Clustering Algorithms and their Application to Facial Image Analysis

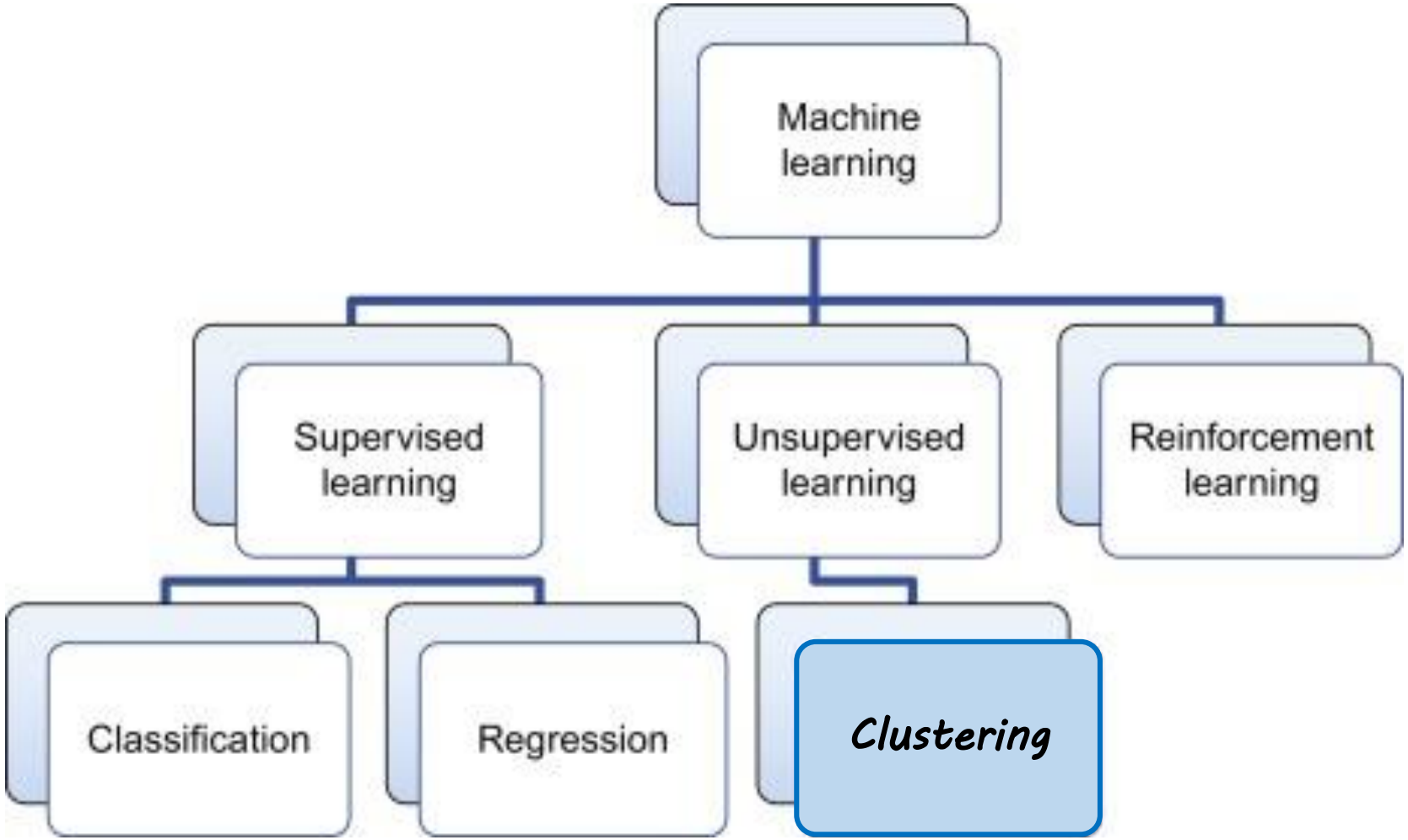
Hamid Sadeghi

nextera, FaceCup 1400 (2022)

Contents

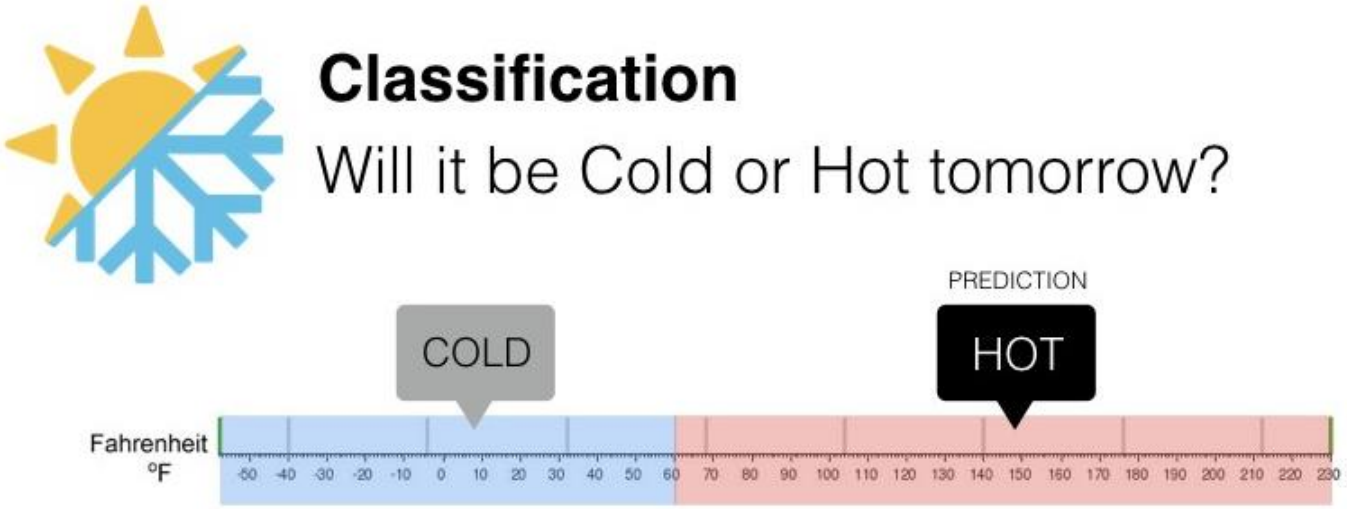
- Introduction
- Clustering Algorithms
- Evaluation
- Face Analysis

Introduction



Handbook of Statistics, Chapter 8: Machine Learning, Elsevier, 2018

Introduction (Classification vs Regression)

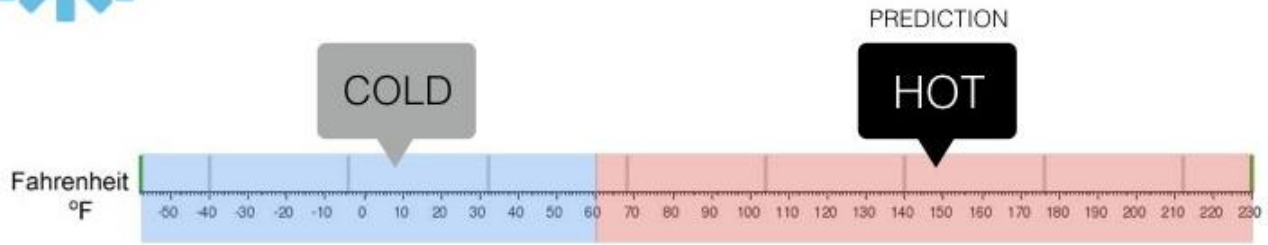


www.pinterest.com

Introduction (Classification vs Regression)

Classification

Will it be Cold or Hot tomorrow?

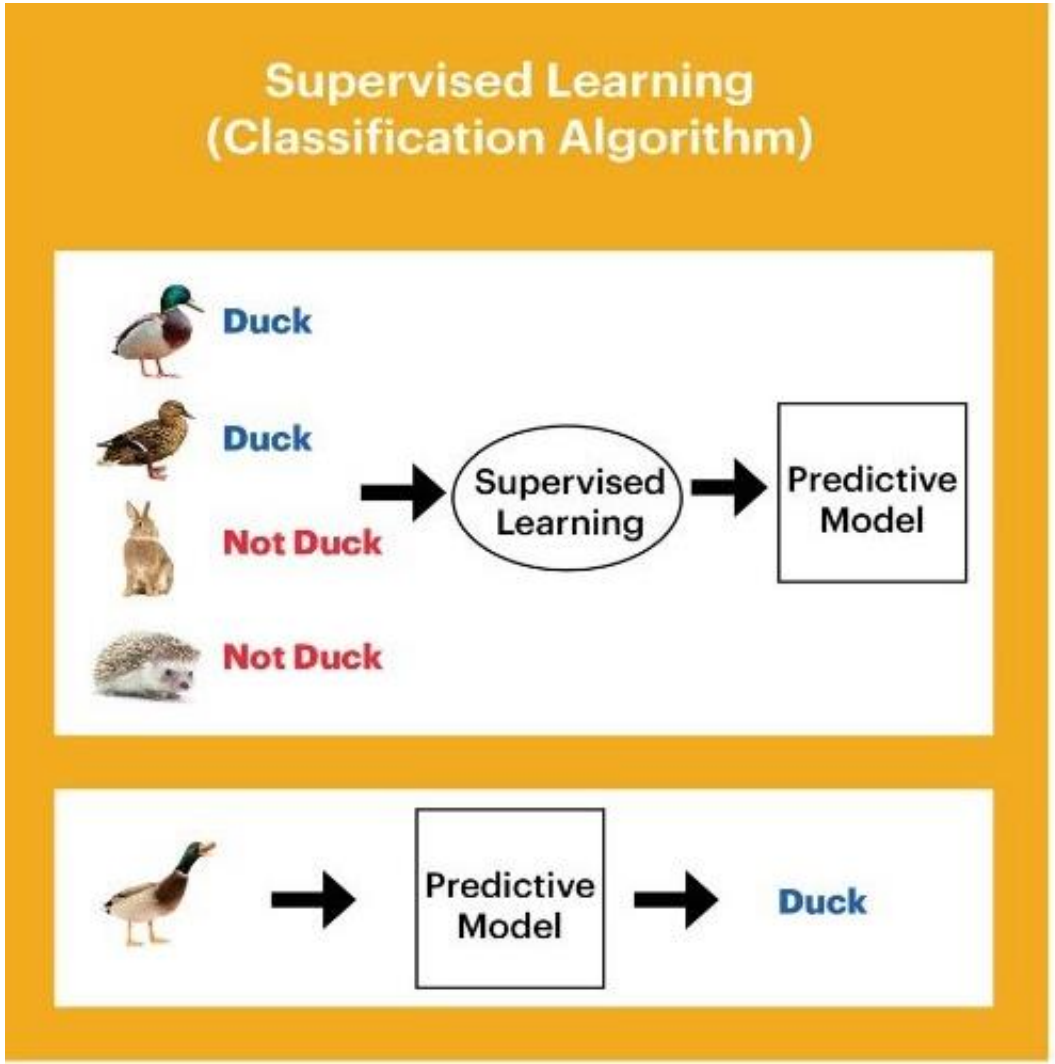


Regression

What is the temperature going to be tomorrow?

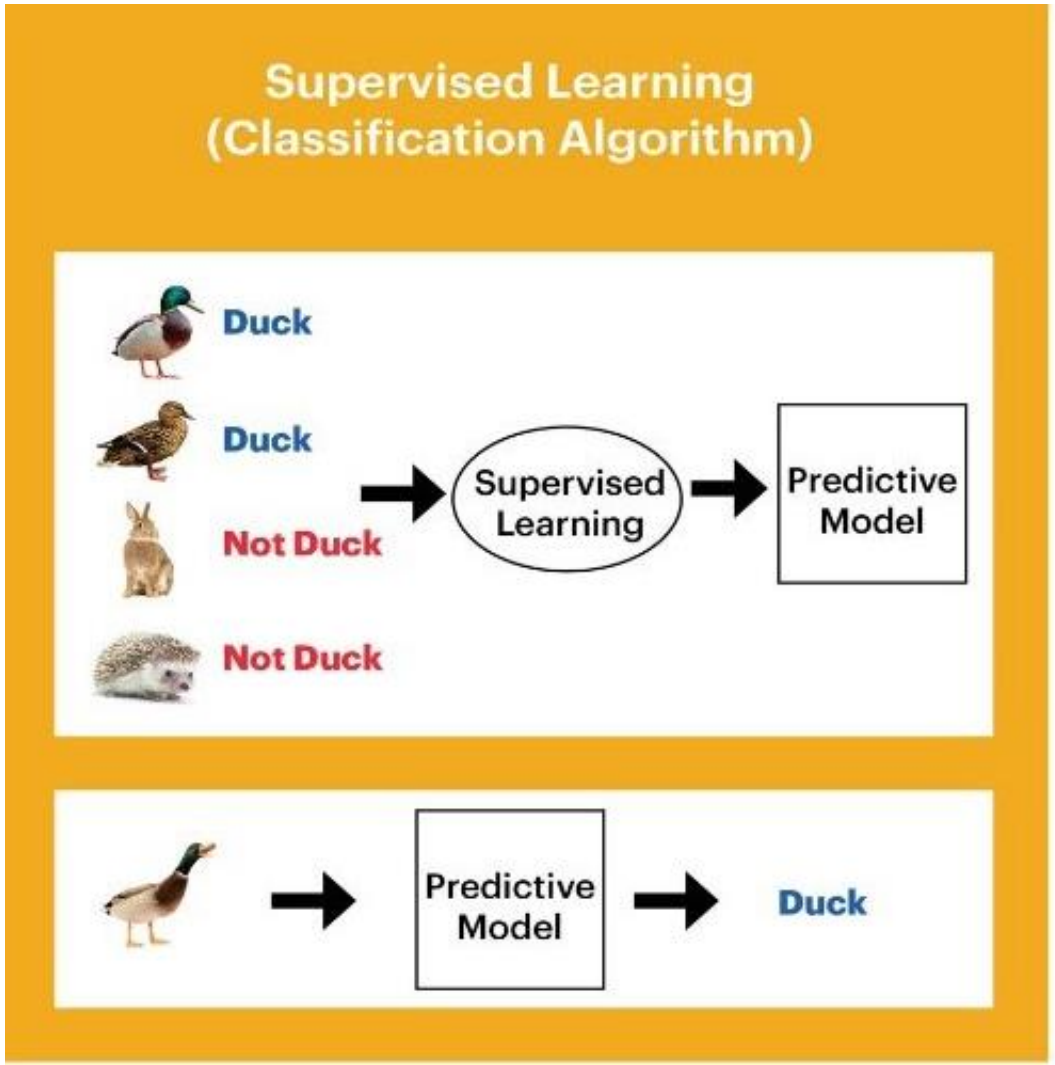
www.pinterest.com

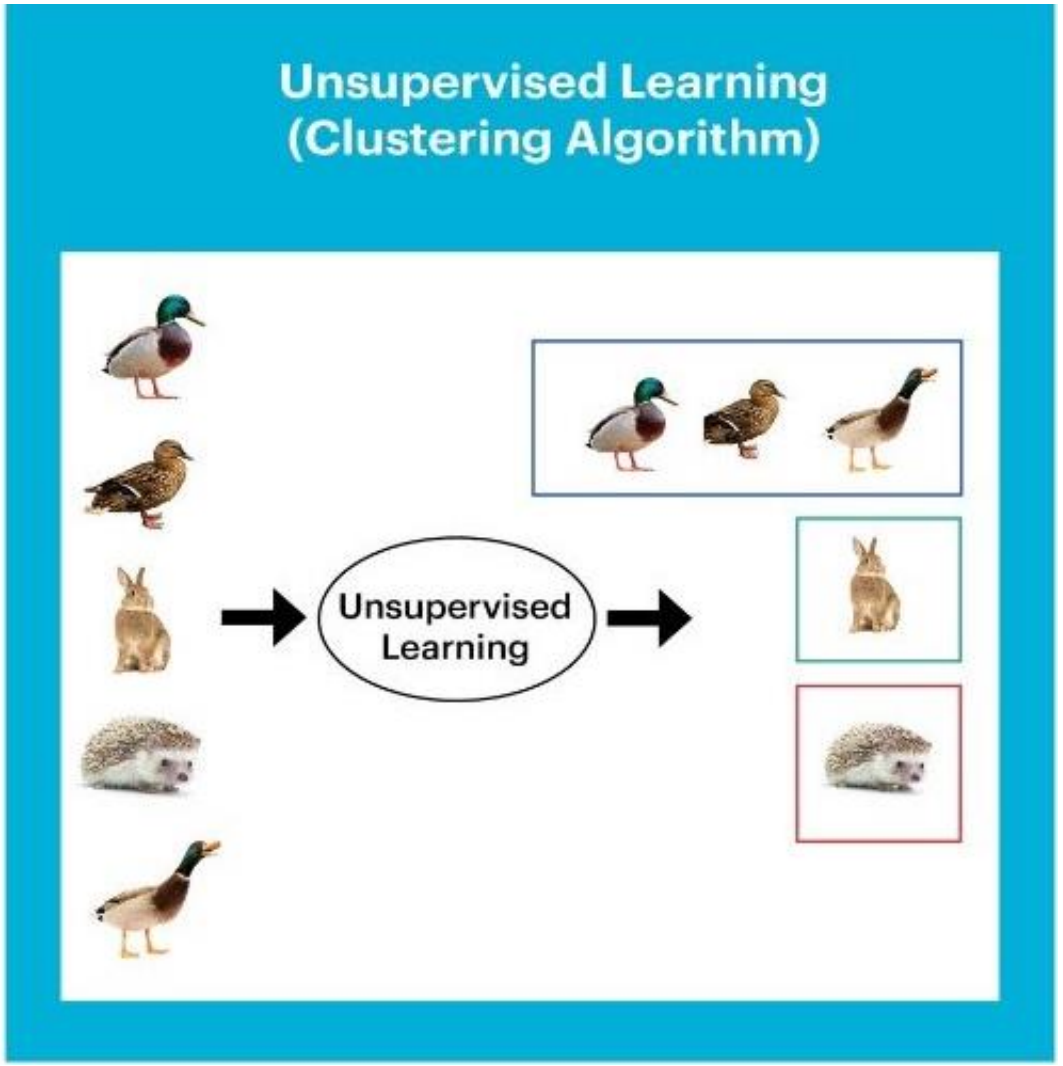
Introduction (Classification vs Clustering)



blog.westerndigital.com

Introduction (Classification vs Clustering)





blog.westerndigital.com

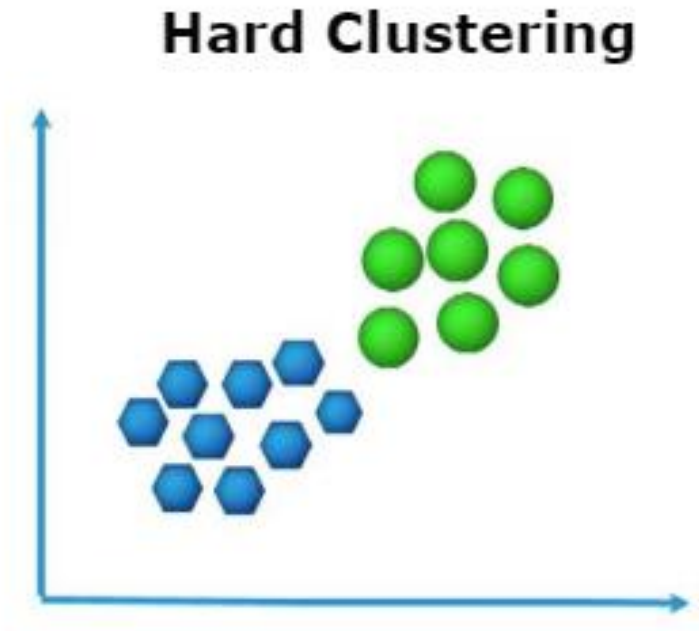
Introduction

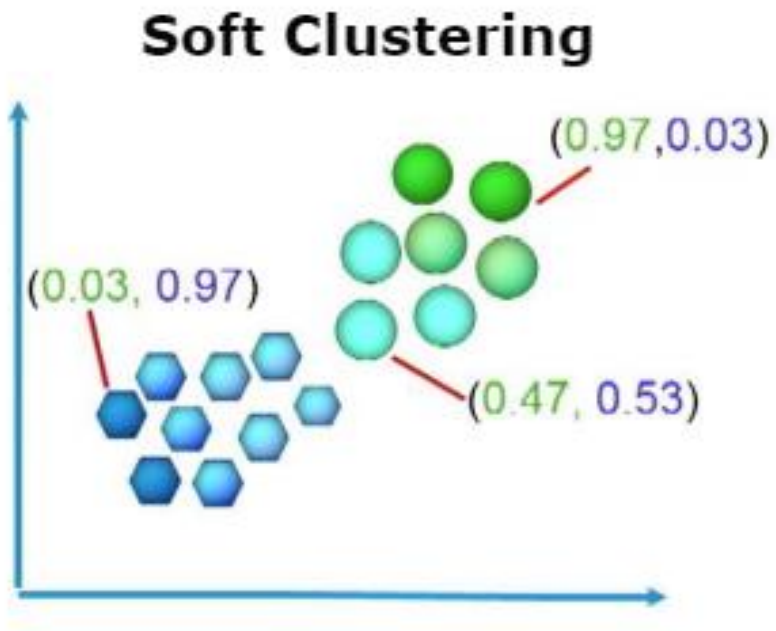
Clustering

Clustering Algorithms

- Introduction
- K-means Clustering
- DBSCAN Clustering
- Agglomerative Clustering

