Pattern Classification and Object Detection

Dr. Mongkol Ekpanyapong

Introduction

- The **goal** of pattern classification techniques: to assign a class to each image (or object within an image) based on a numerical representation of the image's (or object's) properties that is most suitable for the problem at hand.

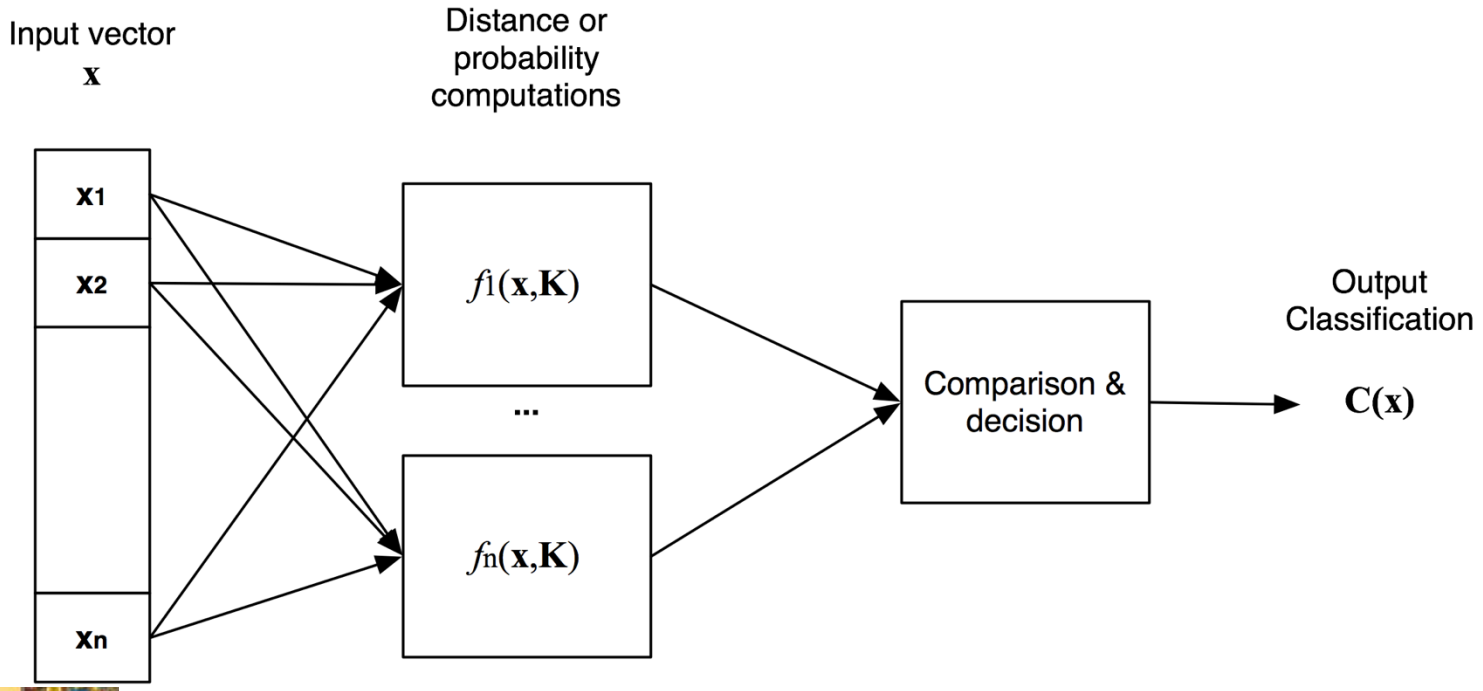
- Pattern classification techniques:
  - **statistical**
    - Each object or class can be represented as a *feature vector* and make decisions on which class to assign to a certain pattern based on distance calculations or probabilistic models.
  - structural (or syntactic).

Fundamentals

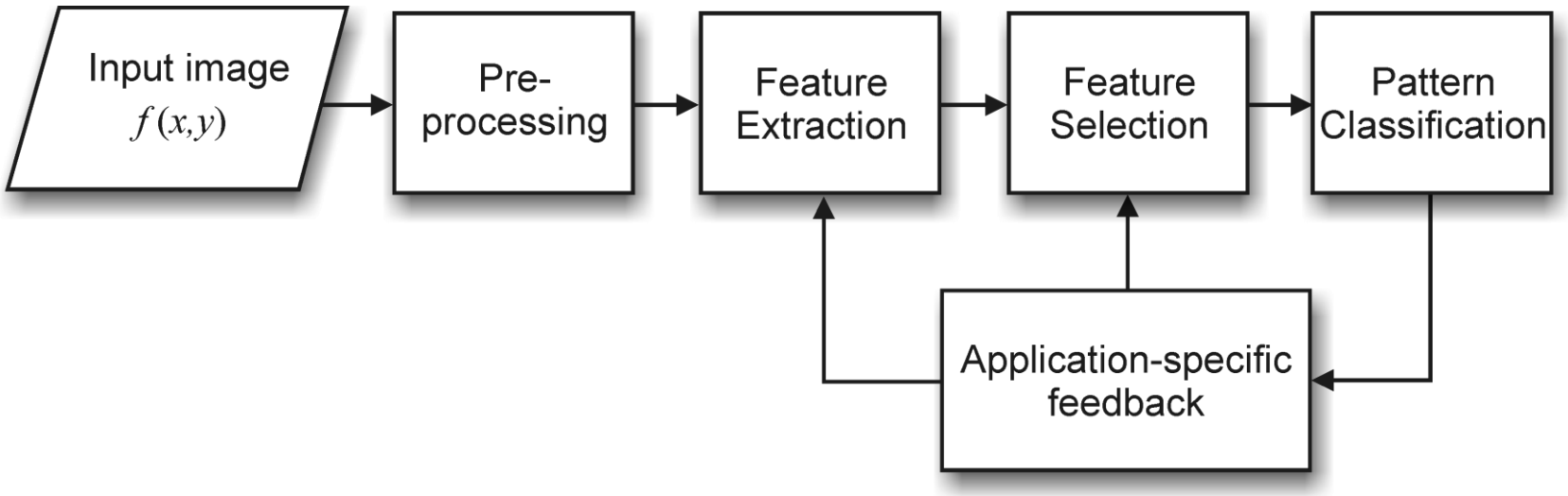
- The common goal of pattern classification techniques is to assign a *class* to an unknown *pattern* based on previously acquired *knowledge* about *objects* and the *classes* to which they belong.

Fundamentals

- A statistical pattern classifier processes numerical information from the feature vectors, computes a series of distances or probabilities, and uses those results to make decisions regarding which class label C(**x**) should be assigned to each input pattern **x**.



Design and implementation of a visual pattern classifier



Design and implementation of a visual pattern classifier

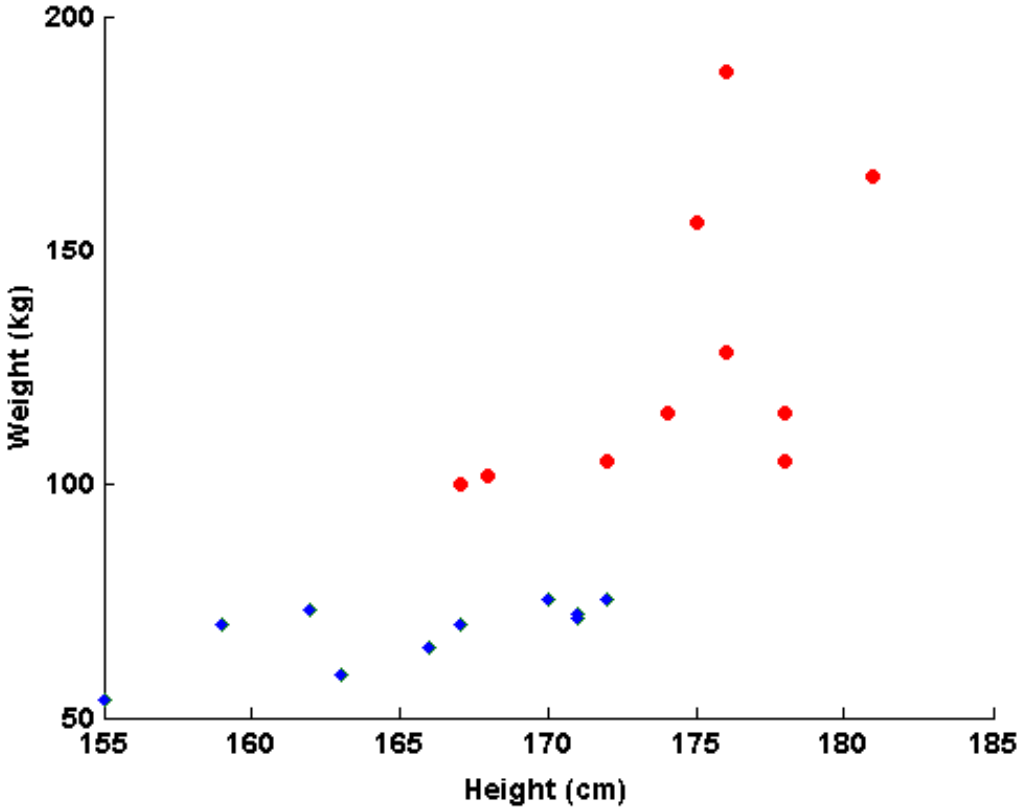
- Steps:
  1. Define the problem and determine the number of classes involved.
  2. Extract features that are most suitable to describe the images and allow the classifier to label them accordingly.
  3. Select a classification method or algorithm.
  4. Select a dataset.
  5. Select a subset of images and use them to train the classifier.
  6. Test the classifier.
  7. Refine and improve the solution.

