Machine Learning Fundamentals – DTSC102

Lecture 4
K Nearest Neighbors - KNN

Course Instructors: Dr.-Ing. Maggie Mashaly
maggie.ezzat@guc.edu.eg
C3.220

Dr.- Ing. Maggie Mashaly

GUC German University in Cairo

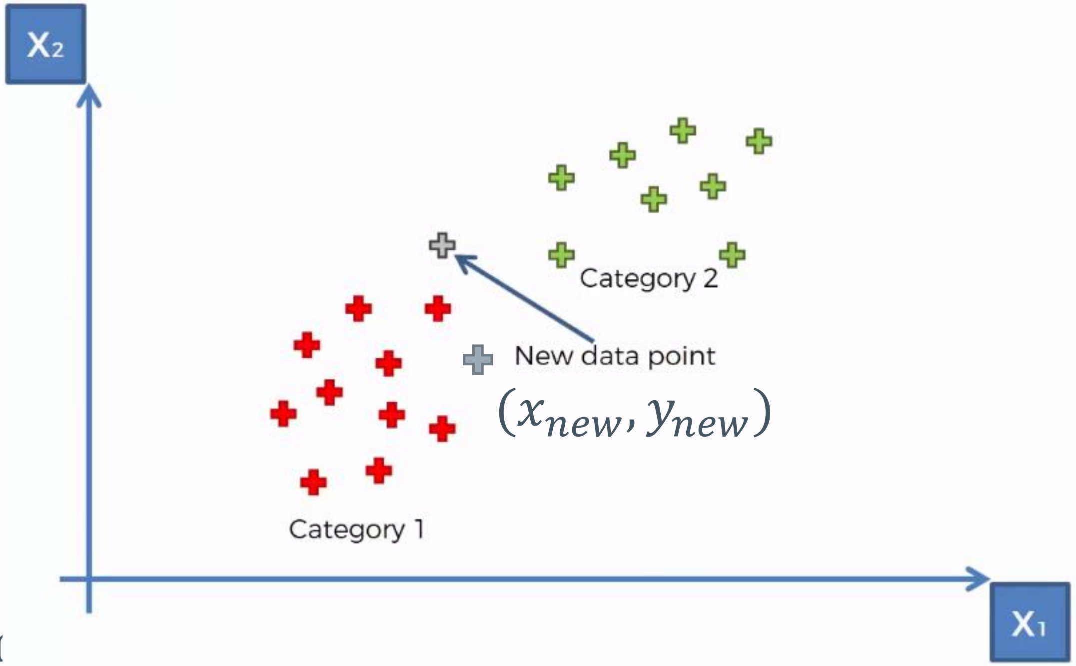
K-Nearest Neighbors : Intuition

Givens:
- Set of points (x_i, y_i), $\mathrm{i} = 1, .., \mathrm{m}$
- Two output classes

Requirement:
- Predict the class of a new given point (x_{new}, y_{new})
- Perform **Classification**

How to do it?

.. the intuitive solution:
- near points are mostly red, so n likely (x_{new}, y_{new}) as well is red

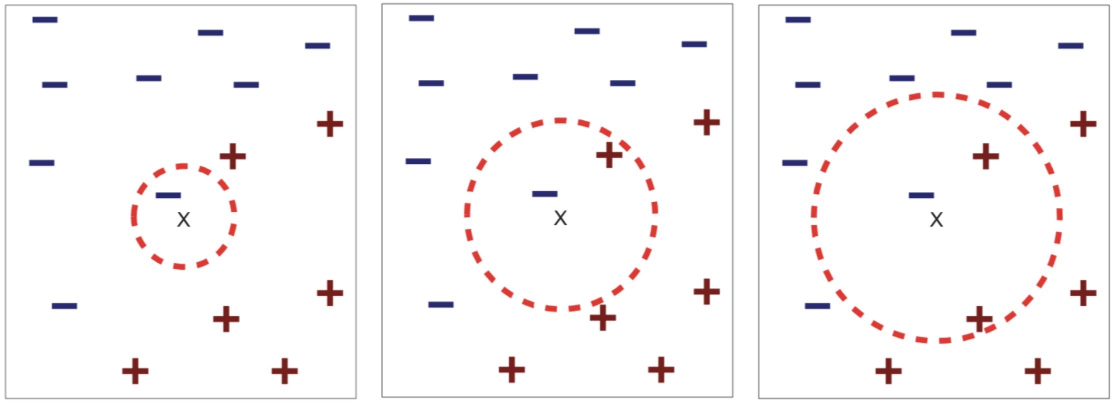


K-Nearest Neighbors : Introduction

- Amongst the simplest of all machine learning algorithms.
- No explicit training or model.
- A supervised learning technique, can be used both for classification and regression.

Idea:

- Use x's K-Nearest Neighbors to vote on what x's label should be.
- Classify using the majority vote of the k closest training points



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-Nearest Neighbors : How does it work?

- K-NN algorithm does not explicitly compute decision boundaries.
- The boundaries between distinct classes form a subset of the Voronoi diagram of the training data.

