NUS AI SUMMER EXPERIENCE 2019

Pattern Recognition

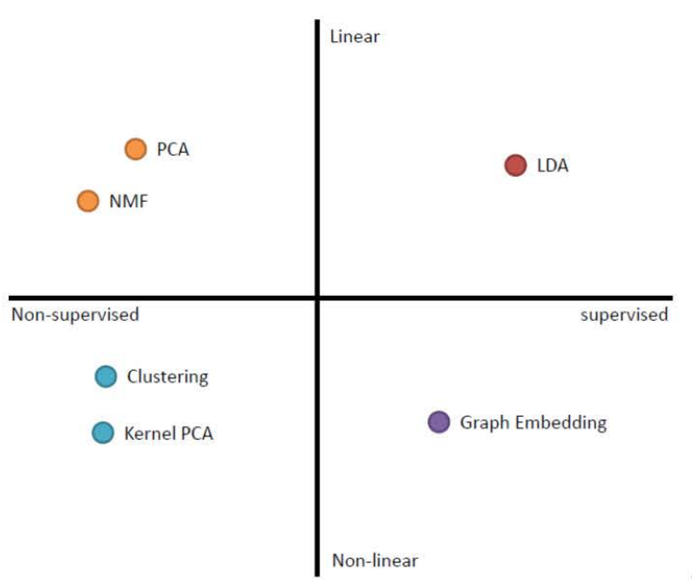
Tan Sing Kuang
陈星旷

NATIONAL UNIVERSITY OF SINGAPORE
DEPARTMENT OF ISEM

Pattern Recognition

- Dimension Reduction
  - Principal Component Analysis (PCA)
  - Non-Negative Matrix Factorization (NMF)
  - Graph Embedding
- Classification
  - Linear Discriminant Analysis (LDA)
  - Support Vector Machine (SVM)
  - Boosting

- Clustering
  - K-Means
  - Hierarchical Clustering
  - Gaussian Mixture Model (GMM)



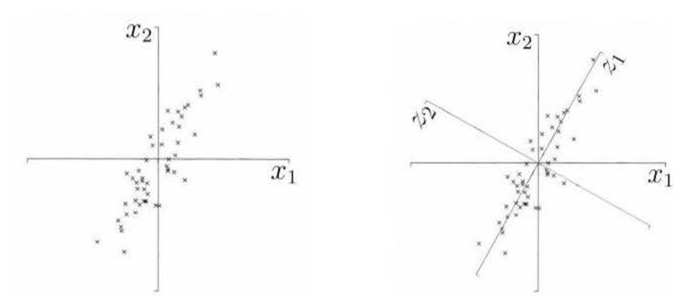
Dimension Reduction

- To prevent the curse of dimensionality
  - Datapoints become so sparse as number of dimension increases
  - No longer able to do pattern recognition using approach such as k nearest neighbours (knn)
- Reduce time and storage space
- Better interpretation of the data
- Easier to visualize the data (especially in 2D or 3D)

- Clustering can also be used for dimension reduction
  - 1000 datapoints, after clustering, reduced to 10 datapoints

Principal Component Analysis (PCA)

- Definition (principal component)
  - Find $z_1 = a_1^T x$ where var[z_1] is maximum
  - Data is a stretched point cloud
  - Trying to find the linear projection that gives us the most information



- To find a_1,
  - $var[z_1] = E\left((z_1 - \bar{z_1})^2\right) = \frac{1}{n}\sum_{i=1}^{n}(a_1^T x_i - a_1^T \bar{x})^2$
  - $= \frac{1}{n}\sum_{i=1}^{n} a_1^T(x_i - \bar{x})(x_i - \bar{x})^T a_1 = a_1^T S a_1$
  - Trying to find the projection of data x_i to a_1 so that has the largest variance
  - Where $S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$ is the covariance matrix
    - $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the mean
  - Solution
    - a_1 is the eigenvector of S with the largest eigenvalue
    - Then find the next orthogonal principal component

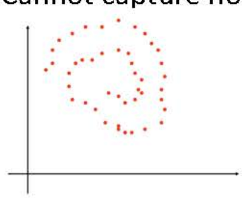
Practical Computation

- In practice, we use singular value decomposition (SVD) to find the principal components
- Centered data matrix:
  - $X_{d,n} = [(x_1 - \bar{x}), \dots, (x_n - \bar{x})]$
- Compute its SVD:
  - $X = U_{d,d} D_{d,n} \left(V_{n,n}\right)^T$
- $S = XX^T = UD^2U^T$
  - The columns of U are the eigenvectors of S
  - Diagonal elements of D_2 are the eigenvalues
  - Select the eigenvectors with the top k eigenvalues as the principal components

Classification with PCA

- Project training and test data into principal components space
- For the test data, use nearest neighbours (NN) for classification
- Accuracy is sensitive to the number of principal components

- Disadvantage
  - PCA is based on covariance of the samples, disregard the class-membership
  - Cannot capture non-linear structures such as manifold