Clustering Algorithms: Introduction

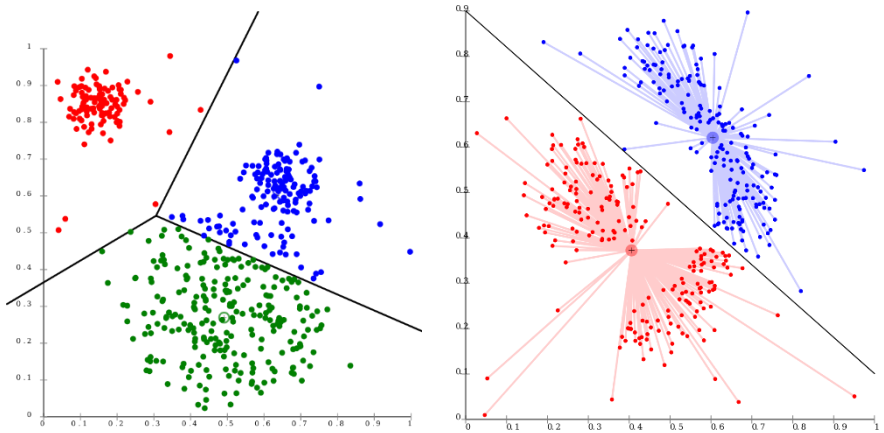




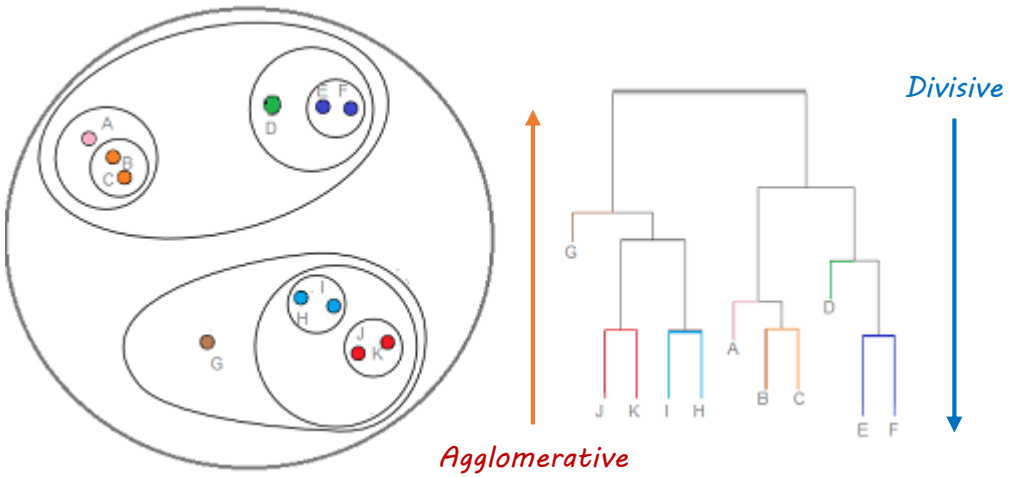
S. Kumar, towardsdatascience.com

Clustering Algorithms: Introduction

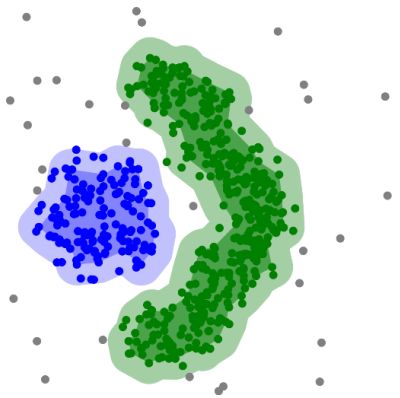
Centroid-based (Partitional) clustering



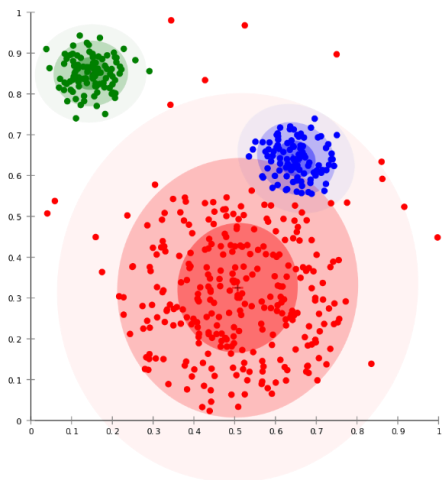
Connectivity-based (hierarchical) clustering



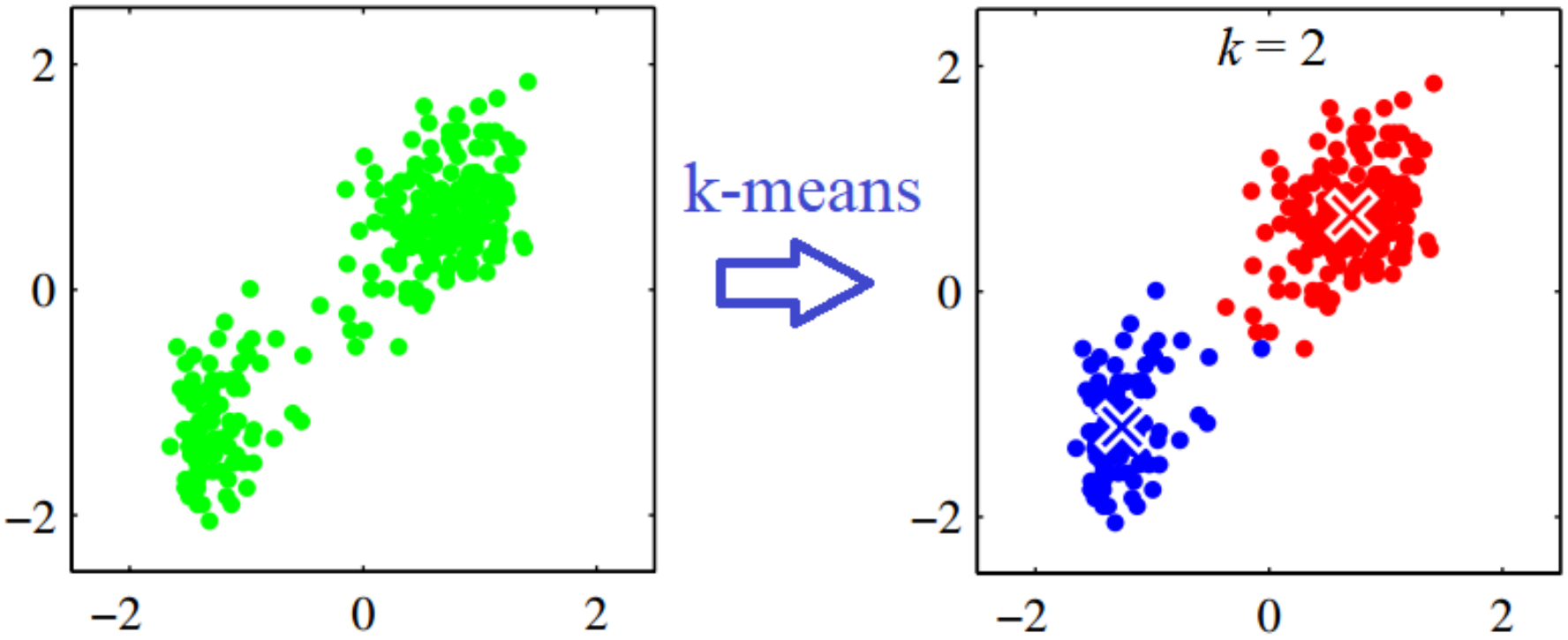
Density-based clustering



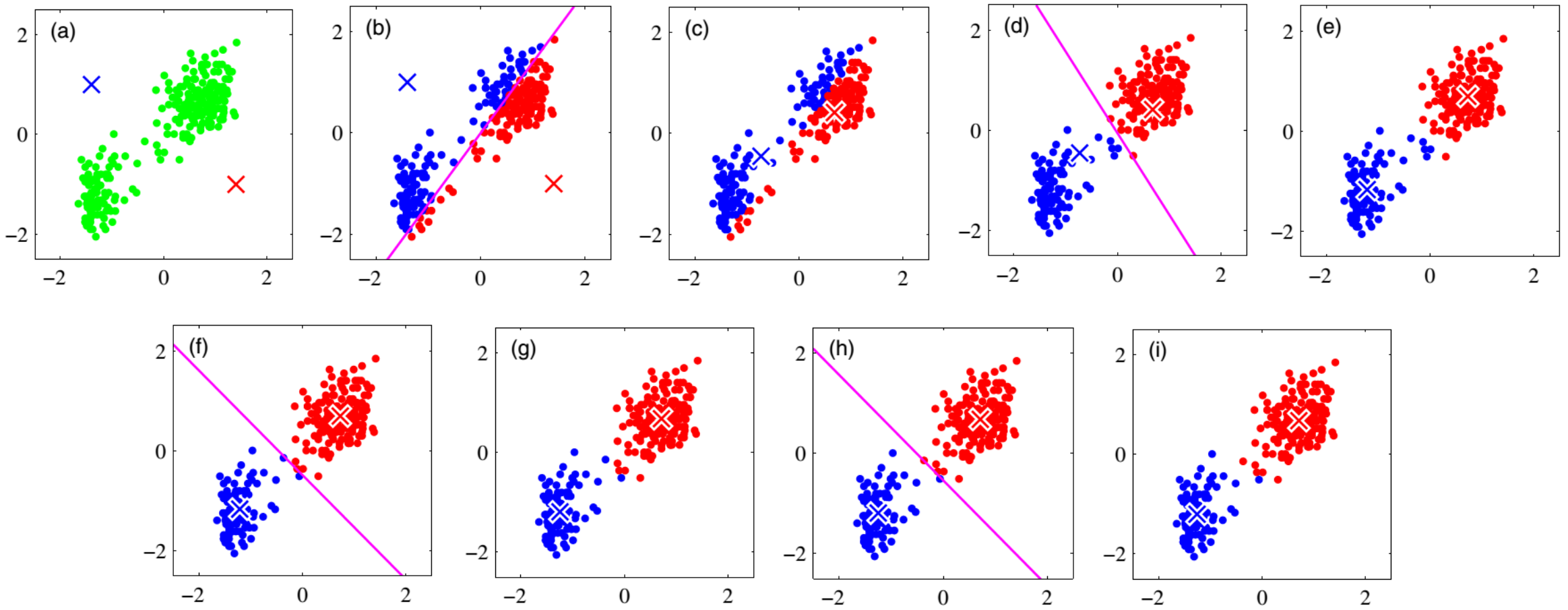
Distribution-based clustering



Clustering Algorithms: K-means

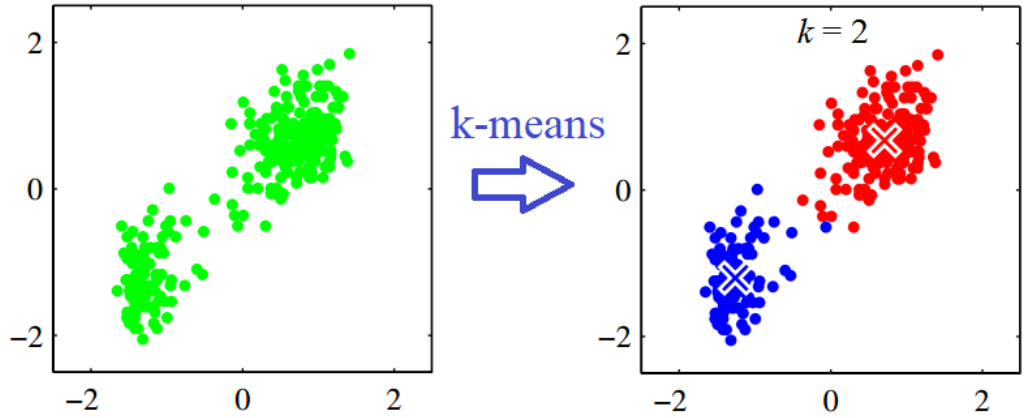


Clustering Algorithms: K-means



C. M. Bishop, Pattern Recognition and Machine Learning, Ch. 9: Mixture Models and EM

Clustering Algorithms: K-means



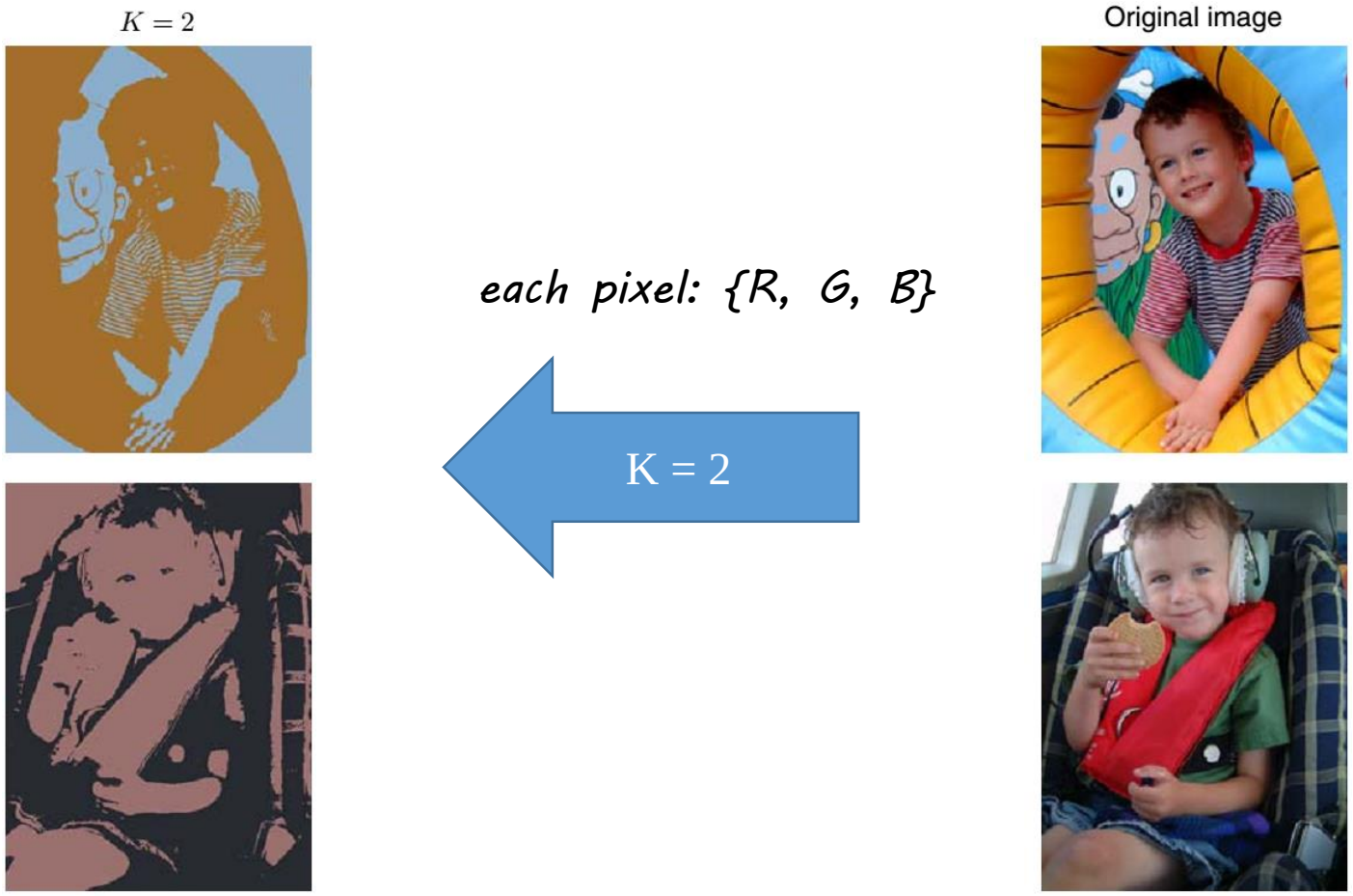
Algorithm 1 k-means algorithm

1: Specify the number k of clusters to assign.
2: Randomly initialize k centroids.
3: **repeat**
4: **expectation:** Assign each point to its closest centroid.
5: **maximization:** Compute the new centroid (mean) of each cluster.
6: **until** The centroid positions do not change.

realpython.com/k-means-clustering-python/

Clustering Algorithms: K-means

image segmentation and image compression

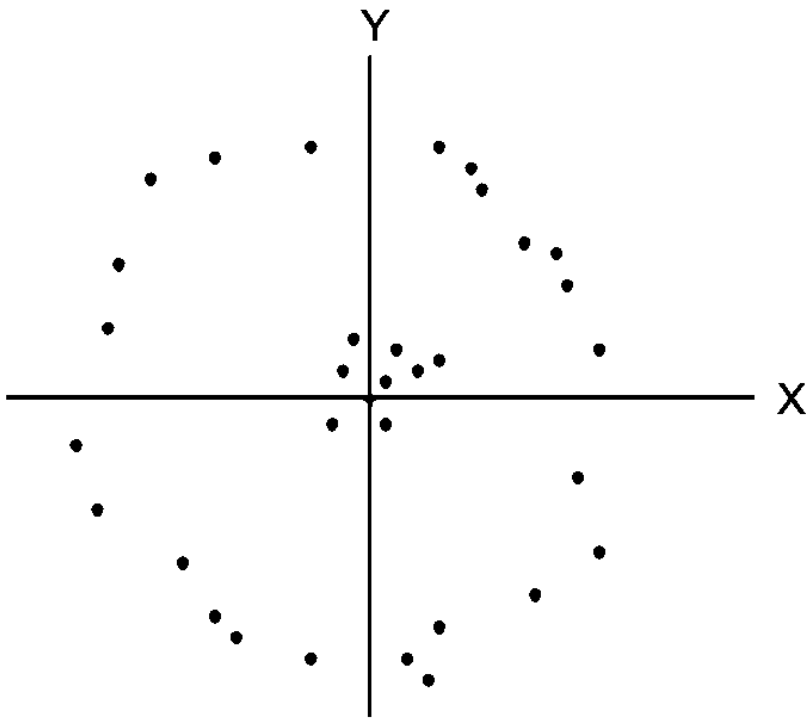


Clustering Algorithms: K-means

image segmentation and image compression

C. M. Bishop, Pattern Recognition and Machine Learning, Ch. 9: Mixture Models and EM

Clustering Algorithms: K-means

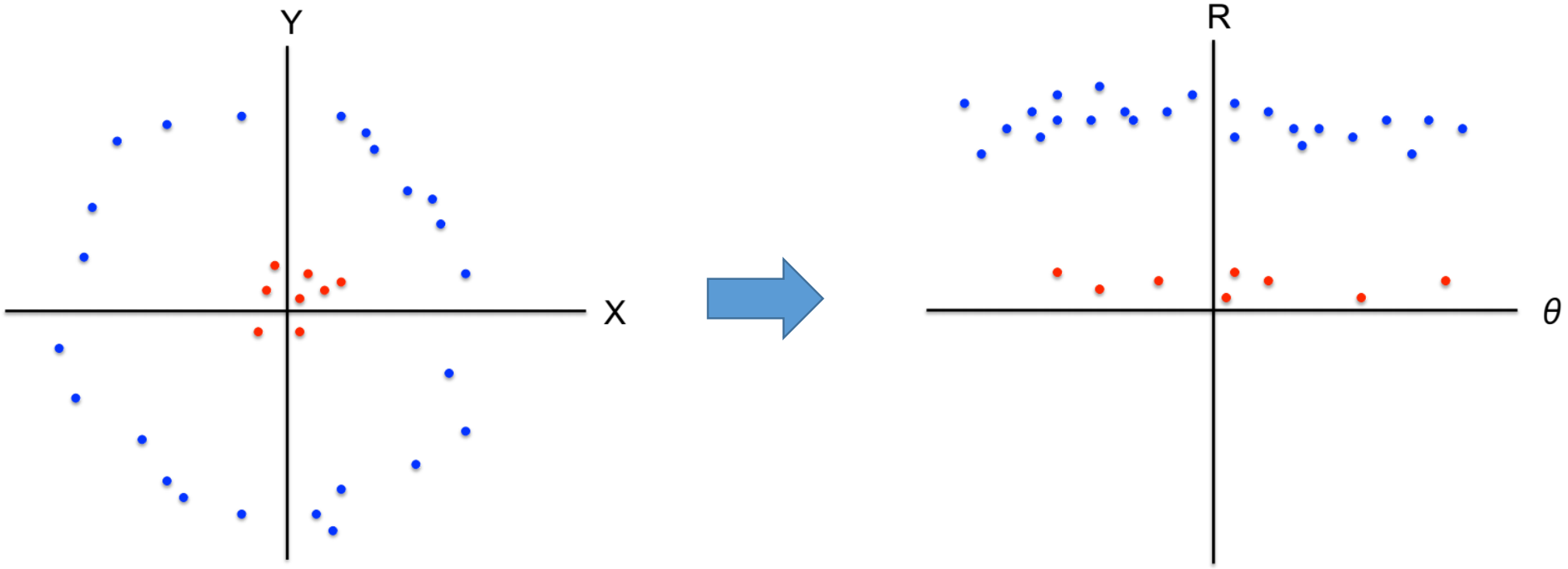


K-means not able to properly cluster

D. Sontag, http://people.csail.mit.edu/dsontag/courses/ml13/

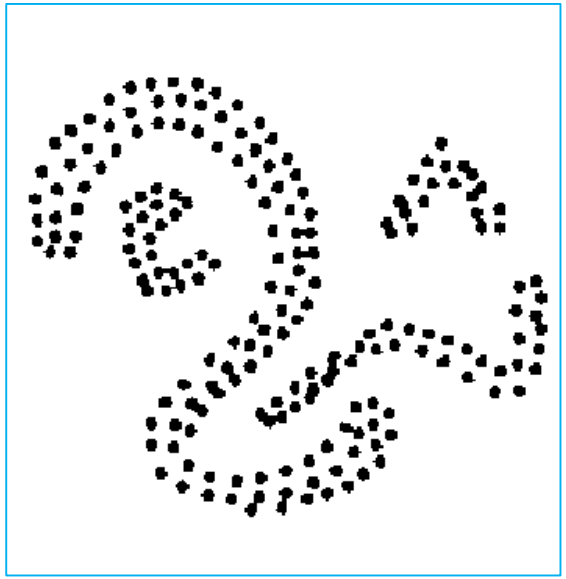
Clustering Algorithms: K-means

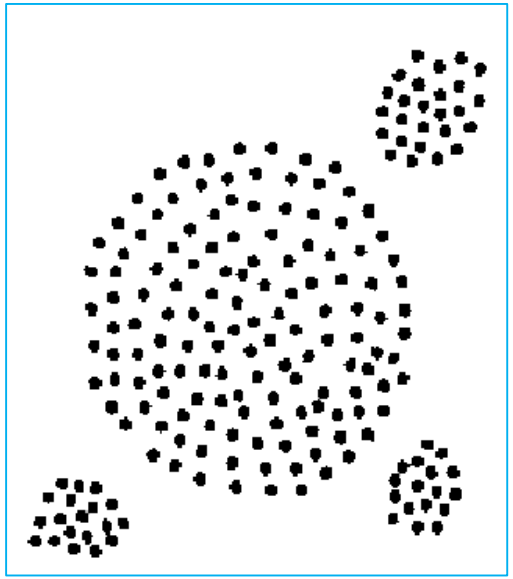
Changing the features (distance function)