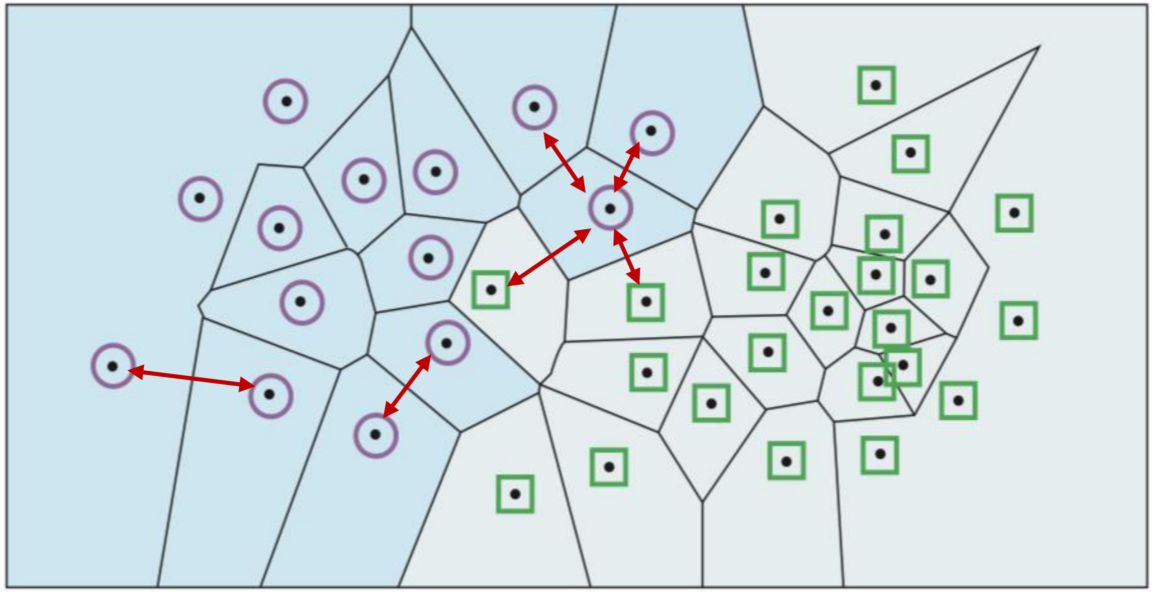


Image by MIT OpenCourseWare

- Each line segment is equidistant to neighboring points.