Patterns and pattern classes

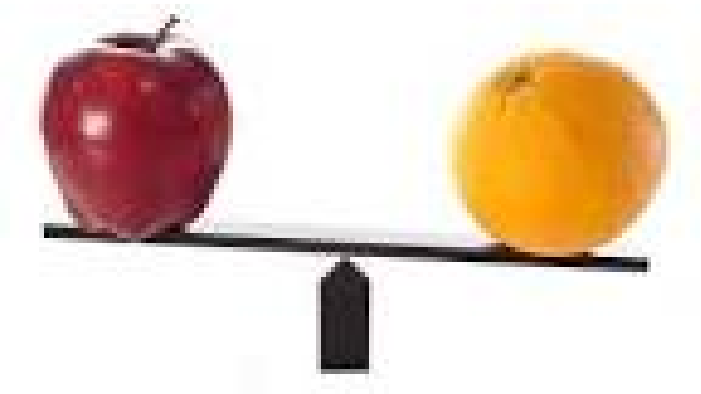
- A *pattern* can be defined as an arrangement of *descriptors* or *features*.
  - Patterns are usually encoded in the form of feature vectors, strings, or trees.
- A *class* is a set of patterns that share some common properties.
  - An ideal class is one in which its members are very similar to one another (i.e., the class has **high intra-class similarity**) and yet significantly different from members of other classes (i.e., **inter-class differences are significant**)

Patterns and pattern classes

- Sumo wrestlers and table tennis players



Apple and Orange Comparison



Patterns and pattern classes

- Data preprocessing
  - **Noise removal**: data samples that deviate too far from the average value for a class are removed, under the rationale that: (a) there may have been a mistake while measuring (or extracting) that particular sample; (b) the sample is a poor example of the underlying structure of the class.
  - **Normalization**: feature vectors may need to be normalized before distance, similarity, and probability calculations take place.
  - **Insertion of missing data**: (optional).

Training and test sets

- The process of development and testing of pattern classification algorithms usually requires that the dataset be divided in two subgroups:
  - **training set**: used for algorithm development and fine-tuning
  - **test set**: used to evaluate the algorithm's performance.
- The training set contains a small (typically 20% or less), representative subsample of the dataset, selected manually or automatically.
- The size of the training set and the method used to build it are often dependent on the selected pattern classification technique.
- The goal of having two separate sets is to avoid bias in reporting the success rates of the approach.

Confusion matrix

- A 2D array of size $K \times K$ (where K is the total number of classes) used to report raw results of classification experiments.
- The value in row *i*, column *j* indicates the number of times an object whose true class is *i* was labeled as belonging to class *j*.
- The main diagonal of the confusion matrix indicates the number of cases where the classifier was successful; a perfect classifier would show all off-diagonal elements equal to zero.

Confusion matrix

- Example :

Predict

True class

	ω_1	ω_2	ω_3	ω_4
ω_1	97	0	2	1
ω_2	0	89	10	1
ω_3	0	0	100	0
ω_4	0	3	5	92

- Example :
  - Overall error rate: 5.5 % (success rate = 94.5%)

Hit rates, false alarm rates, and ROC curves

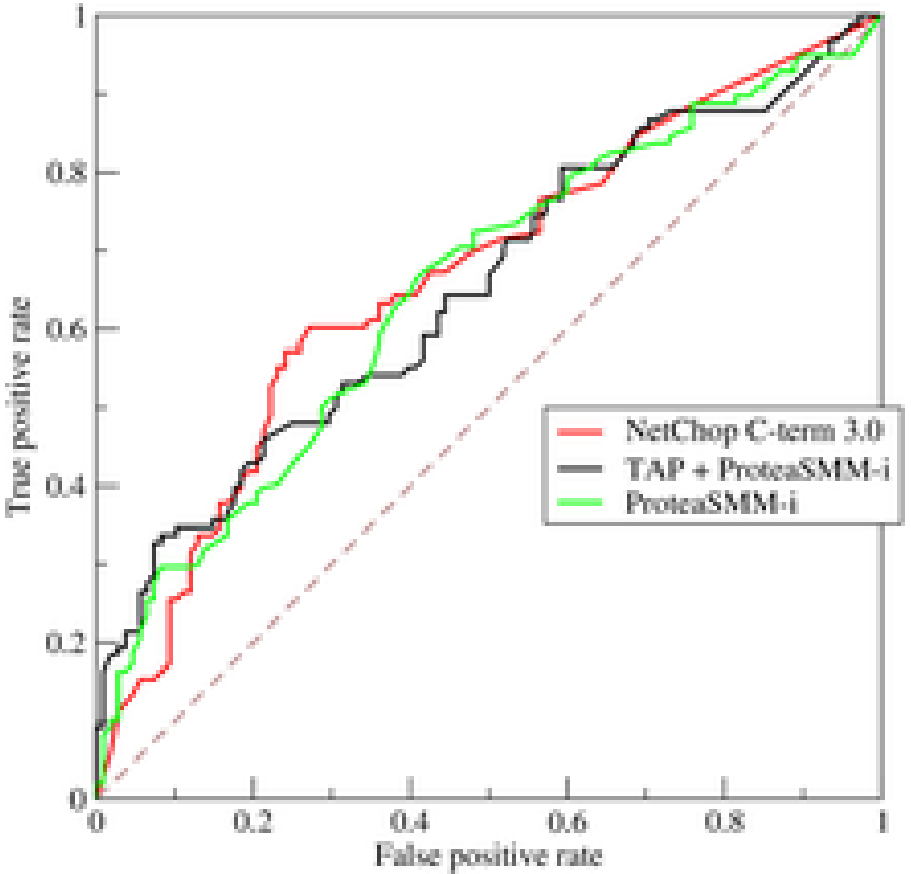
- A classical computer vision example: the *object detection* task
  - A CV algorithm is presented with an image and the question: "Is the object present in this image or not?"
    - If the algorithm successfully answers *yes* (and points to where in the image the object is located) when the object is present, it is called a *true positive*.
    - If the algorithm correctly answers *no* when the object is absent, it is called a *true negative*.
  - There are two possible errors the algorithm can make:
    - Answering *yes* in the absence of an object (a *false alarm* or *false positive*)
    - Answering *no* when the object is present, i.e., missing the object (a *false negative*).

Hit rates, false alarm rates, and ROC curves

- The cost of a false positive or a false negative is application-dependent.
- The *receiver operating characteristic* (ROC) curve is a plot that shows the relationship between the correct detection (true positive) rate (also known as *hit rate*) and the *false alarm* (false positive) *rate*.
- The ideal ROC curve is one in which the “knee” of the curve is as close to the top-left corner of the graph as possible, suggesting hit rate close to 100% with a false alarm rate close to zero.

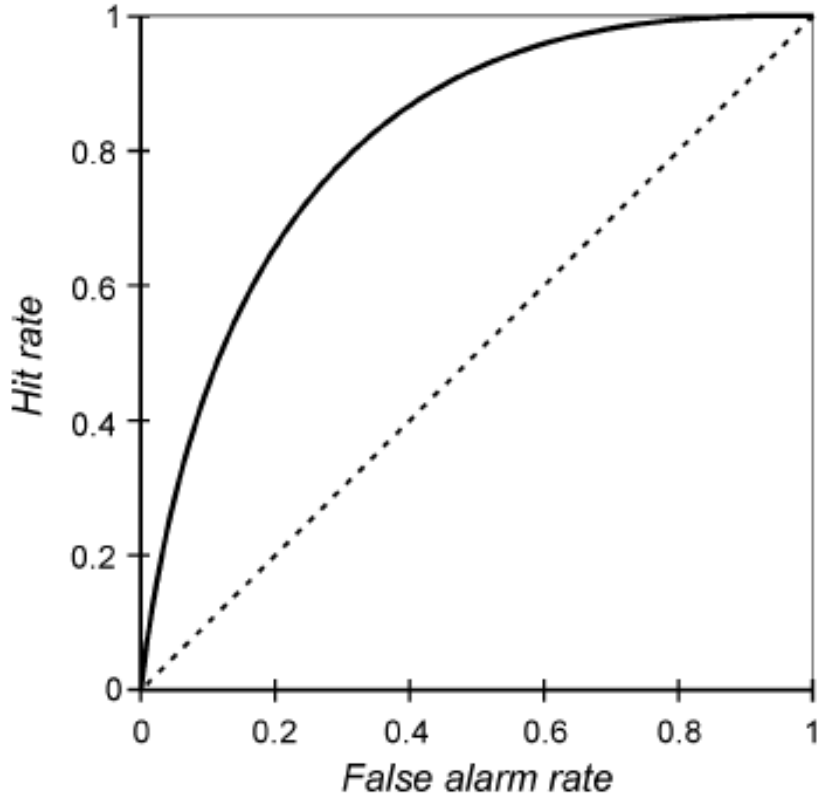
Receiver Operating Characteristics (ROC) Curve

- The curve is created by plotting the true positive rate against the false positive rate



Hit rates, false alarm rates, and ROC curves

- Example of ROC curve



Precision and recall

- Certain image processing applications, notably image retrieval, have as their goal to retrieve *relevant* images while not retrieving *irrelevant* ones.
- The measures of performance used in image retrieval borrow from the field of (document) information retrieval and are based on two primary figures of merit: *precision* and *recall*.
  - **Precision** is the number of relevant documents retrieved by the system divided by the total number of documents retrieved (i.e., true positives plus false alarms).
  - **Recall** is the number of relevant documents retrieved by the system divided by the total number of relevant documents in the database (which should, therefore, have been retrieved).