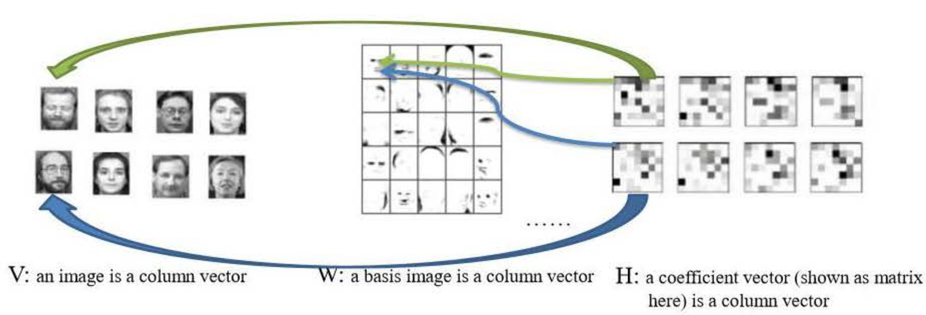
Number of Principal Components to Keep

- We can use the following measure to decide the number of principal components p to keep
  - $\frac{\sum_{i=1}^{p} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq Threshold$ (e.g. 0.95)

Non-negative Matrix Factorization NMF

- PCA do adding and subtraction of basis vectors
- Subtracting does not make sense in some of the applications
  - How do we subtracts a face?
  - What is the meaning of subtracting in the context of document classification?

- Matrix factorization: $V \approx WH$
  - V is a matrix where its columns contain facial images
  - W is a matrix where its columns contain basis images
  - H is a matrix where its columns contain encodings



Interpretation

- Using non-negative basis vectors make intuitive sense
  - Has physical interpretations
- Leads to nice basis vectors
  - During reconstruction of the image, we simply add in more basis vectors
  - Each basis vectors represent the parts of the object in the image

Learning the Basis Vectors

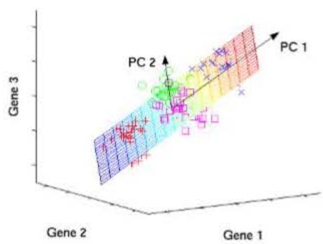
- Definition
  - $V_{iu} = (WH)_{iu} = \sum_{a=1}^{r} W_{ia} H_{au}$
- Gradient Descent Rule:
  - $H_{au} \leftarrow H_{au} + \eta_{au}[(W^T V)_{au} - (W^T WH)_{au}]$
  - Set $\eta_{au} = \frac{H_{au}}{(W^T WH)_{au}}$
- The update rule becomes
  - $H_{au} \leftarrow H_{au} \frac{(W^T V)_{au}}{(W^T WH)_{au}}$

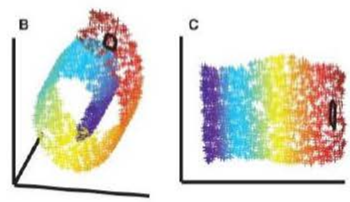
The derivation of the rules is very complex
You can try it yourself

For the details, please read the original paper
http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.pdf

Graph Embedding (GE)

- Linear subspace vs manifold



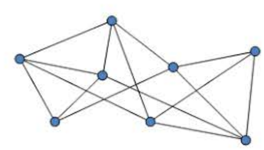


Unfolding the Swiss roll

How to flatten the Swiss roll?

Mathematics

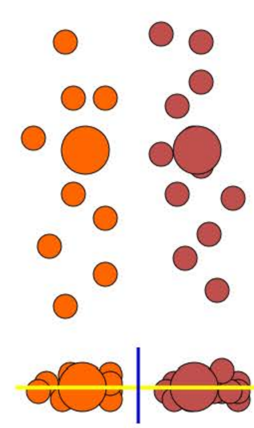
- A graph G=(x_i,S_{ij})
  - x_i is a datapoint and
  - S_{ij} is a similarity matrix
    - Is 1 when point x_i connects to point x_j
    - Can be found by nearest neighbours of x_i
- Define L=D-S, $D_{ii} = \sum_{j \neq i} S_{ij}$
- y_i is a low dimension representation (assume 1D)
- y*= $\min_{y^T y=1} \sum_{i \neq j} \left\| y_i - y_j \right\|^2 S_{ij}$
  - which is similar to $\min_{y^T y=1} y^T L y$



- Assume there is a linear mapping from x_i to y_i,
  - $y = X^T w$
- Objective function for the linearization
  - $w^* = \min_{w^T w=1} w^T X L X^T w$
- XLX^T is semi positive definite
  - So the solution can be found by minimum eigenvalue solution

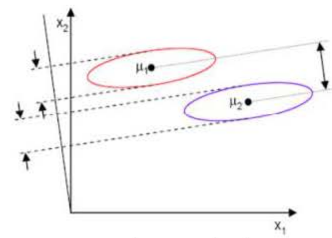
Linear Discriminant Analysis (LDA)

