R}^s$ ($r = s$ or $r \neq s$), consider *linear dimension reduction* of each of two random vectors,

$$\xi = \mathbf{g}'\mathbf{X} = g_1 X_1 + \cdots + g_r X_r,$$
$$\omega = \mathbf{h}'\mathbf{Y} = h_1 Y_1 + \cdots + h_s Y_s.$$

CCA finds the random variables (ξ, ω) or the projection vectors $(\mathbf{g}, \mathbf{h})$ that give *maximal correlation* between ξ and ω,

$$\mathrm{Corr}(\xi, \omega) = \frac{\mathrm{Cov}(\xi, \omega)}{\sqrt{\mathrm{Var}(\xi)\mathrm{Var}(\omega)}}$$

- CCA formulation

$$\underset{\mathbf{u}\in R^p,\mathbf{v}\in R^q}{\operatorname{argmax}} \quad \frac{\mathbf{u}^T\mathbf{X}^T\mathbf{Y}\mathbf{v}}{\sqrt{(\mathbf{u}^T\mathbf{X}^T\mathbf{X}\mathbf{u})(\mathbf{v}^T\mathbf{Y}^T\mathbf{Y}\mathbf{v})}}$$

  - **X** is n by p: n samples in p-dimensional space
  - **Y** is n by q: n samples in q-dimensional space
  - The n samples are ***paired*** in **X** and **Y**
- How to solve? Generalized eigenproblems !

$$\mathbf{X}^T\mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{Y}^T\mathbf{X}\mathbf{u} = \lambda\mathbf{X}^T\mathbf{X}\mathbf{u}$$

$$\mathbf{Y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}\mathbf{v} = \lambda\mathbf{Y}^T\mathbf{Y}\mathbf{v}$$

Motivation - Cocktail Party Problem

- Simple scenario:
  - Two people speaking simultaneously in a room.
  - Speeches are recorded by two microphones in separate locations.

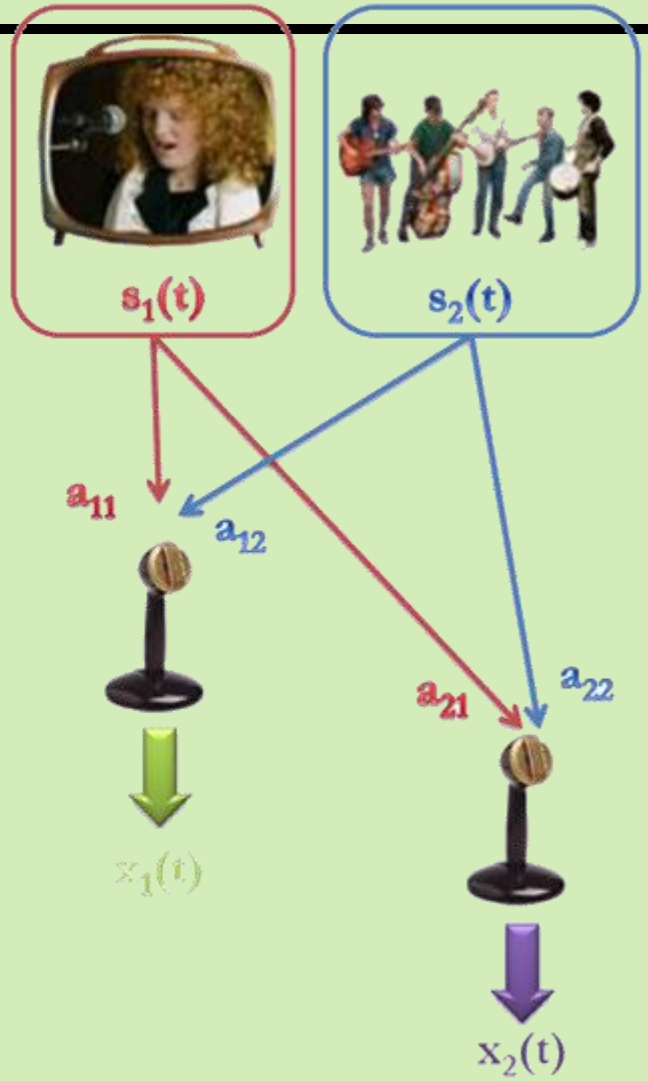
- Let $\mathbf{s_1(t)}$, $\mathbf{s_2(t)}$ be the speech signals emitted by the two speakers.
- Recorded time signals, by the two microphones, are denoted by $\mathbf{x_1(t)}$, $\mathbf{x_2(t)}$.
- The recorded time signals can be expressed as a linear equation:

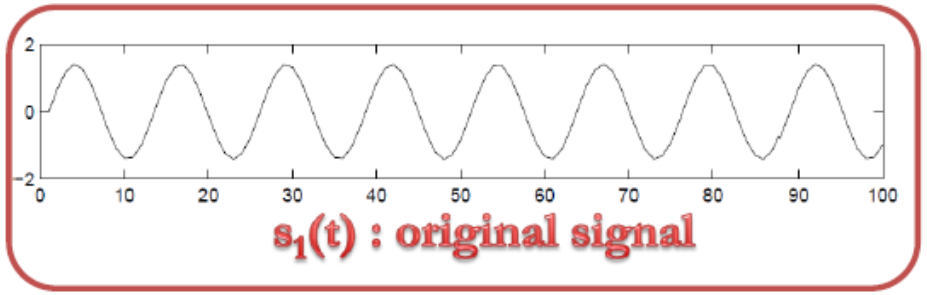
$$x_1(t) = a_{11}s_1(t) + a_{12}s_2(t)$$

$$x_2(t) = a_{21}s_1(t) + a_{22}s_2(t)$$

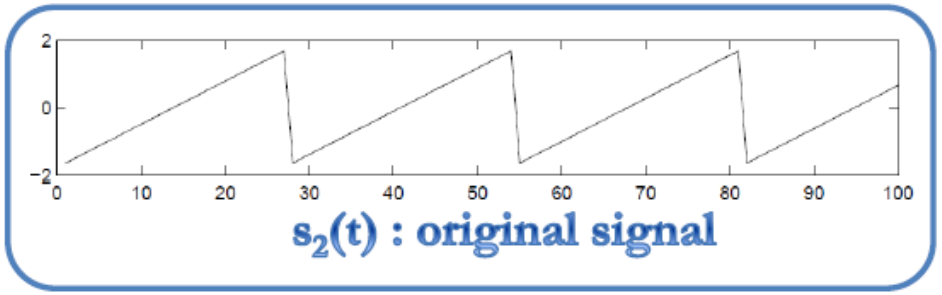
where parameters in matrix **A** depend on distances of the microphones to the speaker, along with other microphone properties

- Assume $\mathbf{s_1(t)}$ and $\mathbf{s_2(t)}$ are *statistically independent.*