Precision and recall

- Example :

	ω_1	ω_2	ω_3	ω_4
ω_1	97	0	2	1
ω_2	0	89	10	1
ω_3	0	0	100	0
ω_4	0	3	5	92

$$P = \frac{tp}{tp + fp}$$

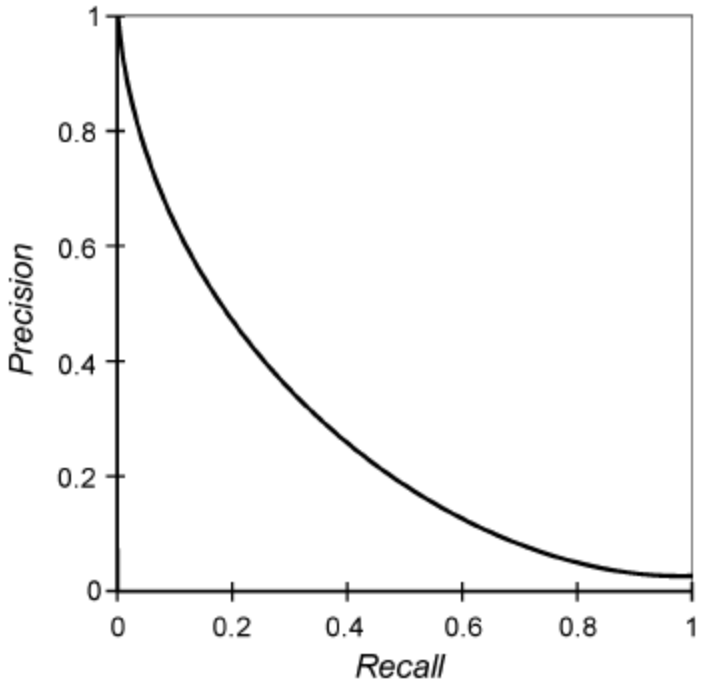
$$R = \frac{tp}{tp + fn}$$

$$P_1 = 97/(97 + 0 + 0 + 0) = 100\%$$
$$P_2 = 89/(0 + 89 + 0 + 3) = 96.74\%$$
$$P_3 = 100/(2 + 10 + 100 + 5) = 85.47\%$$
$$P_4 = 92/(1 + 1 + 0 + 92) = 97.87\%$$
$$R_1 = 97/(97 + 0 + 2 + 1) = 97\%$$
$$R_2 = 89/(0 + 89 + 10 + 1) = 89\%$$
$$R_3 = 100/(0 + 0 + 100 + 0) = 100\%$$
$$R_4 = 92/(0 + 3 + 5 + 92) = 92\%$$

Precision and recall

- Precision can be interpreted as a measure of *exactness*, whereas recall provides a measure of *completeness*.
  - A perfect precision score of 1.0 means that every retrieved document (or image, in our case) was relevant, but does not provide any insight as to whether all relevant documents were retrieved.
  - A perfect recall score of 1.0 means that all relevant images were retrieved, but says nothing about how many irrelevant images might have also been retrieved.

Precision and recall graph



- P-R graph
  - Obtained by calculating the precision at various recall levels.
  - The ideal P-R graph shows perfect precision values at every recall level until the point where all relevant documents (and only those) have been retrieved; from that point on it falls monotonically until the point where recall reaches one.

Precision and recall

- F1: a more compact representation of the precision and recall properties of a system.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

Distance and similarity measures (Feature Comparison)

- Two features can be compared with each other by calculating the similarity
  - Euclidean distance:

$$d_E = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

  - Manhattan (or city-block) distance:

$$d_M = \sum_{i=1}^{n} |a_i - b_i|$$

  - Minkowski distance:

$$d_M = \left[\sum_{i=1}^{n} |a_i - b_i|^r \right]^{1/r}$$

Distance and similarity measures

- Similarity measures
  - Vector inner product:

$$\sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$$

  - Tanimoto metric:

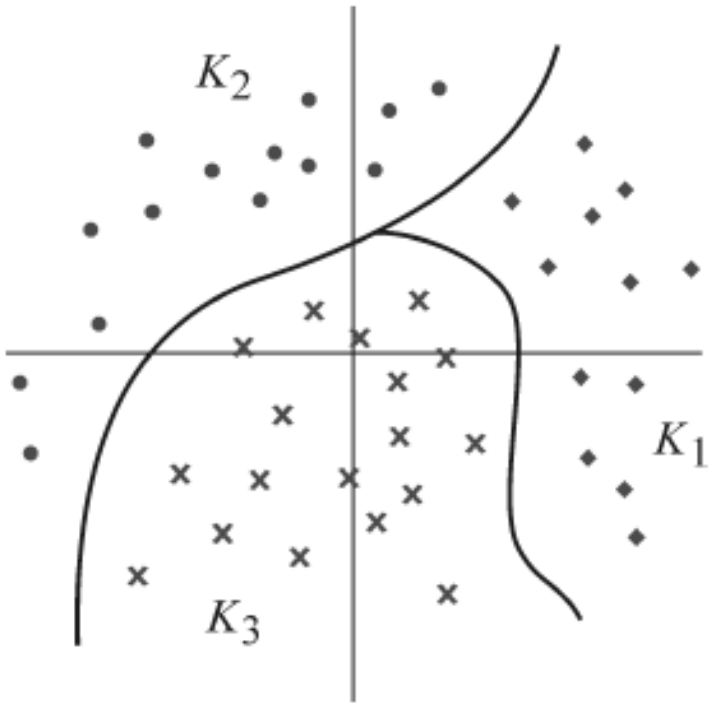
$$\frac{\sum_{i=1}^{n} a_i b_i}{\sum_{i=1}^{n} a_i^2 + \sum_{i=1}^{n} b_i^2 - \sum_{i=1}^{n} a_i b_i}$$

Statistical pattern classification techniques

- So far, objects' properties are represented using *feature vectors* which are projected onto a *feature space*.
- The resulting points in the *n*-dimensional feature space should ideally be distributed in a way that correlates proximity in the feature space with similarity among the actual objects.
  - In other words, feature vectors associated with objects from the same class will appear close together as *clusters* in the feature space.

Statistical pattern classification techniques

- The job of a statistical pattern classification technique is to find a discrimination curve (or a *hyper-surface*, in the case of an *n*-dimensional feature space) that can tell the clusters (and, therefore, classes) apart.

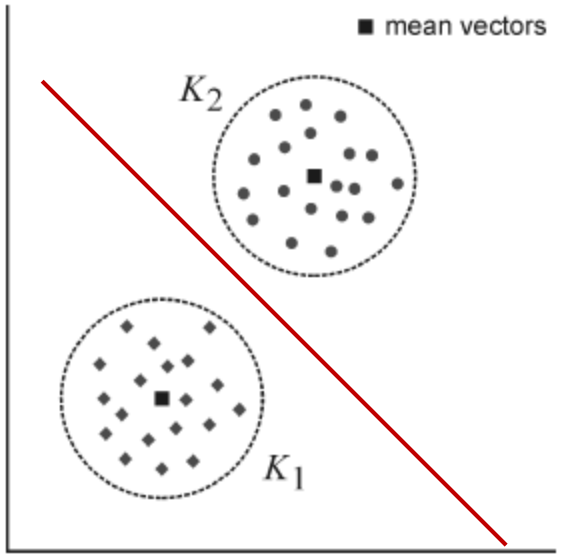


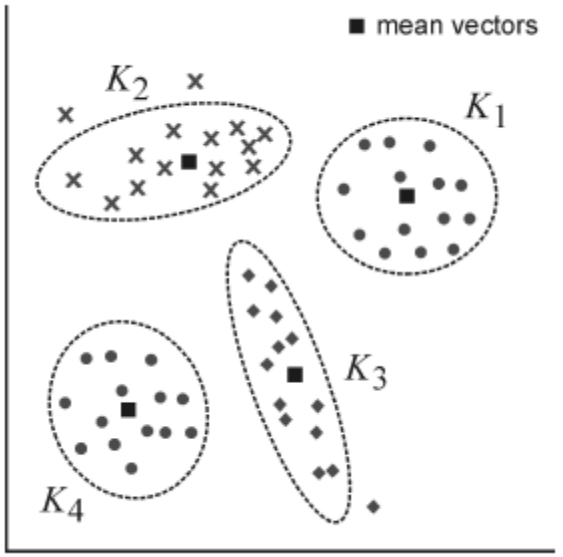
Statistical pattern classification techniques

