



Feature Selection & Feature Extraction

Outline

- ❑ Introduction of Feature
- ❑ Feature Selection
- ❑ Feature Extraction
- ❑ Feature Learning
- ❑ Summary

Introduction of Feature



Equus Zebra (山斑马)



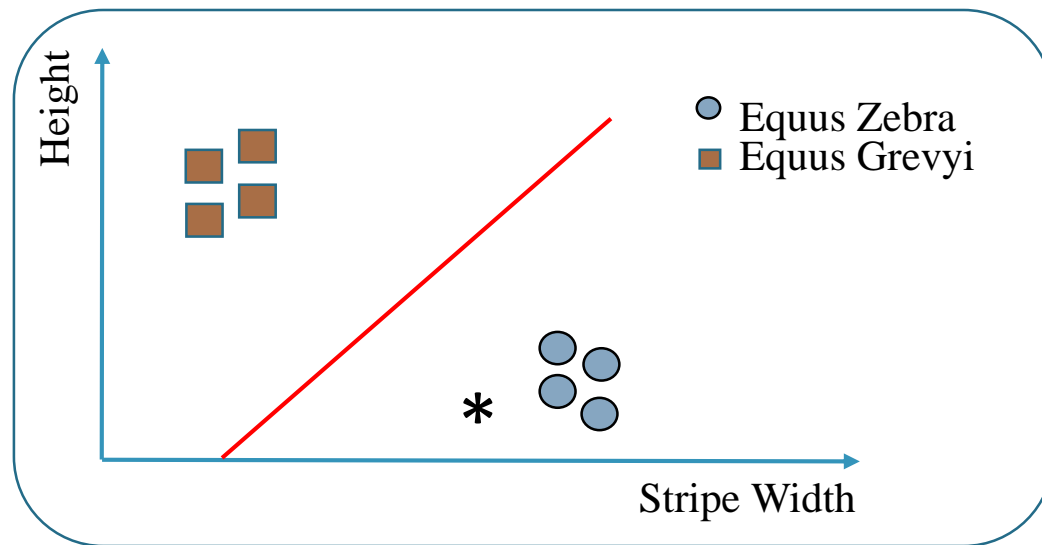
Equus Grevyi (细纹斑马)

Introduction of Feature

We suppose that there are two differences between Equus Zebra and Equus Grevyi:

- 1) Stripe Width (条纹)
- 2) Height (身高)

According to these two differences, the two kinds of zebra are distinct from each other:



Introduction of Feature

- ❑ Let us now assume that we are given a new image of Zebra (✱) that we don't know to which class it belongs.
- ❑ It is reasonable to say that we observe the height and width of stripes, and we plot the corresponding point in the above figure.

The measurements used for the classification, stripe width and height, are known as *feature*.

In more general case l features, $x_i, i = 1, 2, \dots, l$, are used and they form the *feature vector*

$$\mathbf{x} = [x_1, x_2, \dots, x_l]^T$$

Each of the feature vectors identifies *uniquely* a single object

Introduction of Feature

Several basic questions in a classification task:

❑ How are the features generated?

It is *problem dependent*, and it concerns the *feature generation stage*

❑ What is the best number l of features to use?

It concerns the *feature selection stage*. In practice, a larger than necessary number of feature candidates is generated and then the ‘best’ of them is adopted

❑ Having adopted the appropriate features, how does one design the classifier?

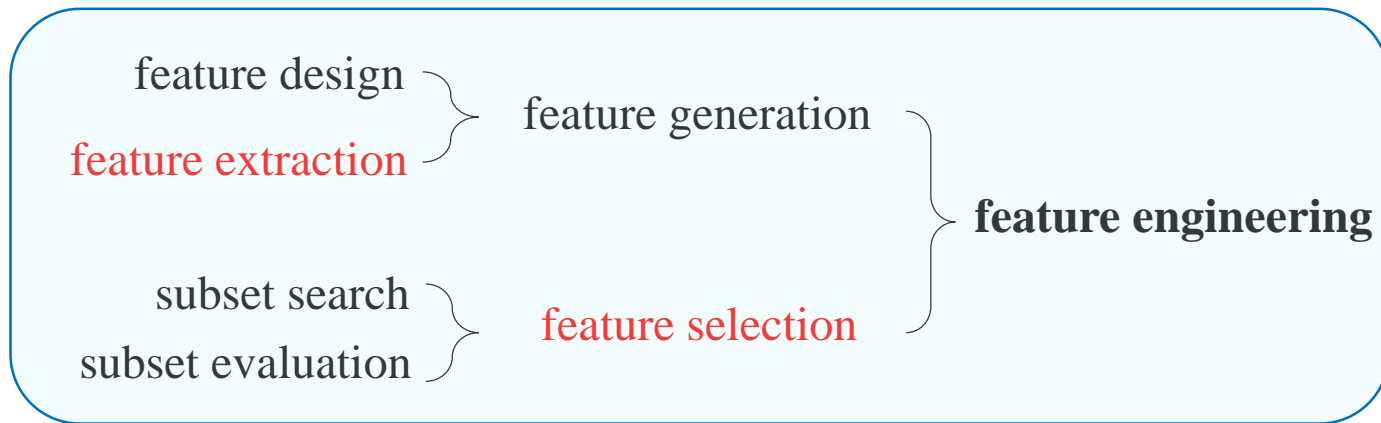
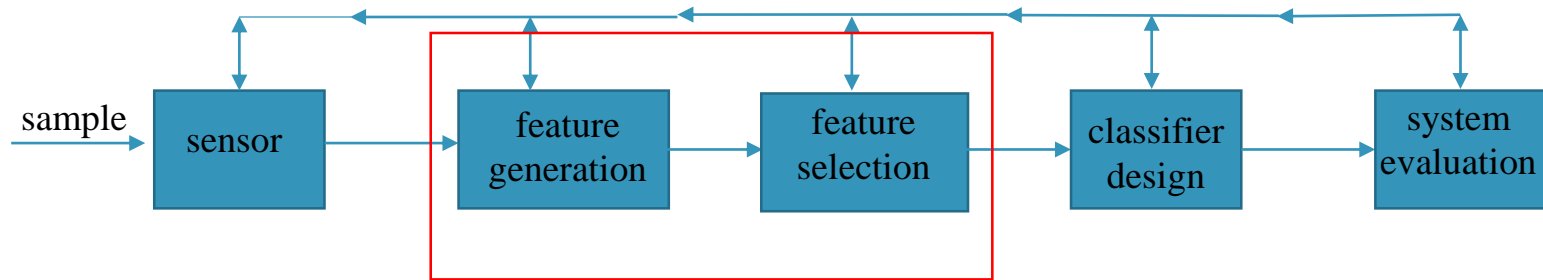
In the proceeding example, it is linear classifier. In general, the surface dividing the space in the various class regions are nonlinear. What type of classifier is optimizing criterion for the specific task is *classifier design stage*

❑ How can one assess the performance of designed classifier?

This is the task of the *system evaluation stage*

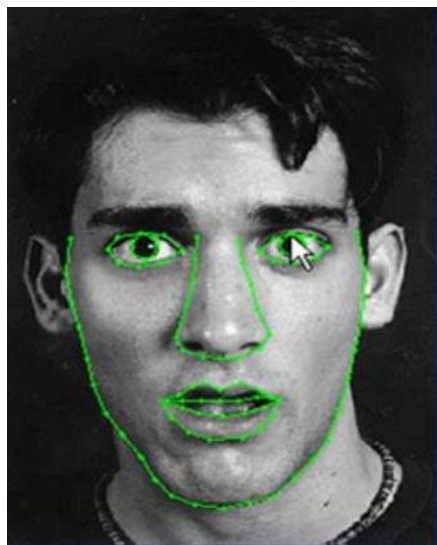
Introduction of Feature

The basic stages involved in the design of a classification system:

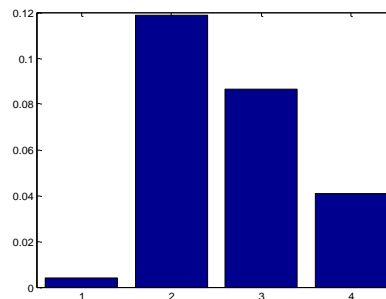
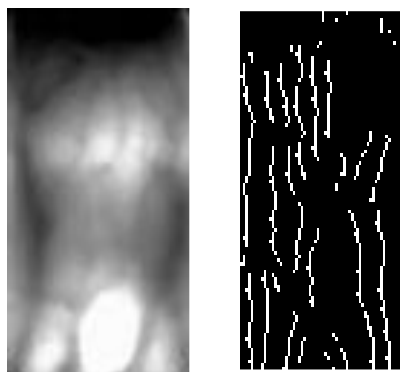


Introduction of Feature

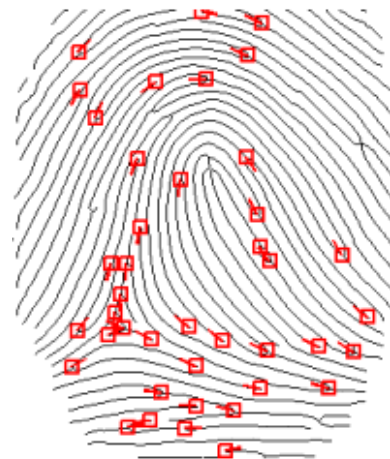
Examples of feature design:



feature of face



feature of finger vein



feature of finger print

Outline

- ❑ Introduction of Feature
- ❑ Feature Selection
- ❑ Feature Extraction
- ❑ Feature Learning
- ❑ Summary

Feature Selection

For example: Watermelon {color, root, sound, texture, touch...}

→ **designed feature**

Root: curled
Sound: crisp
Texture: clear



Experienced farmers say:

**This is a good
watermelon ! ! !**

Feature Selection

Relevant Features: attributes which are **related to our task**, and are **not relate to each other**

{root, sound, texture} 好而不同

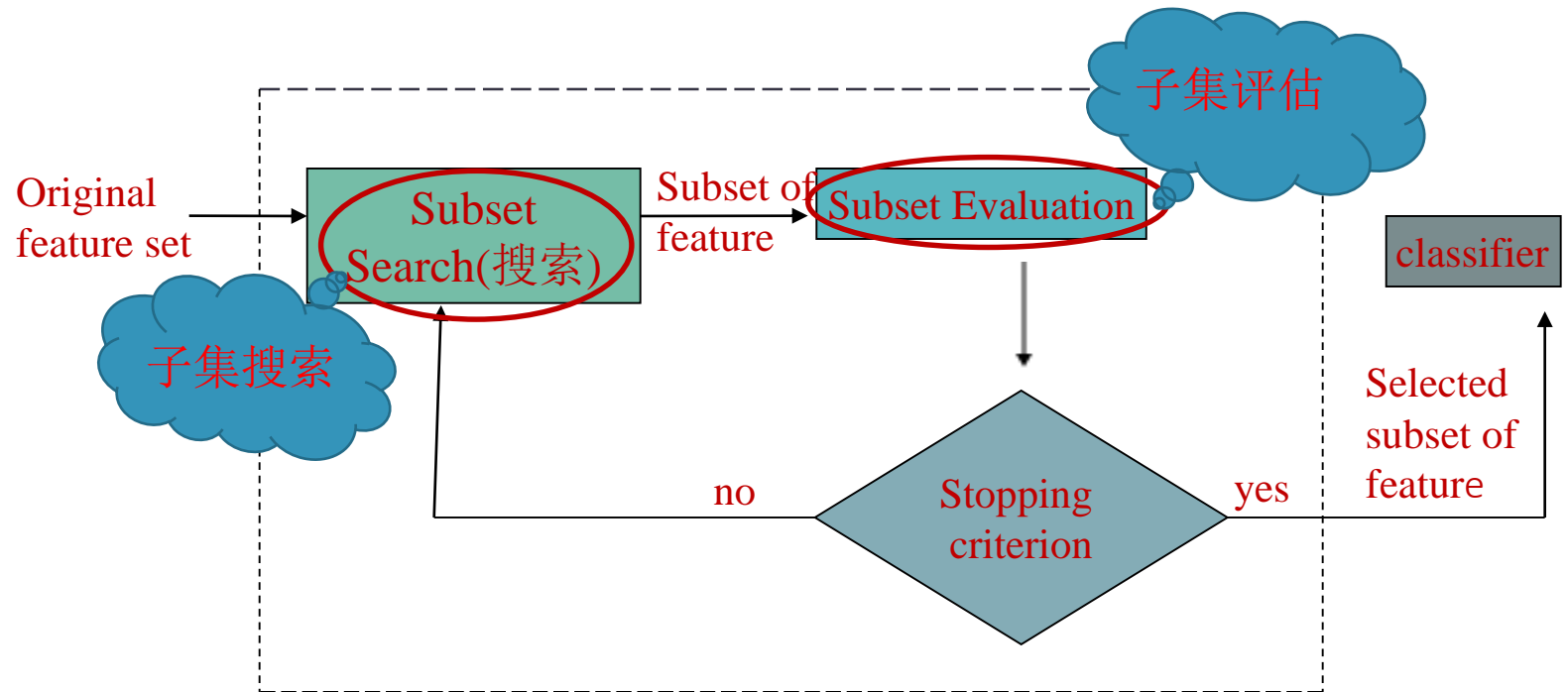
Irrelevant Features: attributes which are not relate to our task

{color, touch...}

Feature Selection is select out **relevant features** from all features for our task and features are **not related to each other**

Feature Selection

Originally, feature selection include two steps (一般来说, 特征选择步骤如下) :



Feature Selection

Feature selection strategy:

- ❑ Filter(过滤式)
- ❑ Wrapper（包裹式）
- ❑ Embedding（嵌入式）

Feature Selection

- ❑ **Filter** (of irrelevant attributes) occurs before the training stage

Method: Relief

- ❑ **Wrapper** is different from filter, it directly takes performance of learners as evaluation standard

Method: LVW

- ❑ **Embedding** is an optimized process which confuse feature selection and training. It means that feature selection is completed automatically during training

$$\min_{\omega} \sum_{i=1}^m (y_i - \omega^T x_i)^2 + \lambda \|\omega\|_1$$

↓
L1 norm

易获得稀疏解，
是一种嵌入式
特征选择方法

Filter

Filter (of irrelevant attributes) occurs before the training stage

Relief (Relevant Features) [Kira and Rendell, 1992]

- Give a 'relevant statistic' to *each original feature* to measure the importance of feature
- Set a **threshold** t , if relevant statistic of a feature is bigger than t , add it to subset
- The importance of subset feature is **equal to sum of feature subset relevant statistic**

Relevant statistic is a
feature evaluation method

The methods are often univariate and consider the feature independently, or with regard to the dependent variable

Wrapper

Wrapper is different from filter, it directly takes performance of learners as evaluation standard.

LVW(Las Vegas Wrapper) is a typical wrapper method.

- **Randomly** generate series subsets of candidate features
- Run the **learning method** with each ofm
- Use the **accuracy** of the results for evaluation (either training set or a separate validation set)

Wrapper

INPUT: Data Set D ; Feature Set A ; Learning Algorithm \mathcal{L} ; Stopping Criteria T

Process:

$E = \infty$;

$d = |A|$;

$A^* = A$;

$t = 0$;

终止条件

while $t < T$ do

Random Feature Set A' ;

$d' = |A'|$;

$E' = \text{CrossValidation}(\mathcal{L}(D^{A'}))$;

循环的条件

if $(E' < E) \vee ((E' = E) \wedge (d' < d))$ then

分类错误率比
上一轮减小

$t = 0$;

$E = E'$;

$d = d'$;

$A^* = A'$;

分类错误率跟上一轮相等，但特征维数减少

else

$t = t + 1$;

end if

end while

OUTPUT: Subset A^*

If it is not update continuously for T times, the program will break the loop(连续 T 次不更新).

Wrapper

- ❑ **LVW** can reduce the number of features and improve the accuracy
- ❑ Not be recommended in applications where time is critical factor
- ❑ Slowness is caused by learning algorithm

Embedding

Embedding is an optimization which confuse feature selection and training. It means that feature selection is completed automatically during training.

Optimal Object

$$\min_{\omega} \sum_{i=1}^m (y_i - \omega^T x_i)^2 + \lambda \|\omega\|_1$$

L1 norm

feature selection + feature extraction

易获得稀疏解，
是一种嵌入式
特征选择方法

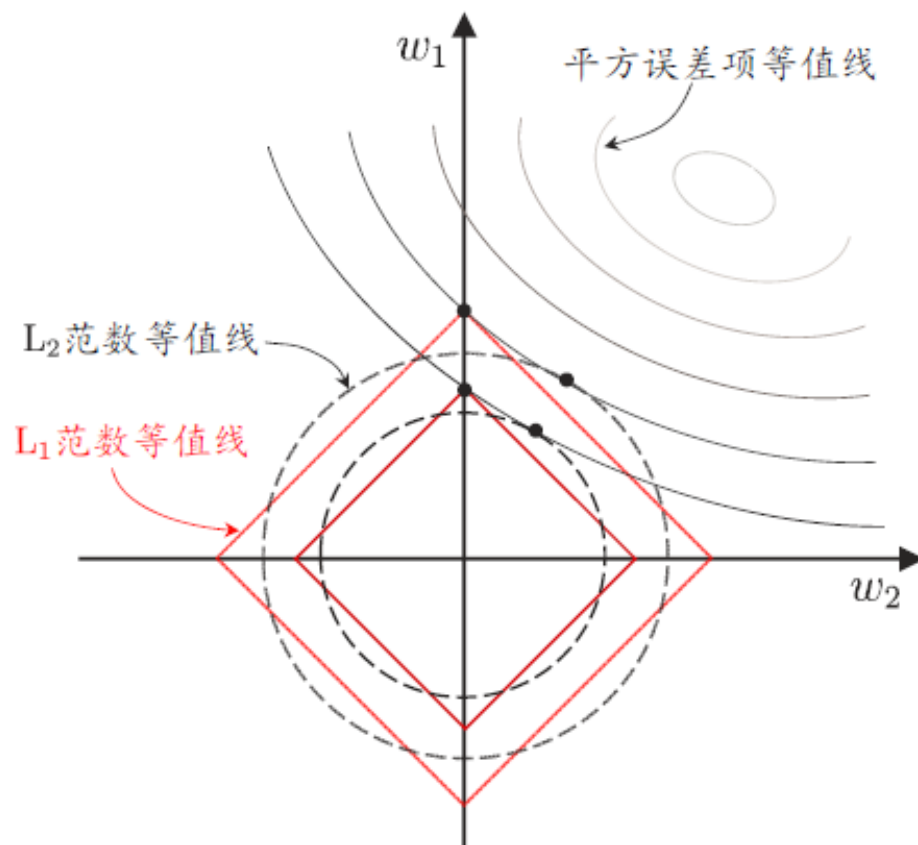
并行的思路

L1-norm is easier to get the sparse solution than L2-norm!

$$\min_{\omega} \sum_{i=1}^m (y_i - \omega^T x_i)^2 + \lambda \|\omega\|_2^2$$

L2 norm

Embedding



假设 \mathbf{x} 仅有两个属性, 那么 \mathbf{w} 有两个分量 w_1 和 w_2 . 那么目标优化的解要在平方误差项与正则化项之间折中, 即出现在图中平方误差项等值线与正则化等值线相交处.