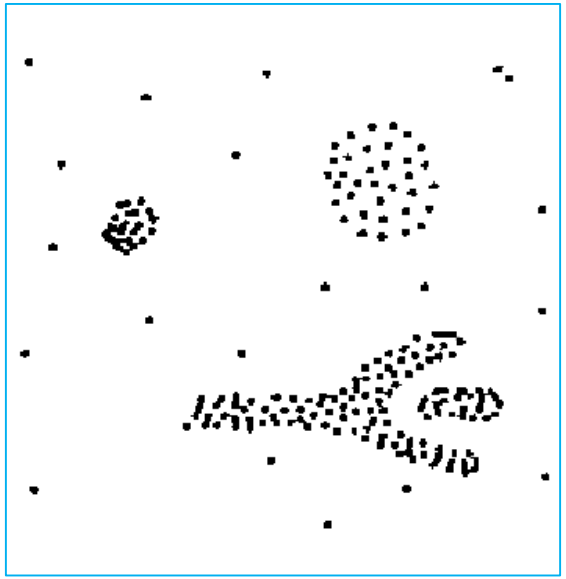


D. Sontag: http://people.csail.mit.edu/dsontag/courses/ml13/

Clustering Algorithms: K-means



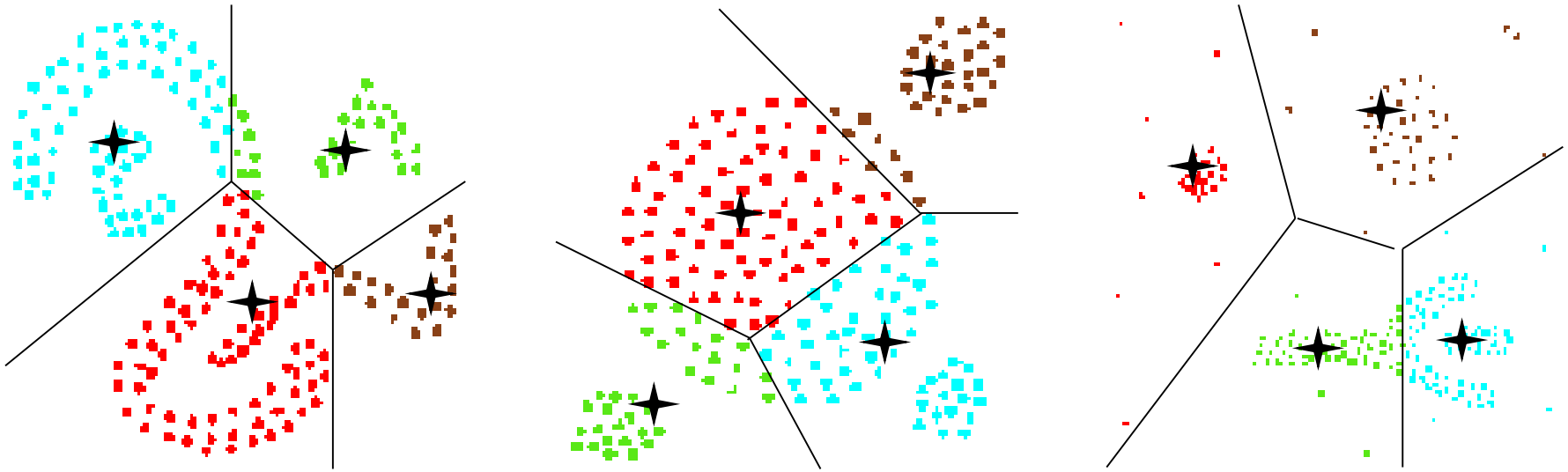




M. Ester, et al., A density-based algorithm for discovering clusters in large spatial databases with noise, 1996.

Clustering Algorithms: K-means

Results of k-medoids algorithm for k=4



(In contrast to the k-means algorithm, k-medoids chooses actual data points as centers·)

www.cse.buffalo.edu/faculty/azhang/cse601/density-based.ppt

Clustering Algorithms: K-means

- Implementations:

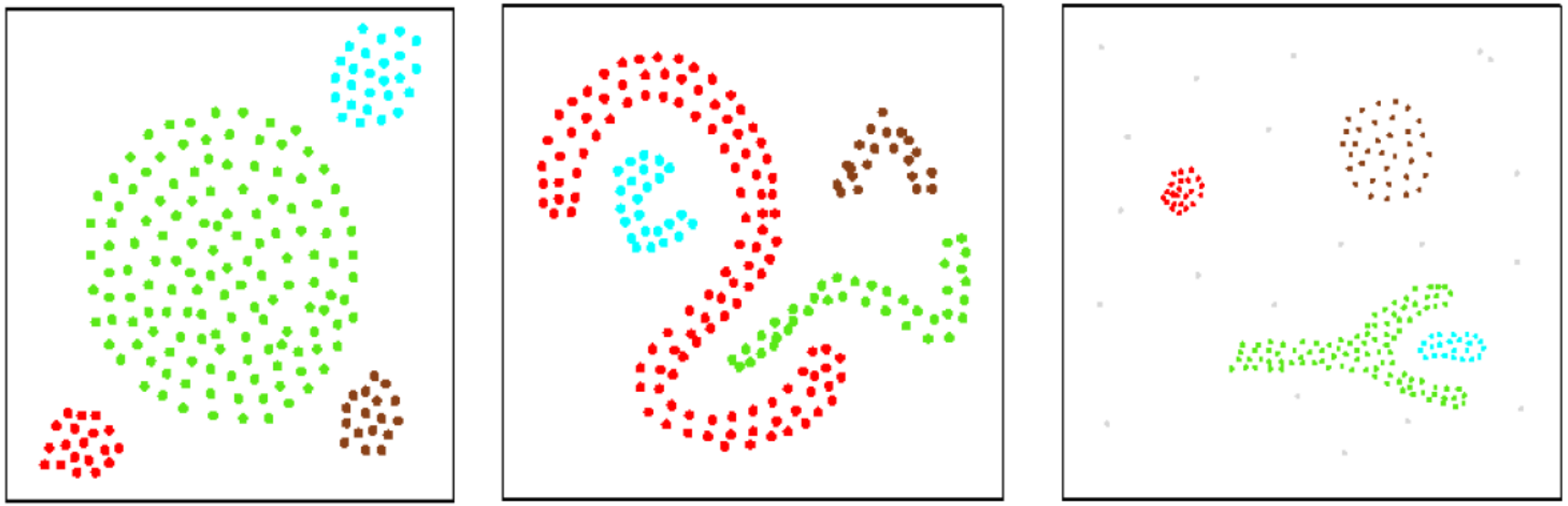
  GitHub https://github.com/hamidsadeghi68/face-clustering

- K-means:

  https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/clustering_kmeans.ipynb

Clustering Algorithms: DBSCAN

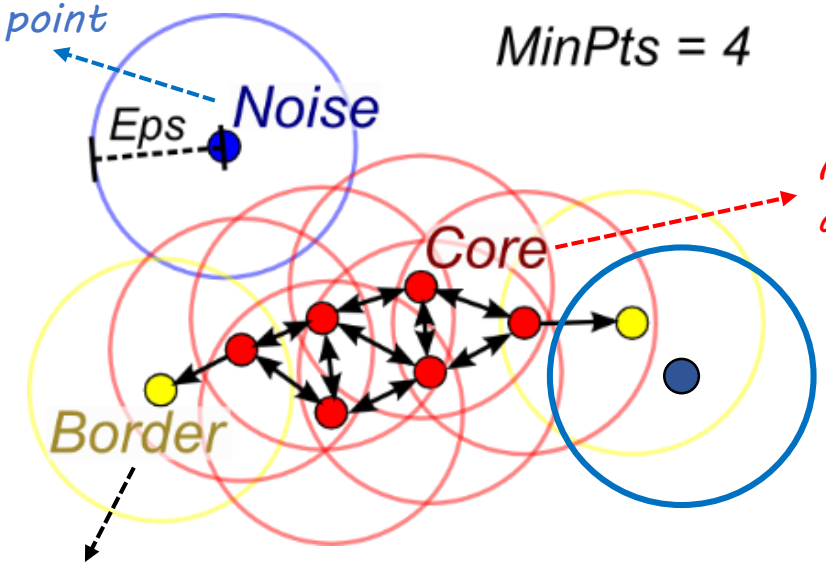
Results of a DBSCAN clustering



www.kdnuggets.com/2021/10/clustering-what-is-how-works.html

Clustering Algorithms: DBSCAN

not reachable from any core point

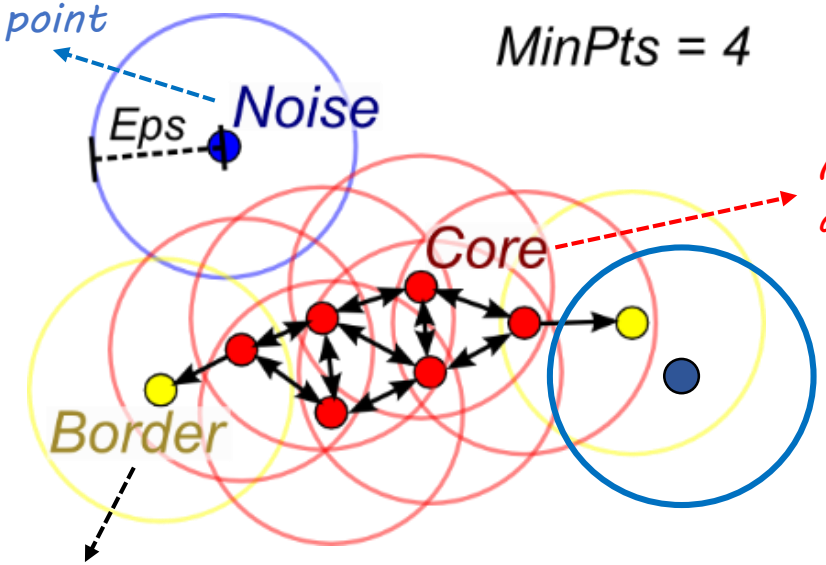


At least minPts points (including p) are within distance ε of it

The point is within distance ε from core point p

Clustering Algorithms: DBSCAN

not reachable from any core point



At least minPts points (including p) are within distance ε of it

The point is within distance ε from core point p

ALGORITHM 2: Abstract DBSCAN Algorithm

```
1 Compute neighbors of each point and identify core points      // Identify core points
2 Join neighboring core points into clusters                    // Assign core points
3 foreach non-core point do
4     Add to a neighboring core point if possible               // Assign border points
5     Otherwise, add to noise                                   // Assign noise points
```

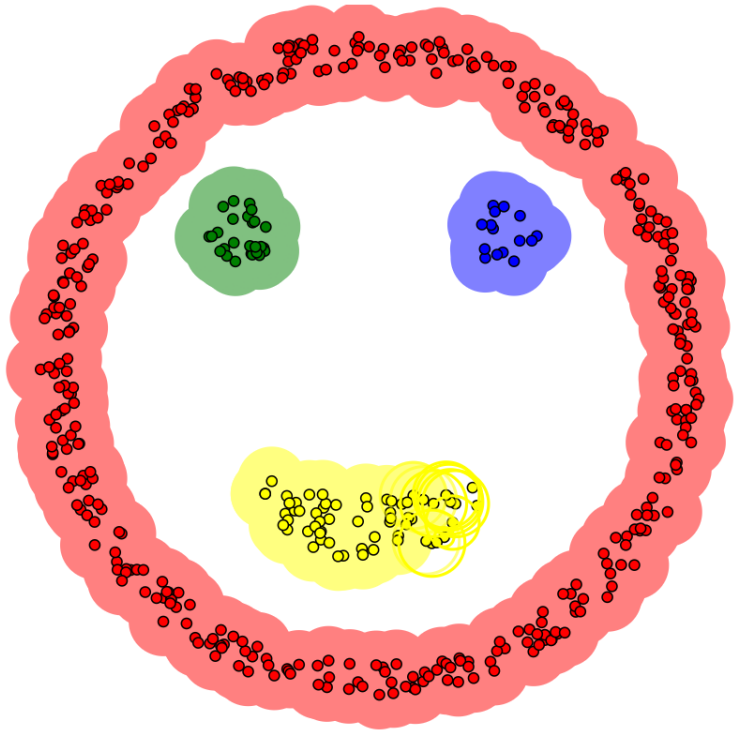
Clustering Algorithms: DBSCAN

```
DBSCAN (database: DB, distFunc, eps, minPts) {
    C ← 0                                                   /* Cluster counter */
    for each point P in DB {
        if label(P) ≠ undefined then continue               /* Previously processed in inner loop */
        Neighbors N ← RangeQuery(DB, distFunc, P, eps)      /* Find neighbors */
        if |N| < minPts then {                              /* Density check */
            label(P) ← Noise                                /* Label as Noise */
            continue
        }
        C ← C + 1                                           /* next cluster label */
        label(P) ← C                                        /* Label initial point */
        SeedSet S ← N \ {P}                                 /* Neighbors to expand */
        for each point Q in S {                             /* Process every seed point Q */
            if label(Q) = Noise then label(Q) ← C           /* Change Noise to border point */
            if label(Q) ≠ undefined then continue           /* Previously processed (e.g., border point) */
            label(Q) ← C                                    /* Label neighbor */
            Neighbors N ← RangeQuery(DB, distFunc, Q, eps)  /* Find neighbors */
            if |N| ≥ minPts then {                          /* Density check (if Q is a core point) */
                S ← S ∪ N                                   /* Add new neighbors to seed set */
            }
        }
    }
}
```

Clustering Algorithms: DBSCAN

- Online visualization:

  https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/



epsilon = 1.00
minPoints = 3

Clustering Algorithms: DBSCAN

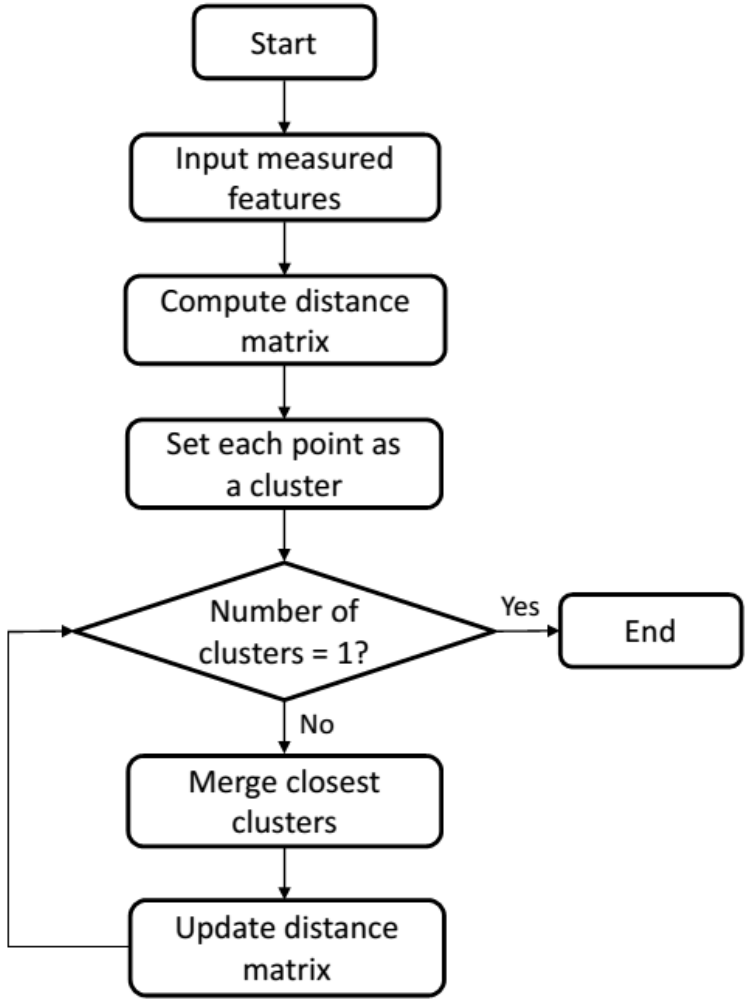
- *implementation:*

https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/clustering_dbscan.ipynb

Clustering Algorithms: Agglomerative

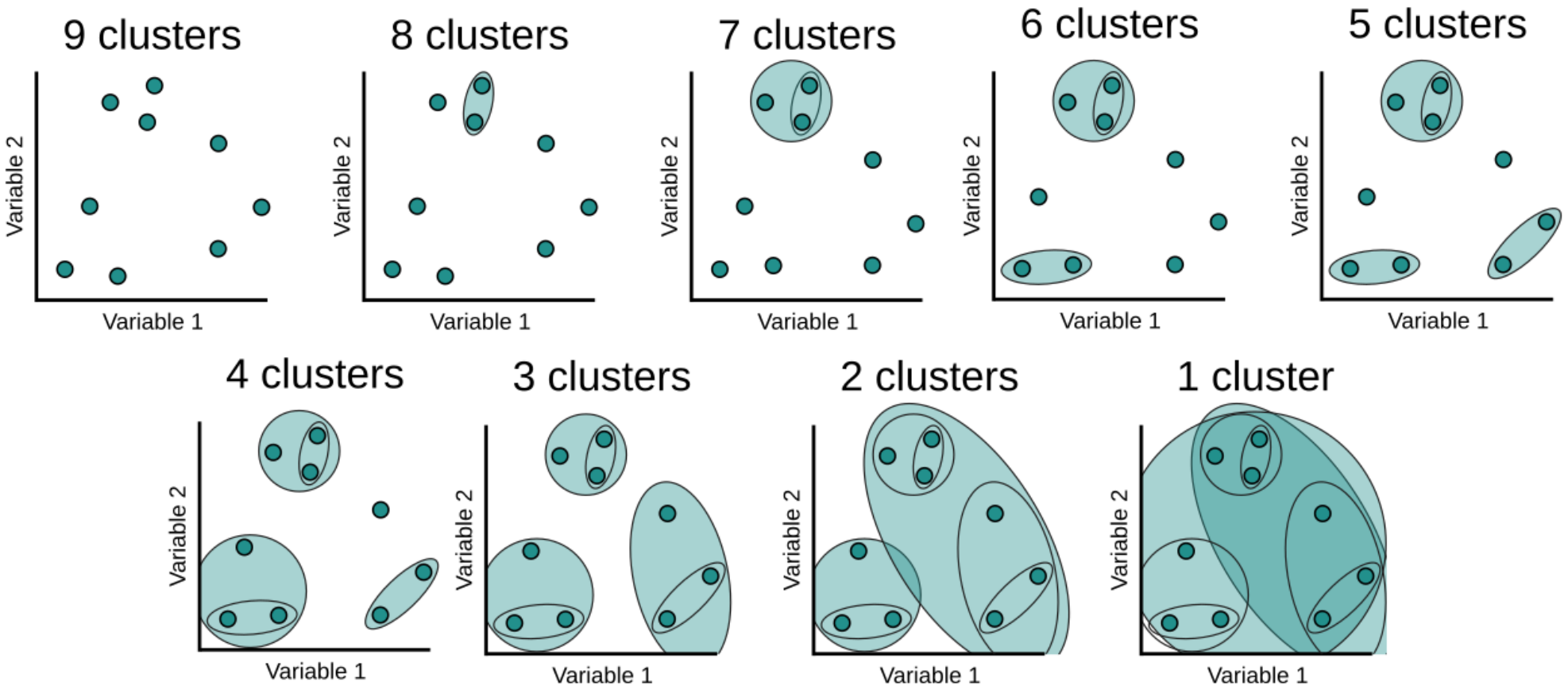
- each observation starts in its own cluster
- pairs of clusters are merged

Clustering Algorithms: Agglomerative



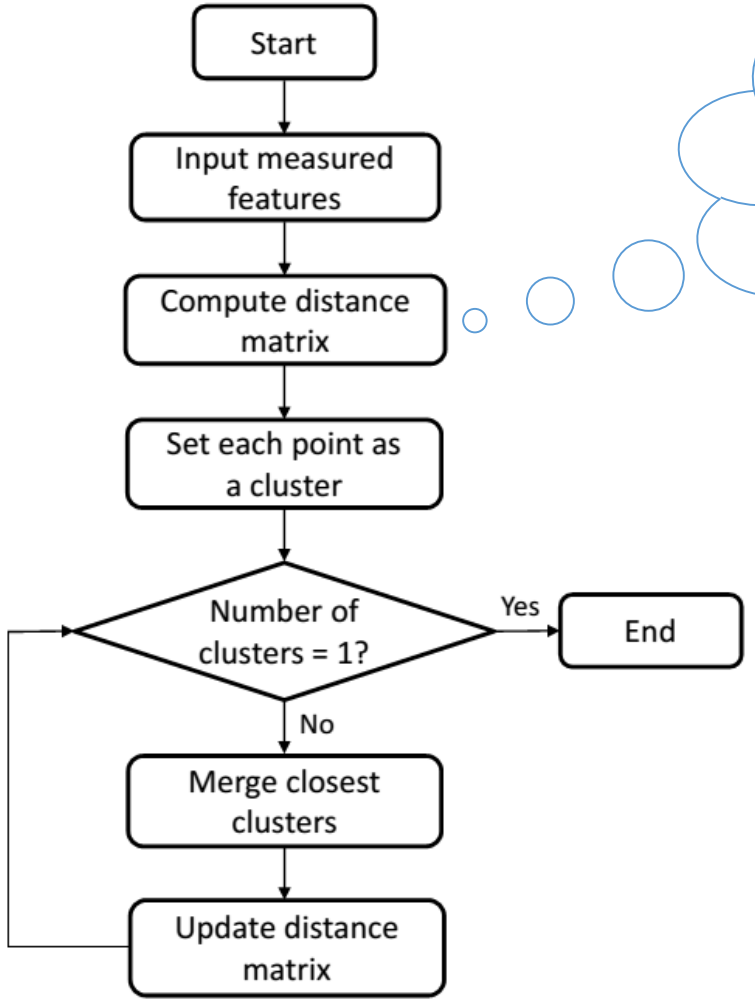
M. Silva, et al. Mechanical Systems and Signal Processing, 2017