- LDA is to find the most discriminative projection
  - Maximizing between-class distance
  - Minimizing within class distance



Mathematics

- Definitions
  - Project samples x on a line to get scalar
    - $y = w^T x$
  - Projected means between two classes
    - $J(w) = |\widetilde{u_1} - \widetilde{u_2}| = |w^T(u_1 - u_2)|$
  - Variance of a class
    - $\widetilde{S_i^2} = \sum_{y \in c_i} (y - \widetilde{u_i})^2$
  - Fisher linear discriminant
    - $J(w) = \frac{|\widetilde{u_1} - \widetilde{u_2}|}{\widetilde{S_1^2} + \widetilde{S_2^2}}$



LDA best projections

- Define
  - Convariance matrix of samples x
    - $S_i = (x - u_i)(x - u_i)^T$
  - $S_1 + S_2 = S_w$
  - The within class scatter of projection y
    - $\widetilde{S_i^2} = \sum_{y \in c_i} (y - \widetilde{u_i})^2 = \sum_{y \in c_i} (w^T x - w^T u_i)^2$
    - $= \sum_{y \in c_i} w^T (x - u_i)(x - u_i)^T w = w^T S_i w$
    - $\widetilde{S_1^2} + \widetilde{S_2^2} = w^T S_w w$

- The difference of projected means
  - $(\widetilde{u_1} - \widetilde{u_2})^2 = (w^T u_1 - w^T u_2)^2$
  - $w^T (u_1 - u_2)(u_1 - u_2)^T w$
  - $w^T S_B w$ Note that S_B is rank 1

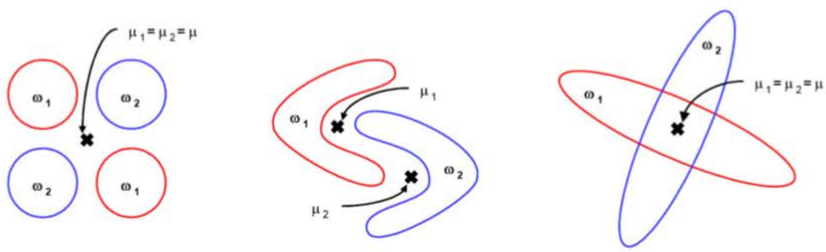
- Fisher criterion, $J(w) = \frac{w^T S_B w}{w^T S_W w}$
- The maximum of J(w) using derivative and set to zero
  - $S_W^{-1} S_B w = J(w) w$
  - w* is the largest eigenvector of $S_W^{-1} S_B$

Multi-class LDA

- Use C-1 projections instead of 1 projection
- $W=[w_1|w_2|\ldots|w_{C-1}]$:
  - $y_i = w_i^T x$
  - $y = W^T x$
- $J(W) = \frac{|W^T S_B W|}{|W^T S_W W|}$
- $W^* = \arg\max \frac{|W^T S_B W|}{|W^T S_W W|}$ implies $(S_B - \lambda_i S_W) w_i^* = 0$
  - Where $\lambda_i = J(w_i) = scalar$
  - W* has columns which are the eigenvectors corresponding to the largest eigenvalues

Limitations

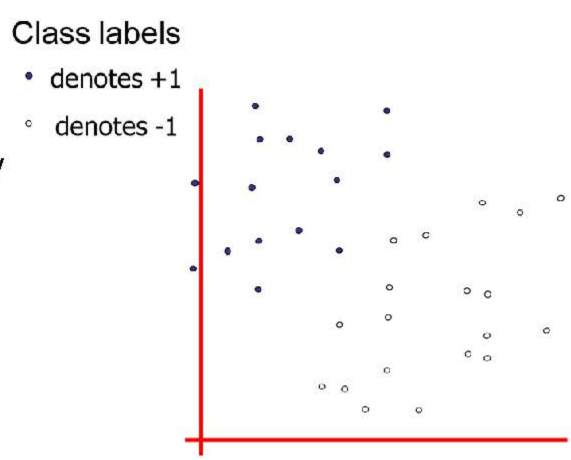
- LDA produces at most C-1 projections of the features
  - Where C is the number of classes
  - Need more features to discriminate the classes if needed
- LDA assumes unimodal Gaussian likelihoods