从图中看出, 采用L₁范数时交点常出现在坐标轴上, 即产生 w_1 或者 w_2 为0的稀疏解.

Feature Selection in Sklearn

```
1 # Recursive Feature Elimination
2 from sklearn import datasets
3 from sklearn.feature_selection import RFE
4 from sklearn.linear_model import LogisticRegression
5 # load the iris datasets
6 dataset = datasets.load_iris()
7 # create a base classifier used to evaluate a subset of attributes
8 model = LogisticRegression()
9 # create the RFE model and select 3 attributes
10 rfe = RFE(model, 3)
11 rfe = rfe.fit(dataset.data, dataset.target)
12 # summarize the selection of the attributes
13 print(rfe.support_)
14 print(rfe.ranking_)
```

RFE

```
1 # Feature Importance
2 from sklearn import datasets
3 from sklearn import metrics
4 from sklearn.ensemble import ExtraTreesClassifier
5 # load the iris dataset
6 dataset = datasets.load_iris()
7 # fit an ExtraTreesClassifier
8 model = ExtraTreesClassifier()
9 model.fit(dataset.data, dataset.target)
10 # display the feature importance
11 print(model.feature_importances_)
```

Feature

Iris dataset:(鸢尾花数据集)

Feature: sepal length, sepal width, petal length, petal width

Label: setosa, versicolor, virginica

Number: 150

Results on Iris dataset:

| | |
|-----|--|
| d=4 | [True True True True] [1 1 1 1] |
| d=3 | [False True True True] [2 1 1 1] |
| d=2 | [False True False True] [3 1 2 1] |
| d=1 | [False False False True] [4 2 3 1] |

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN <http://sklearn.apachecon.org/cn/0.19.0/> (sklearn document)

SKlearn



首页 安装 文档 ▾ 示例 时光轴 项目相关 ▾ 贡献者 GitHub



scikit-learn

Python 中的机器学习

- 简单高效的数据挖掘和数据分析工具
- 可供大家使用，可在各种环境中重复使用
- 建立在 NumPy, SciPy 和 matplotlib 上
- 开放源码，可商业使用 - BSD license

分类

识别某个对象属于哪个类别

应用: 垃圾邮件检测, 图像识别

算法: SVM, nearest neighbors, random forest, ... — 示例

回归

预测与对象相关联的连续值属性

应用: 药物反应, 股价

算法: SVR, ridge regression, Lasso, ... — 示例

聚类

将相似对象自动分组

应用: 客户细分, 分组实验结果

算法: k-Means, spectral clustering, mean-shift, ... — 示例

降维

减少要考虑的随机变量的数量

应用: 可视化, 提高效率

算法: PCA, feature selection, non-negative matrix factorization. — 示例

模型选择

比较, 验证, 选择参数和模型

目标: 通过参数调整提高精度

模型: grid search, cross validation, metrics. — 示例

预处理

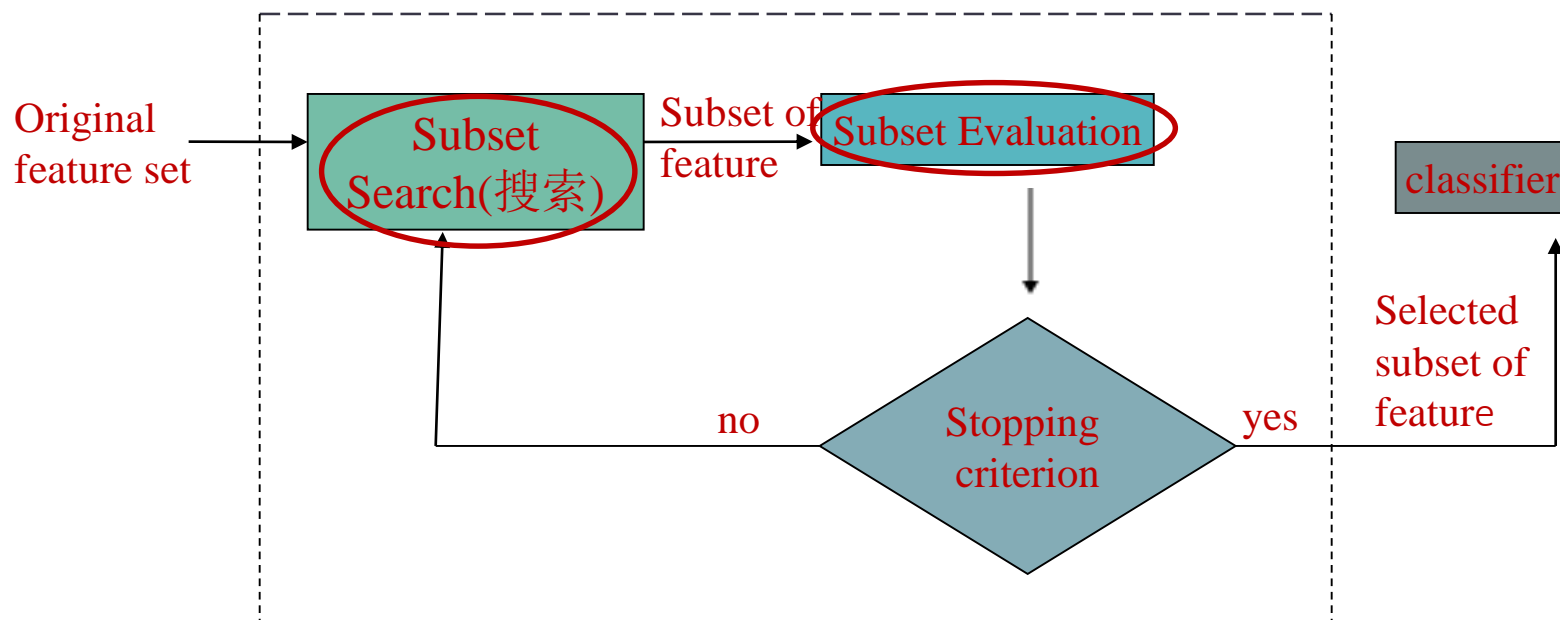
特征提取和归一化

应用: 把输入数据 (如文本) 转换为机器学习算法可用的数据

算法: preprocessing, feature extraction.

Extension

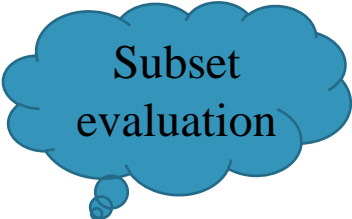
Originally, feature selection include two steps（一般来说，特征选择步骤如下）：



Subset Search

1) **Forward Search**: gradually increase **relevant** feature

Optimal feature: $\{ \} \Rightarrow \{a_2\} \Rightarrow \{a_2, a_4\} \dots$



Subset
evaluation

2) **Backward Search**: gradually reduce **irrelevant** feature

Optimal feature: $\{a_1, a_2, \dots, a_d\} \Rightarrow \{a_1, a_2, \dots, a_{d-1}\} \Rightarrow \{a_1, a_2, \dots, a_k, \dots, a_{d-2}\} \Rightarrow \dots$

These strategies are greedy, only consider optimization of this round
这些方法是贪心的策略，因为是在上一轮的基础上考虑本轮最优，
所以不一定得到最优特征组合

其他子集搜索方法：

<http://www.cnblogs.com/heaad/archive/2011/01/02/1924088.html>

Question: **How** to evaluate the searched feature?

Subset Evaluation

Separation Criterion(类可区分性判据) is for subset evaluation
Measure class distinguishing ability of feature subsets

Distance based separation criterion
基于距离的类可区分性判据

Probability distributions based separation criterion
基于概率分布的类可区分性判据

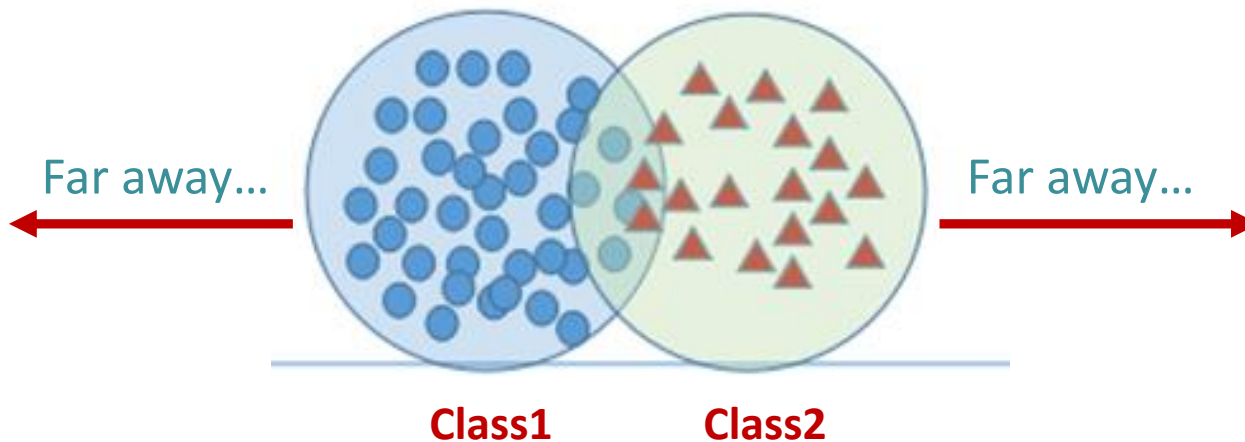
Entropy based separation criterion
基于熵的类可区分性判据