- Three main techniques:
  - Minimum distance classifier
  - k-nearest neighbors
  - SVM classifier

Statistical pattern classification techniques

- Minimum distance classifier



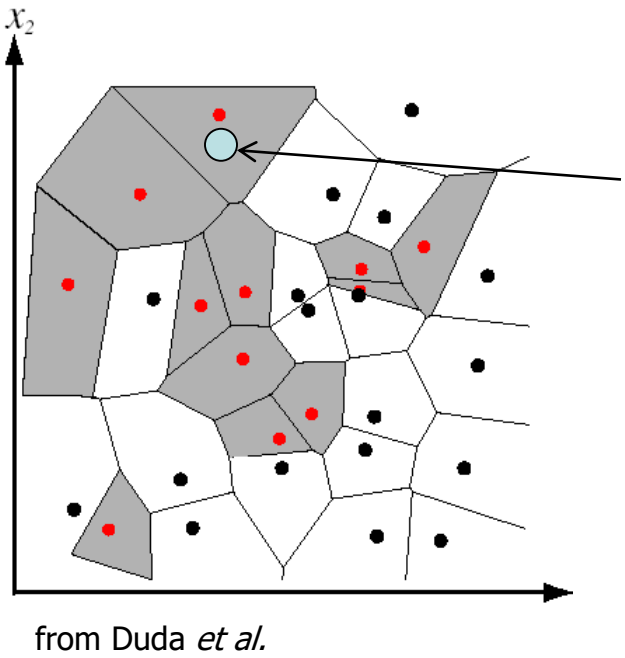


$$d_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_j\| \qquad \mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x}_j$$

Nearest Neighbor classification

- Assign label of nearest training data point to each test data point

Black = negative
Red = positive



from Duda *et al.*

Novel test example

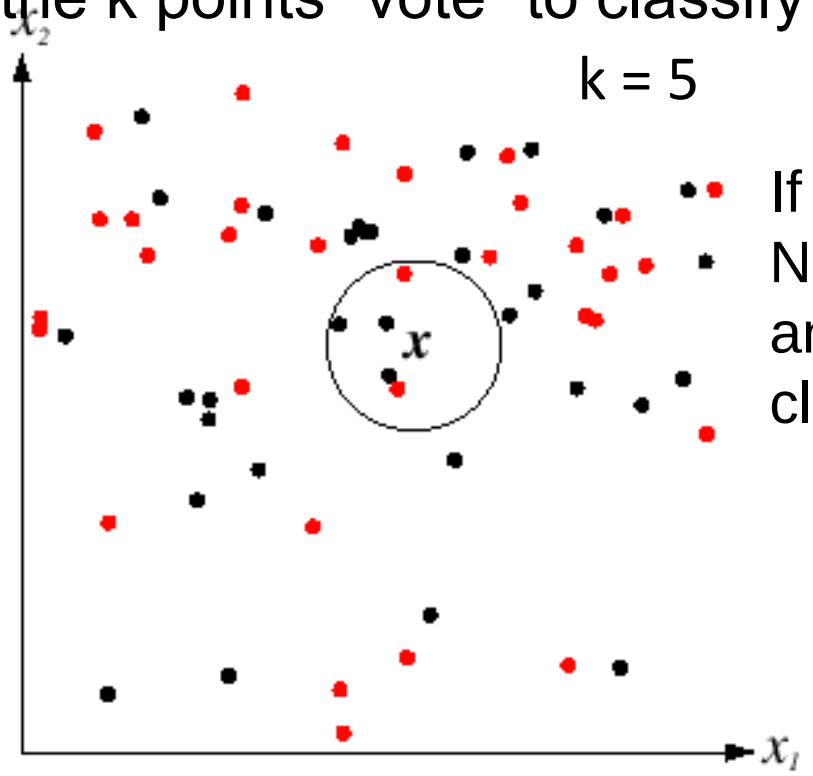
Closest to a positive example from the training set, so classify it as positive.

Voronoi partitioning of feature space
for 2-category 2D data

K-Nearest Neighbors classification

- For a new point, find the k closest points from training data
- Labels of the k points “vote” to classify