Clustering Algorithms: Agglomerative



Hefin I. Rhys, Machine Learning with R, the tidyverse, and mlr, Ch. 17, 2020

Clustering Algorithms: Agglomerative



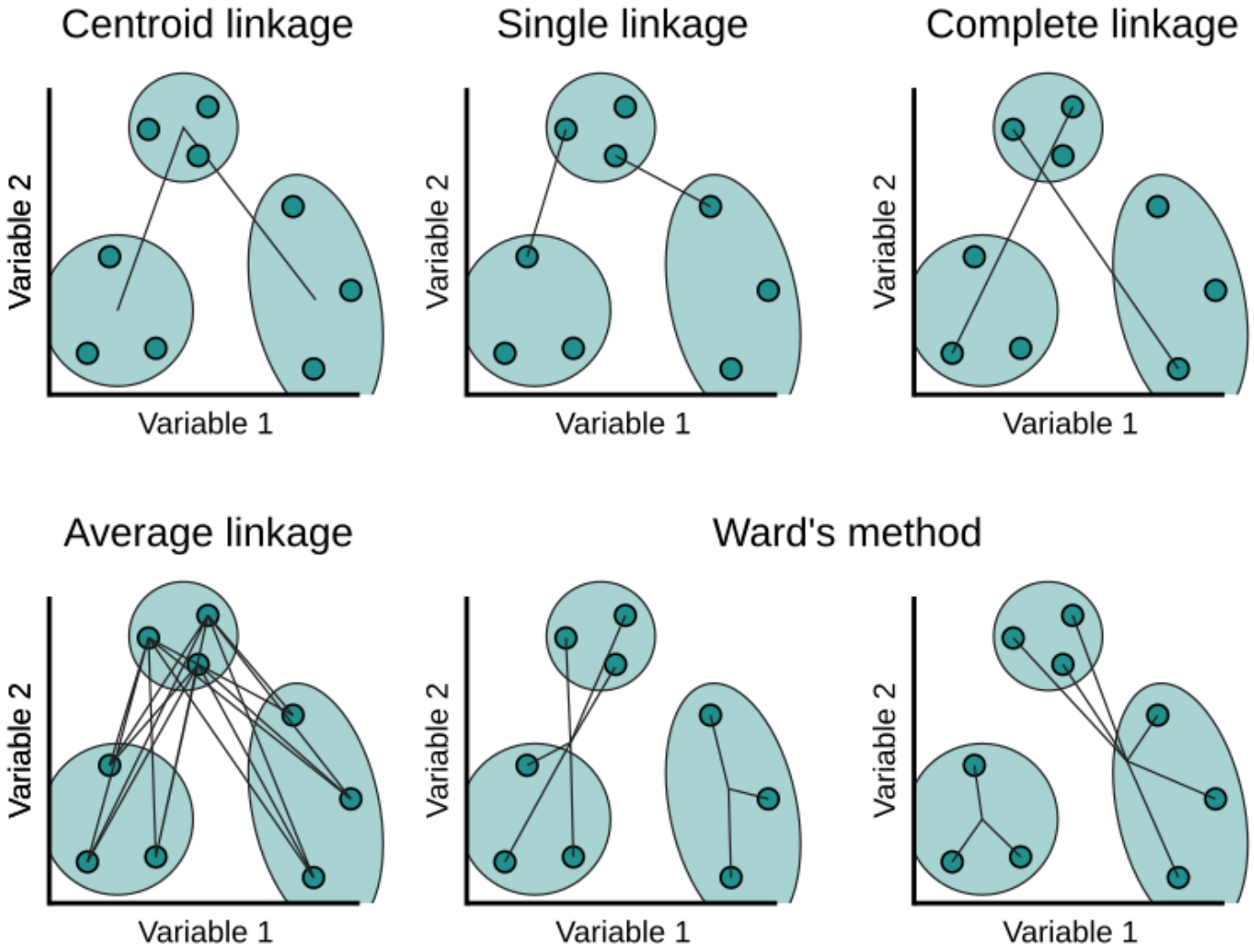
- ward
- average
- complete or maximum
- single or minimum
- centroid

M. Silva, et al. Mechanical Systems and Signal Processing, 2017

Clustering Algorithms: Agglomerative

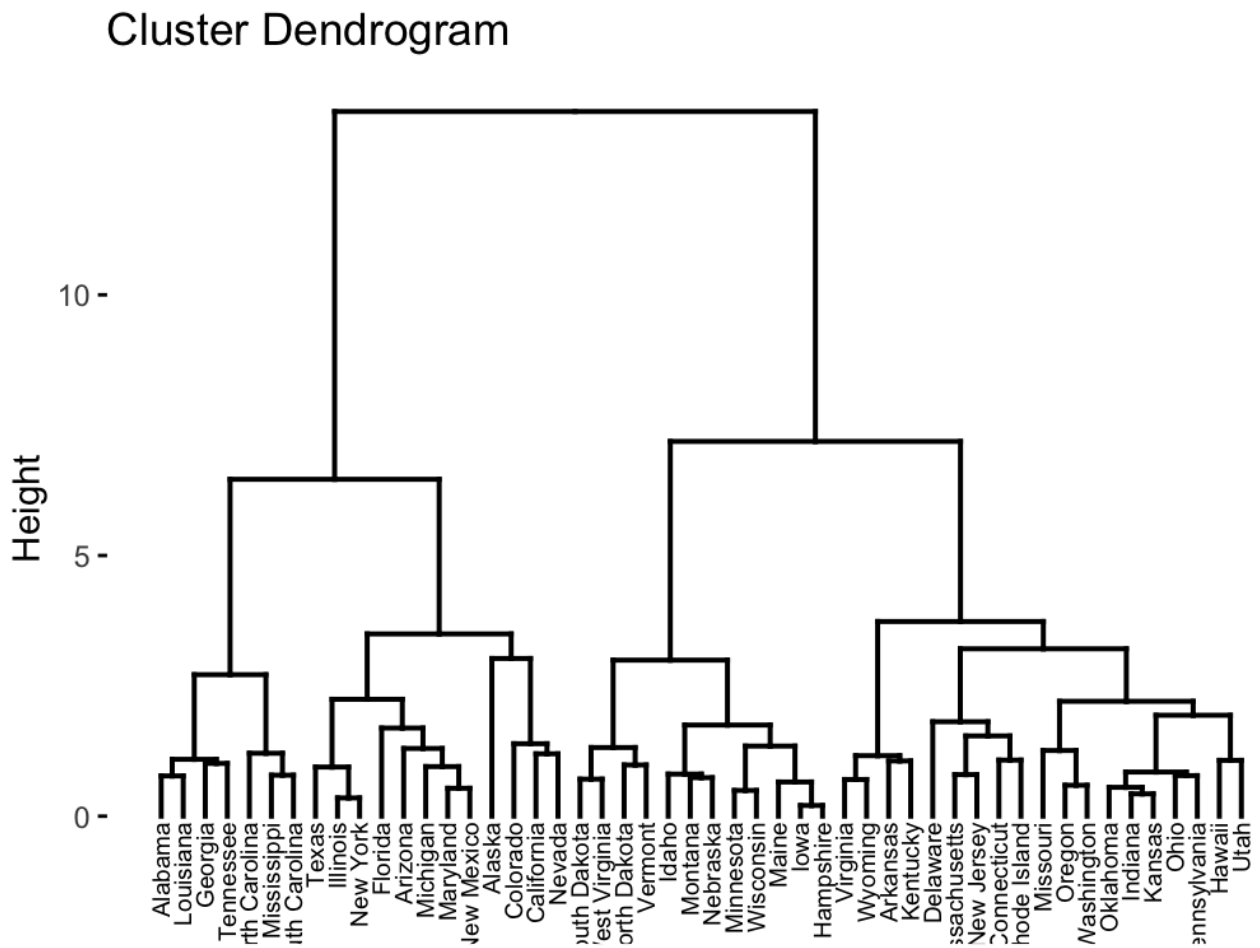
- '**ward**' minimizes the variance of the clusters being merged.
- '**average**' uses the average of the distances of each observation of the two sets.
- '**complete**' or '**maximum**' linkage uses the maximum distances between all observations of the two sets.
- '**single**' or '**minimum**' uses the minimum of the distances between all observations of the two sets.
- '**centroid**': uses the distance between the centroid for cluster 1 (a mean vector of length p variables) and the centroid for cluster 2

Clustering Algorithms: Agglomerative



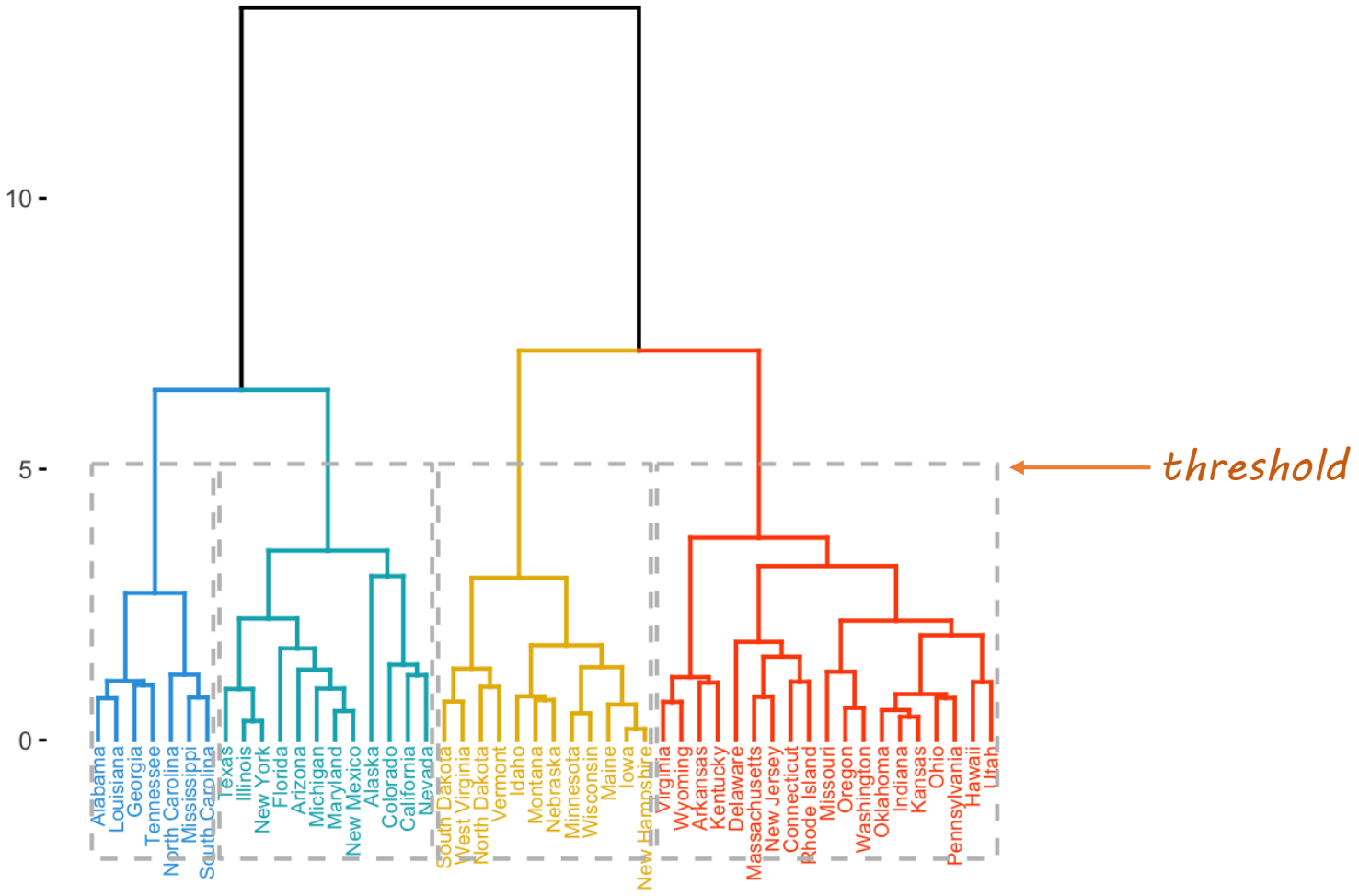
Hefin I. Rhys, Machine Learning with R, the tidyverse, and mlr, Ch. 17, 2020

Clustering Algorithms: Agglomerative



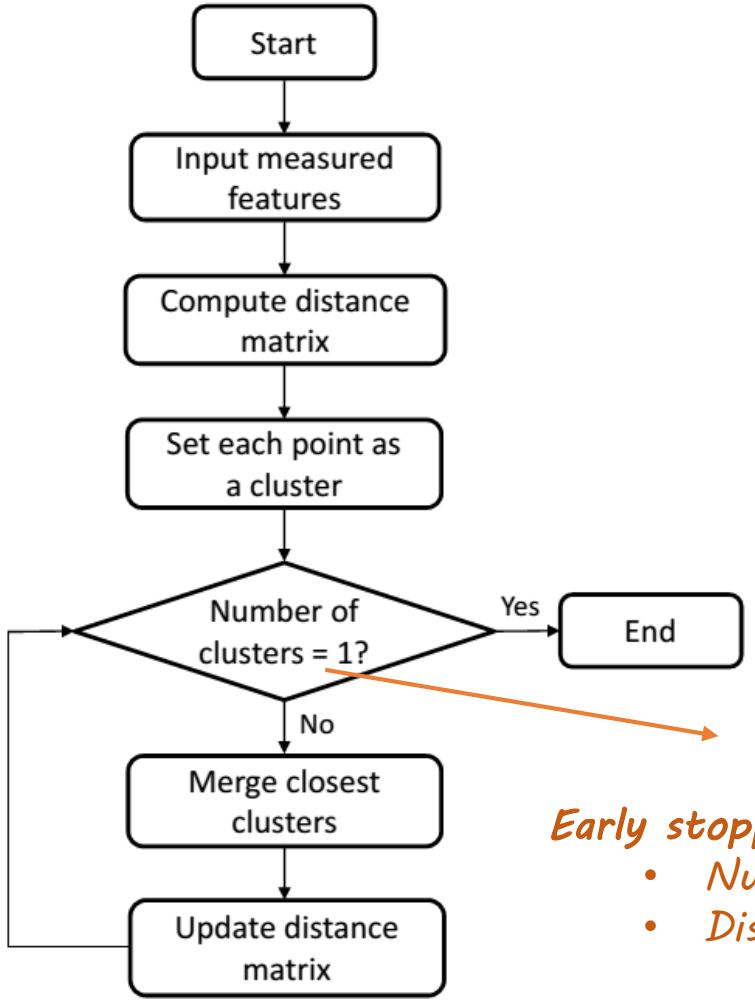
https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/

Clustering Algorithms: Agglomerative



https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/

Clustering Algorithms: Agglomerative



Full tree

Early stopping can be performed based on:

- Num. of clusters
- Distance/similarity criteria

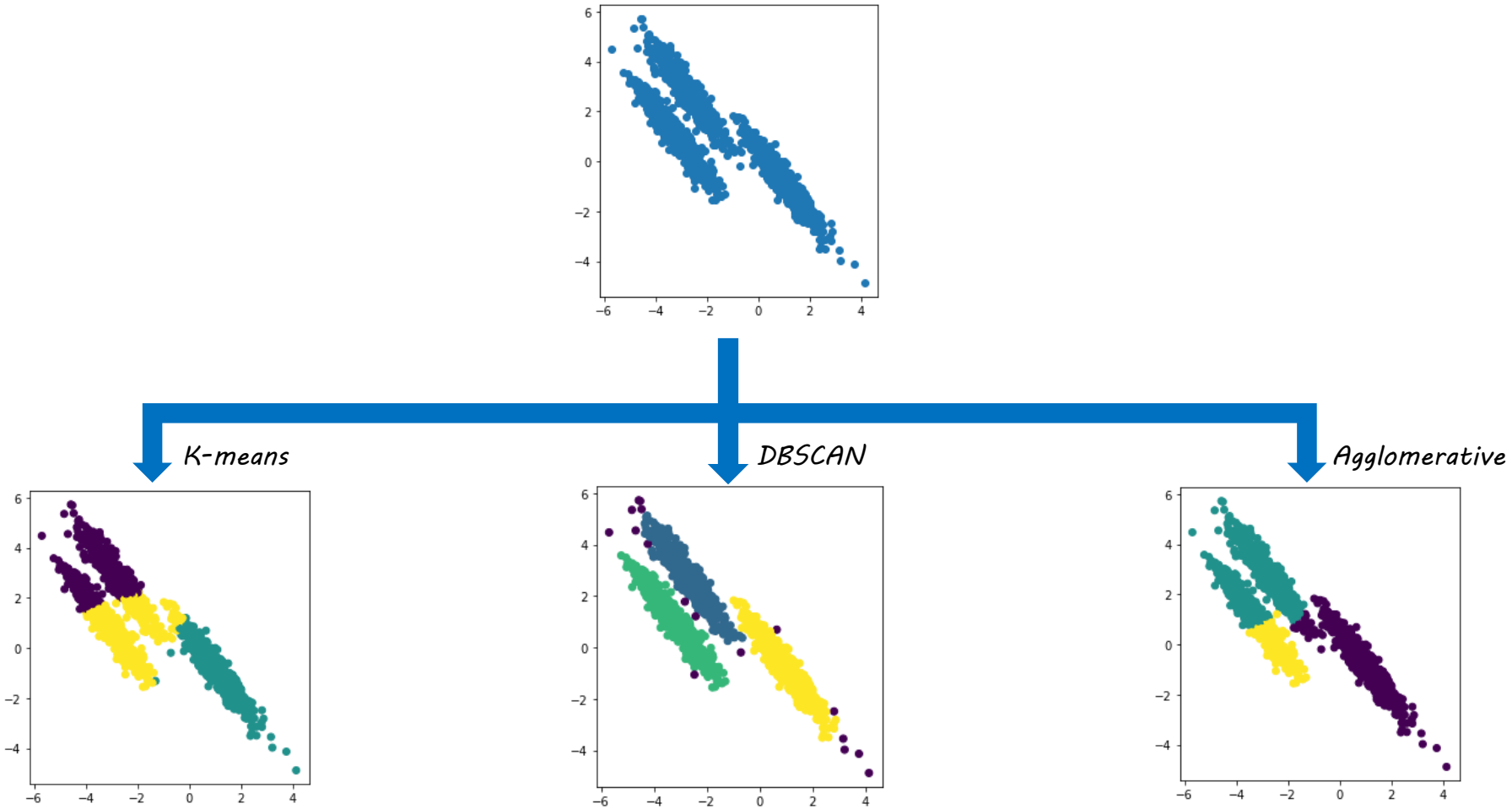
M. Silva, et al. Mechanical Systems and Signal Processing, 2017

Clustering Algorithms: Agglomerative

- *implementation:*

https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/clustering_agglomerative.ipynb

Clustering summary



Evaluation

How to evaluate the quality of clusters?

Internal Evaluation (unsupervised)

Using clustered data itself

- Silhouette coefficient
- Davies-Bouldin index
- Dunn index

External Evaluation (supervised)

Using ground truth or gold standard

- Purity
- Rand Index
- Normalized Mutual Information (NMI)
- F-measure

Evaluation

- Purity

$$purity(\Omega, C) = \frac{1}{N}\sum_{k} \max_{j} |\omega_k \cup c_j|$$

Evaluation

- Purity

$$purity(\Omega, C) = \frac{1}{N}\sum_{k}\max_{j}|\omega_k \cup c_j|$$

clusters

classes

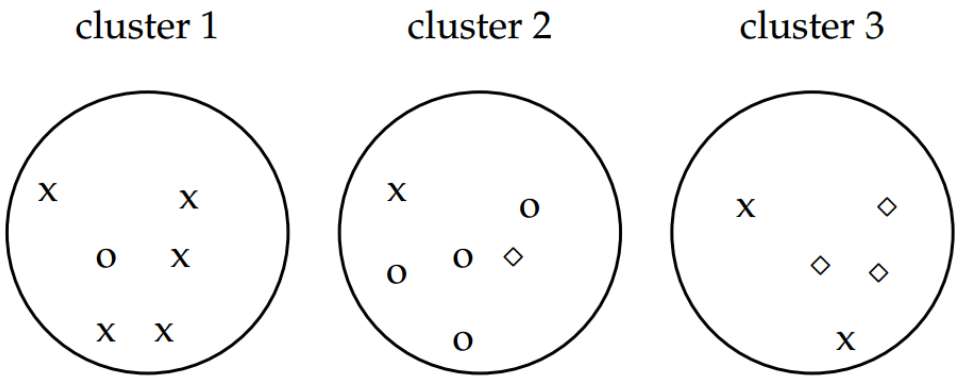
Evaluation

- Purity

$$purity(\Omega, C) = \frac{1}{N}\sum_{k}\max_{j}|\omega_k \cup c_j|$$

clusters

classes



► **Figure 16.4** Purity as an external evaluation criterion for cluster quality. Majority class and number of members of the majority class for the three clusters are: x, 5 (cluster 1); o, 4 (cluster 2); and ◇, 3 (cluster 3). Purity is $(1/17) \times (5+4+3) \approx 0.71$.

Ch. D. Manning , et al. An Introduction to Information Retrieval, Ch. 16: Flat clustering, 2009

Evaluation

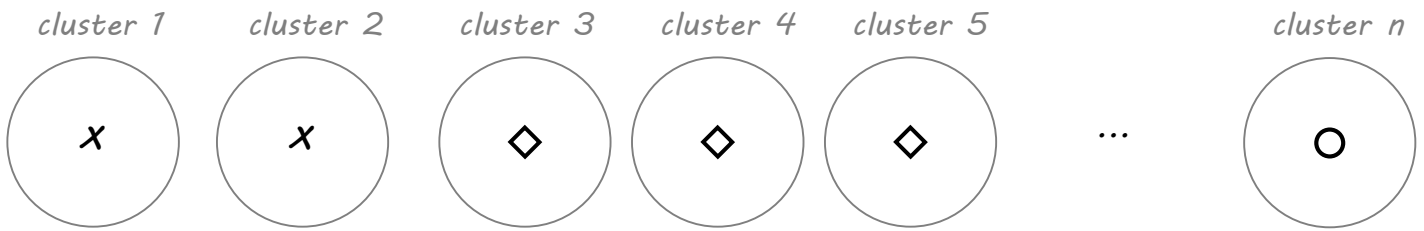
- Purity

$$purity(\Omega, C) = \frac{1}{N}\sum_{k}\max_{j}|\omega_k \cup c_j|$$

clusters

classes

If each data gets 1 cluster ➡ purity = 1 (!)