Distance based Methods

Searching a feature subset, we hope :

with-in class distance is as smaller as possible

between class distance is as larger as possible



Distance based Methods

Mean Vector:

$$u_i = \frac{1}{N_i} \sum_{x \in D_i} x, \quad (i = 1, 2)$$

Covariance:

$$S_i = \sum_{x \in D_i} (x - u_i)(x - u_i)^T, \quad (i = 1, 2)$$

Within-class scatter matrix:

$$S_w = S_1 + S_2$$

Between-class scatter matrix:

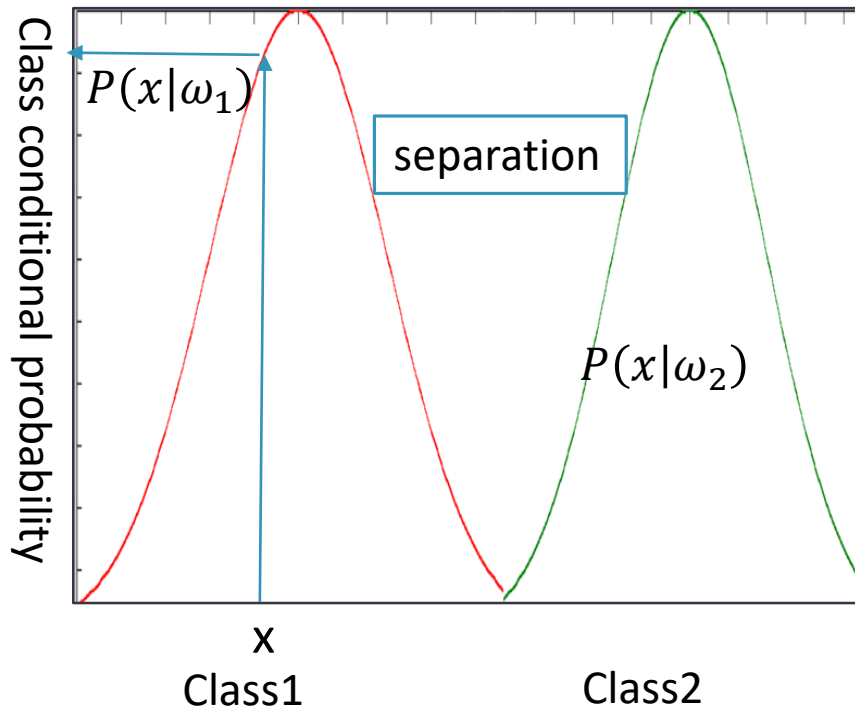
$$S_b = (u_1 - u_2)(u_1 - u_2)^T$$

Separation Criterion:

$$\mathbf{Max} \frac{\text{tr}(S_b)}{\text{tr}(S_w)}$$

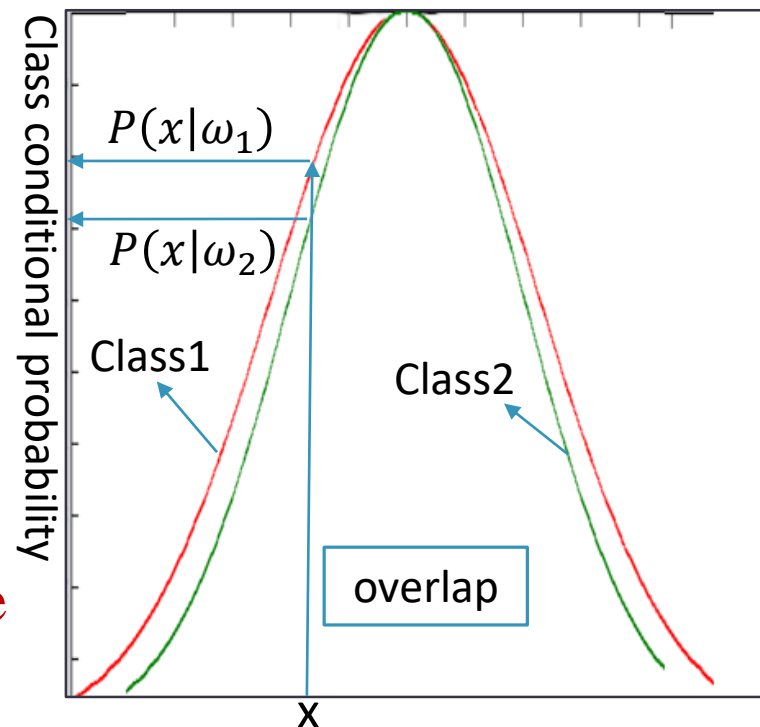
注：协方差矩阵的迹等同于方差

Probability Distributions based



Q: Are two classes overlap base on feature set?

Class conditional probability density curve (get by MLE 极大似然估计)



Probability Distributions based

Overlap degree can be represent by distance J between two probability distribution

$$J = \int g[\underbrace{P(x|\omega_1)}_{\text{Class conditional probability}}, \underbrace{P(x|\omega_2)}_{\text{Class conditional probability}}, \underbrace{P_1, P_2}_{\text{Priori probability}}] dx$$

Conditions for J :

- 1) $J \geq 0$
- 2) If $P(x|\omega_1) = 0$ & $P(x|\omega_2) \neq 0$, or $P(x|\omega_2) = 0$ & $P(x|\omega_1) \neq 0$, $J = \max$
- 3) If $P(x|\omega_1) = P(x|\omega_2)$, $J = 0$;

Probability Distributions based

□ Any function which meet these conditions can be take as probability distance metric !!!

□ Common function for probability distance:

- 1) Bhattacharyya distance (巴氏距离)
- 2) Chernoff bound (Chernoff 界限)
- 3) Divergence (散度)

Reference book: 边肇祺 《模式识别》 第8章

Entropy based Methods

Entropy (熵) :

$$Ent(D) = - \sum_{k=1}^{|Y|} p_k \log_2 p_k$$

样本类别确定: $P_k = 1, Ent = 0;$
样本类别不确定: $P_k < 1, Ent > 0;$

The greater entropy value, the more uncertain
熵值越大，说明样本的类别不确定性越大

Entropy based Methods

In Bayes classifier, classification results is determined by **posterior probability**

For a sample, if **posterior probability** of all classes is **same** base on some features, we **can not know** the **category** of this sample

For example:

$$\text{if } P(w_i|x) = \frac{1}{C}, \quad C \text{ is number of categories}$$

$$\text{Error rate of classification: } e = 1 - \frac{1}{C} = \frac{C-1}{C}$$

$$\text{if } P(w_i|x) = 1, P(w_j|x) = 0, \quad j \neq i$$

$$\text{Error rate of classification: } e = 0$$

Entropy can measure the distribution of posterior probability!

Entropy based Methods

Shannon Entropy（香农熵）：

$$E(D) = - \sum_{i=1}^c P(w_i|x) \log_2 P(w_i|x)$$

Square Entropy（平方熵）：

$$J_C^2 = [P(w_1|x), P(w_2|x), \dots, P(w_c|x)] = 2[1 - \sum_{i=1}^c P^2(w_i|x)]$$

The more dispersed of posterior probability, the bigger of Entropy, the higher of error rate of classification

Outline

- ❑ Introduction of Feature
- ❑ Feature Selection
- ❑ Feature Extraction
- ❑ Feature Learning
- ❑ Summary

Feature Extraction

- ❑ Feature extraction is different from feature selection
- ❑ Feature extraction is to transform original features to a new set of features by constructing combinations of the features
- ❑ It is a process of feature engineering

Methods of Feature Extraction

□ Linear Methods

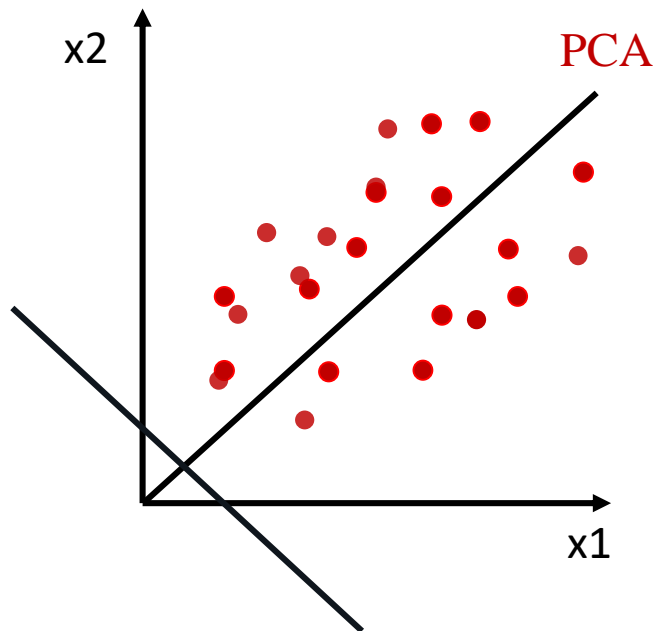
- Principal Component Analysis (PCA) [Pearson , 1901]
- Linear Discriminant Analysis (LDA) [Ronald Fisher , 1936]
[Belhumeur, 1996]

□ Nonlinear Methods

- Multidimensional Scaling (MDS) [Torgerson, W.S. et al. ,1958]
- Kernel principal component analysis (KPCA) [Scholkopf et al., 1998]
- Principal Curves [Hastie, 1989]
- Self-Organizing Feature Map (SOM) [Kohonen et al., 1995]
- Generative topographic map (GTM) [Bishop et al., 1998]
- Manifold Learning: Isomap,LLE,LE.....
-

PCA

PCA:(Principle Component Analysis)



$$z = w_1 * x_1 + w_2 * x_2$$



$$z = w^T x$$

线性组合就相当于几何中的投影

z is 1-dimension

W is projection **vector**

$x=(x_1;x_2)$ is a vector

w is **unknown**

What kind of w is best?