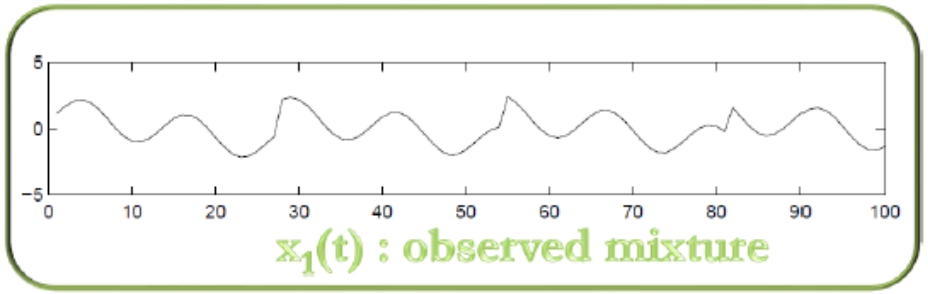




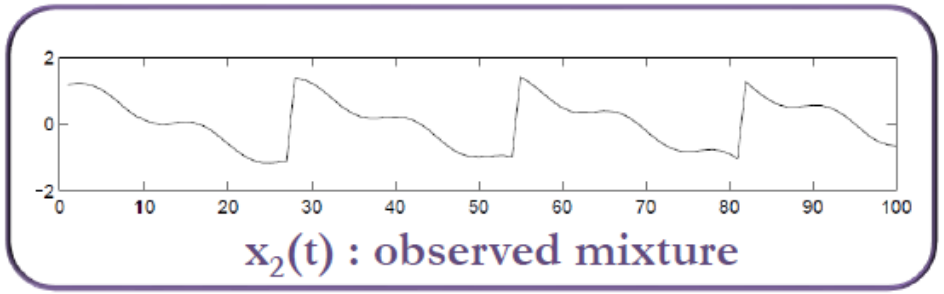
$s_1(t)$: original signal



$s_2(t)$: original signal

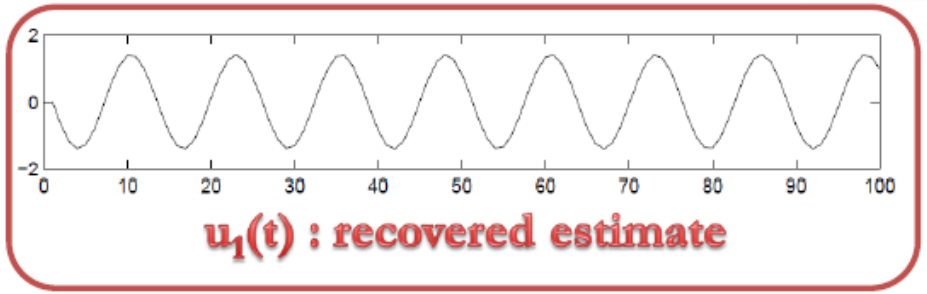


$x_1(t)$: observed mixture

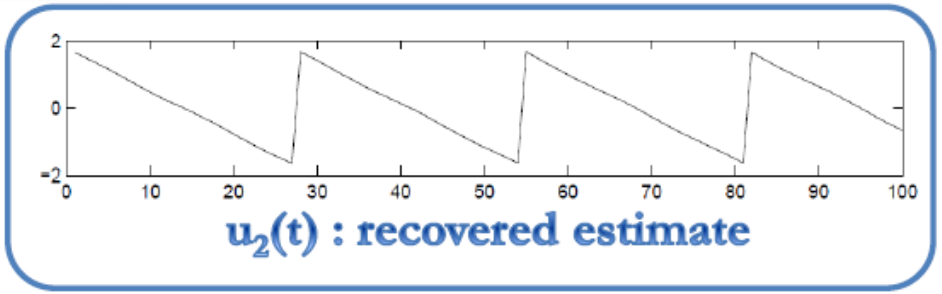


$x_2(t)$: observed mixture

ICA



$u_1(t)$: recovered estimate



$u_2(t)$: recovered estimate

The original signals were very accurately estimated, up to multiplicative signs

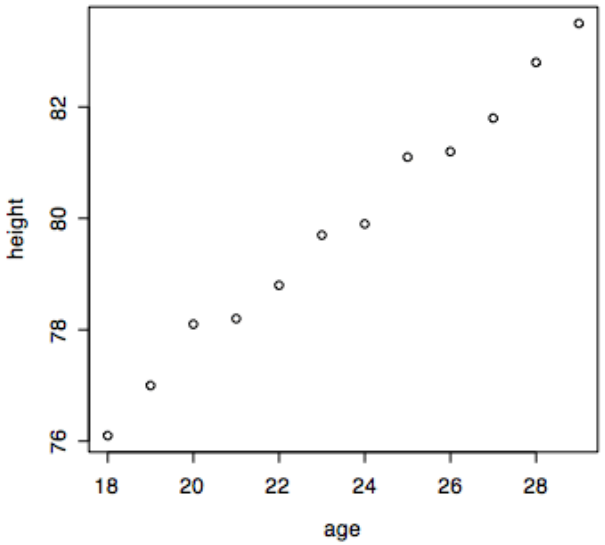
来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

Regression

- **Regression Curve – curve with best possible fit for data; describes how y changes with x**
  - Single variable regression
    - Describes relationship between **x** and estimated means of **y**

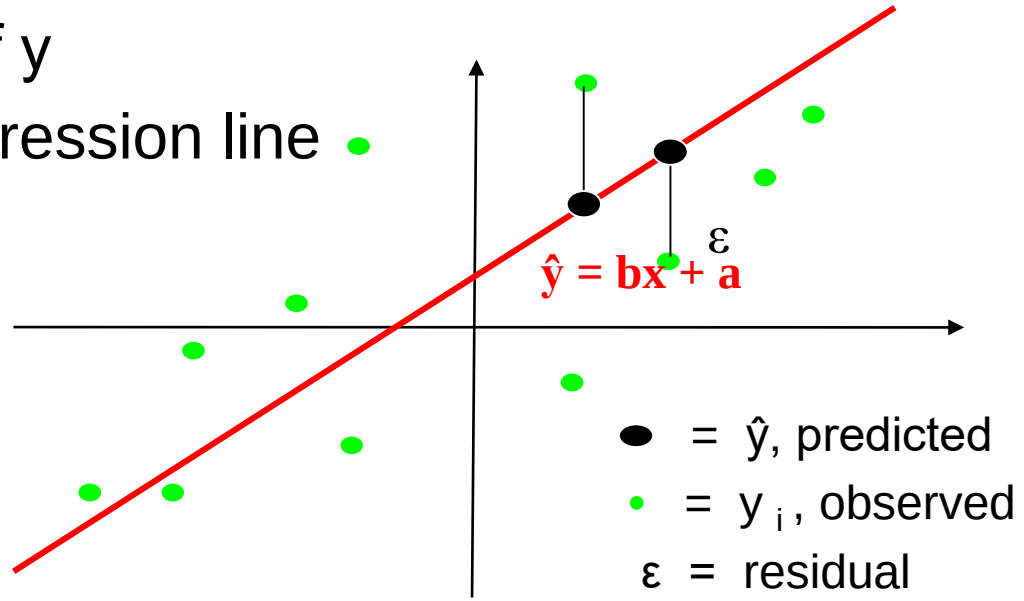
$$\checkmark y = \beta_0 + \beta_1 x + \varepsilon$$



  - Multivariable regression

$$y = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_p x_{pi} + \varepsilon_i$$

- **Here, $\hat{y} = \beta_0 + \beta_1 x$**
  - $\hat{y}$: predicted value of y
  - β_1 : slope [斜率] of regression line
  - β_0 : intercept [截距]



  - Residual error (**ε**): Difference between obtained and predicted values of y (i.e. y- ŷ).
    - $\varepsilon \sim N(0, \sigma^2)$

- **How to compute those two parameters – β_0, β_1**
  - We have many pairs – (x_i, y_i)
  - That means we have many equations

$$\begin{cases} y_1 = \hat{\beta}_0 + \hat{\beta}_1 x_1 \\ y_2 = \hat{\beta}_0 + \hat{\beta}_1 x_2 \\ y_3 = \hat{\beta}_0 + \hat{\beta}_1 x_3 \\ \cdots \\ y_n = \hat{\beta}_0 + \hat{\beta}_1 x_n \end{cases}$$

  - How to compute $\hat{\beta}_0, \hat{\beta}_1$ with so many equations?