Base Case: 1-Nearest Neighbor

- Voronoi diagram defines the classification boundary

Any point within this area follows the class of the data point inside

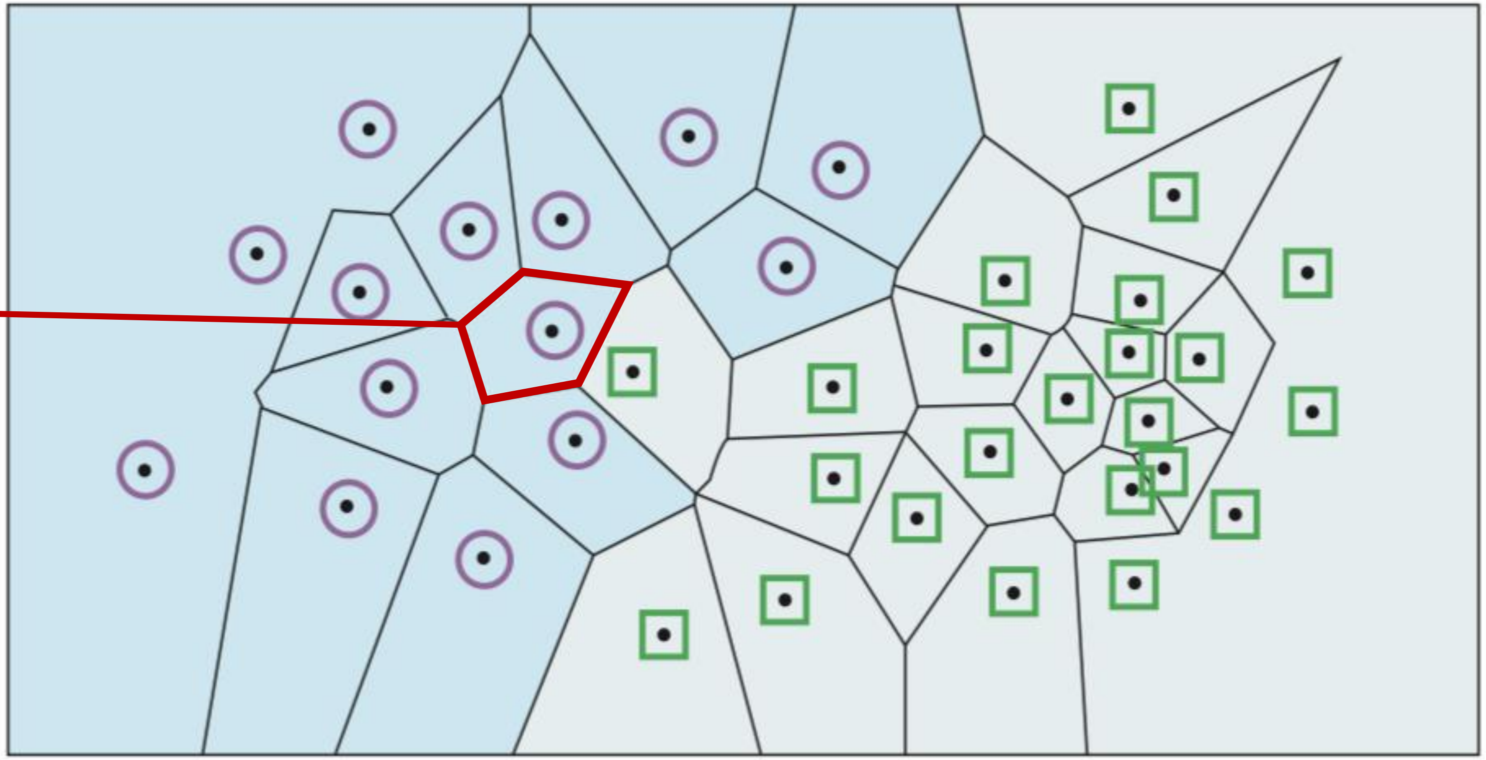


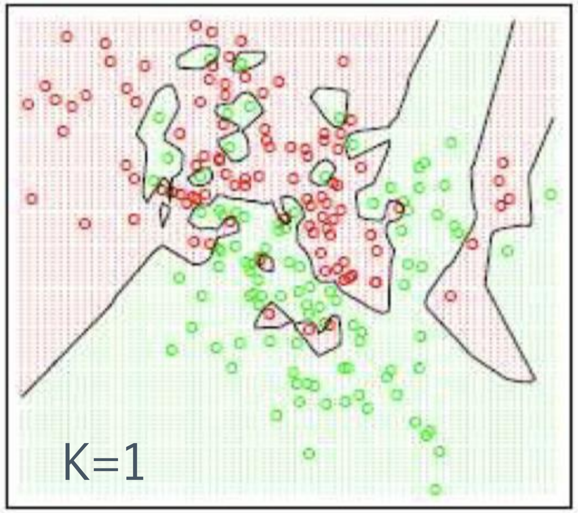
Image by MIT OpenCourseWare

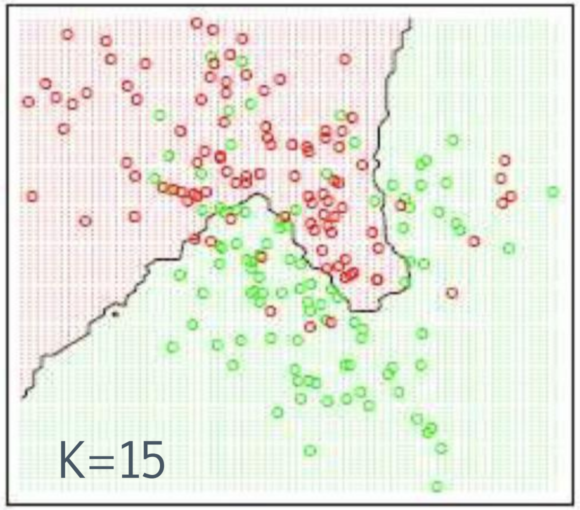
K-Nearest Neighbors : Parameter Selection

How to Select K?

- K too small: we'll model the noise
  e.g.: K=1 yields y=piecewise constant labeling

- K too large: neighbors include too many points from other classes
  e.g.: K=N yields y=majority labeling
  (Majority class will always be predicted)





Generally:

- Larger K produces smoother boundary effect and can reduce the impact of class label noise
- Typically the **k value** is set to the square root of the number of records in your training set.
  e.g.: if your training set is 10,000 records, then the **k value** should be set to sqrt(10000) =100

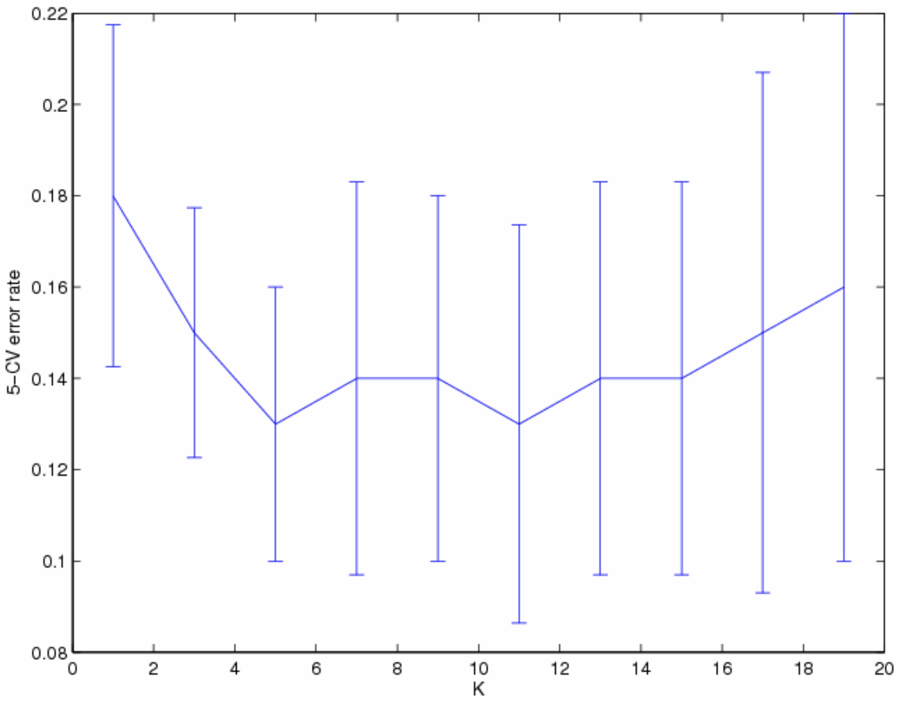
K-Nearest Neighbors : Parameter Selection

How to Select K?