PCA

sample $x = (x_1; x_2; x_3; \dots; x_n)$

$W = (w_1; w_2; w_3; \dots; w_d)$

$Z = W^T x$, $Z = (z_1; z_2; \dots; z_d), d \ll n$

Where, $Z = (z_1; z_2; \dots; z_d)$ are the Principle Component , z_i is scalar
 $w_j = (w_{j1}, w_{j2}, \dots, w_{jn})$ is a vector W is matrix $d \times n$

Question: How to compute W , how to get Z ?

Notes:

- 1) Every principle is the linear combination of original features
- 2) Number of principle component is lower than original feature's dimensions
- 3) The principle component can keep most information of original features
- 4) The principle components are uncorrelated with each other

PCA

Target: Maximum Separability

Object Function:

$$\max_W \text{tr}(W^T X X^T W) \quad s.t. \quad W^T W = I$$

Maximum variance

Lagrange multiplier method

$$\min_W -\text{tr}(W^T X X^T W) + \lambda(W^T W - I)$$

$$X X^T W = \lambda W$$

eigen decomposition

λ (可理解成向量)对应的是信息量的大小, w 对应的是投影方向

Solution: Eigen Decomposition

$$W = (w_1; w_2; w_3; \dots; w_d)$$

前d个最大的特征值 λ 对应的特征向量组成W

PCA

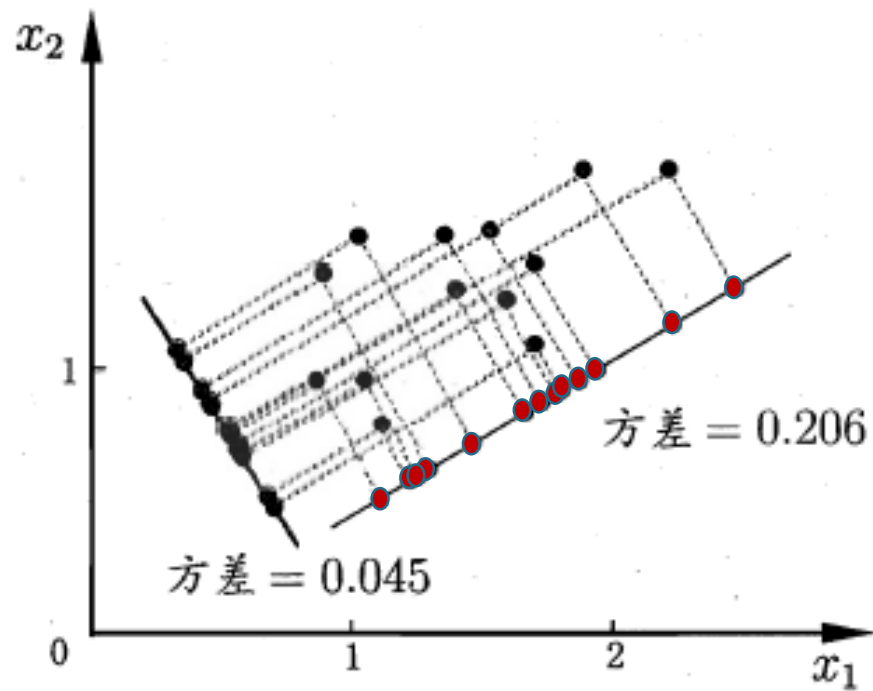
Algorithm

$$\max_W \text{tr}(W^T X X^T W) \quad \text{s.t.} \quad W^T W = I \quad \longrightarrow \quad X X^T W = \lambda W$$

- Let \bar{X} be the mean vector (taking the mean of all rows) 中心化
- Adjust the original data by the mean $X = X - \bar{X}$ 去中心化
- Compute the covariance matrix $X X^T$ of adjusted X
- Find the eigenvectors and eigenvalues of $X X^T$
- Get a matrix W consisted of d ordered eigenvectors
- $Z = W^T x$ is the result that want to get

PCA

For 2 dimensions data set



PCA

优点:

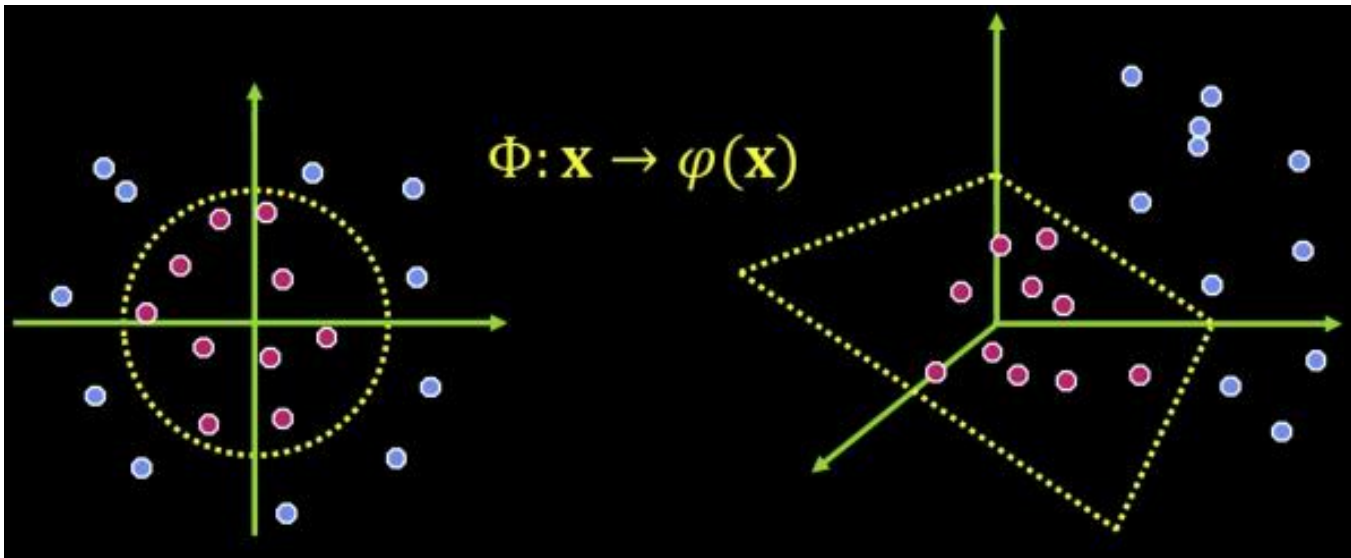
- PCA seeks to reduce the dimension of the data by finding a few orthogonal linear combinations(the PCs) of the original variables with the largest variance.(no class information)
(保证新空间中特征之间不相关的情况下，使变换后的特征维数更少实现降维和特征提取)

局限:

unsupervised

neglected component may contain category separate information (被忽略掉的成分可能也包含一些相对独立的信息)

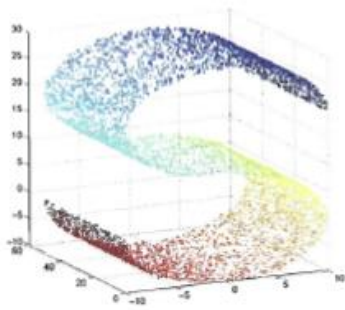
Kernel-PCA



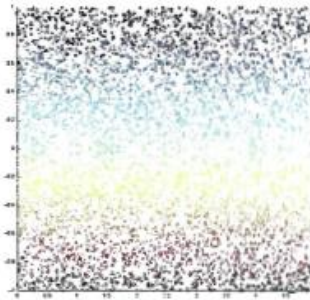
Nonlinearly separable
in low dimension

Linearly separable in
higher dimension

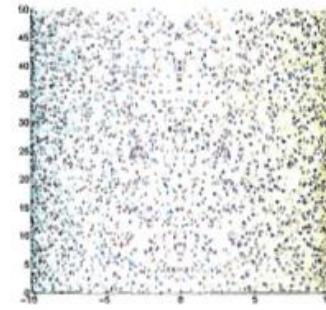
Kernel-PCA



↓
Embedding
in 3D space



↓
Intrinsic



↓
PCA

Kernel-PCA

PCA object function:

$$\max_W \operatorname{tr}(W^T X X^T W) \quad s.t. \quad W^T W = I$$

Kernel PCA object function:

$$\max_W \operatorname{tr}(W^T \underbrace{\varphi(X) \varphi(X)^T}_K W) \quad s.t. \quad W^T W = I$$



$$\max_W \operatorname{tr}(W^T \underbrace{K(X, X)}_K W) \quad s.t. \quad W^T W = I$$

Kernel trick:

低维空间上的Kernel变换代替高维空间上的内积

Kernel-PCA

$$z_i = \phi(x_i)$$

$$\left(\sum_{i=1}^m z_i z_i^T\right)W = \lambda W$$

$$\left(\sum_{i=1}^m \phi(x_i)\phi(x_i)^T\right)W = \lambda W$$

此时， W 没法求解。但是可以证明， W 是新的特征空间中样本的线性组合

$$W = \sum_{i=1}^m \phi(x_i)\alpha_i$$

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

$$KA = \lambda A$$

→ Eigenvalue Decomposition

For a new x , the j -th dimension after projection

$$j = 1, 2, \dots, d'$$

$$\begin{aligned} z_j &= w_j^T \phi(x) = \sum_{i=1}^m \alpha_i^j \phi(x_i)^T \phi(x) \\ &= \sum_{i=1}^m \alpha_i^j k(x_i, x) \end{aligned}$$

z_j 通过求得的 α 和核函数求解

Kernel-PCA

Common kernel function:

- Linear Kernel

$$k(x, y) = x^t y$$

- Polynomial Kernel

$$k(x, y) = (a x^t y + c)^d$$

- Gaussian Kernel

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- Exponential Kernel

$$k(x, y) = \exp\left(-\frac{\|x - y\|}{2\sigma^2}\right)$$

-

Kernel-PCA

- An extension of principal component analysis (PCA) using techniques of kernel methods
- Using a kernel, the originally linear operations of PCA are done in a Reproducing Kernel Hilbert Space with a non-linear mapping

Kernel-PCA

- 优点:

Solve the problem of **nonlinearly mapping** in low dimensional space
Applying kernel trick, don't increase computing complexity

- 局限:

There are no a fixed method to **choose a suitable Kernel function**

Kernel-PCA in Sklearn

```
print(__doc__)

# Authors: Mathieu Blondel
#          Andreas Mueller
# License: BSD 3 clause

import numpy as np
import matplotlib.pyplot as plt

from sklearn.decomposition import PCA, KernelPCA
from sklearn.datasets import make_circles

np.random.seed(0)

X, y = make_circles(n_samples=400, factor=.3, noise=.05)

kpca = KernelPCA(kernel="rbf", fit_inverse_transform=True, gamma=10)
X_kpca = kpca.fit_transform(X)
X_back = kpca.inverse_transform(X_kpca)
pca = PCA()
X_pca = pca.fit_transform(X)

# Plot results

plt.figure()
plt.subplot(2, 2, 1, aspect='equal')
plt.title("Original space")
reds = y == 0
blues = y == 1

plt.scatter(X[reds, 0], X[reds, 1], c="red",
            s=20, edgecolor='k')
plt.scatter(X[blues, 0], X[blues, 1], c="blue",
            s=20, edgecolor='k')
plt.xlabel("$x_1$")
plt.ylabel("$x_2$")

X1, X2 = np.meshgrid(np.linspace(-1.5, 1.5, 50), np.linspace(-1.5, 1.5, 50))
X_grid = np.array([np.ravel(X1), np.ravel(X2)]).T
# projection on the first principal component (in the phi space)
Z_grid = kpca.transform(X_grid)[: , 0].reshape(X1.shape)
plt.contour(X1, X2, Z_grid, colors='grey', linewidths=1, origin='lower')

plt.subplot(2, 2, 2, aspect='equal')
plt.scatter(X_pca[reds, 0], X_pca[reds, 1], c="red",
            s=20, edgecolor='k')
plt.scatter(X_pca[blues, 0], X_pca[blues, 1], c="blue",
            s=20, edgecolor='k')
plt.title("Projection by PCA")
plt.xlabel("1st principal component")
plt.ylabel("2nd component")

plt.subplot(2, 2, 3, aspect='equal')
plt.scatter(X_kpca[reds, 0], X_kpca[reds, 1], c="red",
            s=20, edgecolor='k')
plt.scatter(X_kpca[blues, 0], X_kpca[blues, 1], c="blue",
            s=20, edgecolor='k')
plt.title("Projection by KPCA")
plt.xlabel("1st principal component in space induced by $\phi$")
plt.ylabel("2nd component")

plt.subplot(2, 2, 4, aspect='equal')
plt.scatter(X_back[reds, 0], X_back[reds, 1], c="red",
            s=20, edgecolor='k')
plt.scatter(X_back[blues, 0], X_back[blues, 1], c="blue",
            s=20, edgecolor='k')
plt.title("Original space after inverse transform")
plt.xlabel("$x_1$")
plt.ylabel("$x_2$")

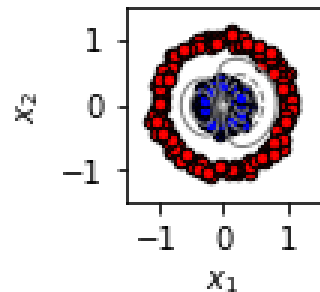
plt.subplots_adjust(0.02, 0.10, 0.98, 0.94, 0.04, 0.35)

plt.show()
```

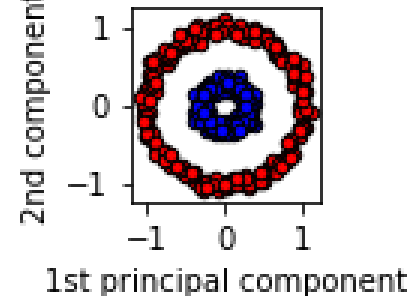
Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN (<http://sklearn.apachecn.org/cn/0.19.0/> (sklearn document))

Kernel-PCA in Sklearn

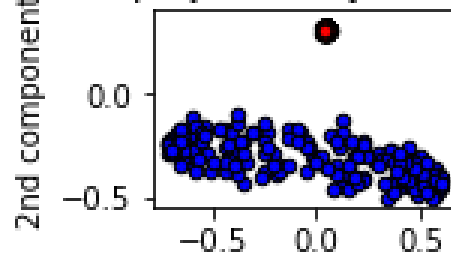
original space



projection by PCA

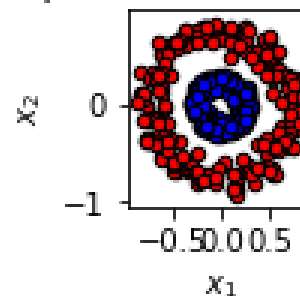


projection by KPCA



1st principal component in space induced by ϕ

original space after inverse transform



Extension

□ Linear Methods

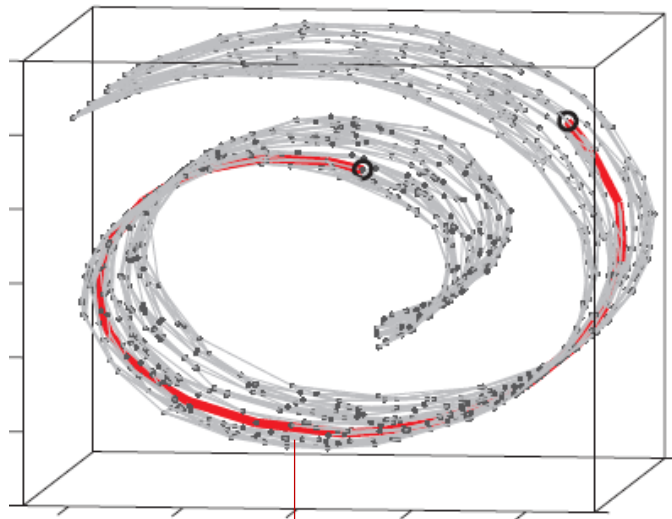
- Principal Component Analysis (PCA)[Pearson , 1901]
- Linear Discriminant Analysis (LDA) [Ronald Fisher , 1936]
[Belhumeur, 1996]

□ Nonlinear Methods

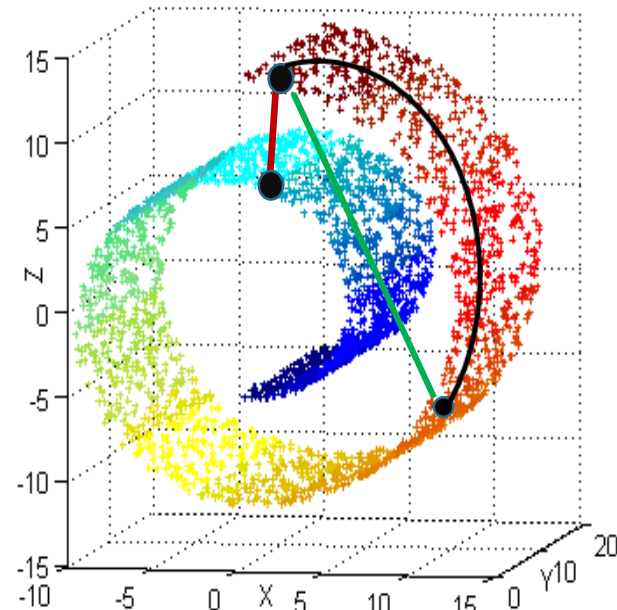
- Multidimensional Scaling (MDS) [Torgerson, W.S. et al. ,1958]
- Kernel principal component analysis (KPCA) [Scholkopf et al., 1998]
- Principal Curves [Hastie, 1989]
- Self-Organizing Feature Map (SOM) [Kohonen et al., 1995]
- Generative topographic map (GTM) [Bishop et al., 1998]
- Manifold Learning: Isomap,LLE,LE.....
-

Isomap

- Isomap (Isometric mapping)
- seeks to preserve the intrinsic geometry of the data, as captured in the geodesic manifold distances between all pairs of data points.