Black = negative
Red = positive



If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

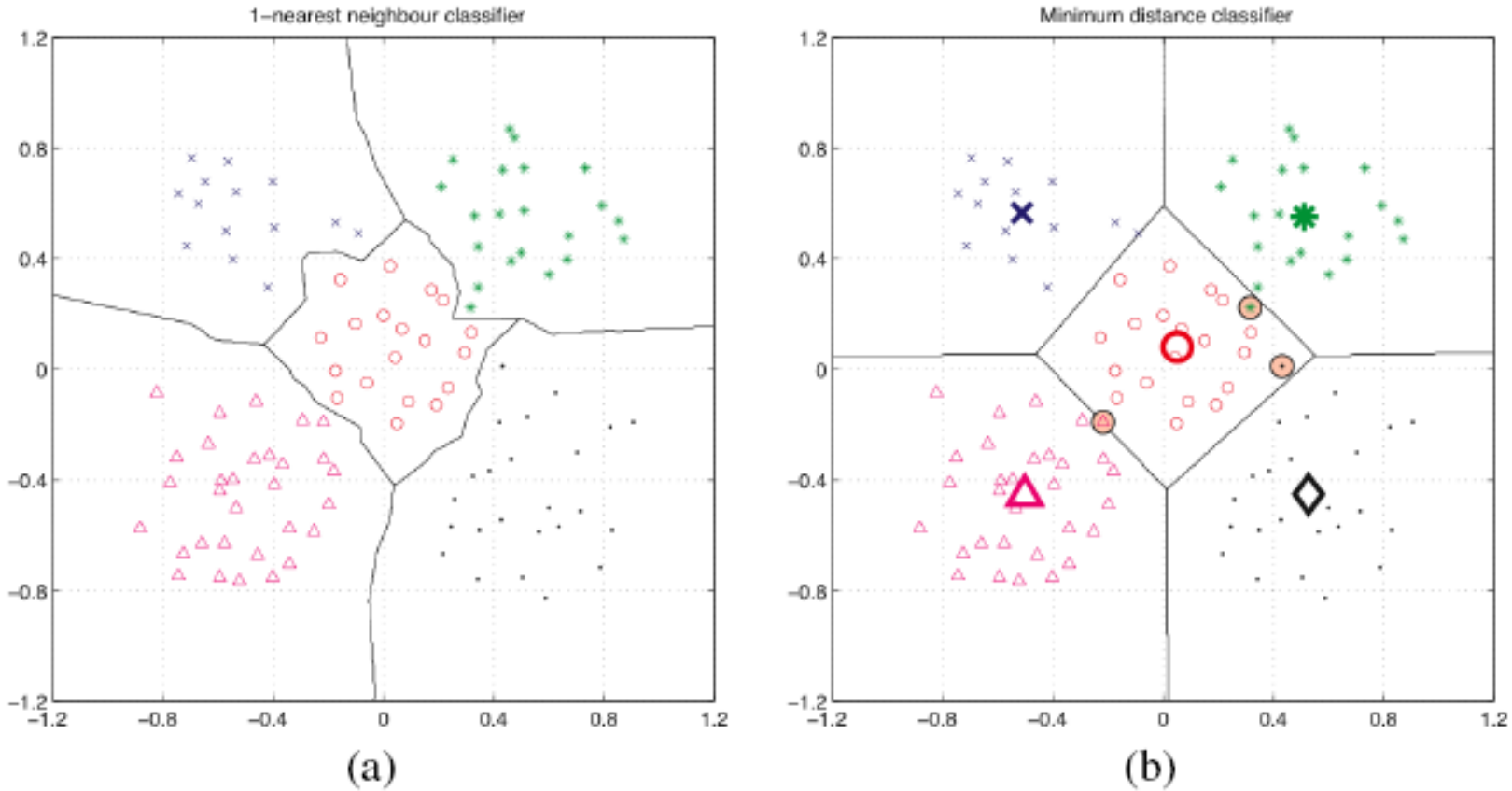
Source: D. Lowe

Statistical pattern classification techniques

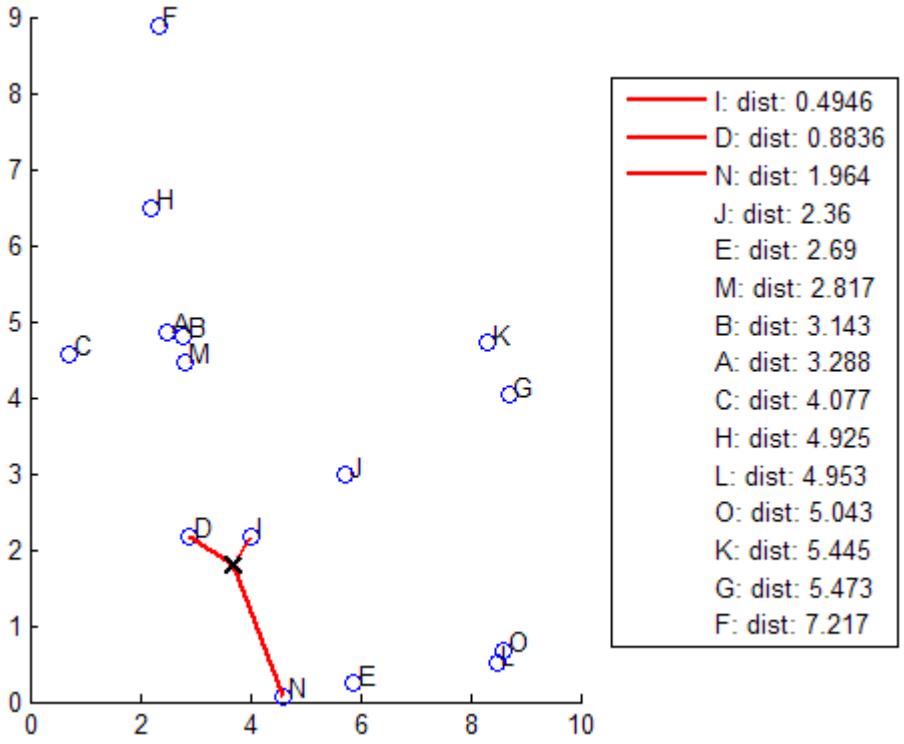
- k-nearest neighbors classifier
  - Works by computing the distance between an unknown pattern's feature vector and the *k* closest *points* (*not* classes) to it in the feature space.
  - It then assigns the unknown pattern to the class to which the majority of the *k* sampled points belong.
  - Advantages: simplicity (e.g., no assumptions need to be made about the probability distributions of each class) and versatility (e.g., it handles overlapping classes or classes with complex structure well).
  - Disadvantages: computational cost.

Statistical pattern classification techniques

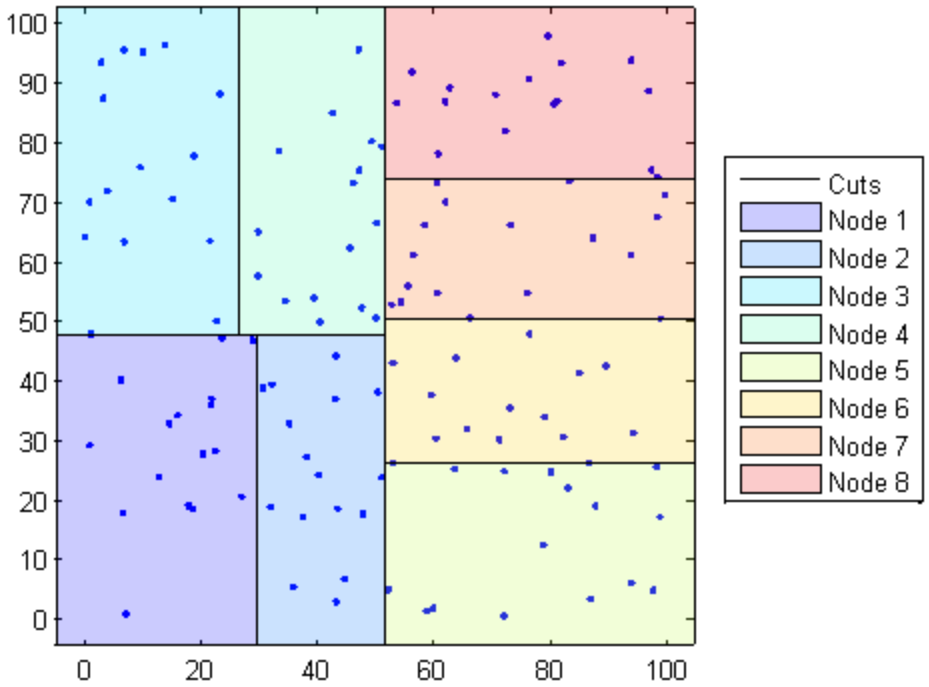
- k-nearest neighbors classifier: example



KNN Search


F
H
AB
C
M
K
G
J
D
I
N
E
O
R
I: dist: 0.4946
D: dist: 0.8836
N: dist: 1.964
J: dist: 2.36
E: dist: 2.69
M: dist: 2.817
B: dist: 3.143
A: dist: 3.288
C: dist: 4.077
H: dist: 4.925
L: dist: 4.953
O: dist: 5.043
K: dist: 5.445
G: dist: 5.473
F: dist: 7.217

KD Tree


100
90
80
70
60
50
40
30
20
10
0
0
20
40
60
80
100
Cuts
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
Node 7
Node 8

KNN Search Example

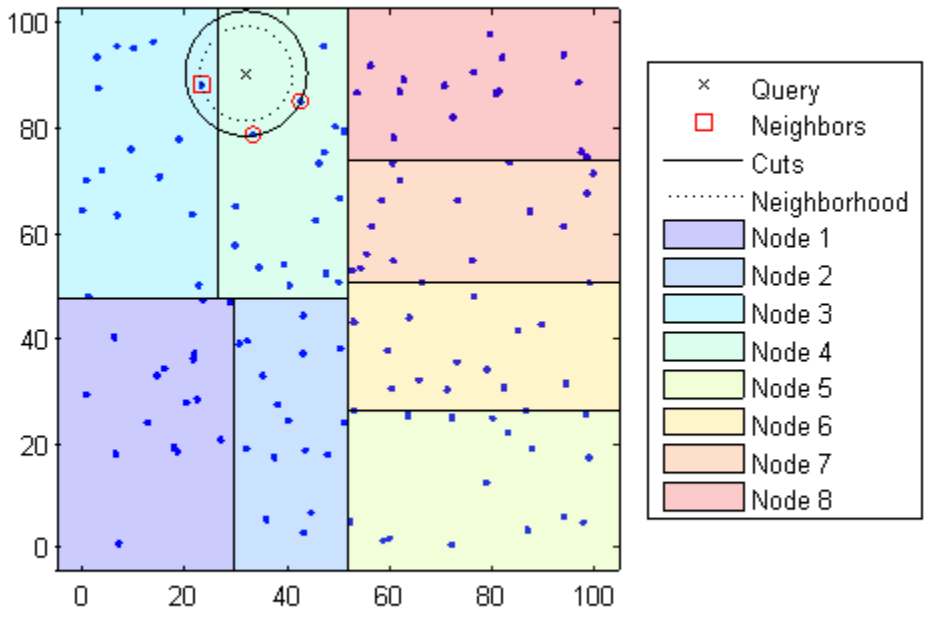
1 Determines the node to which the query point belongs. In the following example, the query point (32,90) belongs to Node 4.

2 Finds the closest k points within that node and its distance to the query point. In the following example, the points in red circles are equidistant from the query point, and are the closest points to the query point within Node 4.

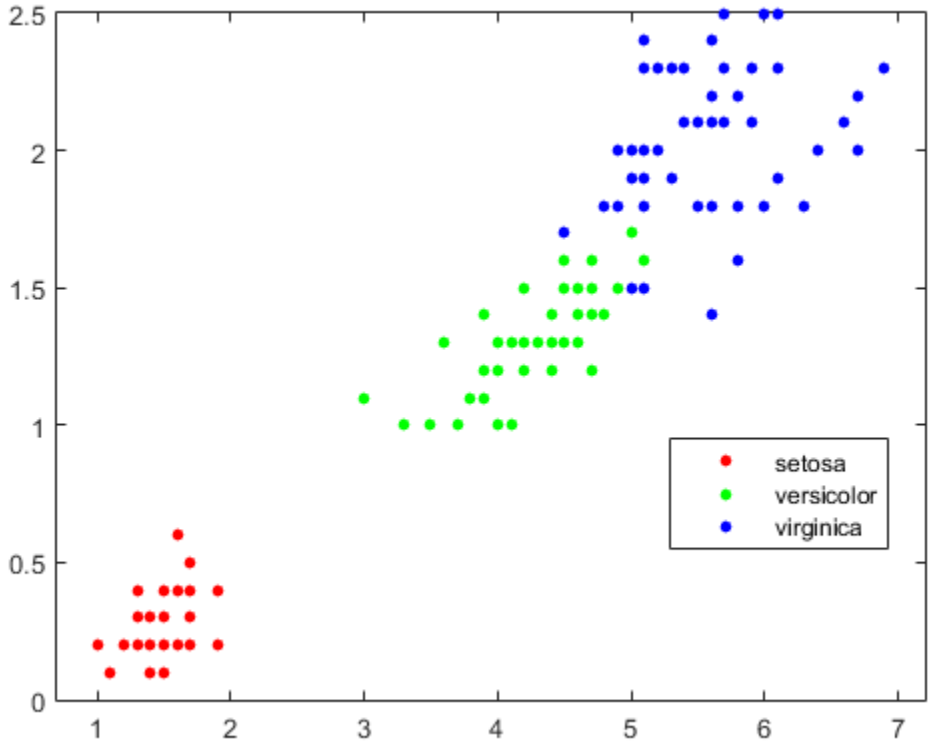
3 Chooses all other nodes having any area that is within the same distance, in any direction, from the query point to the kth closest point. In this example, only Node 3 overlaps the solid black circle centered at the query point with radius equal to the distance to the closest points within Node 4.