➢ Implement Cross Validation and use it to test K values

60% Training	20% CV	20% Test

- Train you model using Training data for several values of K
- Calculate $Error_{CV}$ for every K, and repeat that several times to have a confidence interval
- Choose the one that yields least error
- Calculate the final $Error_{Test}$ using Test Data



Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Function Selection

How to calculate distance to neighbors?

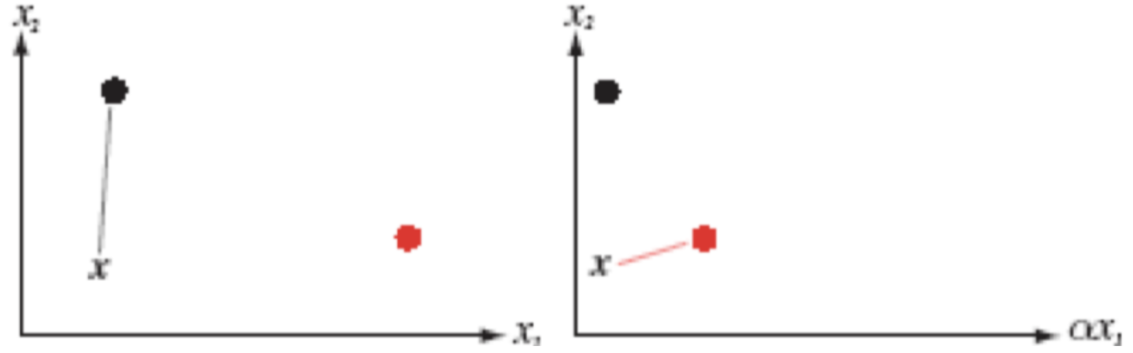
- Could use Euclidean Distance

$$D(x,y)^2 = \|x-y\|^2 = (x-y)^T(x-y) = \sum_{i=1}^{d}(x_i - y_i)^2$$

where d is the dimension of the vector representing a data point

But consider the following:

- If we scale x_1 by $1/3$, Nearest Neighbor changes!



K-Nearest Neighbors : Function Selection

➢ Problems with Euclidean Distance: $D(x,y)^2 = \sum_{i=1}^{d}(x_i - y_i)^2$

Consider the following example:

A: Income ($ 000's)	Lot Size (000's sq ft)
75.0	19.6
52.8	20.8
64.8	17.2
43.2	20.4
84.0	17.6
49.2	17.6

B: Income ($)	Lot Size (000's sq ft)
75,000	19.6
52,800	20.8
64,800	17.2
43,200	20.4
84,000	17.6
49,200	17.6

C: Income ($ 000's)	Lot Size (sq ft)
75.0	19,600
52.8	20,800
64.8	17,200
43.2	20,400
84.0	17,600
49.2	17,600

Do you think the three tables will give the same Euclidean distance between objects?

- Table B will develop distance primarily on the basis of income
    - ➢ Difference in Income is in the order of thousands & it gets squared, while difference in lot size is a one digit value even after squaring

Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Function Selection

➢ Problems with Euclidean Distance: $D(x,y)^2 = \sum_{i=1}^{d}(x_i - y_i)^2$

Consider the following example:

A Income ($ 000's)	Lot Size (000's sq ft)
75.0	19.6
52.8	20.8
64.8	17.2
43.2	20.4
84.0	17.6
49.2	17.6

B Income ($)	Lot Size (000's sq ft)
75,000	19.6
52,800	20.8
64,800	17.2
43,200	20.4
84,000	17.6
49,200	17.6

C Income ($ 000's)	Lot Size (sq ft)
75.0	19,600
52.8	20,800
64.8	17,200
43.2	20,400
84.0	17,600
49.2	17,600

Do you think the three tables will give the same Euclidean distance between objects?

- Table C will develop distance primarily on the basis of Lot size
  - ➢ As Lot size is now in order of thousands while Income is scaled

➢ So Euclidean Distance gets affected by Scale

K-Nearest Neighbors : Function Selection

- Problems with Euclidean Distance: $D(x, y)^2 = \sum_{i=1}^{d}(x_i - y_i)^2$

Consider the following example:

A Income ($ 000's)	Lot Size (000's sq ft)
75.0	19.6
52.8	20.8
64.8	17.2
43.2	20.4
84.0	17.6
49.2	17.6

B Income ($)	Lot Size (000's sq ft)
75,000	19.6
52,800	20.8
64,800	17.2
43,200	20.4
84,000	17.6
49,200	17.6

C Income ($ 000's)	Lot Size (sq ft)
75.0	19,600
52.8	20,800
64.8	17,200
43.2	20,400
84.0	17,600
49.2	17,600

What is the way out?

- Normalization → $z = \frac{x-\mu}{\sigma}$

Now that everything is within the same range, is the problem solved?