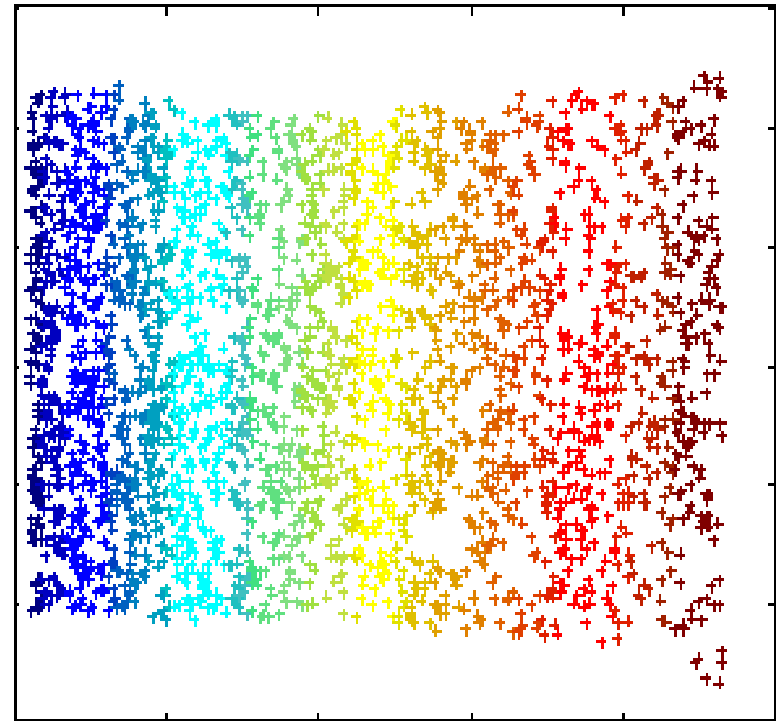
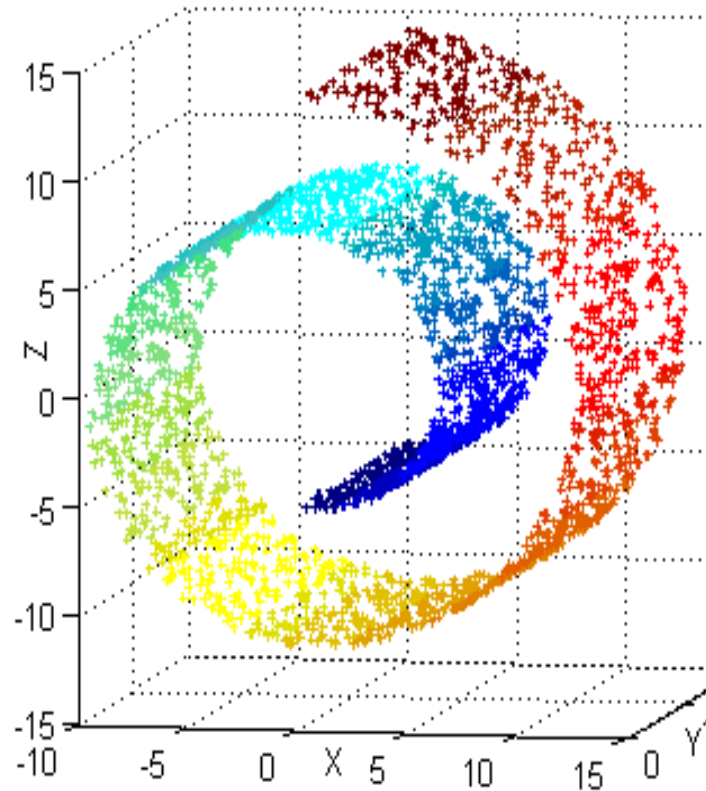


Geodesic distance

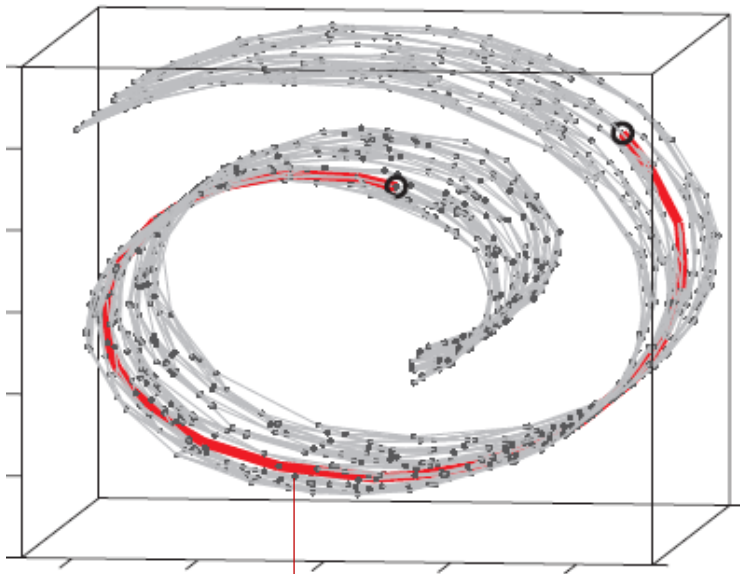


Euclidian distance

Isomap

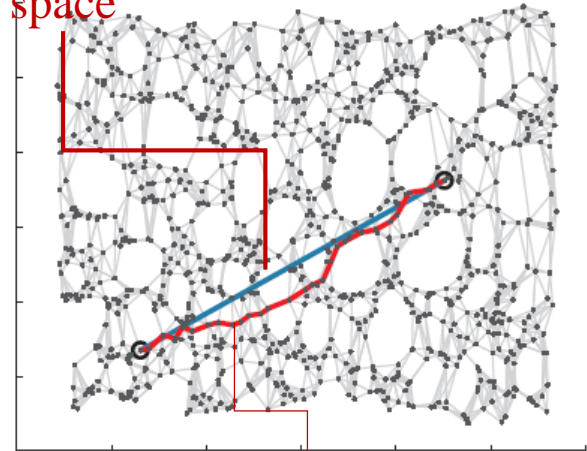


Isomap



Godesic distance

Euclidian distance in
embedding space



Shortest path distance
approximate geodesic distance

Isomap

优点:

For manifold data, the **intrinsic attitude** can **be reflected** in low dimension

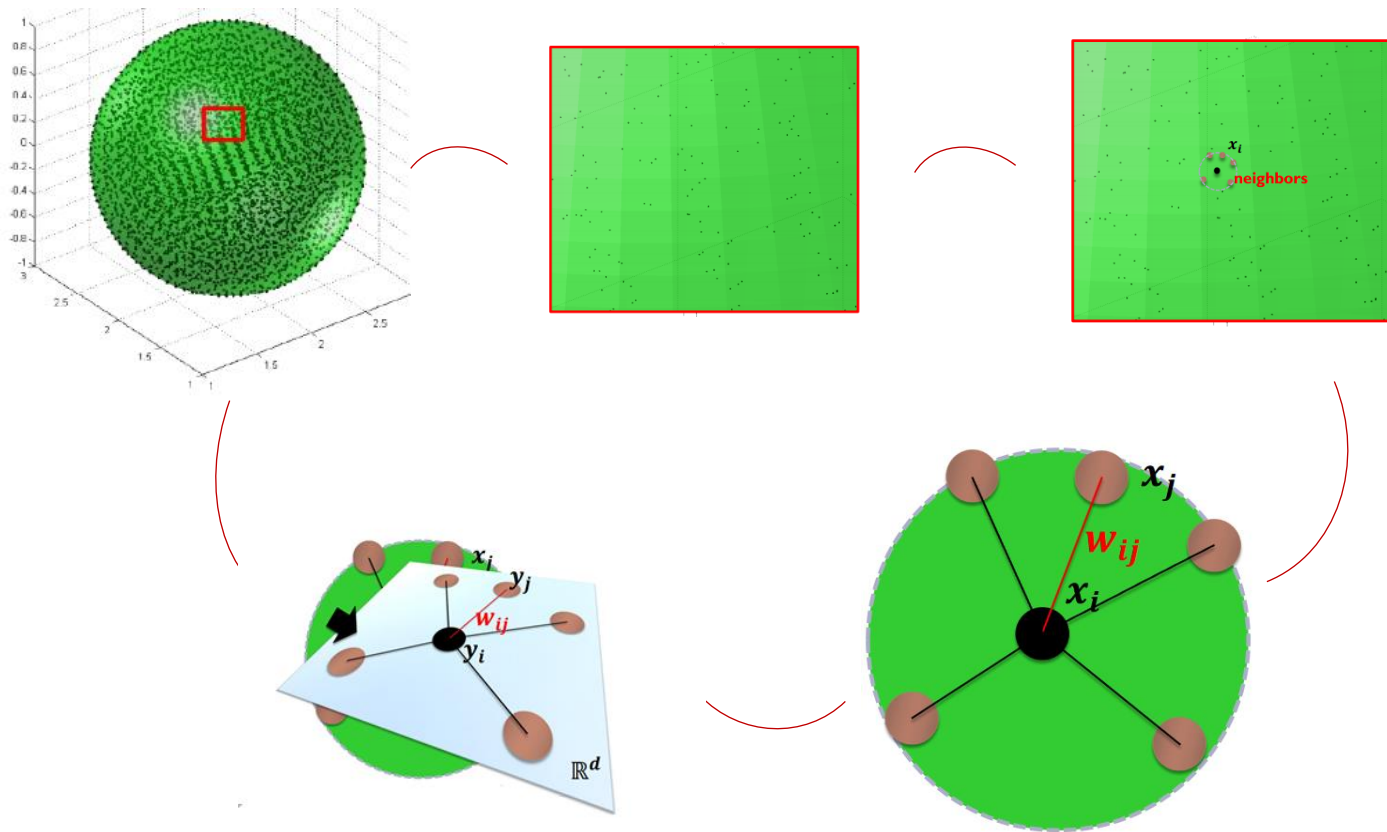
缺点:

Feature in low dimension space **don't contain the category separable information**

Geodesic distance is hard to compute

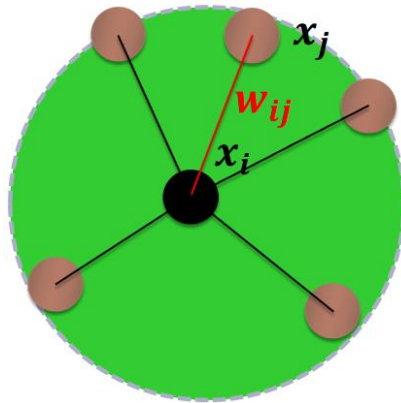
LLE

- Locally Linear Embedding(LLE) [Roweis et al., 2000]