Ch. D. Manning , et al. An Introduction to Information Retrieval, Ch. 16: Flat clustering, 2009

Evaluation

- Rand Index

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

Evaluation

- Rand Index

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

TP: True Positive pairs

TN: True Negative pairs

FP: False Positive pairs

FN: False Negative pairs

Evaluation

- Rand Index

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

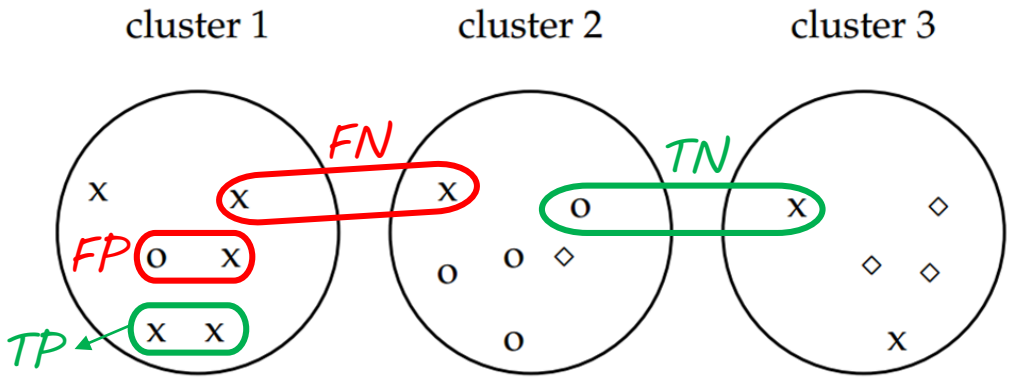
True Positive pairs

True Negative pairs

False Positive pairs

False Negative pairs

Sample pairs:



Ch. D. Manning , et al. An Introduction to Information Retrieval, Ch. 16: Flat clustering, 2009

Evaluation

- Rand Index

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

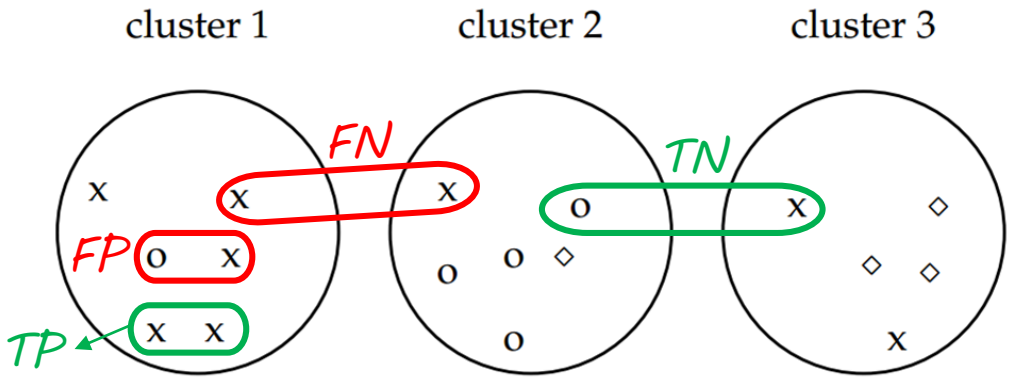
TP: True Positive pairs

TN: True Negative pairs

FP: False Positive pairs

FN: False Negative pairs

Sample pairs:



$$TP + FP = \binom{6}{2} + \binom{6}{2} + \binom{5}{2} = 40 \quad \text{Positive pairs}$$

$$TP = \binom{5}{2} + \binom{4}{2} + \binom{3}{2} + \binom{2}{2} = 20$$

$$FP = 40 - 20 = 20$$

	Same cluster	Different clusters
Same class	TP = 20	FN = 24
Different classes	FP = 20	TN = 72

Ch. D. Manning , et al. An Introduction to Information Retrieval, Ch. 16: Flat clustering, 2009

Evaluation

- Rand Index

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

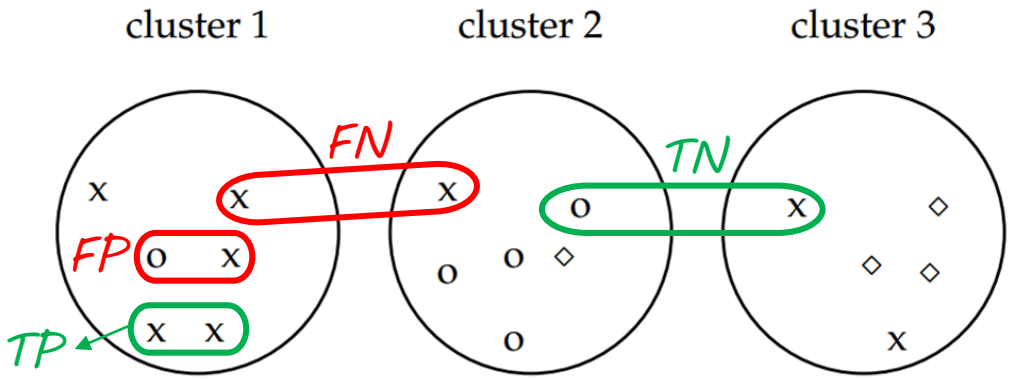
True Positive pairs

True Negative pairs

False Positive pairs

False Negative pairs

Sample pairs:



	Same cluster	Different clusters
Same class	TP = 20	FN = 24
Different classes	FP = 20	TN = 72

$$RI = (20 + 72)/(20 + 20 + 24 + 72) \approx 0.68$$

Evaluation

- Normalized Mutual Information (NMI)

$$NMI(\Omega, \mathrm{C}) = \frac{I(\Omega, \mathrm{C})}{[H(\Omega) + H(\mathrm{C})]/2}$$

Evaluation

- Normalized Mutual Information (NMI)

Mutual Information

clusters

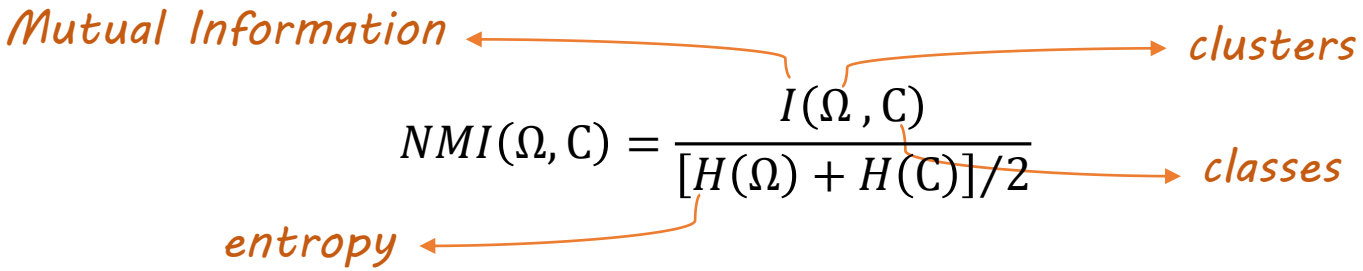
$$NMI(\Omega, C) = \frac{I(\Omega, C)}{[H(\Omega) + H(C)]/2}$$

classes

entropy

Evaluation

- Normalized Mutual Information (NMI)



$$NMI(\Omega, \mathrm{C}) = \frac{I(\Omega, \mathrm{C})}{[H(\Omega) + H(\mathrm{C})]/2}$$

$$I(\Omega, C) = \sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P(\omega_k \cap c_j)}{P(\omega_k)\, P(c_j)} = \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N\, |\omega_k \cap c_j|}{|\omega_k|\, |c_j|}$$

Ch. D. Manning , et al. An Introduction to Information Retrieval, Ch. 16: Flat clustering, 2009

Evaluation

- Normalized Mutual Information (NMI)

Mutual Information

$$NMI(\Omega, C) = \frac{I(\Omega, C)}{[H(\Omega) + H(C)]/2}$$

clusters

classes

entropy

probabilities of a data being in the intersection of ω_k and c_j

$$I(\Omega, C) = \sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P(\omega_k \cap c_j)}{P(\omega_k)\, P(c_j)} = \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N\, |\omega_k \cap c_j|}{|\omega_k|\, |c_j|}$$

probabilities of a data being in cluster ω_k

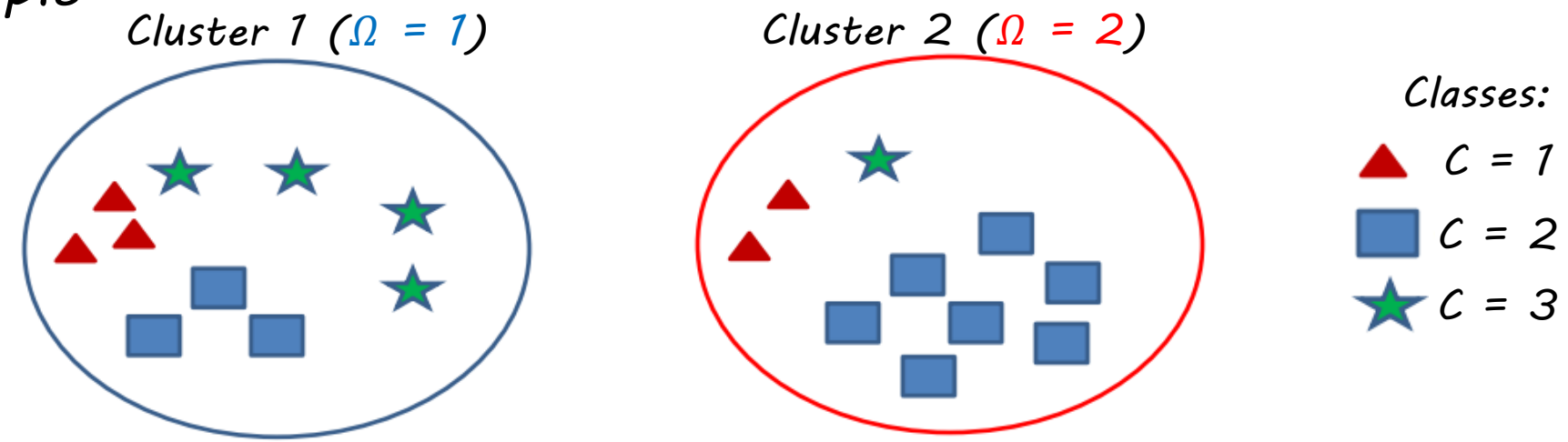
probabilities of a data being in class c_j

$$H(\Omega) = \sum_k P(\omega_k) \log P(\omega_k) = \sum_k \frac{\omega_k}{N} \log \frac{\omega_k}{N}$$

Ch. D. Manning , et al. An Introduction to Information Retrieval, Ch. 16: Flat clustering, 2009

Evaluation

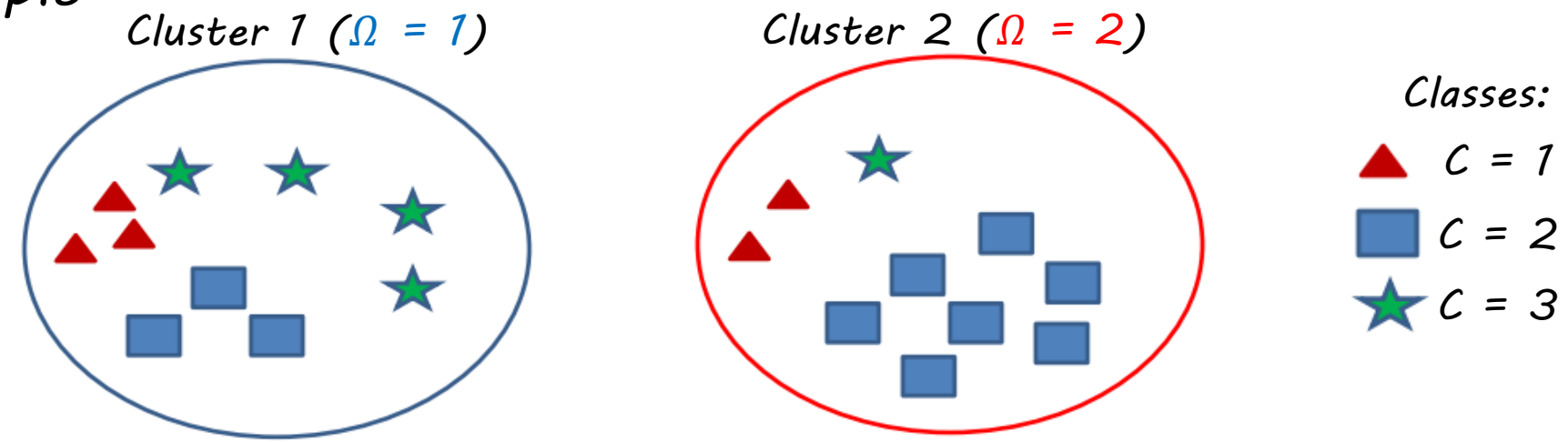
- NMI example



https://course.ccs.neu.edu/cs6140sp15/7_locality_cluster/Assignment-6/NMI.pdf

Evaluation

- NMI example



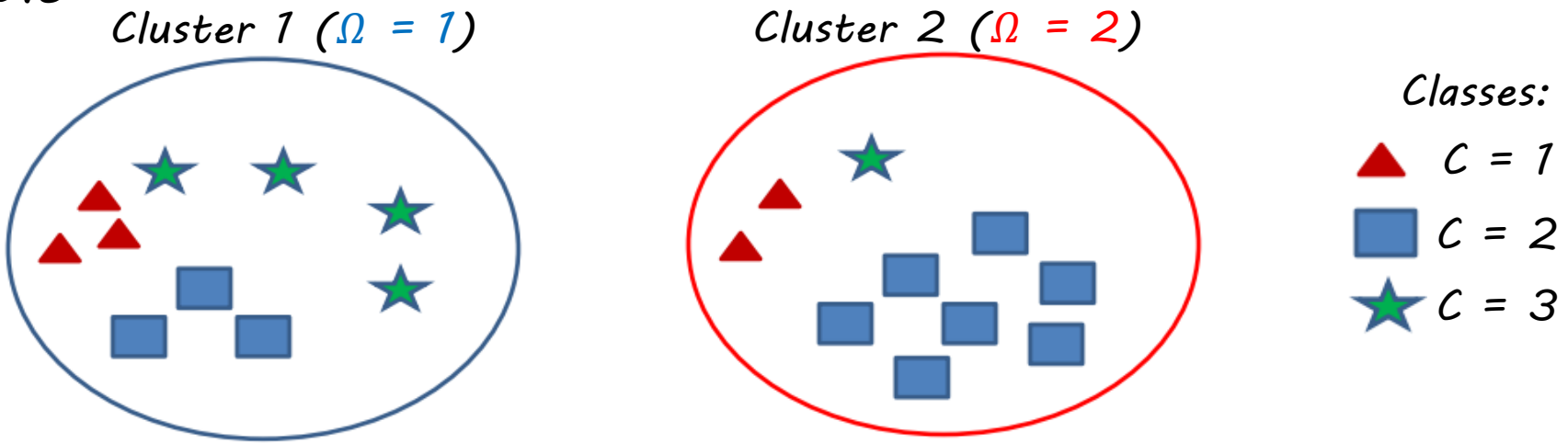
P(▲)=5/20=1/4 P(★)=5/20=1/4 P(■)=10/20=1/2

$$H(C) = -\frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right) = 1.5$$

https://course.ccs.neu.edu/cs6140sp15/7_locality_cluster/Assignment-6/NMI.pdf

Evaluation

- NMI example



$P(\blacktriangle)=5/20=1/4$ $P(\bigstar)=5/20=1/4$ $P(\blacksquare)=10/20=1/2$

$$H(C) = -\frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right) = 1.5$$

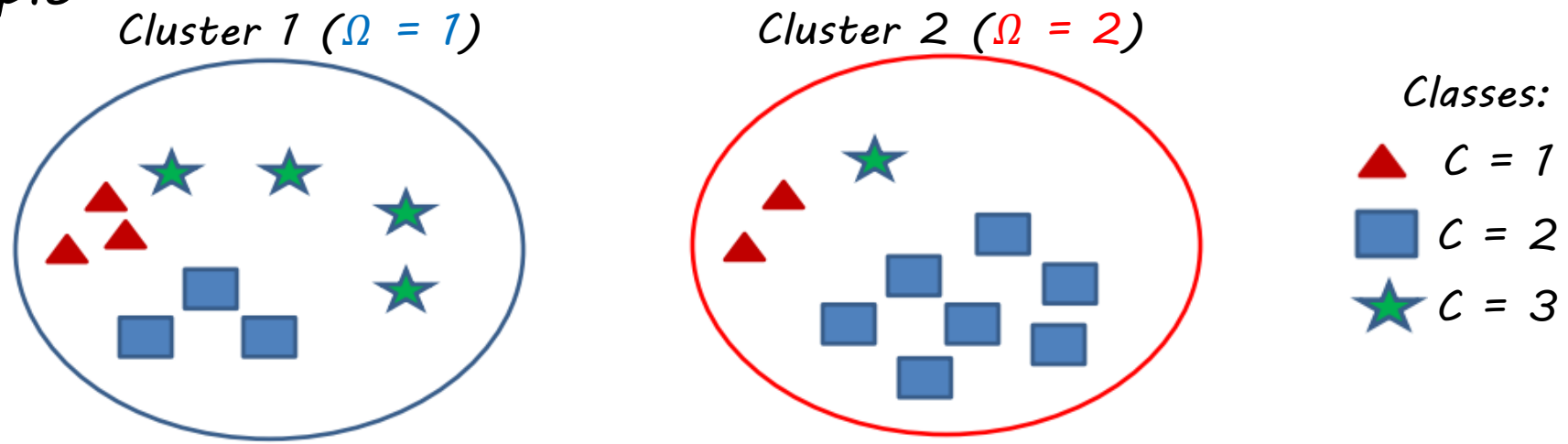
$P(\Omega = 1)=10/20=1/2$ $P(\Omega = 2)= 10/20=1/2$

$$H(\Omega) = -\frac{1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right) = 1$$

https://course.ccs.neu.edu/cs6140sp15/7_locality_cluster/Assignment-6/NMI.pdf

Evaluation

- NMI example



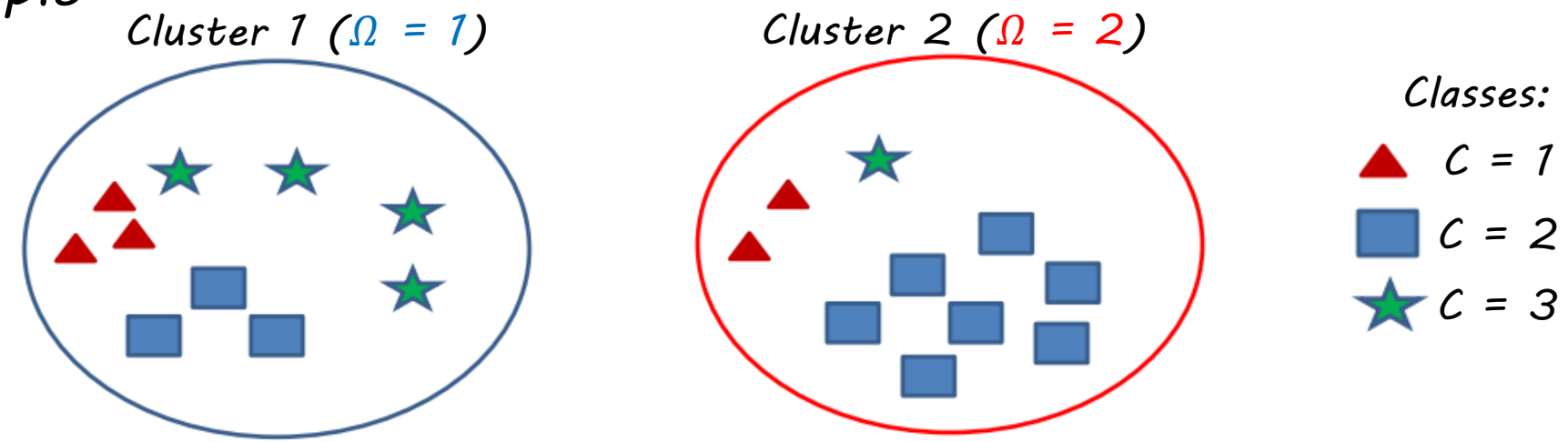
$$I(\Omega, C) = \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} log \frac{N\,|\omega_k \cap c_j|}{|\omega_k|\,|c_j|}$$

$$= \frac{3}{20}\log\left(\frac{20\times 3}{10\times 5}\right) + \frac{3}{20}\log\left(\frac{20\times 3}{10\times 10}\right) + \frac{4}{20}\log\left(\frac{20\times 4}{10\times 5}\right) + \frac{2}{20}\log\left(\frac{20\times 2}{10\times 5}\right) + \frac{7}{20}\log\left(\frac{20\times 7}{10\times 10}\right) + \frac{1}{20}\log\left(\frac{20\times 1}{10\times 5}\right) = 0.1361$$

https://course.ccs.neu.edu/cs6140sp15/7_locality_cluster/Assignment-6/NMI.pdf

Evaluation

- NMI example



$$NMI(\Omega, \mathrm{C}) = \frac{I(\Omega\,, \mathrm{C})}{[H(\Omega) + H(\mathrm{C})]/2} = \frac{0.1361}{[1 + 1.5]/2} = 0.1089$$

https://course.ccs.neu.edu/cs6140sp15/7_locality_cluster/Assignment-6/NMI.pdf

Evaluation

- Precision

(correct rate within clusters) $P = \frac{TP}{TP + FP}$

- Recall

(correct rate within classes/ground truth) $R = \frac{TP}{TP + FN}$

- F-measure

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

$\beta = 0$: $F = P$
$\beta\uparrow$: R

Evaluation

- *implementation:*

https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/clustering_kmeans.ipynb

https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/clustering_dbscan.ipynb

https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/clustering_agglomerative.ipynb

Face Analysis

- Introduction
- Face Detection & Preprocessing
- Face Recognition
- A Complete Face Clustering Algorithm

Face Analysis: Introduction

Face Recognition (It is different from face clustering!)

Identification (1:N)	Verification (1:1)

Face Analysis: Introduction

Face Recognition (It is different from face clustering!)

Identification (1:N)

?