K-Nearest Neighbors : Function Selection

➢ Problems with Euclidean Distance: $D(x, y)^2 = \sum_{i=1}^{d}(x_i - y_i)^2$

Consider the following table:

- Do you suspect any correlation here?
  - Income and Saving are correlated
- Don't you think that in case of correlation; you are counting same impact multiple time?
  - Yes, correlated features should only count once

Income ($ 000's)	Lot Size (000's sq ft)	Saving ($ 000's)
75.0	20	27.9
52.8	21	23.3
64.8	17	28.6
43.2	20	19.3
84.0	18	34.8
49.2	18	23.0

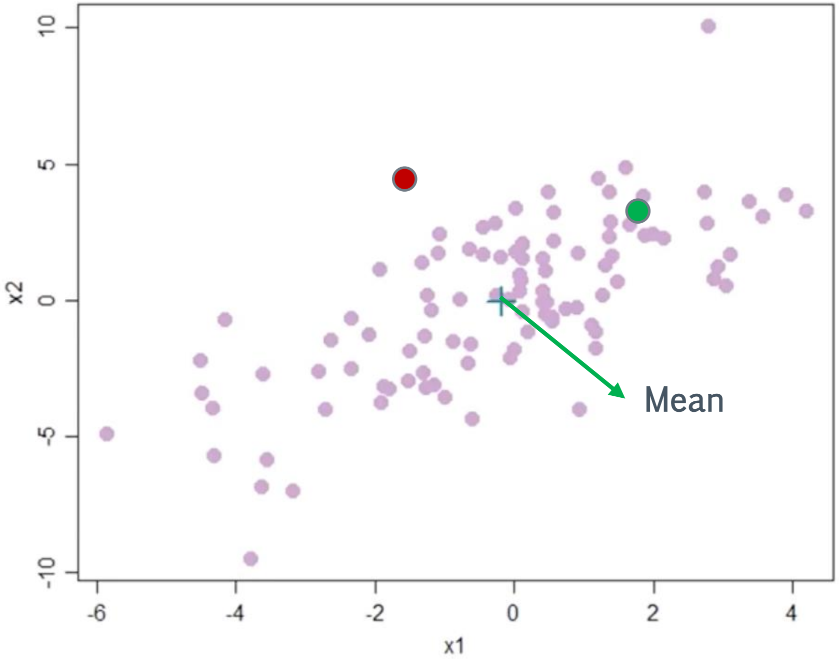
➢ We need another function that avoids both scaling impact as well as correlation impact…

K-Nearest Neighbors : Function Selection

➢ Problems with Euclidean Distance: $D(x,y)^2 = \sum_{i=1}^{d}(x_i - y_i)^2$

Let's visualize the Correlation Problem:

- Consider real data sets which usually have some degree of covariance
  - ➢ Red and Green points both have same Euclidean distance from the mean
  - ➢ Although intuitively, red point seems far away from data points while green point seems very close to them



- It would be easier to calculate distance if we could rescale the coordinates so they didn't have any covariance

K-Nearest Neighbors : Function Selection

- Solution: use Mahalanobis Distance

Measures distance by the following steps:

1. Transforms variables into uncorrelated variables
2. Makes their variance=1
3. Calculates simple Euclidean distance

- The Mahalanobis Distance between two vectors $x = (x_1, x_2, \ldots, x_d)^T$ & $y = (y_1, y_2, \ldots, y_d)^T$ with a Covariance Matrix S is defined as

$$D(x,y)^2 = (x-y)^T S^{-1} (x-y)$$

K-Nearest Neighbors : Function Selection

- The Mahalanobis Distance between two vectors $x = (x_1, x_2, \dots, x_d)^T$ & $y = (y_1, y_2, \dots, y_d)^T$ with a Covariance Matrix S is defined as

Multivariate Normalization, same as $z = \frac{x-\mu}{\sigma}$

$$D(x,y)^2 = (x-y)^T S^{-1} (x-y)$$

Distance between vectors

Covariance Matrix

Note: If the covariance matrix reduces is the Identity Matrix, Mahalanobis Distance reduces to Euclidean Distance

The sample covariance matrix:

$$\underset{p\times p}{S} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{12} & s_{11} & \cdots & s_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ s_{1p} & s_{2p} & \cdots & s_{pp} \end{bmatrix}$$

where

$$s_{ik} = \frac{1}{n-1}\sum_{j=1}^{n}\left(x_{ij}-\bar{x}_i\right)\left(x_{kj}-\bar{x}_k\right)$$

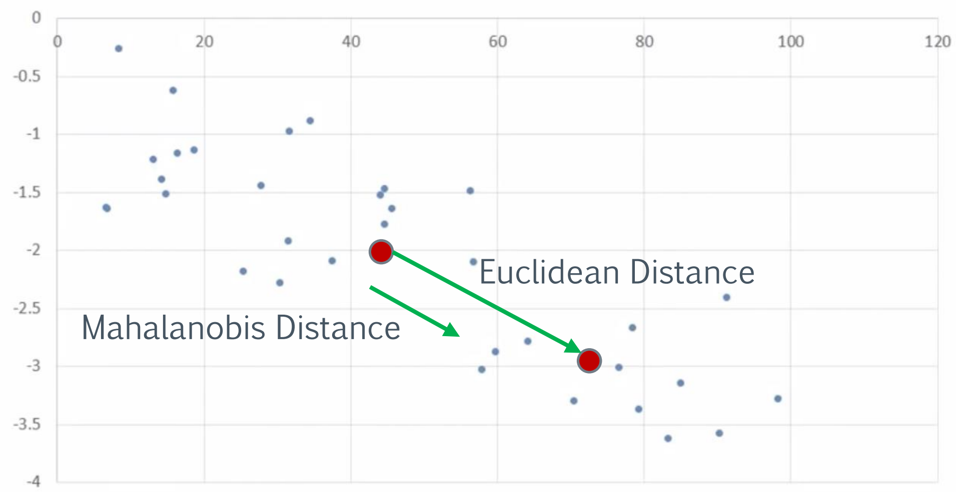
Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Function Selection