LLE

- LLE constructs a neighborhood-preserving mapping

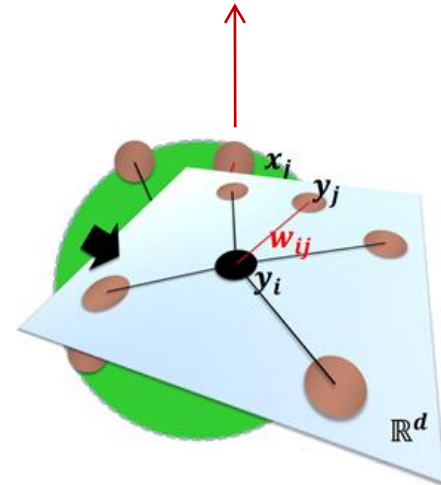


$$\arg \min_{w_i} \left\| x_i - \sum_{j \sim i} w_{ij} x_j \right\|^2$$

Compute the best reconstruct weight w

Compute the best embedding coordinate y

$$\arg \min_y \left\| y_i - \sum_{j \sim i} w_{ij} y_j \right\|^2$$



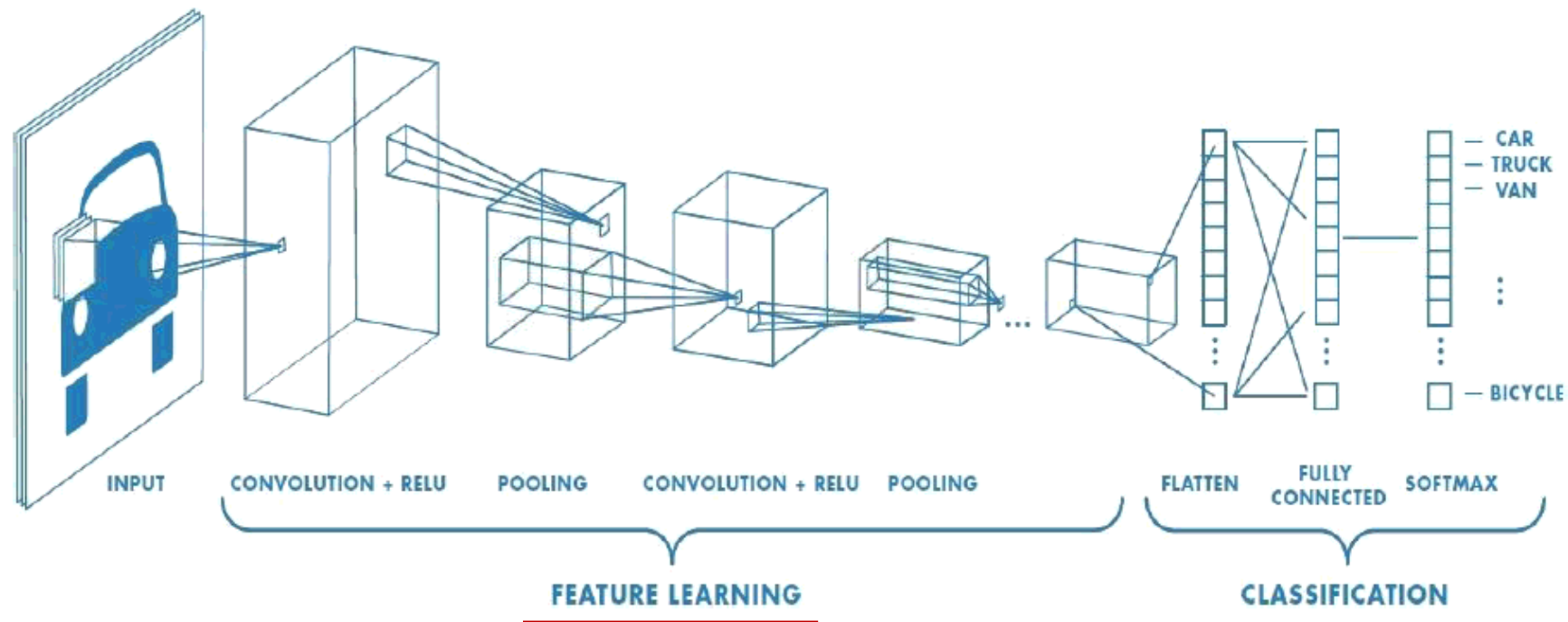
LLE

- 优点:
 - It can process nonlinear signal
 - It can keep the local intrinsic attitude
- 局限:
 - Manifold is locally linear
 - Sensitive to noise

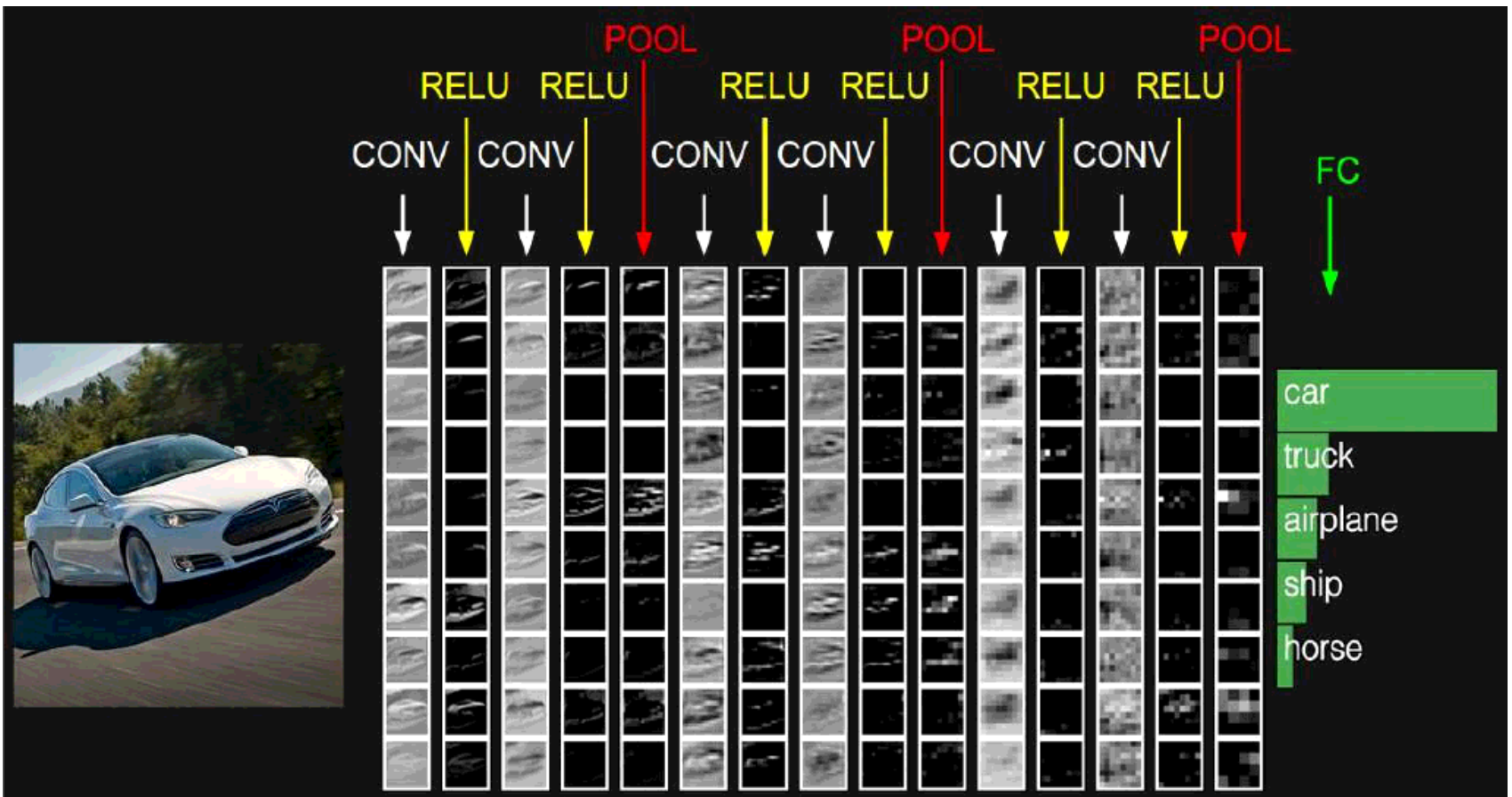
Outline

- ❑ Introduction of Feature
- ❑ Feature Selection
- ❑ Feature Extraction
- ❑ Feature Learning
- ❑ Summary

Feature Learning



Feature Learning



<http://cs231n.github.io/convolutional-networks/>

Outline

- ❑ Introduction of Feature
- ❑ Feature Selection
- ❑ Feature Extraction
- ❑ Feature Learning
- ❑ Summary

Summary

❑ Introduction of Feature

❑ Feature Engineering

➤ Feature Selection

filter, wrapper, embedding

➤ Feature Generation(Feature Design & Feature Extraction)

linear & nonlinear methods

❑ Feature Learning

➤ Deep Learning

Resources

- <http://www.cnblogs.com/heaad/archive/2011/01/02/1924088.html>
- <http://blog.csdn.net/j123kaishichufa/article/details/7679682>
- <http://www.docin.com/p-1446129160.html?docfrom=rrela>
- <http://www.doc88.com/p-9425192872197.html>
- <http://www.cnblogs.com/bayesianML/p/6397911.html>
- <http://www.docin.com/p-670764298.html>
- 张道强, 陈松灿. 高维数据降维方法
<http://www.doc88.com/p-602277996097.html>
- <http://www.doc88.com/p-6072038641028.html>
- 周志华。《机器学习》第十章、第十一章
- 边肇祺, 张学工。《模式识别》第八章
- <http://sklearn.apachecn.org/cn/0.19.0/>
- Sergios Theodoridis, et al. *Pattern Recognition (Second Edition)*

Home Works

1. Please explain the role of feature selection and feature extraction in Machine Learning
2. Please explain the difference between feature selection and feature extraction
3. Please try to derive the mapping process of PCA, LDA and Kernel PCA and do experiment in Sklearn



Thanks !