4 Searches nodes within that range for any points closer to the query point. In the following example, the point in a red square is slightly closer to the query point than those within Node 4.

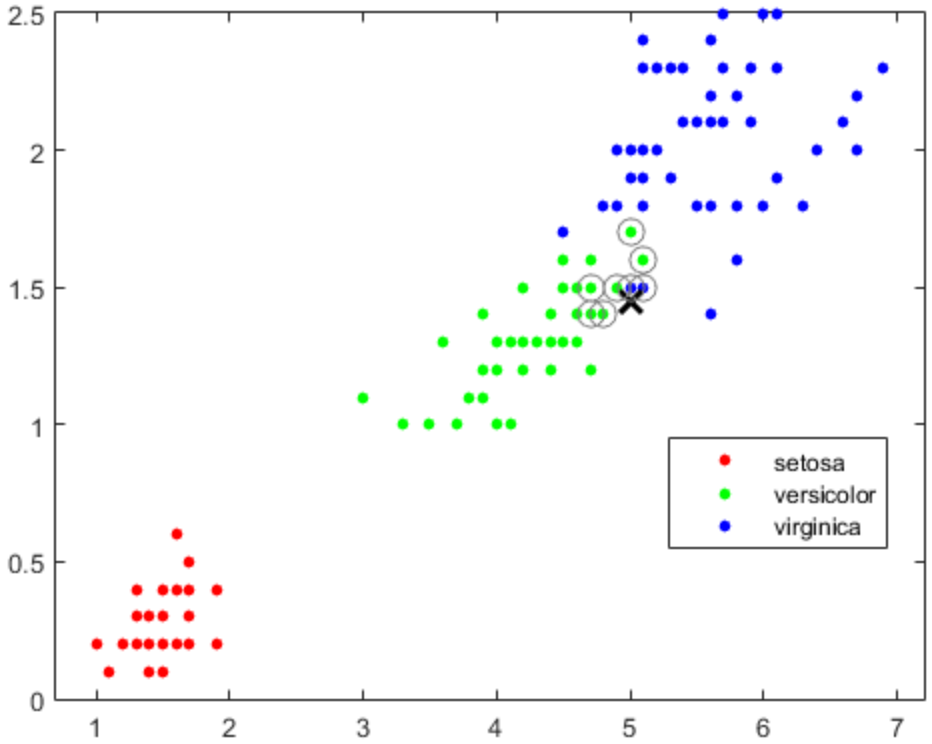
KD Search


100
80
60
40
20
0
0
20
40
60
80
100
Query
Neighbors
Cuts
Neighborhood
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
Node 7
Node 8

Graph



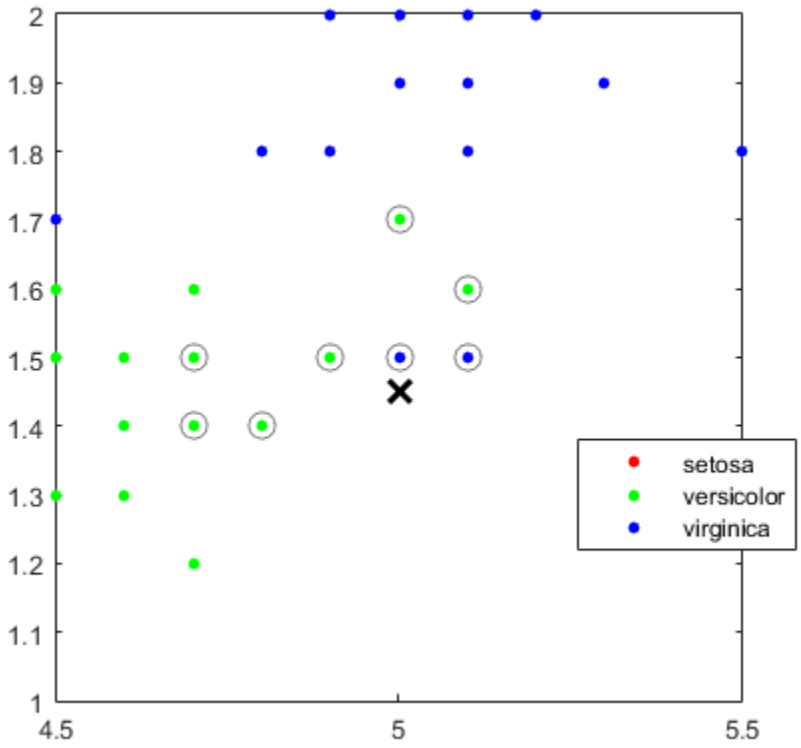
Results



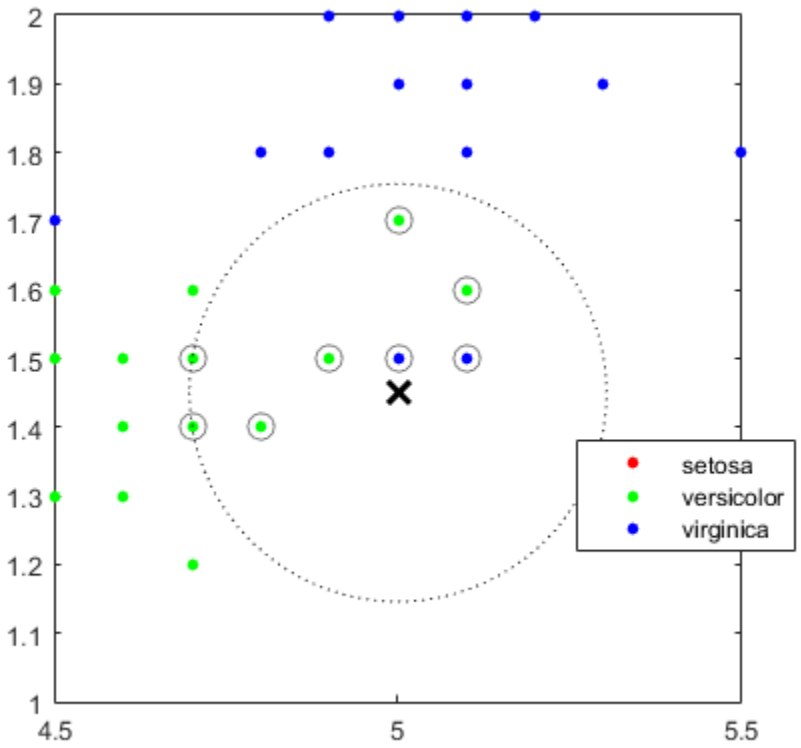
Search Results

```
x(n,:)
ans =

    5.0000    1.5000
    4.9000    1.5000
    4.9000    1.5000
    5.1000    1.5000
    5.1000    1.6000
    4.8000    1.4000
    5.0000    1.7000
    4.7000    1.4000
    4.7000    1.4000
    4.7000    1.5000

% Zoom results
xlim([4.5 5.5]);
ylim([1 2]);
axis square
```

Zoomed Result



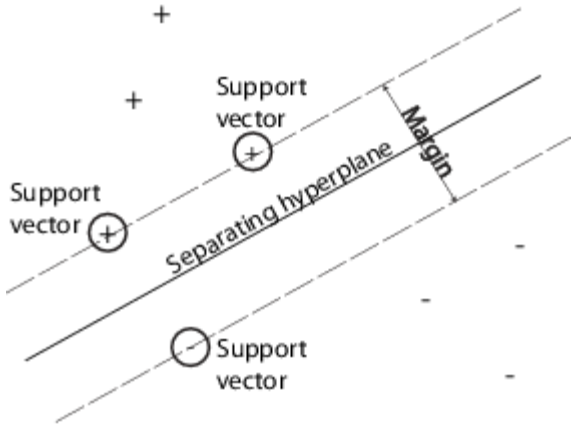
Output


2
1.9
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1
4.5
5
5.5
setosa
versicolor
virginica