➢ Mahalanobis Distance : Intuition

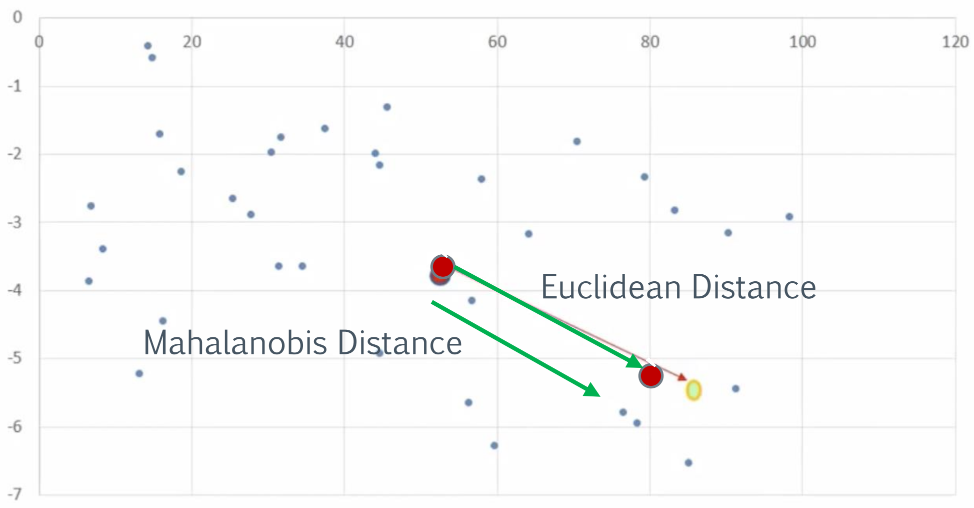
$$D(x,y)^2 = (x-y)^T S^{-1}(x-y)$$

When data has high Covariance



- S will be large
- Dividing by S reduces distance

When data has low Covariance



- S will be small
- Dividing by S doesn't affect distance much

Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Pros & Cons

Pros:

- Simple & Powerful
- No need for tuning complex parameters to build a model.
- Lazy: no training involved
- New training examples can be easily added
- Generic: applies to almost everything as long as you can calculate a distance

K-Nearest Neighbors : Pros & Cons

Cons:

- Expensive and Slow:
  - O(md), m= # examples, d= # dimensions
  - It can be slow to find nearest neighbor in a high dimension space

$$n^* = \arg\min_{n \in D} dist(x, x_n)$$

  - To determine the nearest neighbor of a new point x, must compute the distance to all m training examples.

- Memory intensive, need to store all data

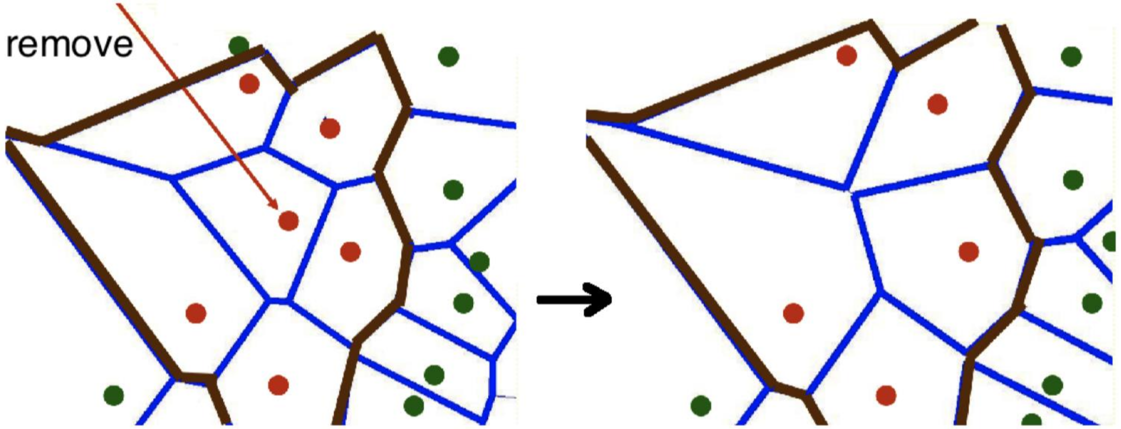
- Need to specify the distance function

Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Pros & Cons

Cons:

- ➢ Runtime performance is slow, but can be improved.
  - ➢ Pre-sort training examples into fast data structures
    ex: Structure the points in a tree (offline computation to save online computation)
  - ➢ Compute only an approximate distance
  - ➢ Remove redundant data (Condensing)
    - If all Voronoi neighbors have the same class, a sample is useless, remove it



Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Pros & Cons

Cons:

- Does not give probabilistic output (why is that important?)
  - Given an input x; KNN returns a single best prediction y
  - A probabilistic classifier would return a probability distribution over outputs $p(y|x) \in [0,1]$
  - If $p(y|x)$ is near 0.5 (very uncertain), the system would better not classify and ask for human help
  - To combine different predictions $p(y|x)$, we need a measure of confidence
  - $p(y|x)$ allows using likelihood as a measure of fit