Failure cases

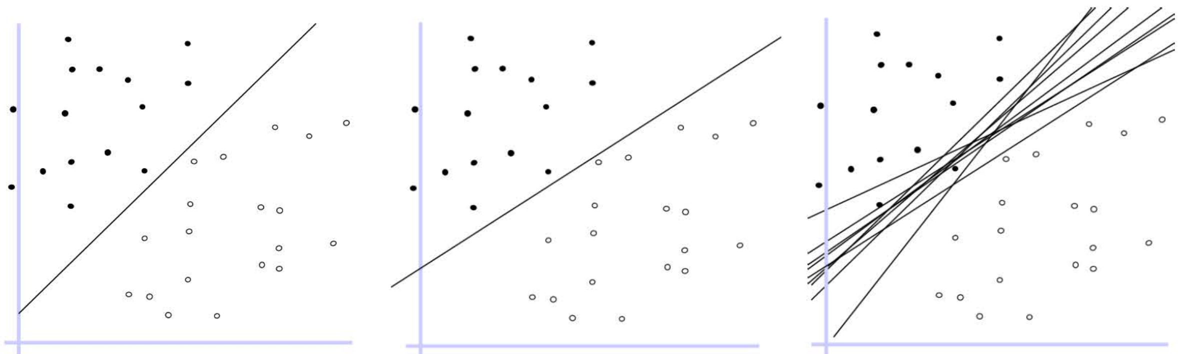
Support Vector Machine (SVM)

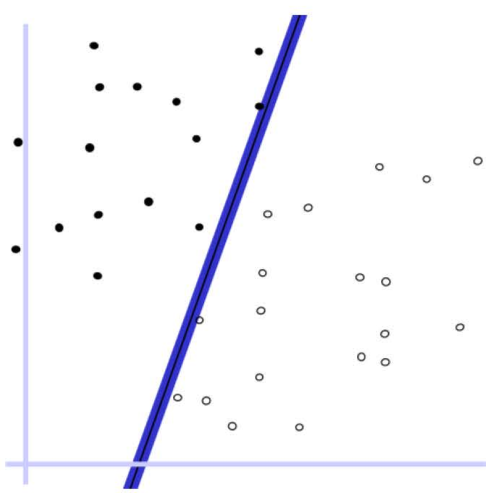
- Given two sets of data points
  - One set of negative class
  - One set of positive class
- What is the best way to classify this data?



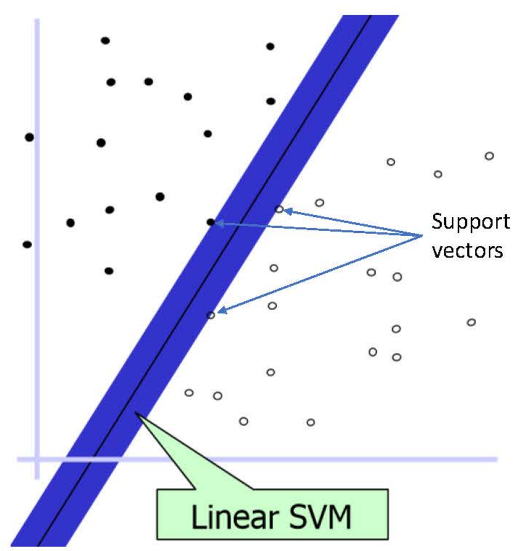
Best Separating Hyperplane?

- Multiple choices for classifying these datapoints
  - Which is the best?





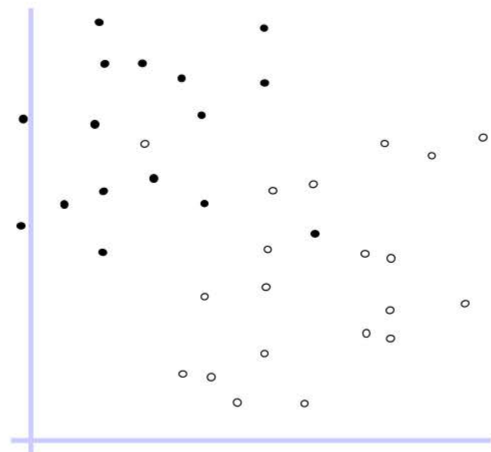
Not so good



The best linear classifier is the one with the largest margin between the two classes of points

Linearly Non-Separable Data

- What if the data is not able to be separated linearly?
- Solutions
  - Ignore a few points that are misclassified
  - Map data into kernel space (higher dimension space)
    - So that it is more linearly separable



More than two class?

- Split the data into N binary classes
  - Class 1 vs the rest of the data
  - Class 2 vs the rest of the data
  - …
  - Class N vs the rest of the data
- Assign the class of a new input to the class that is furthest from the separating plane in the positive region

Mathematics

$$\text{Minimize } \|w\|$$
$$\text{subject to } y_i(w^T x - b) \geq 1$$

- The constraint is to ensure that the datapoints are a least 1 unit distance away from the separating plane
- The minimization of w is used to maximize the margin between the two class of datapoints

$$\text{Minimize } \lambda\|w\| + \frac{1}{n}\sum_i \xi_i$$
$$\text{subject to } y_i(w^T x - b) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

- The ξ_i terms are added to relax the boundary so that some points can be misclassified
- This is needed for data that is not linearly separable with some small set of points that cannot be classified correctly during learning