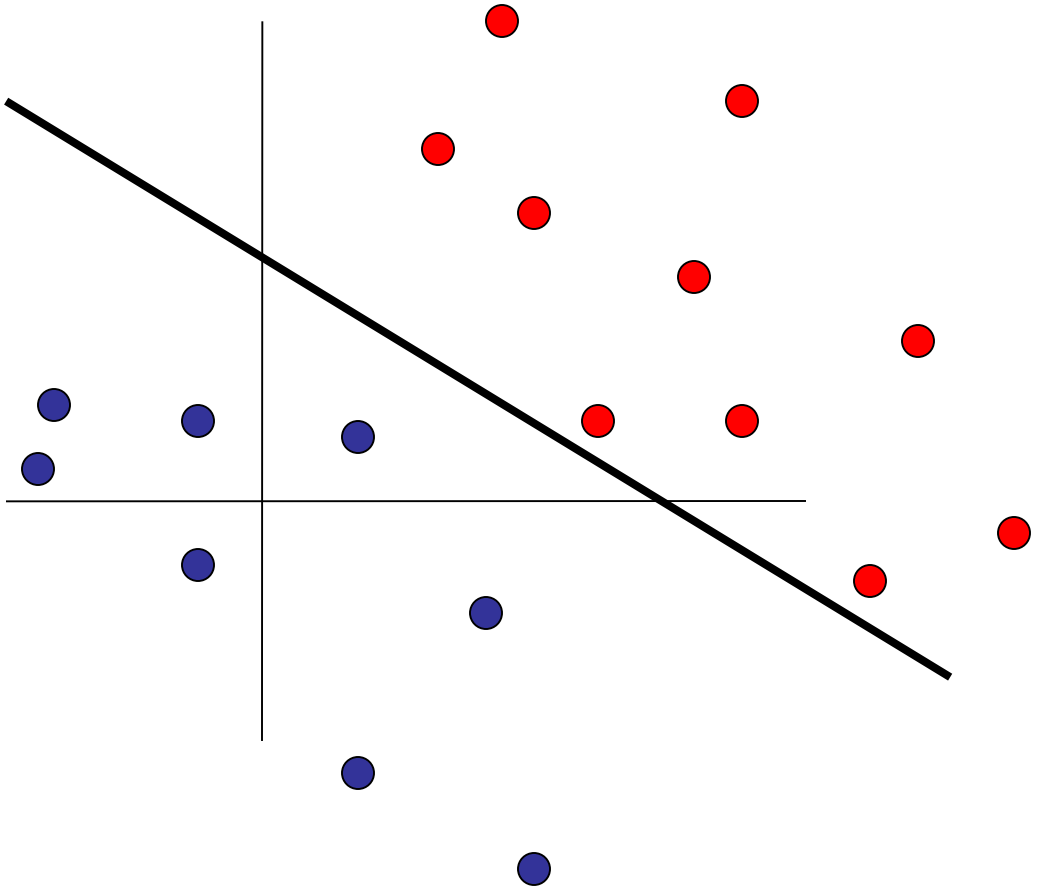
Support Vector Machine (SVM)

- An SVM classifies data by finding the best hyperplane that separates all data points of one class from those of the other class

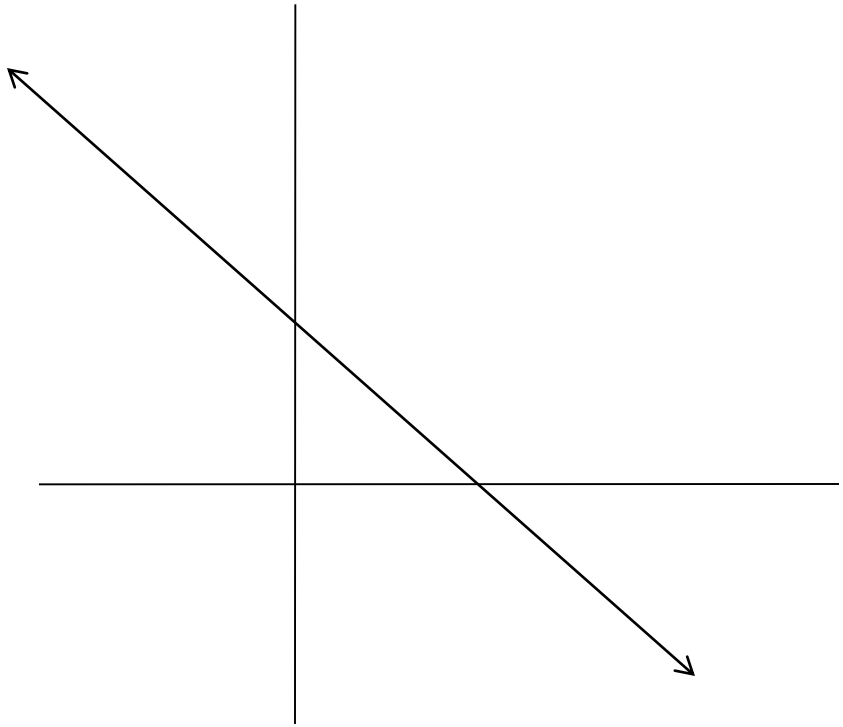
- The best hyperplane means the one with largest margin between two classes



Linear classifiers



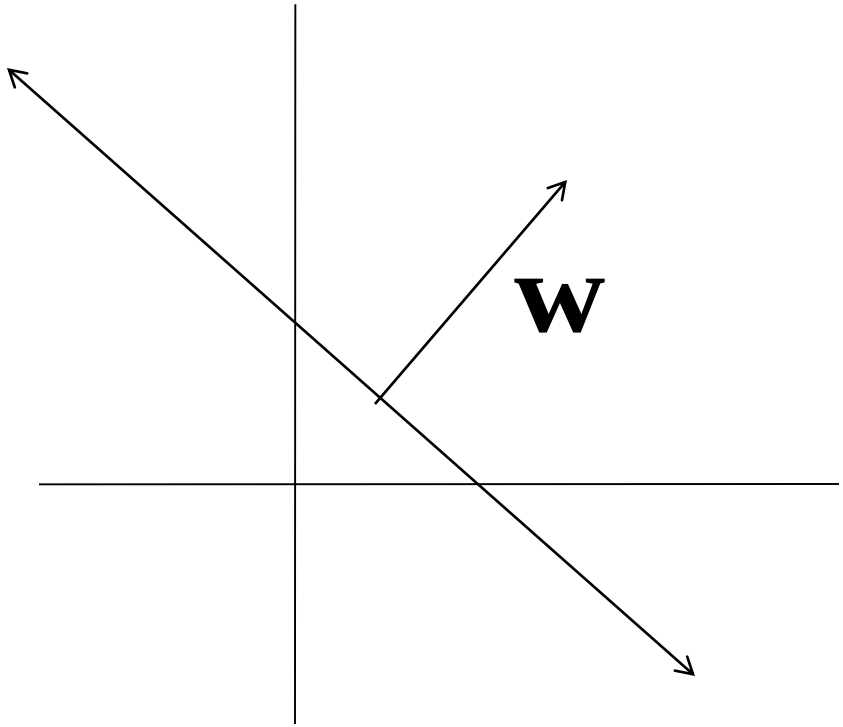
Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + c = 0$$

Lines in R^2



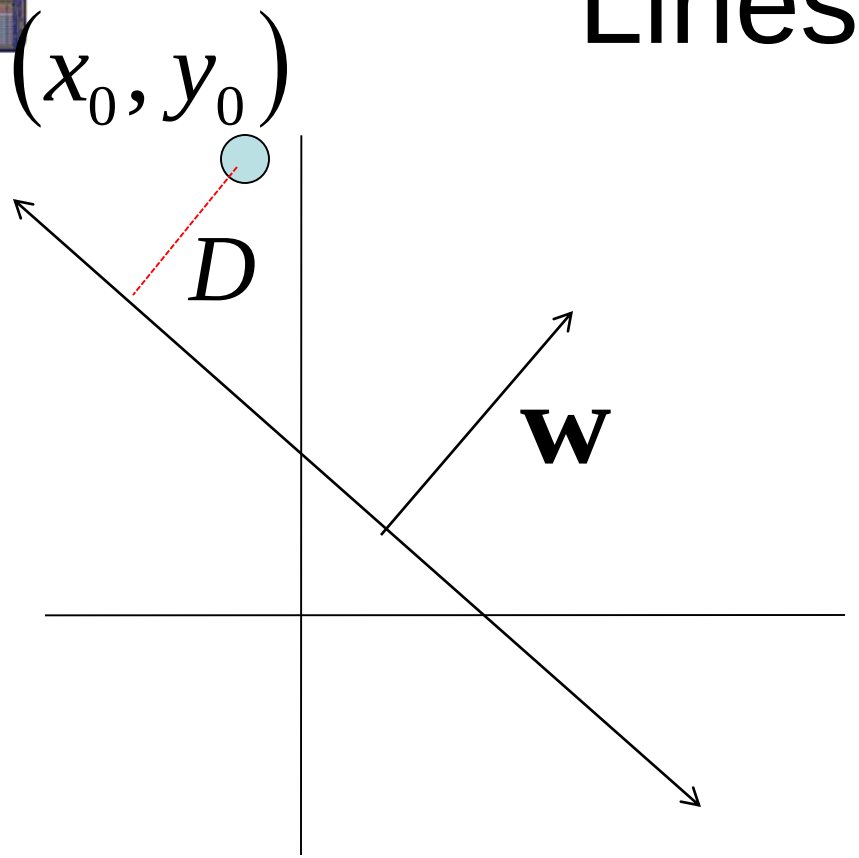
Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + c = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + c = 0$$