Table 1 Average Income & Consumption unit: Yuan

Year	Average Income	Average consumption
1981	393.8	249
1982	419.14	267
1983	460.86	289
1984	544.11	329
1985	668.29	406
1986	737.73	451
1987	859.97	513
1988	1068.8	643
1989	1169.2	690
1990	1250.7	713
1991	1429.5	803
1992	1725.9	947
1993	2099.5	1148

Could you recognize this is an OP with unconstrained?

- **LSM (Least Square Meth… that**
  - $argmin(\widehat{\beta_0}, \widehat{\beta_1}) = \sum_{i=1}^{n}(y_i - \hat{y})^2$

- **We have:**
  - We are trying to **minimize the squared distance** (hence the "**least squares**") between the observations themselves and the predicted values, or (also called the "residuals", or left-over unexplained variability)
  - $\frac{d}{d\widehat{\beta_1}} \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right)\right)^2 = 0$

  ➔ $2\left(\sum_{i=1}^{n}\left(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i\right)(-x_i)\right) = 0$

We get

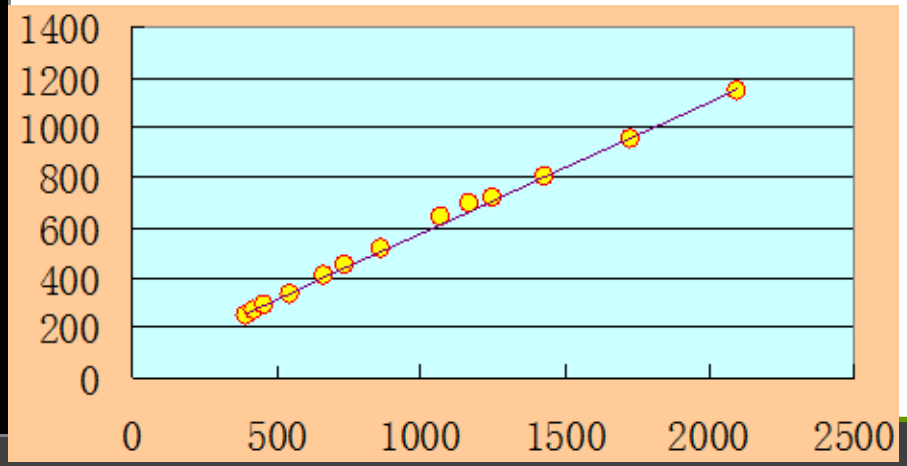
$$\begin{cases} \hat{\beta}_1 = \dfrac{n\sum_{i=1}^{n} x_i y_i - \left(\sum_{i=1}^{n} x_i\right)\left(\sum_{i=1}^{n} y_i\right)}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)} \\ \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} \end{cases}$$

$$\begin{cases} \hat{\beta}_1 = \dfrac{13 \times 9156173.99 - 12827.5 \times 7457}{13 \times 16073323.77 - (12827.5)^2} \\ \hat{\beta}_0 = 573.61538 - 0.52638 \times 986.73077 \end{cases}$$

$$\begin{cases} \hat{\beta}_1 = 0.526378 \\ \hat{\beta}_0 = 54.2229 \end{cases}$$

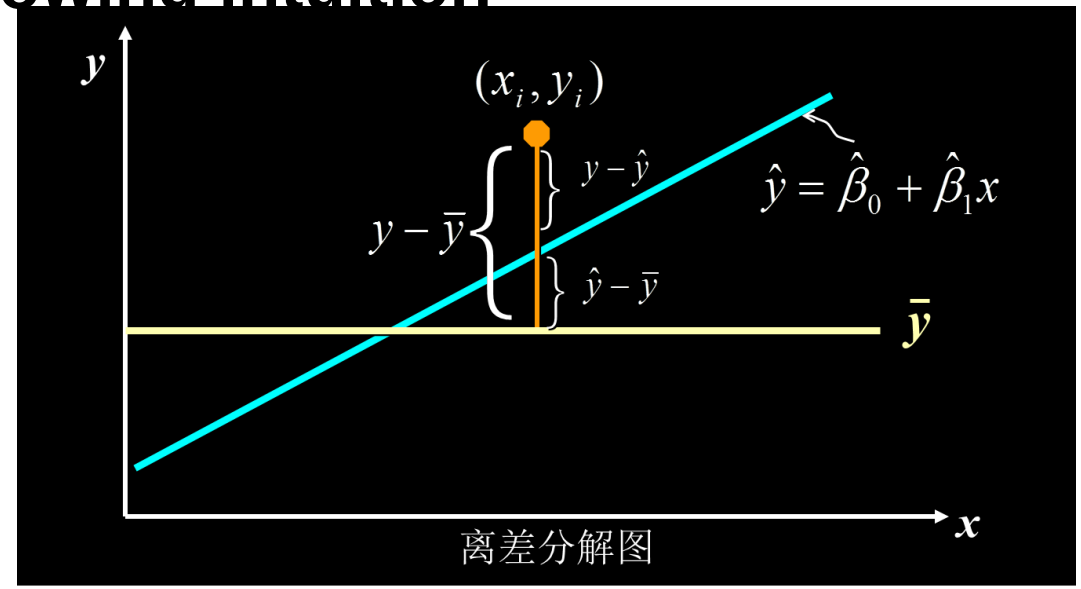
The regression line for “Table 1” is

$\hat{y} = 54.22286+0.52638x$



Modeling is art

- We have the following intuition



离差分解图

$$\sum_{i=1}^{n}(y_i-\bar{y})^2=\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2+\sum_{i=1}^{n}(y_i-\hat{y})^2$$

总变差平方和（*SST*）　回归平方和（*SSR*）　残差平方和（*SSE*）

Significance (显著性) of the regression equation

- We have
  - **SST** (Sum of Squared Total) $\sum_{i=1}^{n}(y_i - \bar{y})^2$
  - **SSR** (Regression) $\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$
  - **SSE** (Error) $\sum_{i=1}^{n}(y_i - \hat{y})^2$
- And they have following relationship
  - $SST = SSR + SSE$
- The significance evaluation problem becomes to check if SST and SSR are correlated or not

$$r^2 = \frac{SSR}{SST} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}$$

  - If correlated ($r^2 \rightarrow 1$), this means the regression equation could cover most given data records

- **We have the following ration to evaluate the intimacy of the regression function**

$$r^2 = \frac{SSR}{SST} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}$$

- **Back to "Table 3"**
  - $r^2 = \frac{946507.2207}{946766.9231} = 0.999725696$
  - ➔ $r = \sqrt{0.999725696} = 0.999862838$

Significance of the independent

- We can use the value of co
  to evaluate the significance
  can help us to select the signific
- It's based on the distribution of β_1, and we need

For single variable regression, this is same as to evaluate the correlation between x and y

$$t = \frac{\hat{\beta}_1}{S_{\hat{\beta}_1}} \sim t(n-2)$$

Where $S_{\hat{\beta}_1} = \frac{S_y}{\sqrt{\sum (x_i - \bar{x})^2}}$

- $|t| > t_{\alpha/2}$, Reject H_0; $|t| < t_{\alpha/2}$, Accept H_0
  - H_0: $\beta_1 = 0$ (x is not significant with y)
  - H_1: $\beta_1 \neq 0$ (x is significant with y)

http://www.real-statistics.com/multiple-regression/testing-significance-extra-variables-regression-model/

Do you feel familiar with the Group Testing based on F-distribution?

$$F = \frac{(SS'_E - SS_E)/m}{MS_E} \sim F(m, df_E)$$

- E.g. suppose we consider the multiple regression model

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4 + b_5x_5$$

  - want to determine whether b_3, b_4 and b_5 add significant benefit to the model (i.e. whether the **reduced** model y = b_0 + b_1x_1 + b_2x_2 is significantly no worse than the **complete** model).
    - ✓ The **null hypothesis** H_0: $b_3 = b_4 = b_5 = 0$ is tested using the statistic F as described in Property 1 where $m = 3$ and SS'_E references the reduced model, while SS_E, MS_E and df_E refer to the complete model.