Dual Problem and Kernel Trick

$$\text{maximize } \sum_i a_i - 0.5 \sum_{i,j} a_i a_j y_i y_j \phi(x_i)^T \phi(x_j)$$
$$\text{Subject to } a_i \geq 0$$
$$\sum_i a_i y_i = 0$$

x_i with non-zero a_i are the support vectors

For linear kernel (linear classification), $\phi(x_i) = x_i$

For non linear kernel, we can replace $\phi(x_i)^T \phi(x_j)$ by $k(x_i,x_j)$

Linear kernel: $k(x_i,x_j)=x_i^T x_j$

Quadratic kernel: $k(x_i,x_j)=\left(x_i^T x_j + 1\right)^2$

Polynomial kernel: $k(x_i,x_j)=\left(x_i^T x_j + 1\right)^n$

Radial Basis Function kernel: $k(x_i,x_j)=e^{-\frac{\|x_i - x_j\|^2}{\sigma}}$

Boosting

- Question posed by Kearns and Valiant (1988, 1989):
  - "Can a set of **weak learners** create a single **strong learner**?"
- Defines a classifier using an additive model:
  - $F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \cdots$

Weak classifier

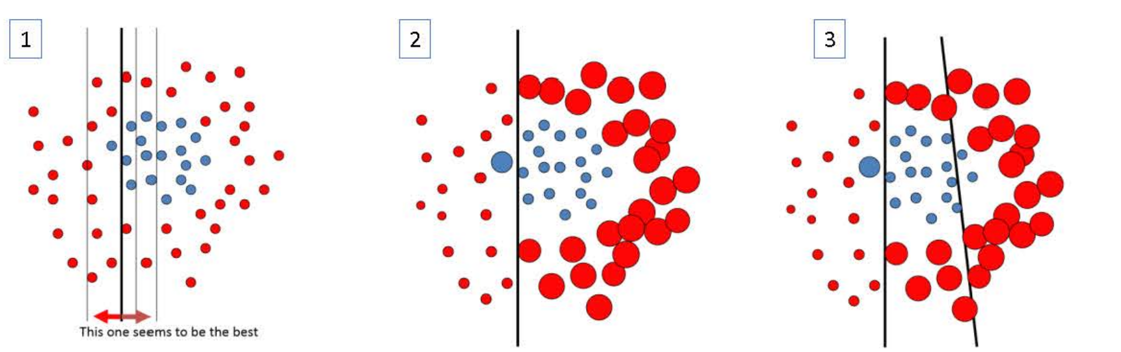
Strong classifier

Features vector

Weight

A weak classifier performs slightly better than chance

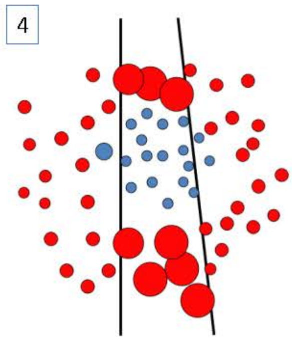
Toy Example



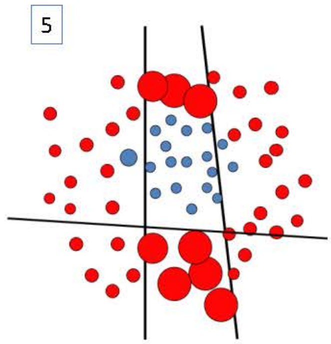
Learn the first weak classifier

Reweighting the samples

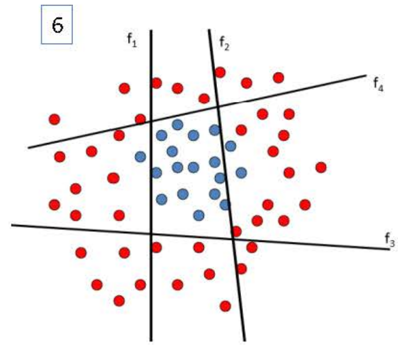
Learn another weak classifier



Reweighting the samples



Learn another weak classifier



Continue learning and add the final weak classifier

Gentle Boosting

- Boosting fits using the additive model
  - $F(x) = f_1(x) + f_2(x) + f_3(x) + \cdots$
- By minimizing this exponential loss
  - $J(F) = \sum_{t=1}^{N} e^{-y_t F(x_t)}$

Training samples

- Sequentially at each step, we add new weak classifier
  - $F(x) \leftarrow F(x) + f_m(x)$
- To minimize the residual loss
  - $(\phi_m) = \arg\min_{\phi} \sum_{t=1}^{N} J(y_t, F(x_t) + f(x_t;\phi))$

Parameters of weak classifier

Desired Output

Input

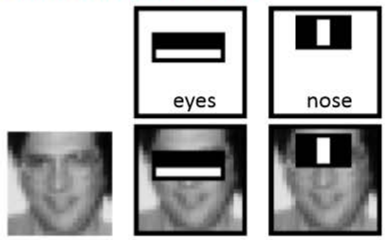
- At each iteration:
  - We select $f_m(x)$ that minimizes the cost:
    - $J(F + f_m) = \sum_{t=1}^{N} e^{-y_t(F(x_t)+f_m(x_t))}$
  - This is the same as minimizing the approximation of the error
    - $J(F) \propto \sum_{t=1}^{N} \boxed{e^{-y_t F(x_t)}} \big(y_t - f_m(x_t)\big)^2$