Emilia Clarke
Angelina Jolie
Abbas Bouazar
.
.
.
N

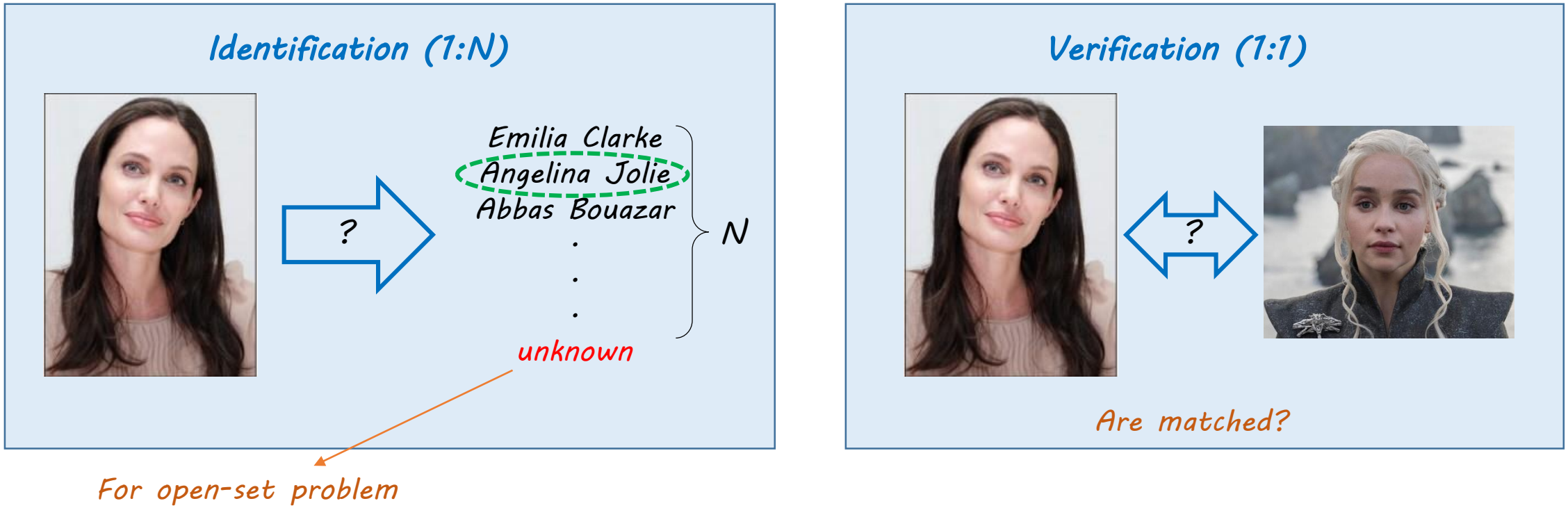
unknown

For open-set problem

Verification (1:1)

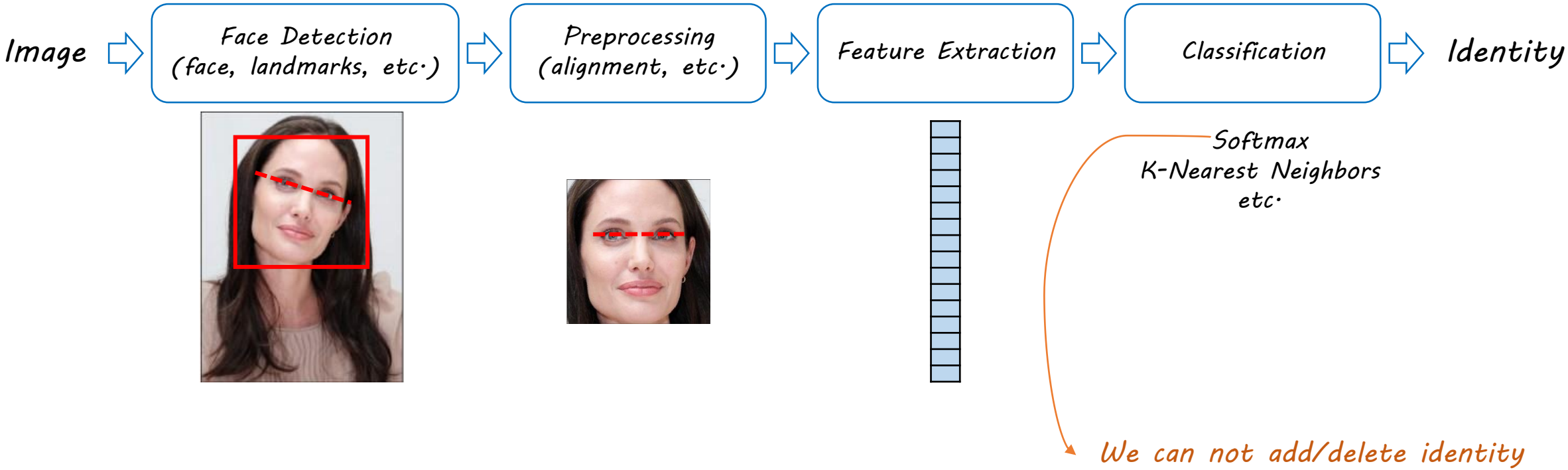
Face Analysis: Introduction

Face Recognition (It is different from face clustering!)

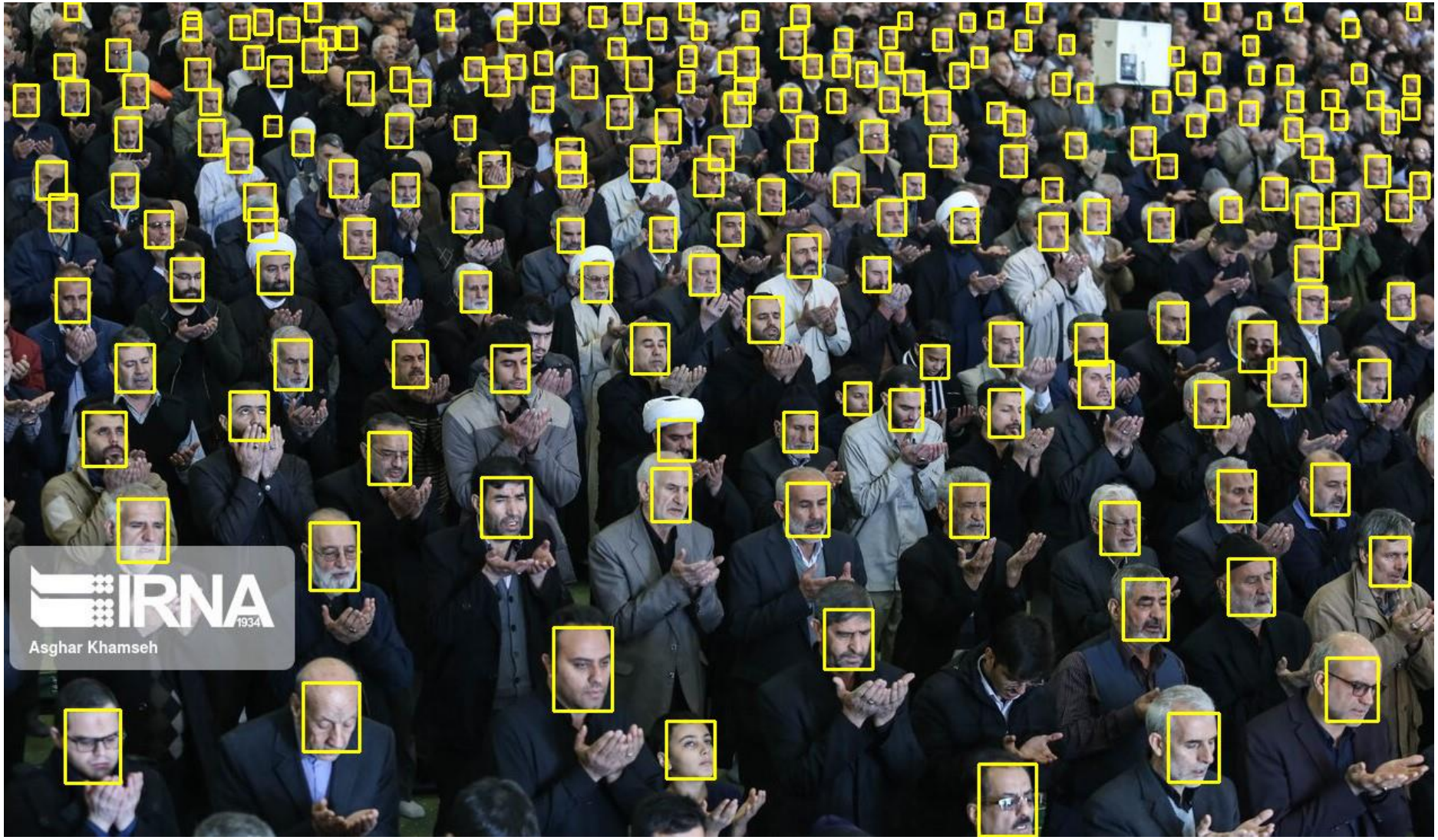


Face Analysis: Introduction

- Block diagram of a face recognition system

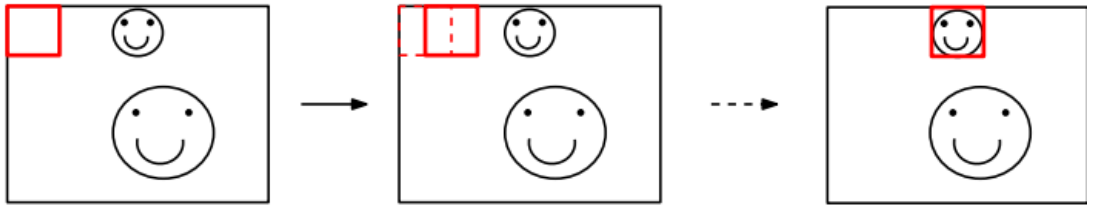


Face Analysis: Face Detection



Face Analysis: Face Detection

- Haar Cascade

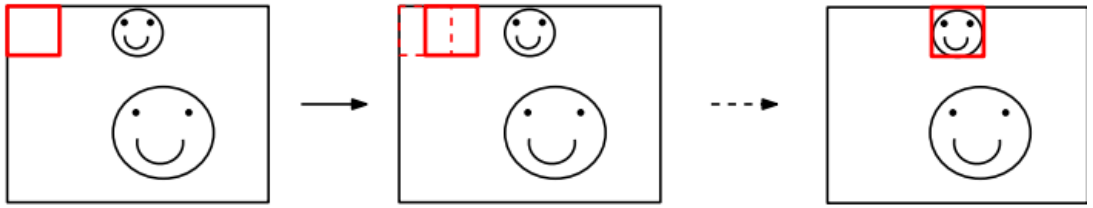


F. Comaschi, et al., RASW: a Run-time Adaptive Sliding Window to Improve Viola-Jones Object Detection

P. Viola & M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", CVPR 2001

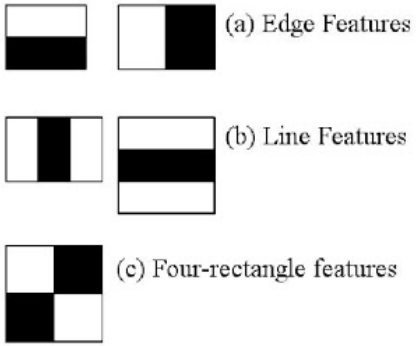
Face Analysis: Face Detection

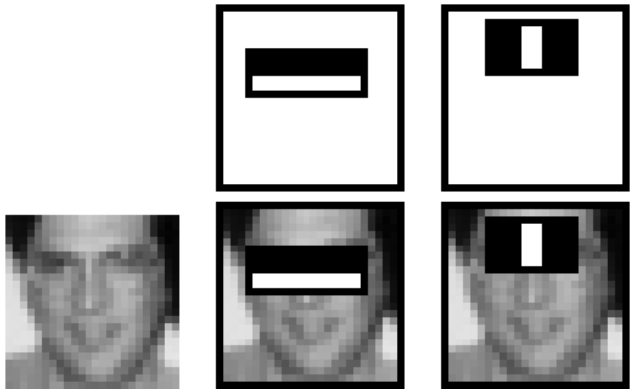
- Haar Cascade

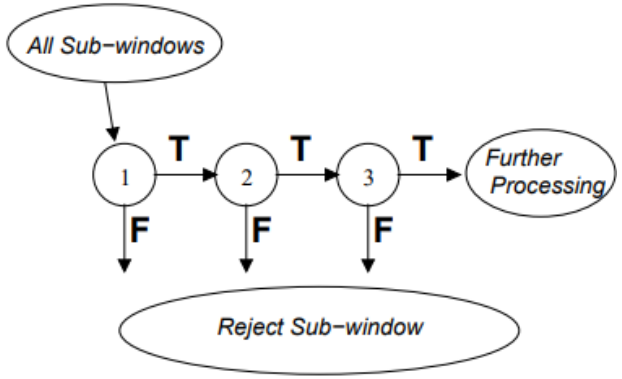


F. Comaschi, et al., RASW: a Run-time Adaptive Sliding Window to Improve Viola-Jones Object Detection

Each feature: ∑(pixels under black rectangle) - ∑(pixels under white rectangle)







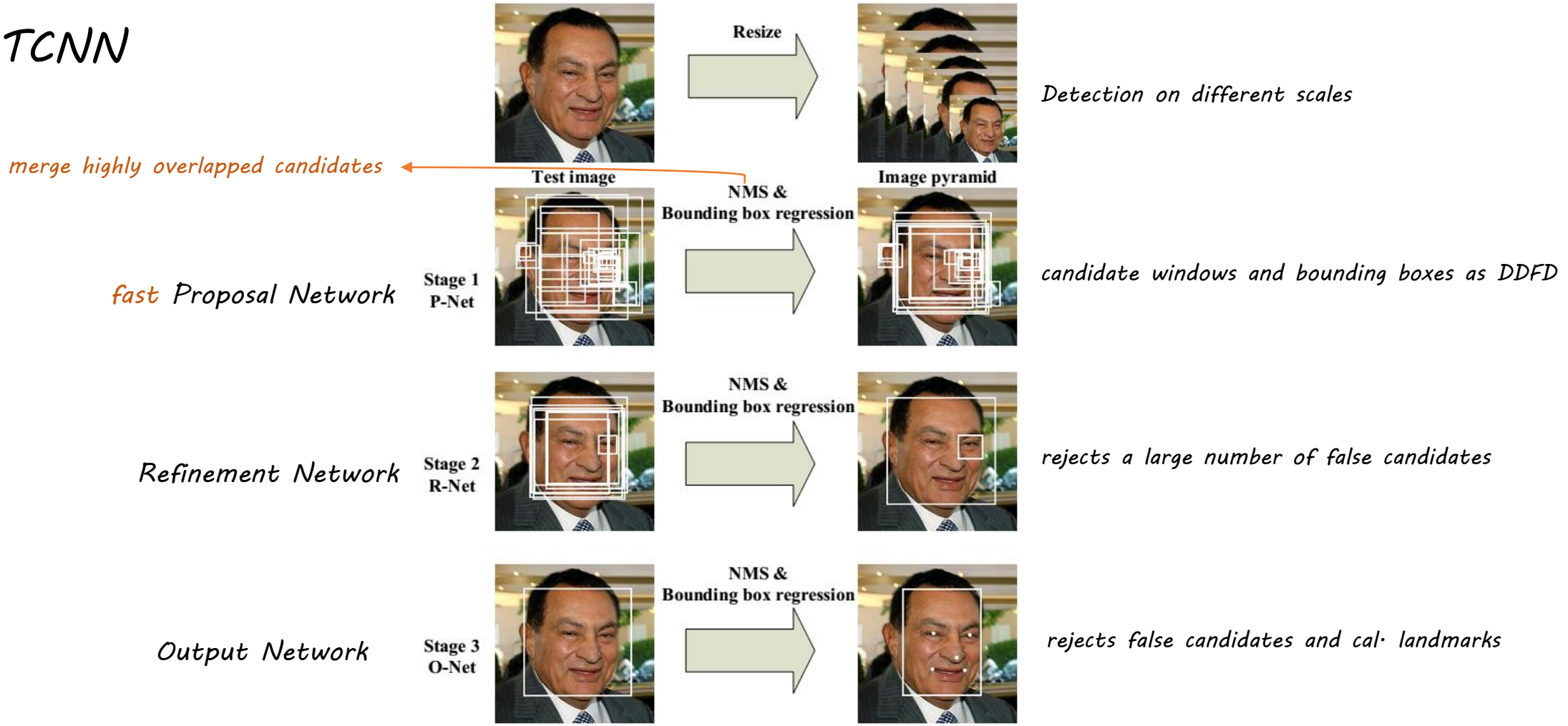
Code:

https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08
https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html

P. Viola & M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", CVPR 2001

Face Analysis: Face Detection

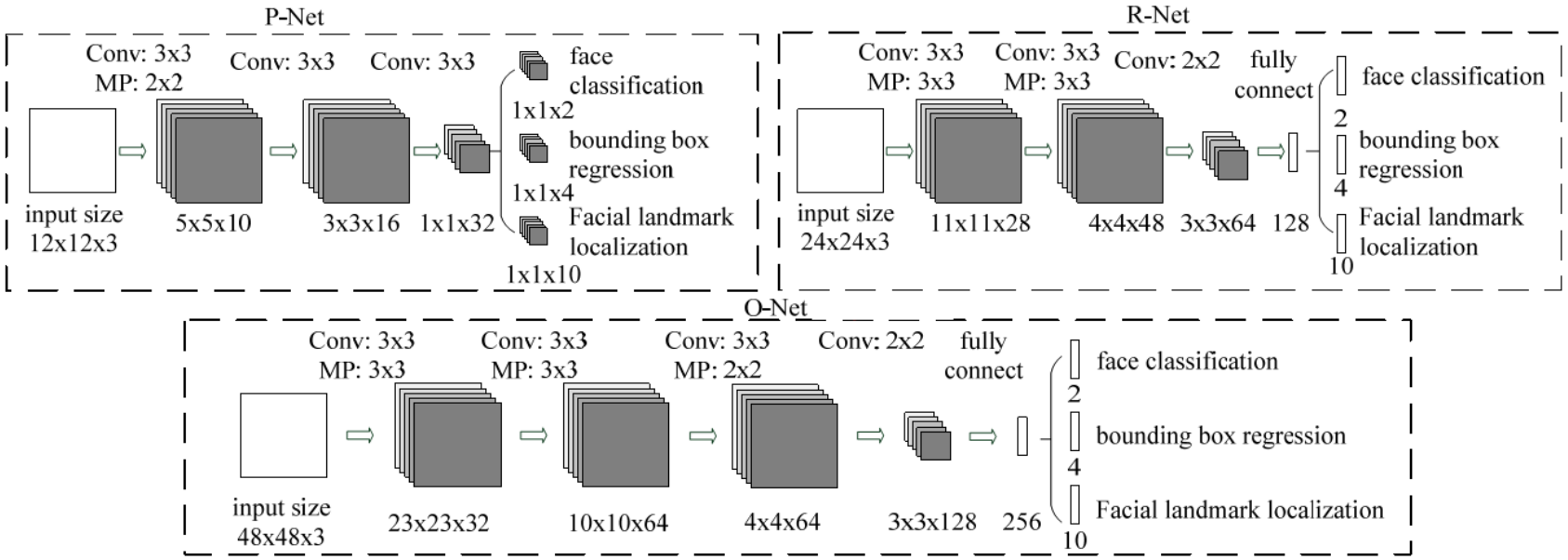
- MTCNN



Zhang, et al., Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks, IEEE Signal Processing Letters, 2016

Face Analysis: Face Detection

- MTCNN



Code:

Python package: https://github.com/ipazc/mtcnn
https://github.com/davidsandberg/facenet/tree/master/src/align
https://github.com/kpzhang93/MTCNN_face_detection_alignment

Face Analysis: Face Detection

- MTCNN

Face classification loss:

$$L_i^{det} = -\left(y_i^{det}\, log(p_i) + (1 - y_i^{det})(log(1 - p_i))\right) \qquad y_i^{det} \epsilon \{0,1\}$$

Bounding box regression loss:

$$L_i^{box} = \left\| \hat{y}_i^{box} - y_i^{box} \right\|_2^2$$

landmark localization loss:

$$L_i^{Landmark} = \left\| \hat{y}_i^{Landmark} - y_i^{Landmark} \right\|_2^2$$

overall learning target:

$$\min \sum_{i=1}^{N} \sum_{j \in (det, box, landmark)} \alpha_j \beta_i^j L_i^j$$

$\beta_i^j \epsilon \{0,1\}$ (sample type indicator)

task importance: $\begin{cases} PNet, RNet: & \alpha_{det} = 1, \alpha_{box} = 0.5, \alpha_{landmark} = 0.5 \\ ONet: & \alpha_{det} = 1, \alpha_{box} = 0.5, \alpha_{landmark} = 1 \end{cases}$

Zhang, et al., Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks, IEEE Signal Processing Letters, 2016

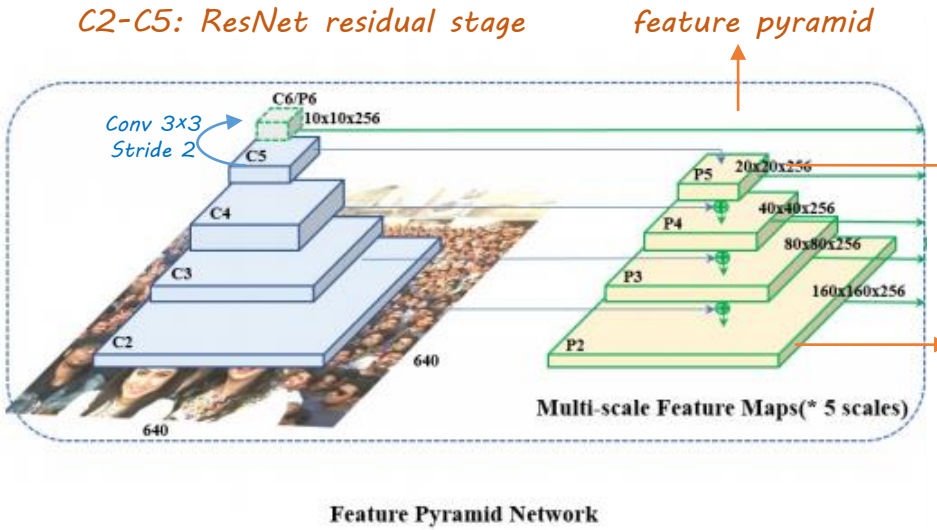
Face Analysis: Face Detection

- Implementation (MTCNN):

https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/face_detection_mtcnn.ipynb