来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

What is Sparsity?

Assume full rank, $N > d$

$$\min_{x} \left(Ax - b \right)^2$$

$$\underbrace{\begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots\cdots & a_N \\ | & | & & | \end{bmatrix}}_{N} \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} | \\ b \\ | \end{bmatrix} \in R^d$$

$\Longrightarrow$

Many x can achieve

$$\min_{x} \left(Ax - b \right)^2 = 0$$

$\Downarrow$

Which do you want?

$\Longleftarrow$

Choose the x with the least nonzero component

$$\arg\min_{x} \left\| x \right\|_0 , \text{ s.t. } \left(Ax - b \right)^2 = 0$$

Why Sparsity?

核磁共振成像（Nuclear Magnetic Resonance Imaging，简称NMRI），又称自旋成像（spin imaging），也称磁共振成像（Magnetic Resonance Imaging，简称MRI）

- **The more concise, the more better**
- **In some domain, there naturally exists a sparse latent vector that controls the data we saw. (ex. MRI, music)**

$$\begin{bmatrix} | \\ \boldsymbol{b} \\ | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots\cdots & \boldsymbol{a}_d \\ | & | & & | \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ x_i \\ \vdots \\ x_j \\ \vdots \\ 0 \end{bmatrix} + (\text{noise})$$

A *k*-sparse domain means that each ***b*** can be constructed by a ***x*** vector with at most *k* nonzero element

- **In some domain, samples from the same class have the sparse property.**
- **The domain can be learned.**

MRI

CS的研究背景—问题提出

2006年，由D. Donoho(美国科学院院士)、E. Candes(Ridgelet, Curvelet创始人)及华裔科学家T. Tao(2006年菲尔兹奖获得者，2008年被评为世界上最聪明的科学家)等人提出了一种新的信息获取指导理论，即，压缩感知。该理论指出：对可压缩的信号可通过远低于Nyquist标准的方式进行采样数据，仍能够精确地恢复出原始信号。该理论一经提出，就在信息论、信号/图像处理、医疗成像、模式识别、地质勘探、光学/遥感成像、无线通信，雷达探测，生物传感，集成电路分析，图像超分辨率重建等领域受到高度关注，并被美国科技评论评为2007年度十大科技进展。D. Donoho 因此还获得了2008年IEEE学会最佳论文奖。

《Robust Uncertainty Principles: Exact Signal Reconstruction From Highly Incomplete Frequency Information》 IEEE Transactions on Information Theory, Feb. 2006

《Quantitative Robust Uncertainty Principles and Optimally Sparse Decompositions》 Foundations of Computational Mathematics, Apr. 2006

《Near Optimal Signal Recovery From Random Projections: Universal Encoding Strategies?》 IEEE Transactions on Information Theory, Dec. 2006

Dave Donoho

Emmanuel Candes

Terence Tao

天津大学

How to Get The Sparse Solution?

- There is no algorithm other than exhaustively searching to solve:

$$\boldsymbol{x}^{*} = \arg\min_{x} \|\boldsymbol{x}\|_0 \text{, s.t. } \left(A\boldsymbol{x} - \boldsymbol{b}\right)^2 = 0$$

- While in some situations (ex. special form of A), the solution of l_1 minimization approaches the one of l_0 minimization

$$\boldsymbol{x}^{***} = \arg\min_{x} \|\boldsymbol{x}\|_1 = \sum_{n=1}^{N} \left| x^{(n)} \right| \text{, s.t. } \left(A\boldsymbol{x} - \boldsymbol{b}\right)^2 = 0$$

$$\boldsymbol{x}^{***} = \boldsymbol{x}^{*}$$

- Have to use L_1 to approximate

$$\hat{\mathbf{a}}_1 = \arg\min_{\mathbf{a}} \{ \|\mathbf{a}\|_1 \text{ s.t. } \mathbf{y} = \mathbf{\Phi\Psi a} \}$$

Sum of absolute values

- Convexity: tractable problem
  - Solvable by Linear (**LP**) or Second-order programming (**SOP**)
  - For C > 0, $\hat{\mathbf{a}}_1 = \hat{\mathbf{a}}$ if:

$$M \geq C \cdot \mu^2(\mathbf{\Phi}, \mathbf{\Psi}) \cdot K \cdot logN$$

- Noisy data: Solve the **LASSO** problem

$$\hat{\mathbf{a}}_1(\epsilon) = \arg\min_{\mathbf{a}} \{ \|\mathbf{a}\|_1 \text{ s.t. } \|\mathbf{y} - \mathbf{\Phi}\mathbf{\Psi}\mathbf{a}\|_2 \leq \epsilon \}$$

- Convex problem solvable via 2nd order cone programming (**SOCP**)
  - If $\delta_{2K} < \sqrt{2} - 1$, then:

$$\|\hat{\mathbf{a}}_1(\epsilon) - \hat{\mathbf{a}}\|_2 \leq C\|\hat{\mathbf{a}} - \hat{\mathbf{a}}_K\|_1/\sqrt{K} + C'\epsilon$$

Dictionary generation

- In preceding sections, we generally assume that the (over-complete) bases *A* is existed and known
- However in practice, we usually need to build it:
  - Wavelet + Fourier + Haar + ……
  - Learning based on data
- How to learn?

Given a training set $\left\{ \boldsymbol{b}^{(i)} \in R^d \right\}_{i=1}^{N}$, form B as $B = \left[\boldsymbol{b}^{(1)} \ \boldsymbol{b}^{(2)} \ \ldots\ldots \ \boldsymbol{b}^{(N)} \right]$

$$\left(A^*, X^* \right) = \arg\min_{A,\, X} \left\| B - AX \right\|_F^2 + \lambda \left\| X \right\|_1, \text{ where } X \left[\boldsymbol{x}^{(1)} \ \boldsymbol{x}^{(2)} \ \ldots\ldots \ \boldsymbol{x}^{(N)} \right]$$

- May result in over-fitting

An important issue

- When using sparse representation as a way of feature extraction, you may wonder, even if there exists the sparsity property in the data, does sparse feature really come up with better results? Does it contain any semantic meaning?
- Successful areas:
  - Face recognition
  - Digit recognition
  - Object recognition (with carful design):