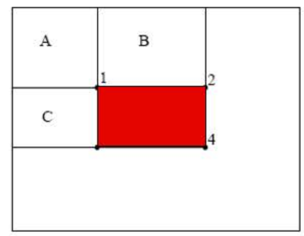
Weights at this iteration

Examples of Weak Detectors

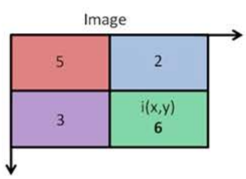
Haar filters and integral image

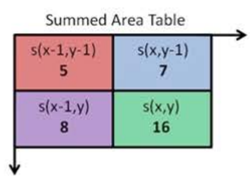
Viola and Jones, ICCV 2001

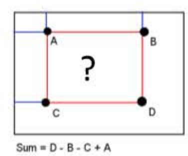




The average intensity in the block is computed with four sums independently of the block size.

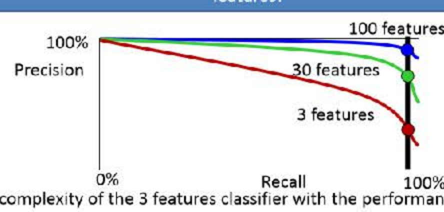




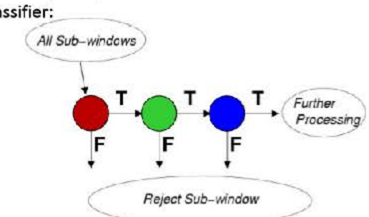


Cascade of Classifier

What is the motivation: some negative samples may be rejected based on few features!



We want the complexity of the 3 features classifier with the performance of the 100 features classifier:

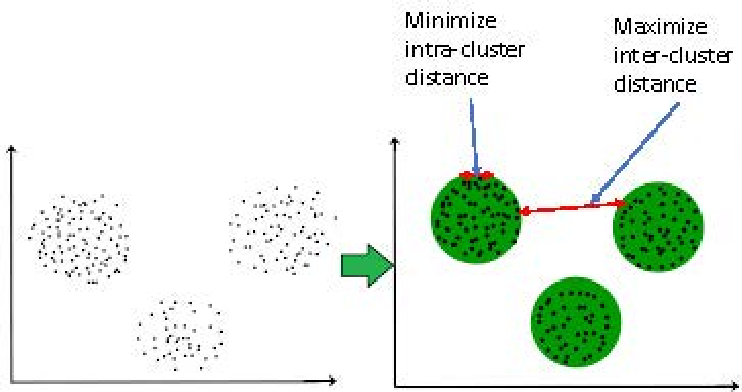


Select a threshold with high recall for each stage.

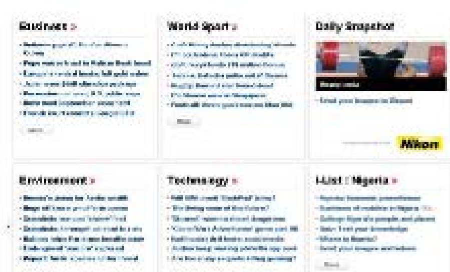
We increase precision using the cascade

What is clustering?

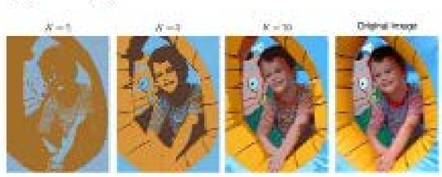
- The objective of clustering is to find objects in a group that are similar to one another and different from other objects in other group



Applications



- Understand data and searching
  - Group documents
  - Genes and Proteins
  - Stocks with similar price fluctuations
- Visualization of data
  - Reduce the size of large data
- Image segmentation

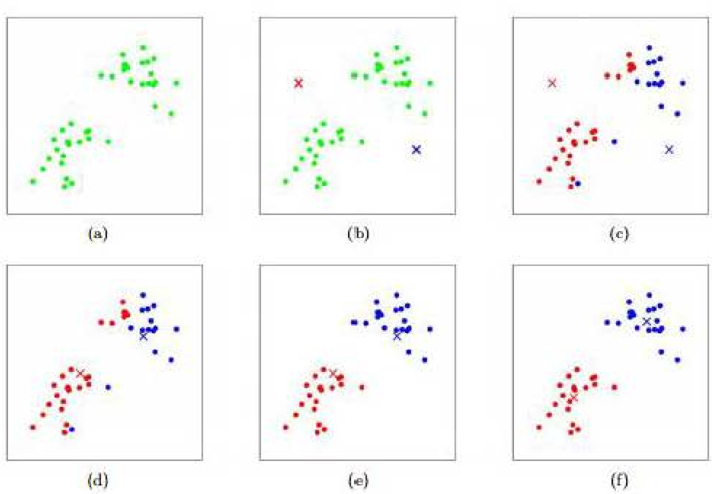


Clustering rain fall amount in Australia

K-means

- K-means is one of the simplest clustering algorithm
  - Can be used to explore of the data
- Algorithm description
  - Initialize with random K centroids
  - Repeat
    - Assign all the points to their nearest centroids
    - Compute the centroid of each cluster
  - Until convergent (the centroids do not change)