Solution: Probabilistic KNN

Dr.- Ing. Maggie Mashaly

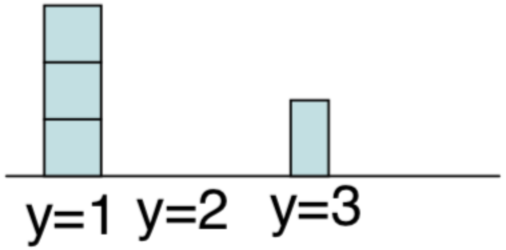
K-Nearest Neighbors : Pros & Cons

Probabilistic KNN

- Compute the empirical distribution over labels in the K-neighborhood

Example: K=4 neighbors, C=3 Classes

$$p = [\frac{3}{4}, 0, \frac{1}{4}]$$

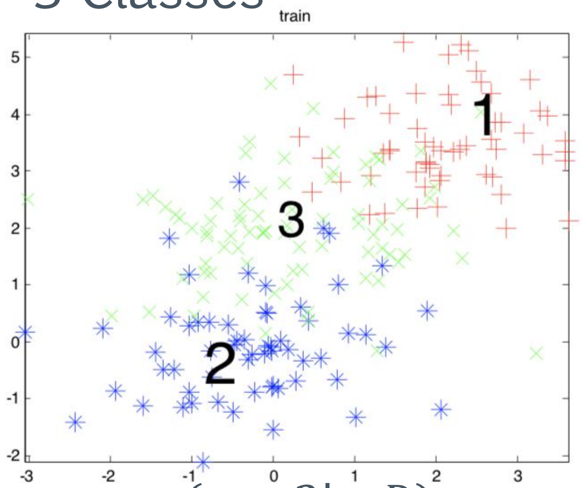


K-Nearest Neighbors : Pros & Cons

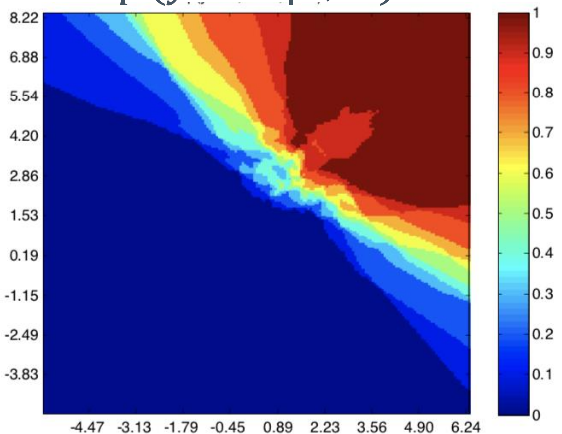
Probabilistic KNN

➢ Compute the empirical distribution over labels in the K-neighborhood

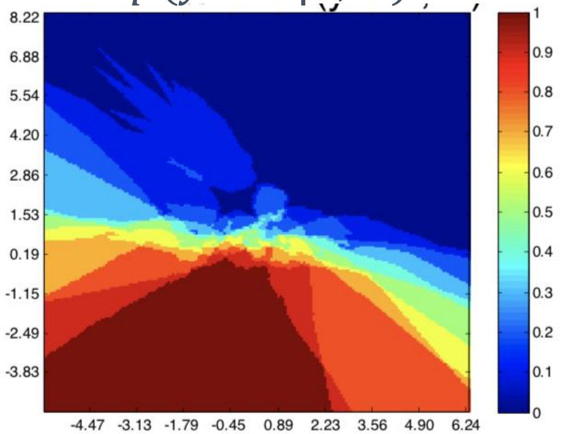
Example: K=4 neighbors, C=3 Classes



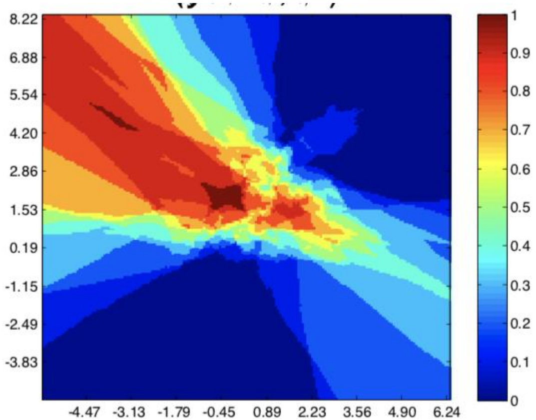
$p(y = 1|x, D)$



$p(y = 2|x, D)$



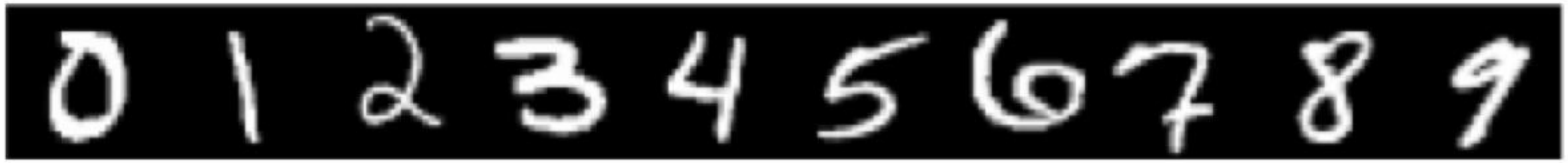
$p(y = 3|x, D)$



K-Nearest Neighbors : Examples

Digit Classification

Decent performance with lots of data



- MNIST Digit Recognition
  - ➢ Handwritten digits
  - ➢ 28*28 pixel images: d=784
  - ➢ 60,000 training samples
  - ➢ 10,000 test samples