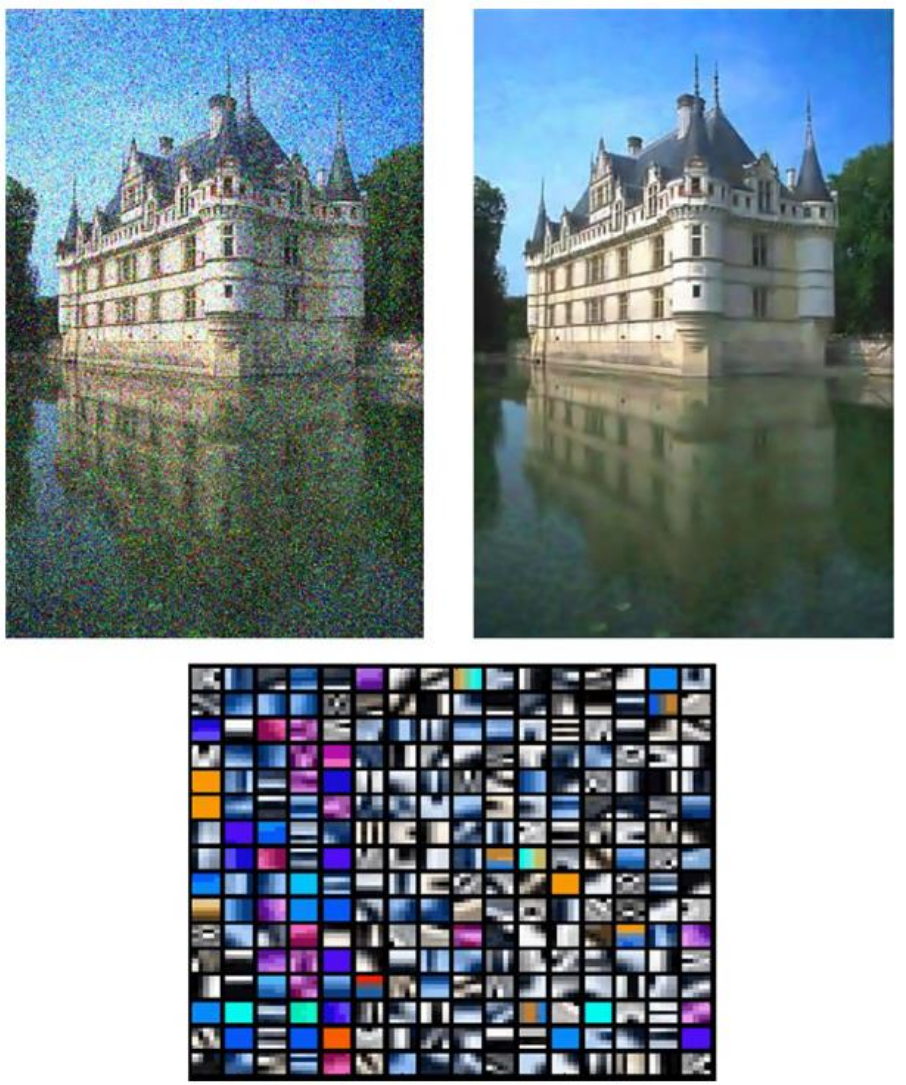
    Ex. K-means → Sparse representation

De-noising

Learn a patch dictionary.
For each patch, compute the sparse representation then use it to reconstruct the patch.

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 + \lambda \|A\boldsymbol{x} - \boldsymbol{b}\|_1$$

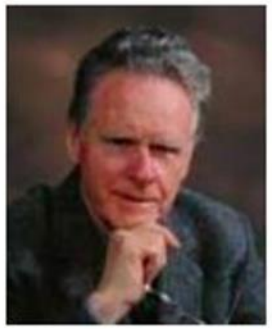
$$\boldsymbol{b} = A\boldsymbol{x}^*$$



来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估

EM



1977年由Dempster等人 总结 提出

由Arthur P. Dempster，Nan Laird和Donald Rubin 1977年正式提出

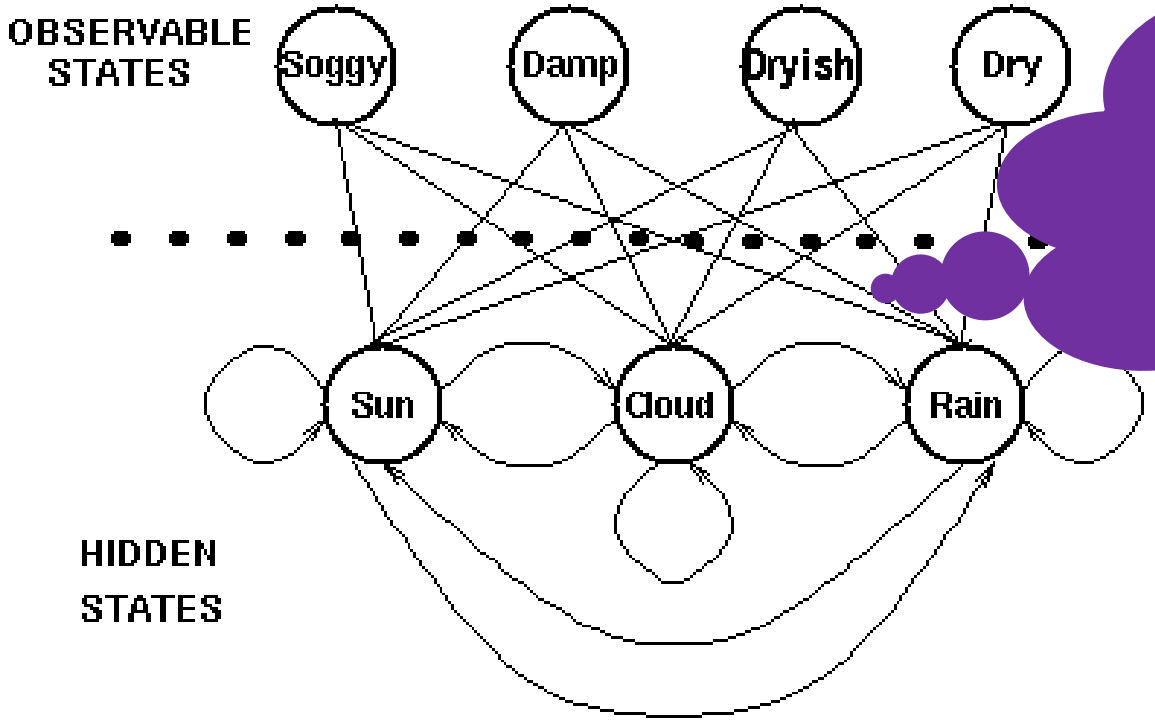
一种优化算法框架，用于含有 隐变量（hidden variable）的概率模型参数的极大似然估计（Maximum Likelihood Estimation, MLE），或极大后验概率估计（Maximum A Posterior estimation, MAP）

方法	处理怎样的数据	模型举例
无监督学习	(X)	k-Means, HMM, GMM, pLSA, LDA…
有监督学习	(X, y)	Naïve Bayes, NN, LR, ME, SVM, GBDT…
半监督学习	(X) + (X, y)	self-training, co-training, S3VM…
强化学习	(action, state, award)	Markov Decision Process（MDP）

模型	模型的隐变量是什么
k-Means	样本所属的聚类中心点
HMM	隐含状态 (E.g. 词性 for 词性标注；状态 for 其他序列标注模型)
GMM	样本所属的高斯分布 (Gaussian Distribution)
topic model (E.g. pLSA, LDA)	topic
IBM Model for Word Alignment	词语对齐 E.g. (布什与沙龙举行了会谈) (Bush held a meeting with Sharon)

By EM (Expectation-Maximization)

- With the given data (pronunciation sound [HMM], documents [LDA – topic], etc.), it's generated by some hidden logics



EM is the method to learn the best mapping of the hidden logics to generate the collected data

Many algorithms are using EM

- **K-means**
- **GMM**
  - Gaussian Mixed Model
- **HMM**
  - Hidden Markov Model
- **pLSA/pLSI**
  - Probabilistic Latent Semantic Analysis(Index)
- **LDA**
  - Latent Dirichlet Allocation
- **Peacock**
  - LDA for Big Data