Visually

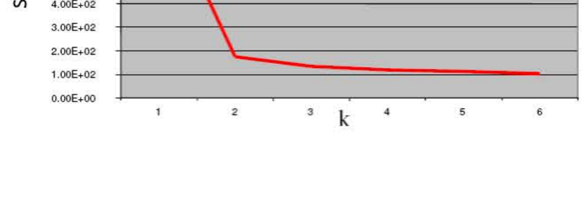


Measure

- One common measure is Sum of Squared Error (SSE)
  - $SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$
- x is a datapoint, C_i is cluster i and m_i is the centroid of cluster i
- We can do a few clustering and choose the best using SSE
  - Since K-means uses random initial centroids which leads to random results, we may need to do a few clustering to get good result

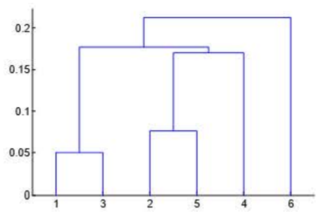
Problems of K-means

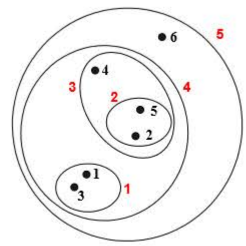
- Choose the optimal number of clusters K
  - We can use the elbow method to select K
- Choosing initial centroids
  - Do multiple runs
  - Use hierarchical clustering to determine initial centroids
  - Use more than K centroids
    - Select the clusters that are the most widely separated among these centroids
  - Post processing
    - Eliminate small clusters as outliers
    - Split and merge clusters



Hierarchical Clustering

- Obtain nested set of clusters





Can be visualized as a dendrogram
With sequences of mergers or splits

Advantages and types

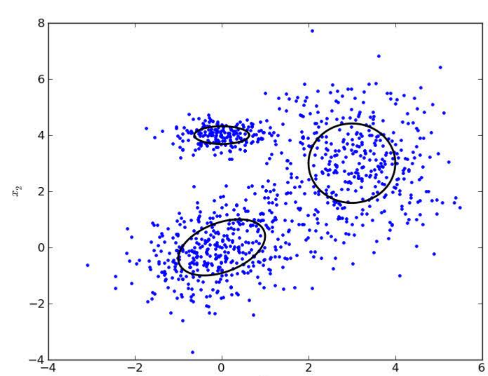
- Advantage
  - No need to decide on the number of clusters
- Types
  - Agglomerative
    - Clustering through repeated merging of small clusters
  - Divisive
    - Clustering through repeated splitting of clusters

Agglomerative Clustering Algorithm

- Basic algorithm is straightforward
- 1. Compute the proximity/distance matrix
- 2. Let each data point be a cluster
- 3. Repeat
  - 4. Merge the two closest clusters
  - 5. Update the proximity/distance matrix
- 6. Until only a single cluster remains

Gaussian Mixture Model (GMM)

- Objective of Gaussian Mixture Model
  - Is to learn the clusters of Gaussian distributed data points
  - Each cluster has their own mean and covariance



- The clusters are determined using the EM algorithm
- EM is a method that alternates between an Expectation (E) step and a Maximization (M) step
- E-step
  - Compute the expected classes of the datapoints
- M-step
  - Re-estimate the parameters
    - Means and covariances of each cluster

Mathematically

- E-step
  - Compute the expected classes (clusters) of the datapoints
  - Keeping the means and covariances fixed

$$z_k^n = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n|\mu_j, \Sigma_j)}$$

- M-step
  - Compute the mean and covariance of each cluster
  - Keeping the classes fixed

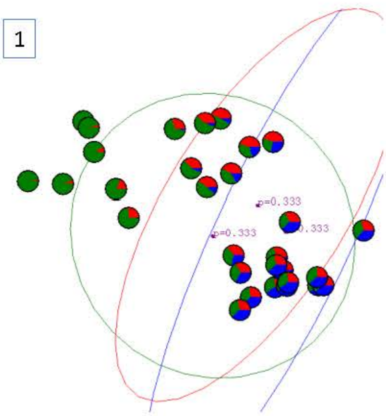
$$\mu_k^{new} = \frac{\sum_{n=1}^{N} z_k^n x_n}{\sum_{n=1}^{N} z_k^n}$$

$$\Sigma_k^{new} = \frac{\sum_{n=1}^{N} z_k^n (x_n - \mu_k^{new})(x_n - \mu_k^{new})^T}{\sum_{n=1}^{N} z_k^n}$$