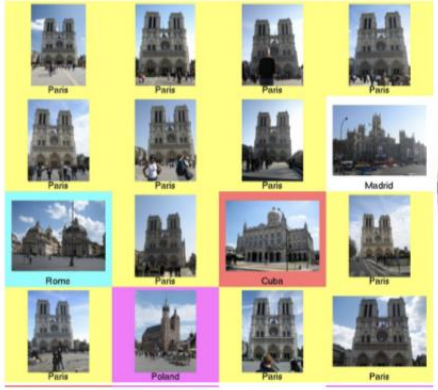
- K - Nearest Neighbor is competitive

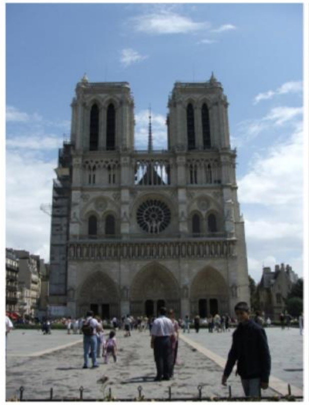
	Test Error Rate (%)
Linear classifier (1-layer NN)	12.0
K-nearest-neighbors, Euclidean	5.0
K-nearest-neighbors, Euclidean, deskewed	2.4
K-NN, Tangent Distance, 16x16	1.1
K-NN, shape context matching	0.67
1000 RBF + linear classifier	3.6
SVM deg 4 polynomial	1.1
2-layer NN, 300 hidden units	4.7
2-layer NN, 300 HU, [deskewing]	1.6
LeNet-5, [distortions]	0.8
Boosted LeNet-4, [distortions]	0.7

K-Nearest Neighbors : Examples

Where on Earth was this photo taken?

- Get 6M images from Flickr with GPs info (dense sampling across world)
- Represent each image with meaningful features
- Do K-NN

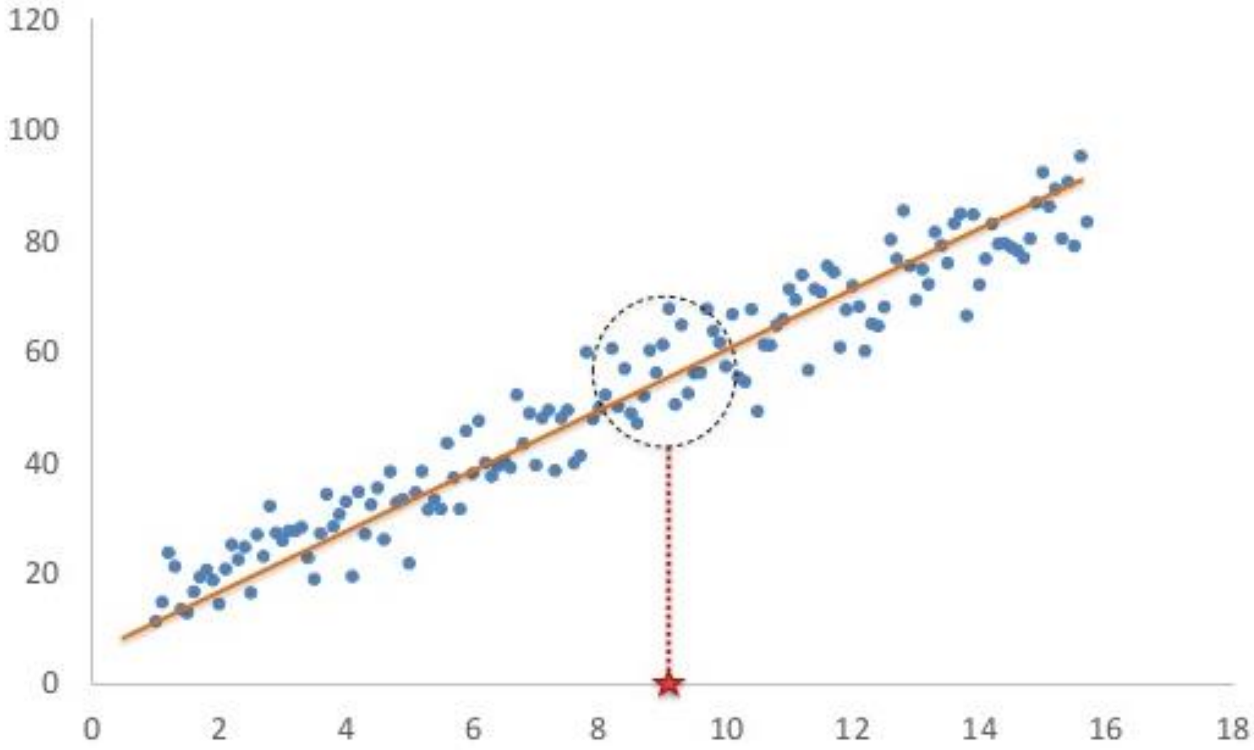




Dr.- Ing. Maggie Mashaly

K-Nearest Neighbors : Regression

- We have explored how to use KNN for Classification
- Could also be used for Regression:
  - The value of the test sample becomes the 'weighted' average of the values of the K neighbors.



Dr.- Ing. Maggie Mashaly

References

- https://www.cs.ubc.ca/~murphyk/Teaching/CS340-Fall07/L4_knn.pdf
- https://www.cs.toronto.edu/~urtasun/courses/CSC411_Fall16/05_nn.pdf
- Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/

For questions please e-mail:
maggie.ezzat@guc.edu.eg