[1] 机器学习十大算法(ICDM2006)（英文）
http://119.90.25.20/www.cs.uvm.edu/~icdm/algorithms/ICDM06-Panel.pdf
[2] 机器学习十大算法(ICDM2006)（中文翻译）
www.itfront.cn/attachment.aspx?attachmentid=1565

Top 10 Algorithms: Summary

- **#1: C4.5** (61 votes), presented by Hiroshi Motoda
- **#2: K-Means** (60 votes), presented by Joydeep Ghosh
- **#3: SVM** (58 votes), presented by Qiang Yang
- **#4: Apriori** (52 votes), presented by Christos Faloutsos
- **#5: EM** (48 votes), presented by Joydeep Ghosh
- **#6: PageRank** (46 votes), presented by Christos Faloutsos
- **#7: AdaBoost** (45 votes), presented by Zhi-Hua Zhou
- **#7: kNN** (45 votes), presented by Vipin Kumar
- **#7: Naive Bayes** (45 votes), presented by Qiang Yang
- **#10: CART** (34 votes), presented by Dan Steinberg

ICDM 2006 Panel 12/21/2006, Coordinators: Xindong Wu and Vipin Kumar

EM: the intuition

- Assume that we have two coins, C1 and C2
- Assume the bias of C1 is θ_1
  (i.e., probability of getting heads with C1)
- Assume the bias of C2 is θ_2
  (i.e., probability of getting heads with C2)
- We want to find θ_1, θ_2 by performing a number of trials
  (i.e., coin tosses)

First experiment

- We choose 5 times one of the coins.
- We toss the chosen coin 10 times

$$\theta_1 = \frac{number\ of\ heads\ using\ C1}{total\ number\ of\ flips\ using\ C1}$$

$$\theta_2 = \frac{number\ of\ heads\ using\ C2}{total\ number\ of\ flips\ using\ C2}$$

Coin A	Coin B
	5 H, 5 T
9 H, 1 T	
8 H, 2 T	
	4 H, 6 T
7 H, 3 T	
24 H, 6 T	9 H, 11 T

$$\theta_1 = \frac{24}{24+6} = 0.8$$

$$\theta_2 = \frac{9}{9+11} = 0.45$$

Assume a more challenging problem

H T T T H H T H T H

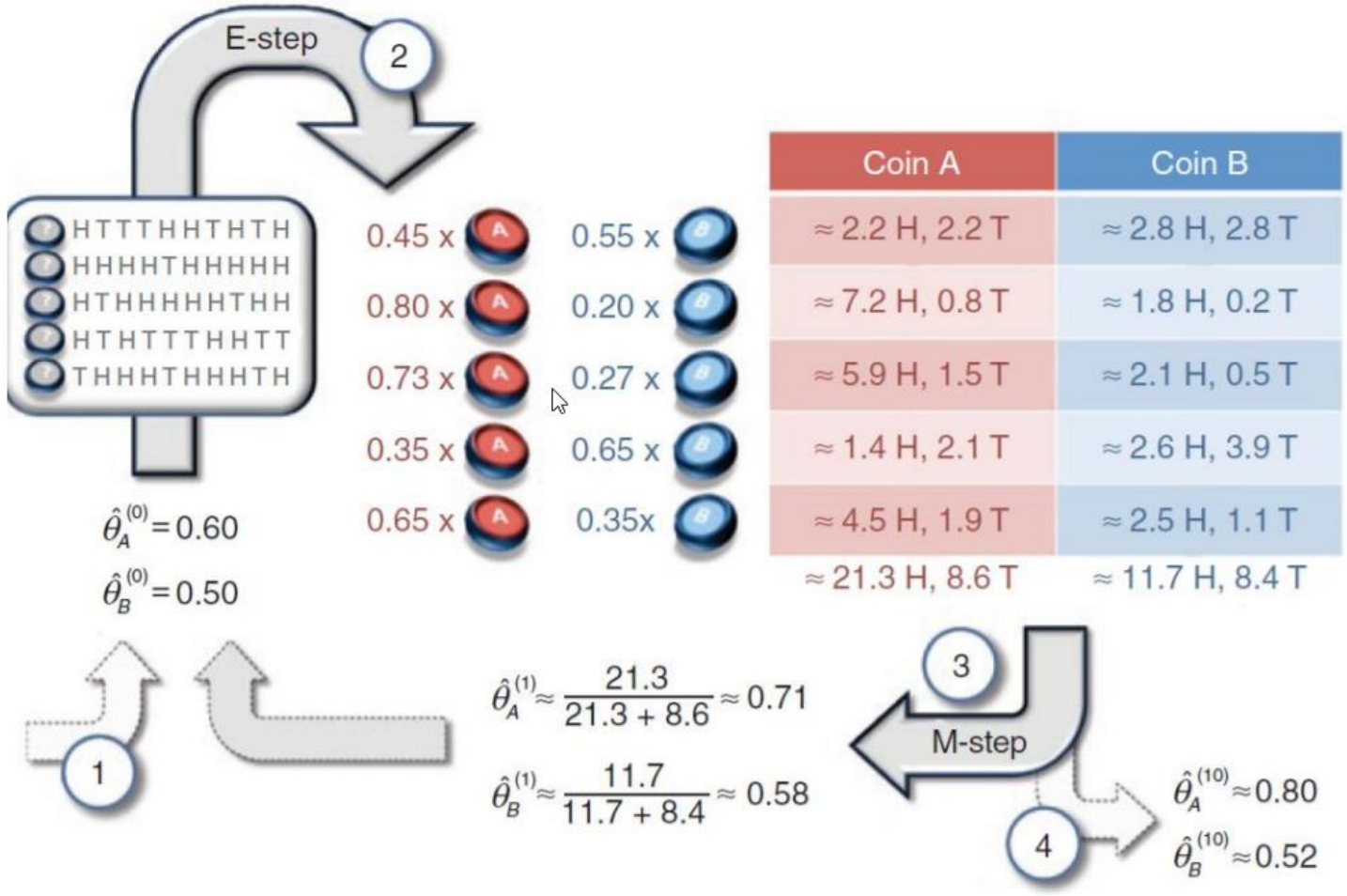
H H H H T H H H H H

H T H H H H H T H H

H T H T T T H H T T

T H H H T H H H T H

•We do not know the identities of the coins used for each set of tosses (we treat them as hidden variables).