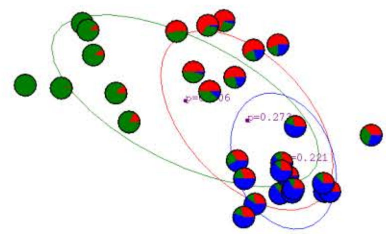
$$\pi_k^{new} = p(\omega_k)^{new} = \frac{\sum_{n=1}^{N} z_k^n}{N}$$

Visually

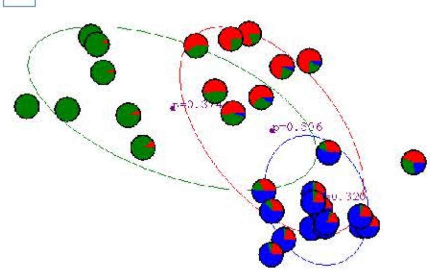
1



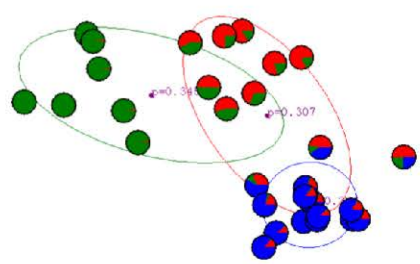
2



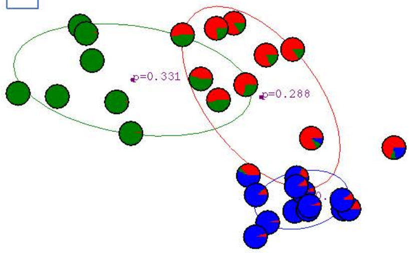
3



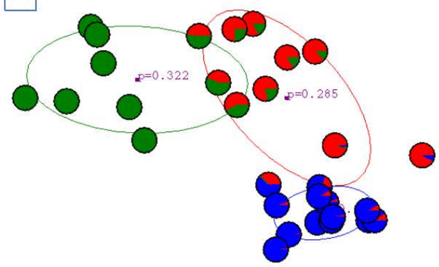
4



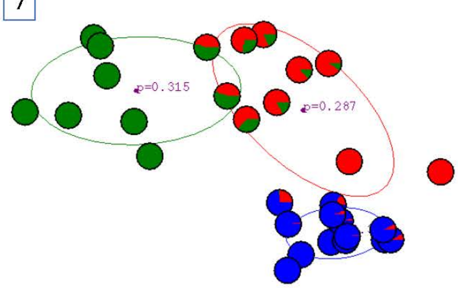
5



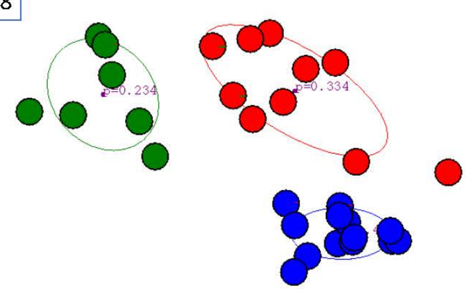
6



7



8



Lab exercises

- Principal Component Analysis
  - https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60
- Manifold learning
  - https://jakevdp.github.io/PythonDataScienceHandbook/05.10-manifold-learning.html
- Boosting
  - https://machinelearningmastery.com/visualize-gradient-boosting-decision-trees-xgboost-python/
- Clustering
  - https://towardsdatascience.com/an-introduction-to-clustering-algorithms-in-python-123438574097
- Support Vector Machine
  - https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html
- Linear Discriminant Analysis
  - https://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html#sphx-glr-auto-examples-decomposition-plot-pca-vs-lda-py

Further exercises

- Experiment with tabular data
  - http://archive.ics.uci.edu/ml/datasets/Travel+Reviews
  - http://archive.ics.uci.edu/ml/datasets/Iris
  - http://archive.ics.uci.edu/ml/datasets/Heart+Disease
- Sklearn datasets
  - https://scikit-learn.org/stable/datasets/index.html
- Use dimension reduction (e.g. clustering, PCA, manifold learning)
  - To visualize the pattern in the data
- Try all classification algorithms (e.g. SVM, LDA, boosting)
  - To see which one is better
- Compare the advantages and disadvantages of all algorithms
- Reading materials
  - https://towardsdatascience.com/3-ways-to-load-csv-files-into-colab-7c14fcbdcb92
- More datasets to try if you have time
  - http://archive.ics.uci.edu/ml/datasets.php

Tan Sing Kuang 陈星旷

isetsk@nus.edu.sg

TanSingKuang

NUS AI SUMMER EXPERIENCE 2019

Thank You

Pattern Recognition

NATIONAL UNIVERSITY OF SINGAPORE
DEPARTMENT OF ISEM