Face Analysis: Face Detection

- RetinaFace



to capture larger faces

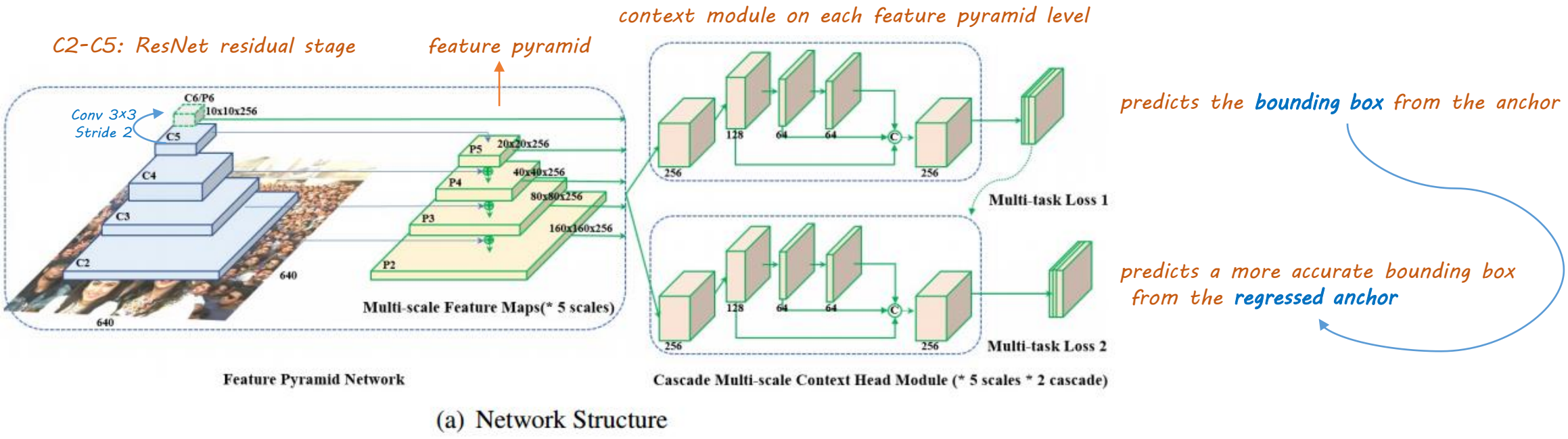
to capture tiny faces by **tiling** small **anchors**

75% of all anchors (102,300 anchors for a 640 x 640 image) are from P2

Deng, et al., RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild, CVPR 2020

Face Analysis: Face Detection

- RetinaFace

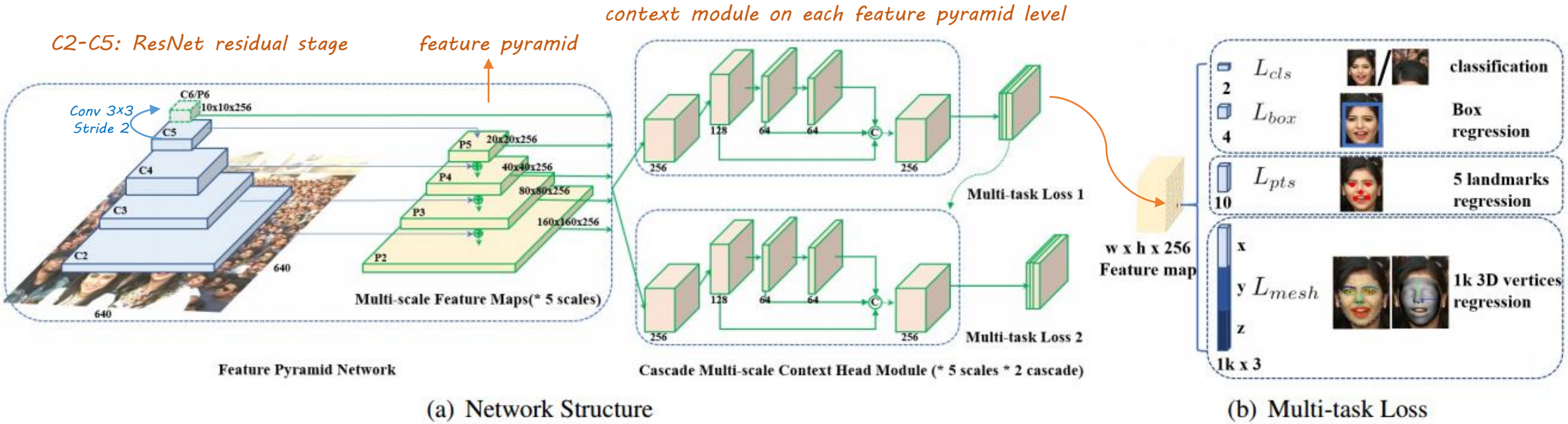


(a) Network Structure

Deng, et al., RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild, CVPR 2020

Face Analysis: Face Detection

- RetinaFace



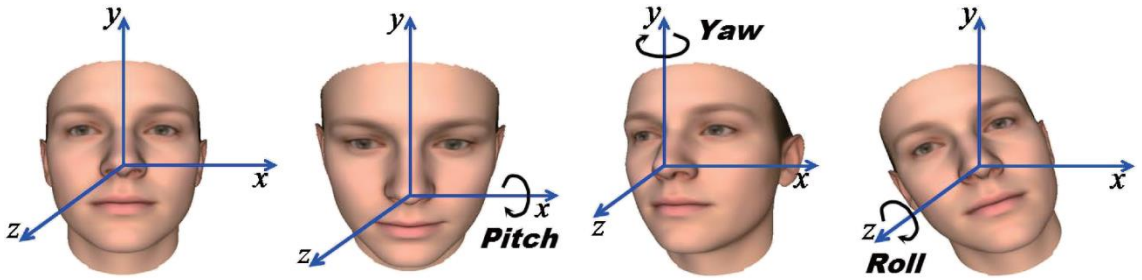
Code:

https://github.com/deepinsight/insightface/tree/master/detection/retinaface

Deng, et al., RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild, CVPR 2020

Face Analysis: Preprocessing

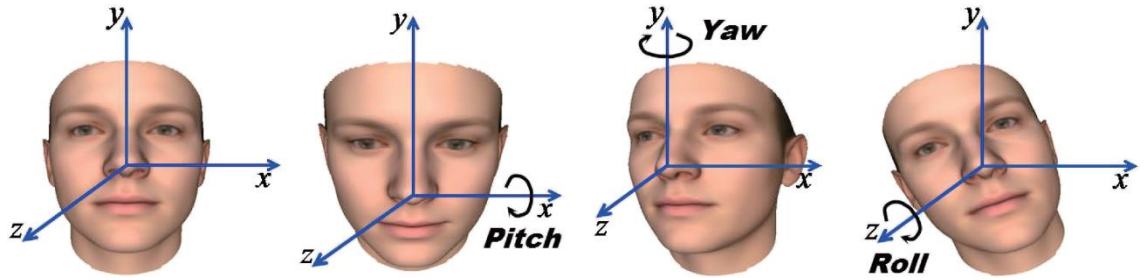
- Head pose



G. Sang, et al., Learning toward practical head pose estimation, SPIE, 2017

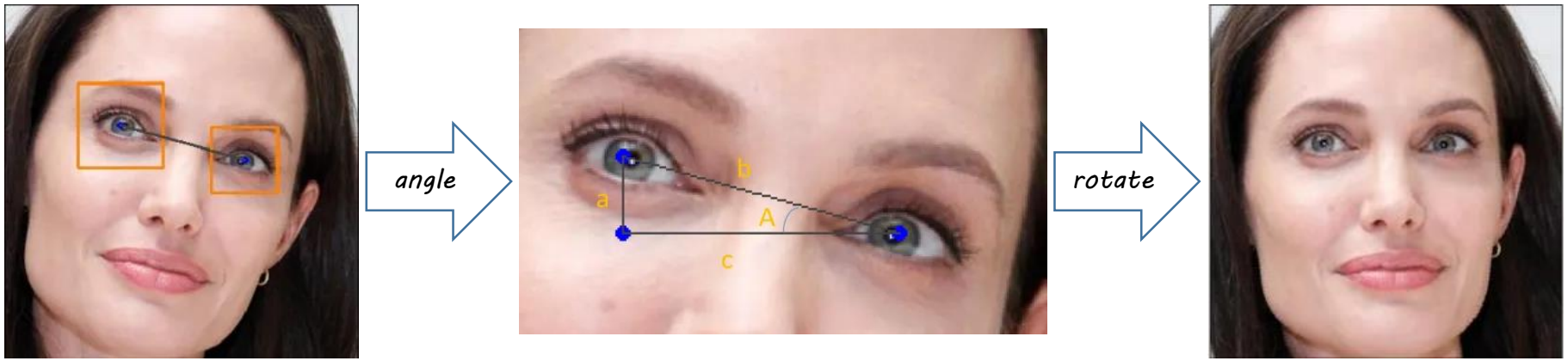
Face Analysis: Preprocessing

- Head pose



G. Sang, et al., Learning toward practical head pose estimation, SPIE, 2017

- Roll:

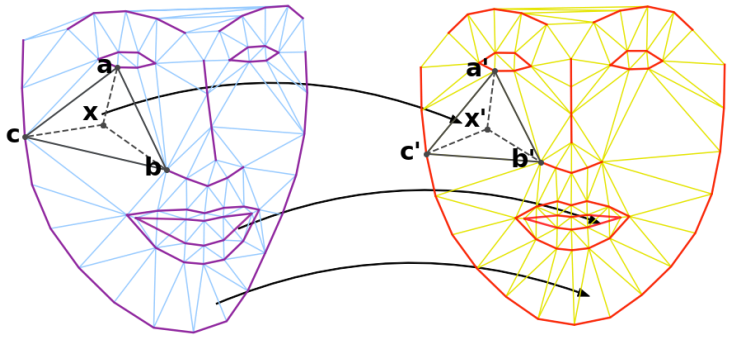


Code: https://sefiks.com/2020/02/23/face-alignment-for-face-recognition-in-python-within-opencv/

Face Analysis: Preprocessing

- Pitch and Yaw

piecewise affine warping



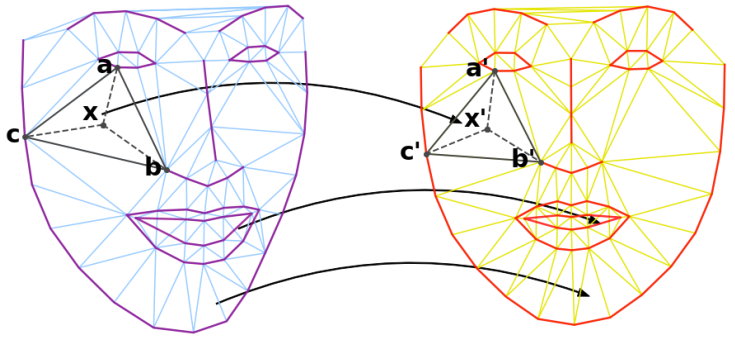
Delaunay triangulation

http://jeankossaifi.com/pages/gagan.html

Face Analysis: Preprocessing

- Pitch and Yaw

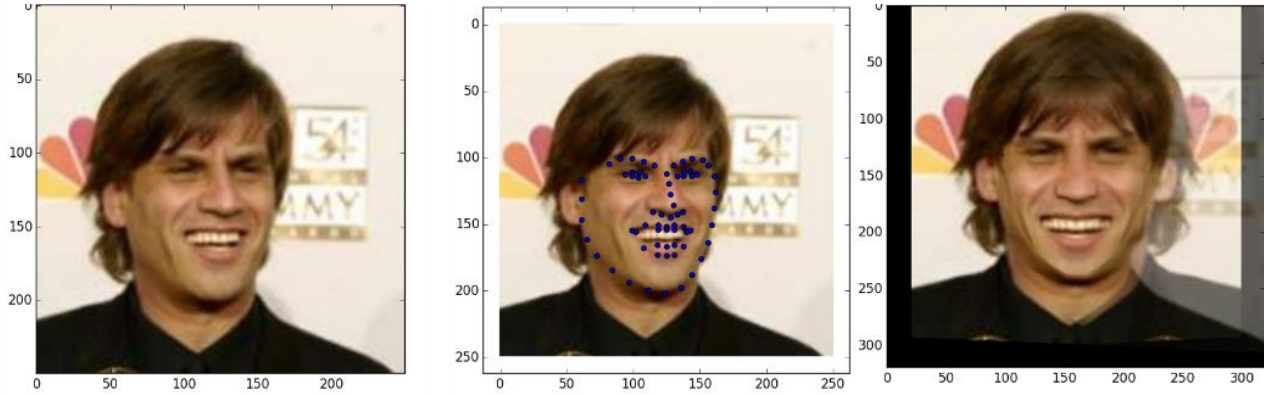
piecewise affine warping



Delaunay triangulation

http://jeankossaifi.com/pages/gagan.html

Face Frontalization

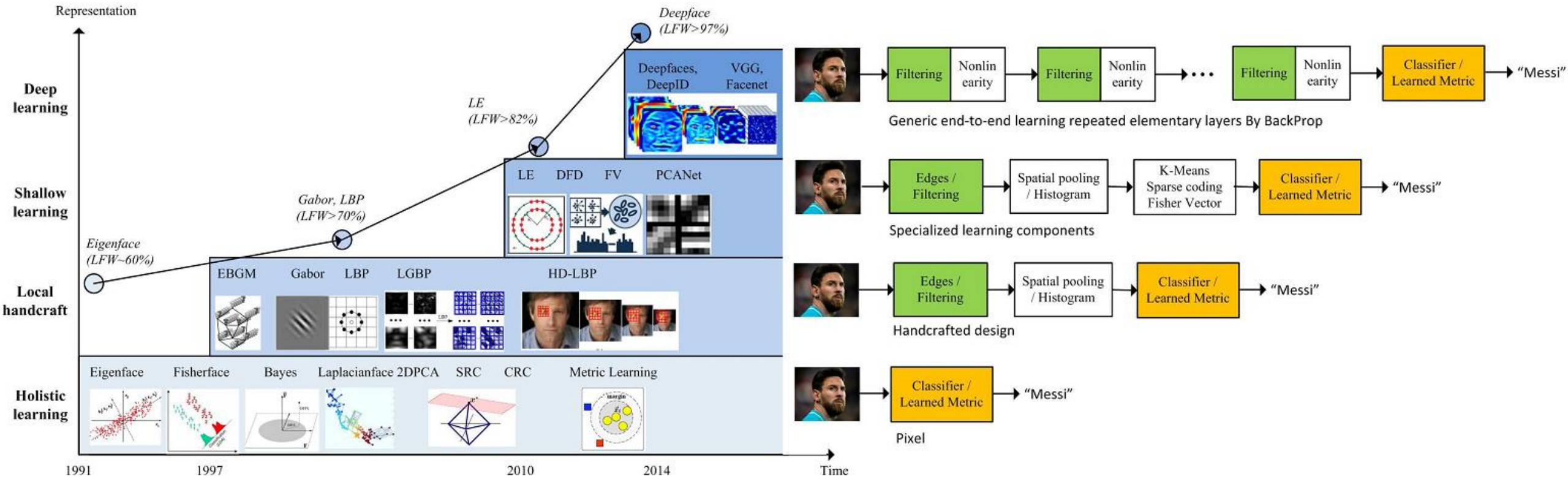


Code: https://github.com/dougsouza/face-frontalization

Insightface Code: https://github.com/deepinsight/insightface/tree/48cc12e3d83c5e98122068eaf04f5d9829c8b399/alignment/coordinateReg

Face Analysis: Face Recognition

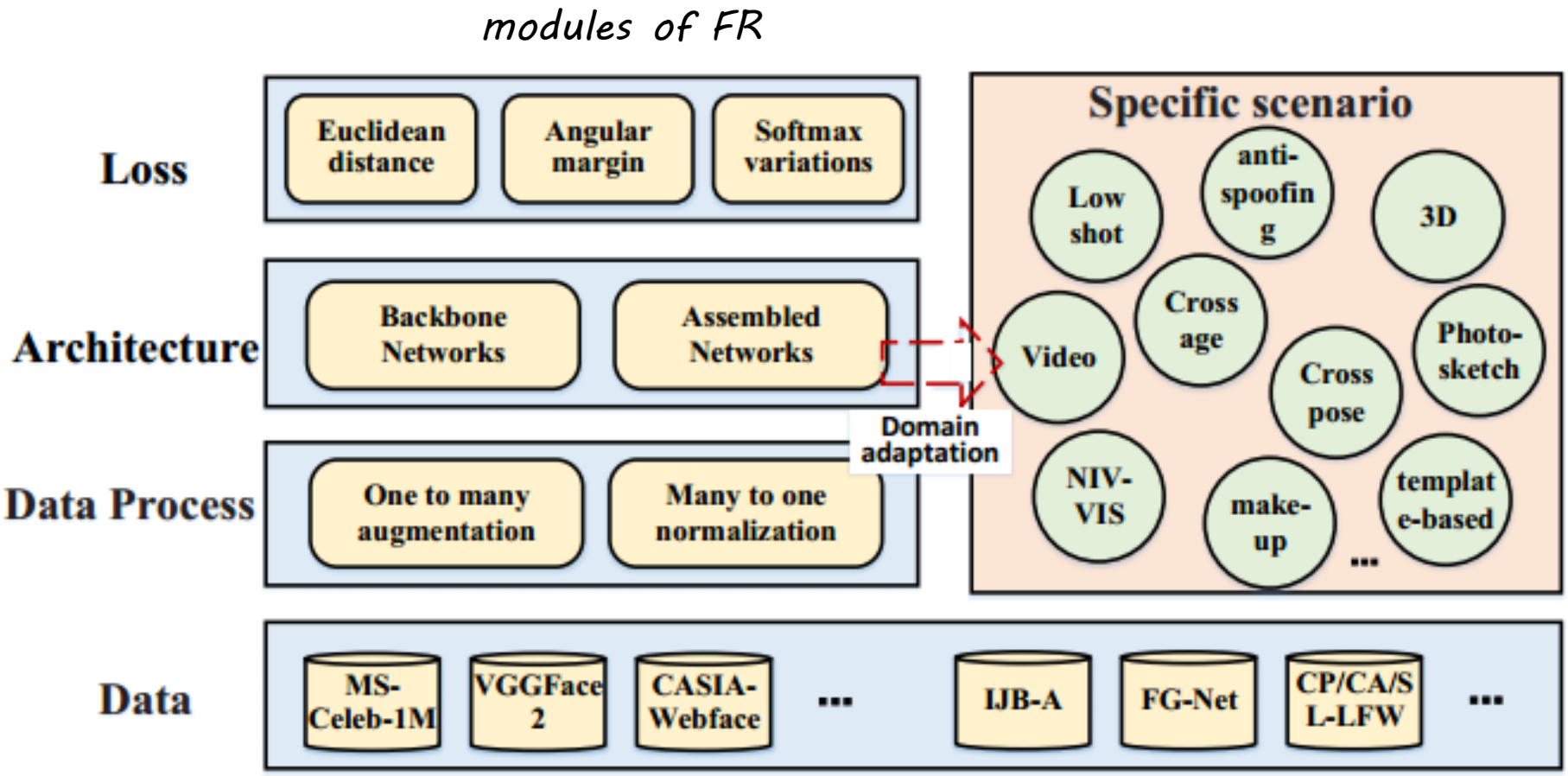
Milestones of face representation for recognition



DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Wang & Deng, Deep face recognition: A survey, Neurocomputing 2021