将两个初始的参数随机设定为$\hat{\theta}_A^{(0)} = 0.6, \hat{\theta}_B^{(0)} = 0.5$，在这两个参数下出现第一轮结果，也就是5正5反的概率就可以表示为

$$P(H^5T^5|A) = 0.6^5 \times 0.4^5, P(H^5T^5|B) = 0.5^{10}$$

对上面的两个似然概率进行归一化可以得出后验概率，两者分别是 0.45 和 0.55，也就是下图中的结果。这说明如果初始结果的随机参数是准确的，拿第一轮结果更可能由硬币 B 产生（0.55>0.45）。同理也可以计算出其他四轮的结果来自不同硬币的后验概率，结果已经在上图中显示。

在已知硬币的选择时，所有正反面的结果都有明确的归属：要么来自 A AA 要么来自 B。利用后验概率可以直接对硬币的选择作出判断：1、4 轮使用的是硬币 B，2、3、5 轮使用的是硬币 A 。

既然硬币的选择已经确定，这时就可以使用最大似然估计，其结果和前文中的最大似然估计结果是相同的，也就是 $\hat{\theta}_A^{(1)} = 0.8, \hat{\theta}_B^{(1)} = 0.45$。利用这组更新的参数又可以重新计算每一轮次抽取不同硬币的后验概率。

虽然这种方法能够实现隐变量和参数的动态更新，但它还不是真正的EM算法，而是**硬输出的k均值聚类**。真正的EM算法并不会将后验概率最大的值赋给隐变量，而是**考虑其所有可能的取值**，在概率分布的框架下进行分析。

在前面的例子中，由于第一轮投掷硬币A的可能性是0.45，那么硬币A对正反面出现次数的贡献就是45%，在5次正面的结果中，来源于硬币A的就是 5×0.45=2.25 次，来源于硬币B的则是2.75次。同理可以计算出其他轮次中A和B各自的贡献，贡献的比例都和计算出的后验概率相对应。

计算出A和B在不同轮次中的贡献，就可以对未知参数做出更加精确的估计。在 50 次投掷中，硬币A贡献了21.3次正面和8.6次反面，其参数估计值$\hat{\theta}_A^{(1)} = 0.71$；硬币$B$贡献了11.7次正面和8.4次反面，其参数估计值$\hat{\theta}_B^{(1)} = 0.58$。利用这组参数继续迭代更新，就可以计算出最终的估计值。

输入： 观察到的数据 $x = (x_1, x_2, \ldots x_n)$，联合分布 $p(x, z; \theta)$，条件分布 $p(z|x, \theta)$，最大迭代次数J。

算法步骤：

（1）随机初始化模型参数θ的初值 θ_0 。

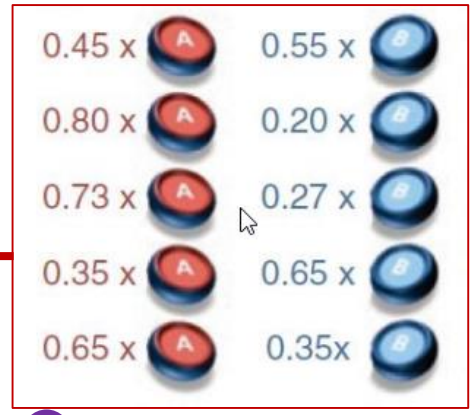
$\hat{\theta}_A^{(0)} = 0.60$

$\hat{\theta}_B^{(0)} = 0.50$

（2）j=1,2,...,J 开始EM算法迭代：

• E步：计算联合分布的条件概率期望：

$$Q_i(z_i) = p(z_i|x_i, \theta_j)$$



$$l(\theta, \theta_j) = \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}$$

• M步：极大化 $l(\theta, \theta_j)$，得到 θ_{j+1} ：

$$\theta_{j+1} = argmax l(\theta, \theta_j)$$

• 如果 θ_{j+1} 已经收敛，则算法结束。否则继续进行E步和M步进行迭代。

输出： 模型参数θ。

Assume a more challenging problem

H T T T H H T H T H

H H H H T H H H H H

H T H H H H H T H H

H T H T T T H H T T

T H H H T H H H T H

$$p(x_i{}^j|\boldsymbol{z}_i, \theta) = \prod_{k=1}^{2} \left[\theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j}\right]^{z_{ik}}$$

EM's challenge is to compose this part!

输入：观察到的数据 $x = (x_1, x_2, \ldots x_n)$，联合分布 $p(x, z; \theta)$，条件分布 $p(z|x, \theta)$，最大迭代次数J。

算法步骤：

（1）随机初始化模型参数θ的初值 θ_0 。

（2）j=1,2,...,J 开始EM算法迭代：

• E步：计算联合分布的条件概率期望：

$$Q_i(z_i) = p(z_i|x_i, \theta_j)$$

$$l(\theta, \theta_j) = \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}$$

• M步：极大化 $l(\theta, \theta_j)$,得到 θ_{j+1} ：

$$\theta_{j+1} = argmax l(\theta, \theta_j)$$

• 如果 θ_{j+1} 已经收敛，则算法结束。否则继续进行E步和M步进行迭代。

输出：模型参数θ。

Assume a more challenging problem

H T T T H H T H T H

H H H H T H H H H H

H T H H H H H T H H

H T H T T T H H T T

T H H H T H H H T H

Coin A	Coin B
≈ 2.2 H, 2.2 T	≈ 2.8 H, 2.8 T
≈ 7.2 H, 0.8 T	≈ 1.8 H, 0.2 T
≈ 5.9 H, 1.5 T	≈ 2.1 H, 0.5 T
≈ 1.4 H, 2.1 T	≈ 2.6 H, 3.9 T
≈ 4.5 H, 1.9 T	≈ 2.5 H, 1.1 T
≈ 21.3 H, 8.6 T	≈ 11.7 H, 8.4 T

$$\hat{\theta}_A^{(1)} \approx \frac{21.3}{21.3 + 8.6} \approx 0.71$$

$$\hat{\theta}_B^{(1)} \approx \frac{11.7}{11.7 + 8.4} \approx 0.58$$

$$\hat{\theta}_A^{(10)} \approx 0.80$$

$$\hat{\theta}_B^{(10)} \approx 0.52$$

EM: the Maths (setting the joint)

$$p(X_1, X_2, \cdots, X_5, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_5 | \theta) \qquad \mathbf{z}_i = \begin{bmatrix} z_{i1} \\ z_{i2} \end{bmatrix} \in \left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\}$$

$$= p(\{x_1{}^1, \cdots, x_1{}^{10}\}, \cdots, \{x_5{}^1, \cdots, x_5{}^{10}\}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_5 | \theta)$$

$$= p(\{x_1{}^1, \cdots, x_1{}^{10}\}, \cdots, \{x_5{}^1, \cdots, x_5{}^{10}\} | \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_5, \theta) p(\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_5)$$

$$= \prod_{i=1}^{5} p(\{x_i{}^1, \cdots, x_i{}^{10}\} | \mathbf{z}_i, \theta) \prod_{i=1}^{5} p(\mathbf{z}_i)$$

$$p(\mathbf{z}_i) = \prod_{k=1}^{2} \pi_\kappa{}^{z_{ik}} \quad \pi_\kappa \text{ is the probability of selecting coin } k \in \{1,2\}$$

$$p(\{x_i{}^1, \cdots, x_i{}^{10}\} | \mathbf{z}_i, \theta) = \prod_{j=1}^{10} p(x_i{}^j | \mathbf{z}_i, \theta)$$

$x_i{}^j = 1$ *If* j *toss of* i *run is head*

$x_i{}^j = 0$ *If* j *toss of* i *run is tail*

$$p(x_i{}^j | \boldsymbol{z}_i, \theta) = \prod_{k=1}^{2} \left[\theta_k{}^{x_i{}^j} (1-\theta_k)^{1-x_i{}^j} \right]^{z_{ik}}$$

then

$$p(X_1, X_2, \cdots, X_5, \boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_5 | \theta)$$

$$= \prod_{i=1}^{5} \prod_{j=1}^{10} \prod_{k=1}^{2} \left[\theta_k{}^{x_i{}^j} (1-\theta_k)^{1-x_i{}^j} \right]^{z_{ik}} \prod_{i=1}^{5} \prod_{k=1}^{2} \pi_\kappa{}^{z_{ik}}$$

$$\ln p(X_1, X_2, \cdots, X_5, \boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_5|\theta)$$

$$= \sum_{i=1}^{5}\sum_{j=1}^{10}\sum_{k=1}^{2} z_{ik} \ln \theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j} + \sum_{i=1}^{5}\sum_{k=1}^{2} z_{ik} \ln \pi_\kappa$$