Face Analysis: Face Recognition

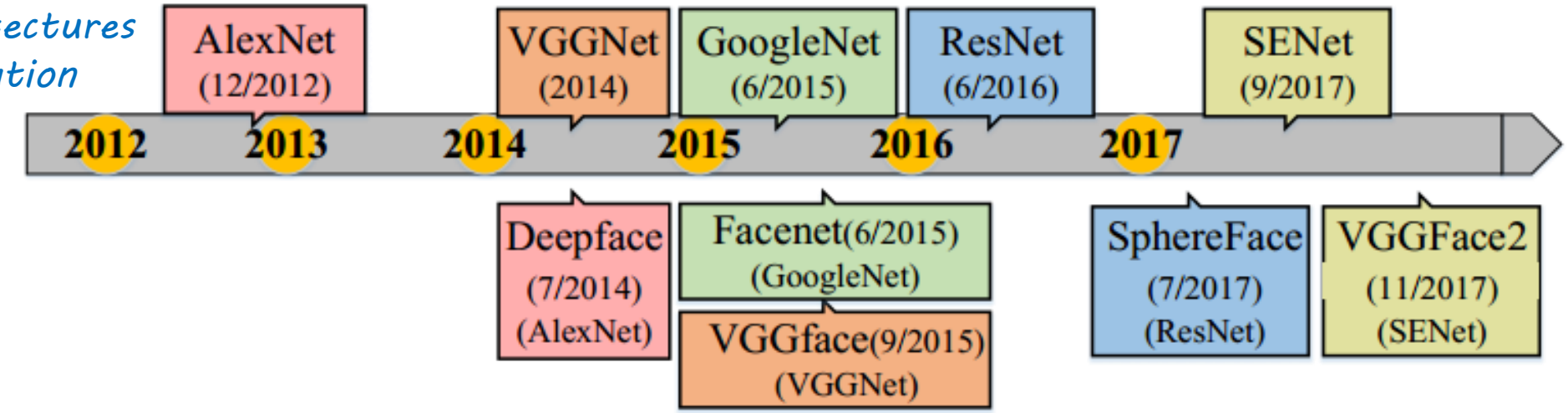


Wang & Deng, Deep face recognition: A survey, Neurocomputing 2021

Face Analysis: Face Recognition

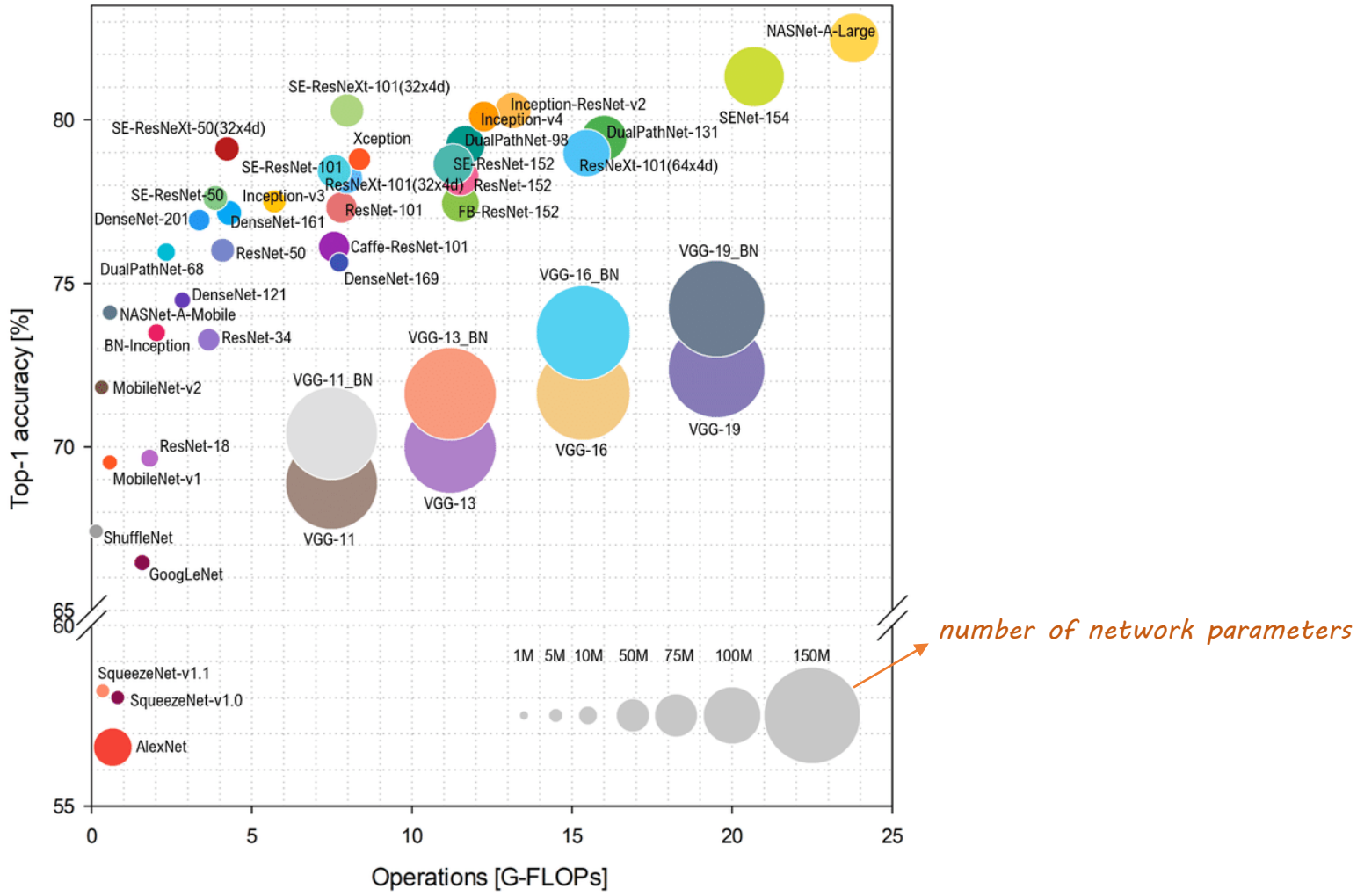
Typical network architectures in object classification

Deep FR algorithms



Wang & Deng, Deep face recognition: A survey, Neurocomputing 2021

Face Analysis: Face Recognition

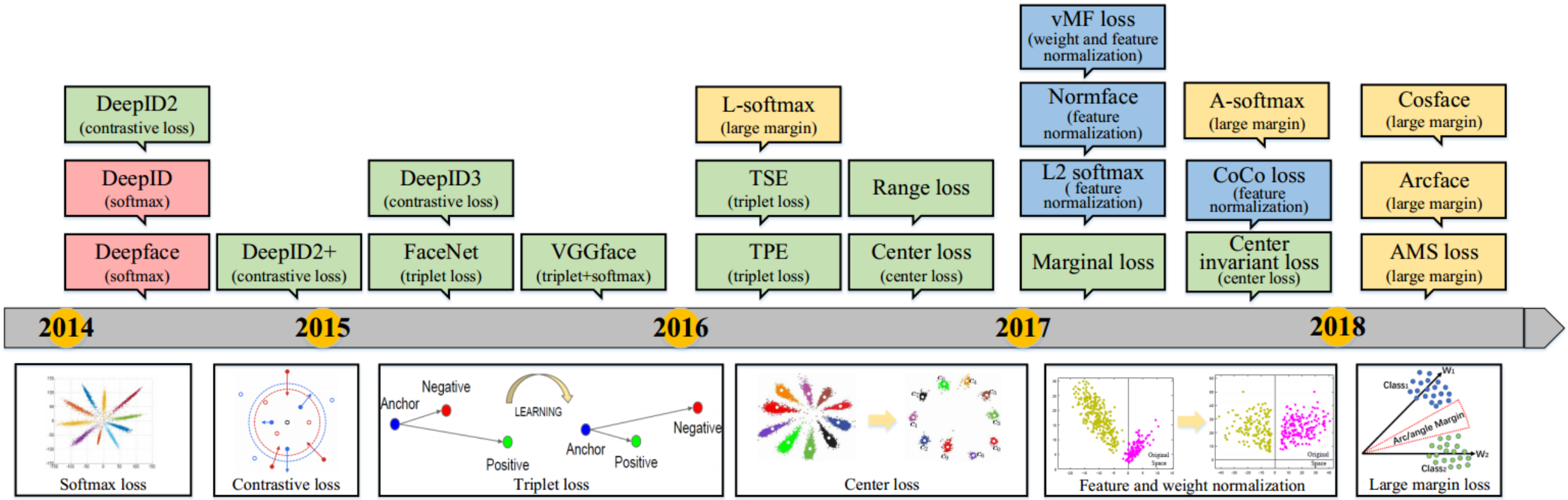


A. Canziani, An analysis of deep neural network models for practical applications, 2016

S. Bianco, Benchmark Analysis of Representative Deep Neural Network Architectures, 2018

Face Analysis: Face Recognition

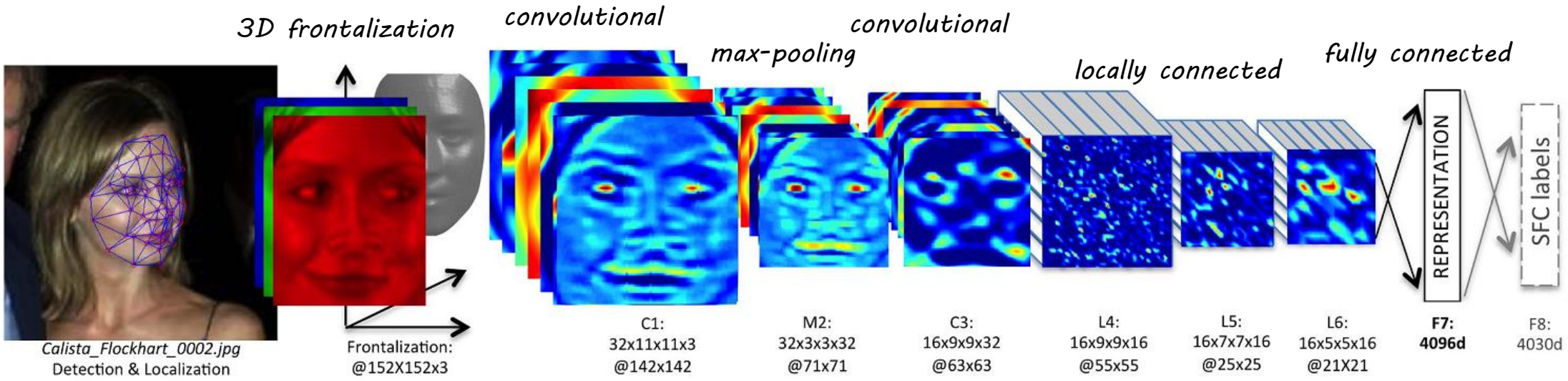
Loss function



Wang & Deng, Deep face recognition: A survey, Neurocomputing 2021

Face Analysis: Face Recognition

- DeepFace

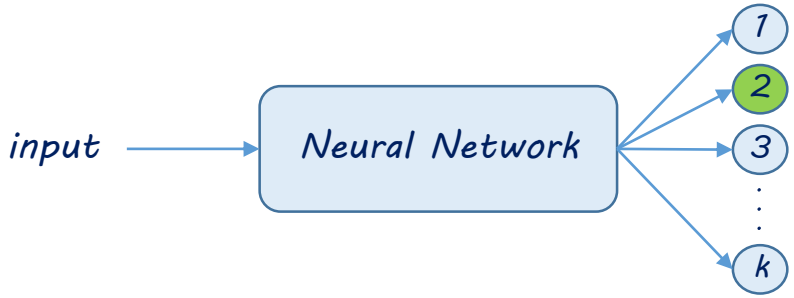


1 - Train the NN using Softmax (for multi-class classification or identification)
2 - Train the two topmost layers using Siamese (for verification)

Taigman, et al. , DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR 2014

Face Analysis: Face Recognition

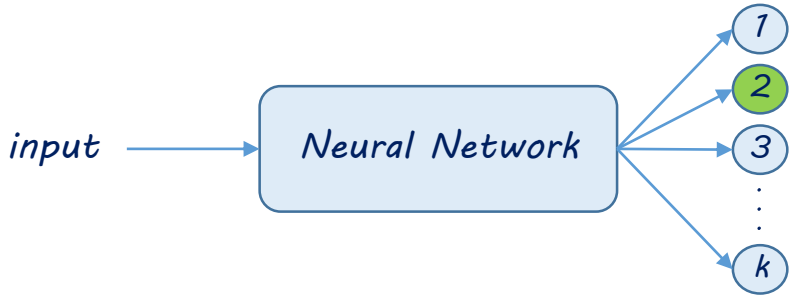
- Softmax



$$loss = -\frac{1}{N}\sum_{i} log\frac{e^{O_i}}{\sum_j e^{O_j}}$$

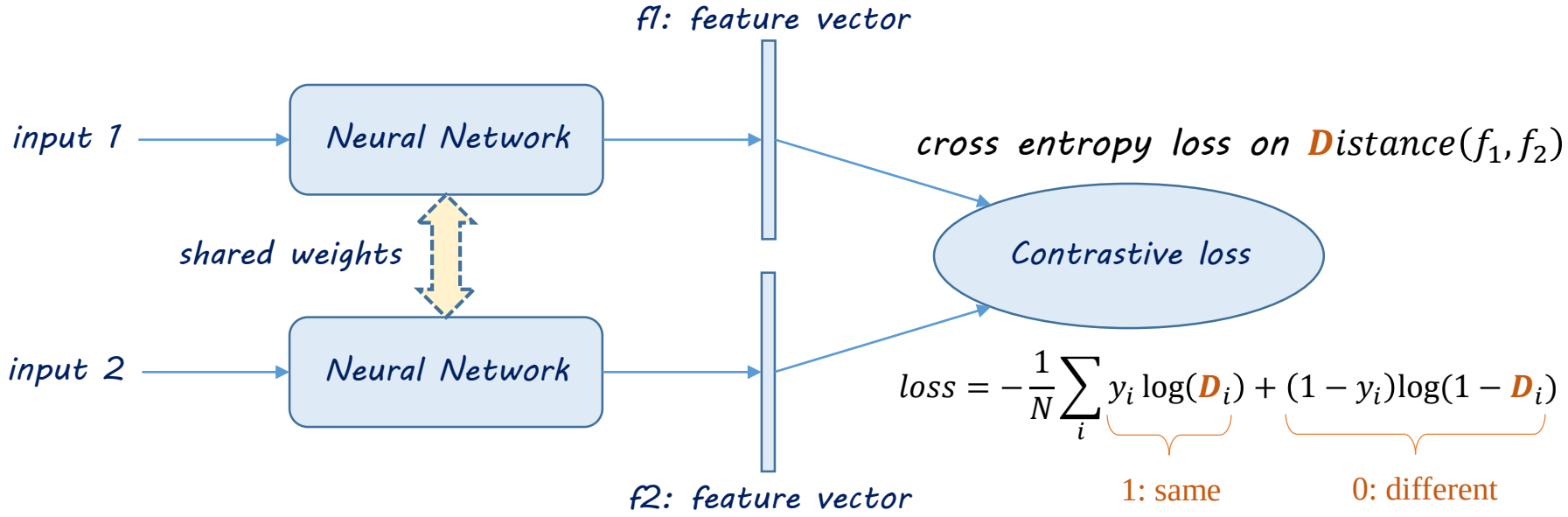
Face Analysis: Face Recognition

- Softmax



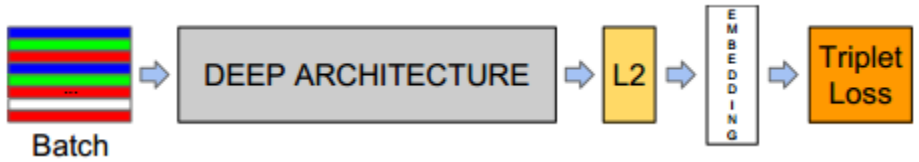
$$loss = -\frac{1}{N}\sum_i log\frac{e^{O_i}}{\sum_j e^{O_j}}$$

- Siamese



Face Analysis: Face Recognition

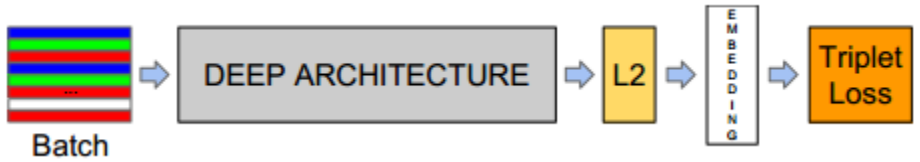
- FaceNet

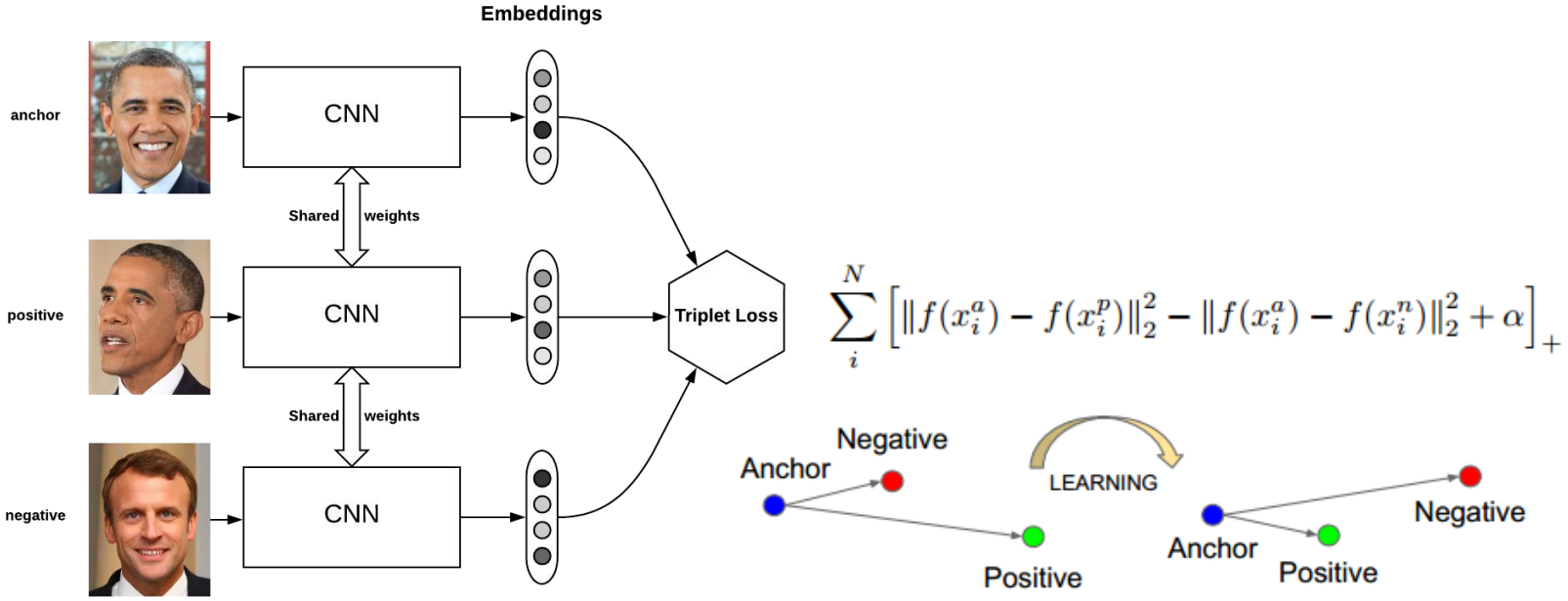


Schroff, et al., FaceNet: A Unified Embedding for Face Recognition and Clustering, CVPR 2015

Face Analysis: Face Recognition

- FaceNet





https://omoindrot.github.io/triplet-loss

Code: https://github.com/davidsandberg/facenet

Face Analysis: Face Recognition

- ArcFace

Softmax:

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}}$$

Deng, et al., ArcFace: Additive Angular Margin Loss for Deep Face Recognition, CVPR 2019

Face Analysis: Face Recognition

- ArcFace

Softmax:

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}}$$

Label of x_i; #classes; i^{th} data; bias; Batch size

Deng, et al.: "Softmax loss function does not explicitly optimise the feature embedding to enforce higher similarity for intra-class samples and diversity for inter-class samples"

Pose, Age, etc.

Let $b = 0,\ \|W_j\| = 1,\ \|x_i\| = s$

$W_j^T x_i = \|W_j\|\|x_i\| \cos\theta_j$

$$L_2 = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s\cos\theta_{y_i}}}{e^{s\cos\theta_{y_i}} + \sum_{j=1, j\neq y_i}^{n} e^{s\cos\theta_j}}$$

margin

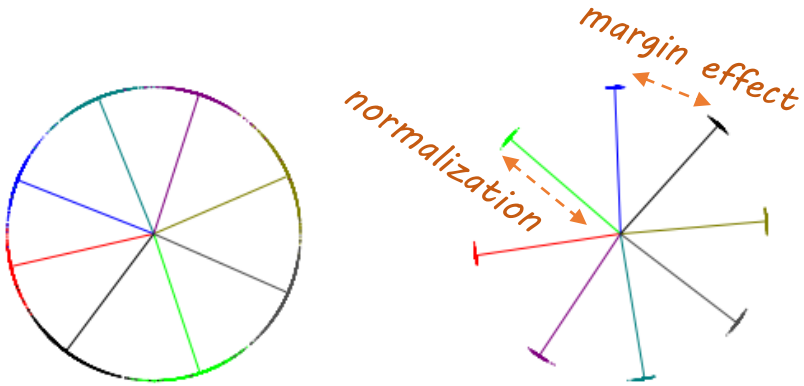
$$L_3 = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j\neq y_i}^{n} e^{s\cos\theta_j}}$$

Deng, et al., ArcFace: Additive Angular Margin Loss for Deep Face Recognition, CVPR 2019

Face Analysis: Face Recognition

- ArcFace

8 identities with 2D normalized features:



(a) Softmax (b) ArcFace

- Code:

https://github.com/deepinsight/insightface

python library, Originally MXNet, newly PyTorch

Face Analysis: Face Recognition

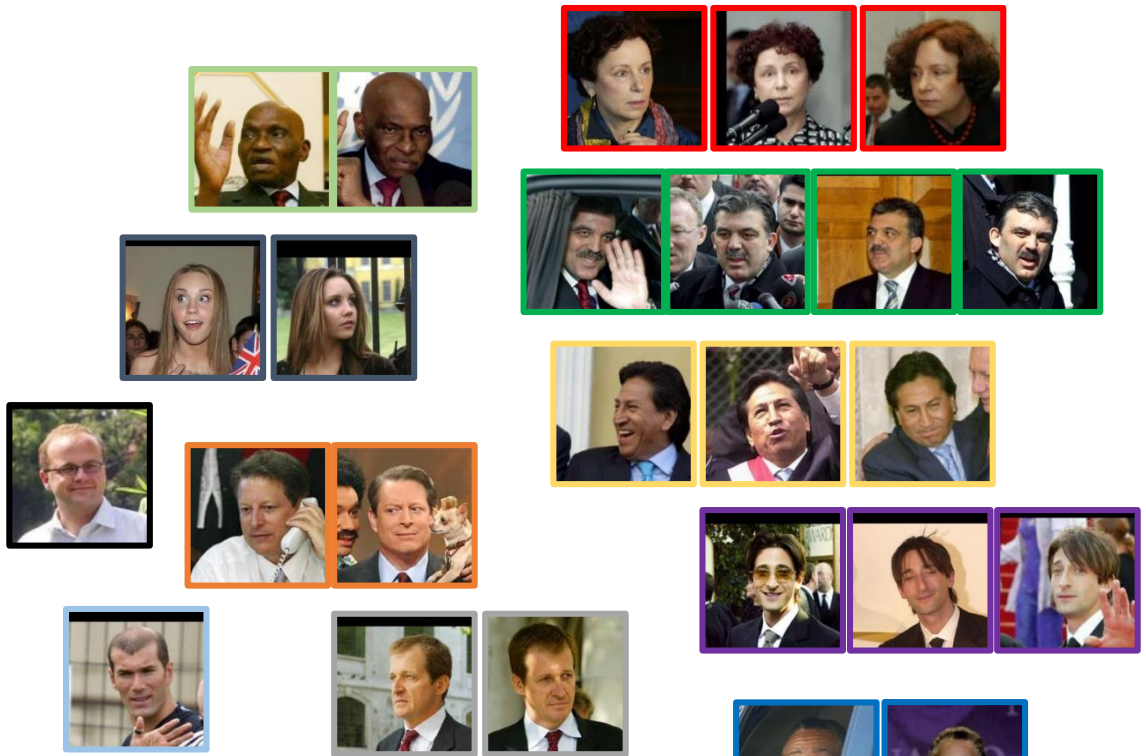
- Implementations:
  - face recognition using dlib:

    https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/face_recognition_using_dlib.ipynb

  - face recognition using ArcFace (insightface):

    https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/arcface.ipynb

A Complete Face Clustering Algorithm



Face Analysis: Face Clustering

- Implementations:
  - Face clustering using ArcFace on a small set:

    https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/face_clustering_arcface.ipynb

  - Face clustering using ArcFace on FaceCup sample dataset:

    https://colab.research.google.com/github/hamidsadeghi68/face-clustering/blob/main/face_clustering_arcface_facecup_samples.ipynb