Taking the expectation of the above

$$E_{p(Z|X)}\left[\ln p(X_1, X_2, \cdots, X_5, \boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_5|\theta)\right]$$

$$= \sum_{i=1}^{5}\sum_{j=1}^{10}\sum_{k=1}^{2} E_{p(Z|X)}[z_{ik}] \ln \theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j}$$

$$+ \sum_{i=1}^{5}\sum_{k=1}^{2} E_{p(Z|X)}[z_{ik}] \ln \pi_\kappa$$

$$p(\boldsymbol{Z}|\boldsymbol{X},\theta) = p\left(\boldsymbol{z}_1,\boldsymbol{z}_2,\cdots,\boldsymbol{z}_5|X_1,X_2,\cdots,X_5,\theta\right)$$

$$E_{p(\boldsymbol{Z}|\boldsymbol{X})}[z_{ik}] = \sum_{\boldsymbol{z}_1}\cdots\sum_{\boldsymbol{z}_5} z_{ik}\, p(\boldsymbol{Z}|\boldsymbol{X},\theta) = \sum_{\boldsymbol{z}_i} z_{ik} p(\boldsymbol{z}_i|\{x_i{}^1,\cdots,x_i{}^{10}\})$$

$$p(\boldsymbol{z}_i|\{x_i{}^1,\cdots,x_i{}^{10}\}) = \frac{p(\{x_i{}^1,\cdots,x_i{}^{10}\}|\boldsymbol{z}_i,\theta)p(\boldsymbol{z}_i)}{p(\{x_i{}^1,\cdots,x_i{}^{10}\}|\theta)}$$

$$= \frac{\prod_{j=1}^{10}\prod_{k=1}^{2}\left[\theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j}\right]^{z_{ik}}\pi_\kappa{}^{z_{ik}}}{\sum_{\boldsymbol{z}_i}\prod_{j=1}^{10}\prod_{k=1}^{2}\left[\theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j}\right]^{z_{ik}}\pi_\kappa{}^{z_{ik}}}$$

$$E_{p(\mathbf{Z}|\mathbf{X})}[z_{ik}] = \sum_{\mathbf{z}_i} z_{ik} \frac{\prod_{j=1}^{10}\prod_{k=1}^{2}\left[{\theta_k}^{{x_i}^j}(1-\theta_k)^{1-{x_i}^j}\right]^{z_{ik}}{\pi_\kappa}^{z_{ik}}}{\sum_{\mathbf{z}_i}\prod_{j=1}^{10}\prod_{k=1}^{2}\left[{\theta_k}^{{x_i}^j}(1-\theta_k)^{1-{x_i}^j}\right]^{z_{ik}}{\pi_\kappa}^{z_{ik}}}$$

$$= \frac{\sum_{\mathbf{z}_i} z_{ik}\prod_{j=1}^{10}\prod_{k=1}^{2}\left[{\theta_k}^{{x_i}^j}(1-\theta_k)^{1-{x_i}^j}\right]^{z_{ik}}{\pi_\kappa}^{z_{ik}}}{\sum_{\mathbf{z}_i}\prod_{j=1}^{10}\prod_{k=1}^{2}\left[{\theta_k}^{{x_i}^j}(1-\theta_k)^{1-{x_i}^j}\right]^{z_{ik}}{\pi_\kappa}^{z_{ik}}}$$

$$= \frac{\pi_\kappa\prod_{j=1}^{10}{\theta_k}^{{x_i}^j}(1-\theta_k)^{1-{x_i}^j}}{\pi_1\prod_{j=1}^{10}{\theta_1}^{{x_i}^j}(1-\theta_1)^{1-{x_i}^j} + \pi_2\prod_{j=1}^{10}{\theta_2}^{{x_i}^j}(1-\theta_2)^{1-{x_i}^j}}$$

EM: Expectation and Maximization Steps

Expectation (E) Step (fix θ):

$$\pi_1 = \frac{1}{2} \quad \pi_2 = \frac{1}{2}$$

$$E_{p(\boldsymbol{Z}|\boldsymbol{X})}[z_{ik}] = \frac{\prod_{j=1}^{10} \theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j}}{\prod_{j=1}^{10} \theta_1{}^{x_i{}^j}(1-\theta_1)^{1-x_i{}^j} + \prod_{j=1}^{10} \theta_2{}^{x_i{}^j}(1-\theta_2)^{1-x_i{}^j}}$$

Maximization Step (fix $E_{p(\boldsymbol{Z}|\boldsymbol{X})}[z_{ik}]$):

$$\max L(\theta) = E_{p(Z|X)}\left[\ln p\left(X_1, X_2, \cdots, X_5, \boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_5 | \theta\right)\right]$$

$$= \sum_{i=1}^{5}\sum_{j=1}^{10}\sum_{k=1}^{2} E_{p(Z|X)}[z_{ik}] \ln \theta_k{}^{x_i{}^j}(1-\theta_k)^{1-x_i{}^j} + \sum_{i=1}^{5}\sum_{k=1}^{2} E_{p(Z|X)}[z_{ik}] \ln \pi_\kappa$$

$$= \sum_{i=1}^{5}\sum_{j=1}^{10}\sum_{k=1}^{2} E_{p(Z|X)}[z_{ik}]\left(x_i{}^j \ln\theta_k + \left(1-x_i{}^j\right)\ln(1-\theta_k)\right) + const$$

EM: Maximization Step

$$\frac{dL(\theta)}{d\theta_1} = \sum_{i=1}^{5}\sum_{j=1}^{10} E_{p(Z|X)}[z_{i1}]\left(x_i{}^j \frac{1}{\theta_1} - \left(1 - x_i{}^j\right)\frac{1}{1-\theta_1}\right) = 0$$

$$\sum_{i=1}^{5}\sum_{j=1}^{10} E_{p(Z|X)}[z_{i1}]\left(x_i{}^j(1-\theta_1) - \left(1 - x_i{}^j\right)\theta_1\right) = 0$$

$$\theta_1 = \frac{\sum_{i=1}^{5}\sum_{j=1}^{10} E_{p(Z|X)}[z_{i1}]x_i{}^j}{\sum_{i=1}^{5} 10\, E_{p(Z|X)}[z_{i1}]} \qquad \theta_2 = \frac{\sum_{i=1}^{5}\sum_{j=1}^{10} E_{p(Z|X)}[z_{i2}]x_i{}^j}{\sum_{i=1}^{5} 10\, E_{p(Z|X)}[z_{i2}]}$$

来自机器学习的数据分析方法 – 基本概念和思想

- **本章只讲点机器学习的基本概念和思想**
  - 距离(Distance) – 要时常琢磨下“真的”距离
    - 各位肯定记得 … **Euclidean 距离**, **PCA (as a transformer)**
    - 很多数据是流形 – 其上的距离很有趣
      - ✓ **KNN** (K Nearest Neighbor), **MDS** (Multi-Dimensional Scaling), IsoMap, LE, LLE, …
    - 间接计算非线性空间中数据的距离 – **Kernel** 的概念
  - 抓住主要矛盾 – 主成分 (Principal Components)
    - **PCA**, FA (Factor Analysis), CCA, ICA, …
    - Regression-based Significance analysis
    - Sparse Coding – Compressed/Compressing Sensing
  - 找出背后的影响因素 – **EM** (Expectation Maximization)
    - K-means, LDA, GMM, HMM, Peacock 等
  - 算法的评估