Lines in R^2



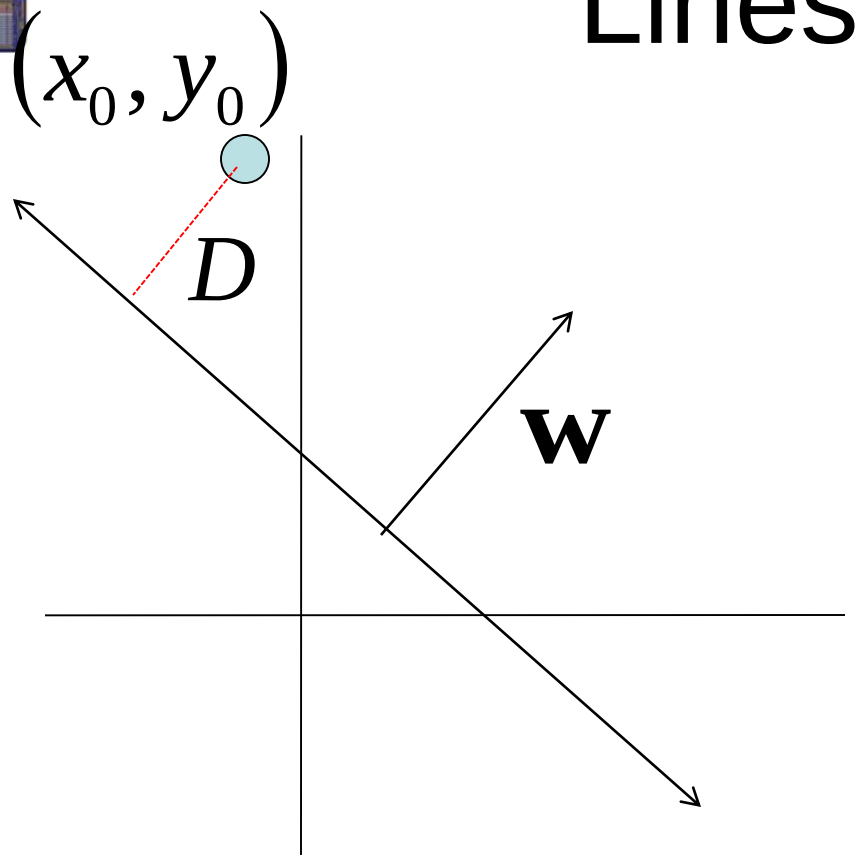
Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + c = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + c = 0$$

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + c = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + c = 0$$

$$D = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

distance from point to line

Proof

- Let (m, n) be a point in the line

  $(y_o-n)/(x_o-m) = b/a$

Hence, $a(y_o-n)-b(x_o-m) = 0$ by squaring we get:

$a^2 (y_o-n)^2 + b^2(x_o-m)^2 = 2ab (y_o-n) (x_o-m)$

Also, from

$(a(x_o-m) + b(y_o-n))^2 = a^2(x_o-m)^2 + 2ab (y_o-n) (x_o-m) + b^2(y_o-n)^2$

Proof

$(a(x_o-m) + b(y_o-n))^2 = (a^2 + b^2) ((x_o-m)^2 + (y_o-n)^2)$

Also:

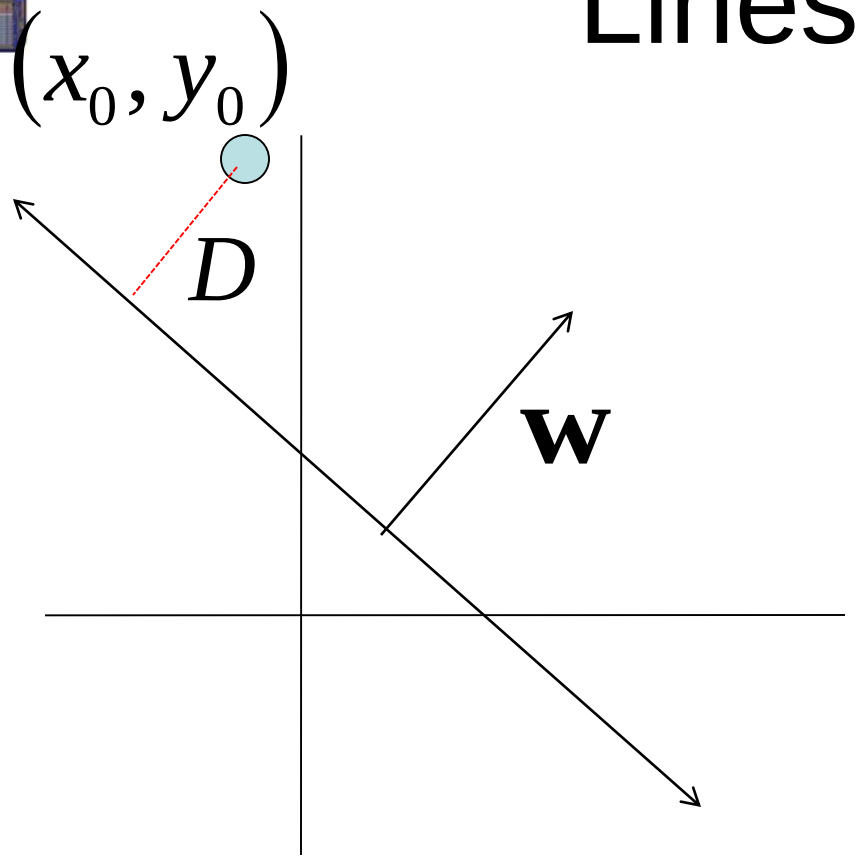
$(a(x_o-m) + b(y_o-n))^2 = (ax_o + by_o - am - bn)^2$

$= (ax_o + by_o + c)^2$

Since $am + bn + c = 0$, thus.

$(a^2 + b^2) ((x_o-m)^2 + (y_o-n)^2) = (ax_o + by_o + c)^2$

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + c = 0$$

$$\updownarrow$$

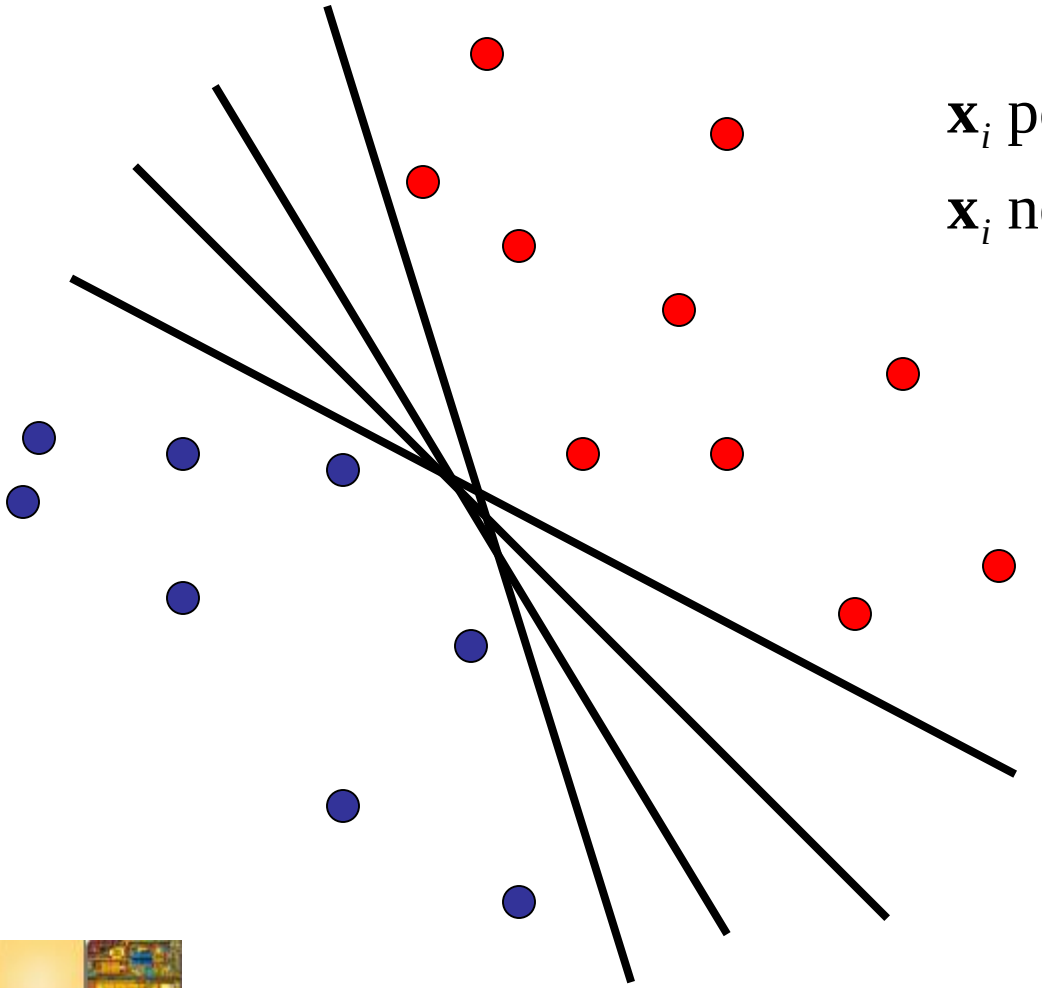
$$\mathbf{w} \cdot \mathbf{x} + c = 0$$

$$D = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} = \frac{\mathbf{w}^T\mathbf{x} + c}{\|\mathbf{w}\|}$$

distance from point to line

Linear classifiers

- Find linear function to separate positive and negative examples

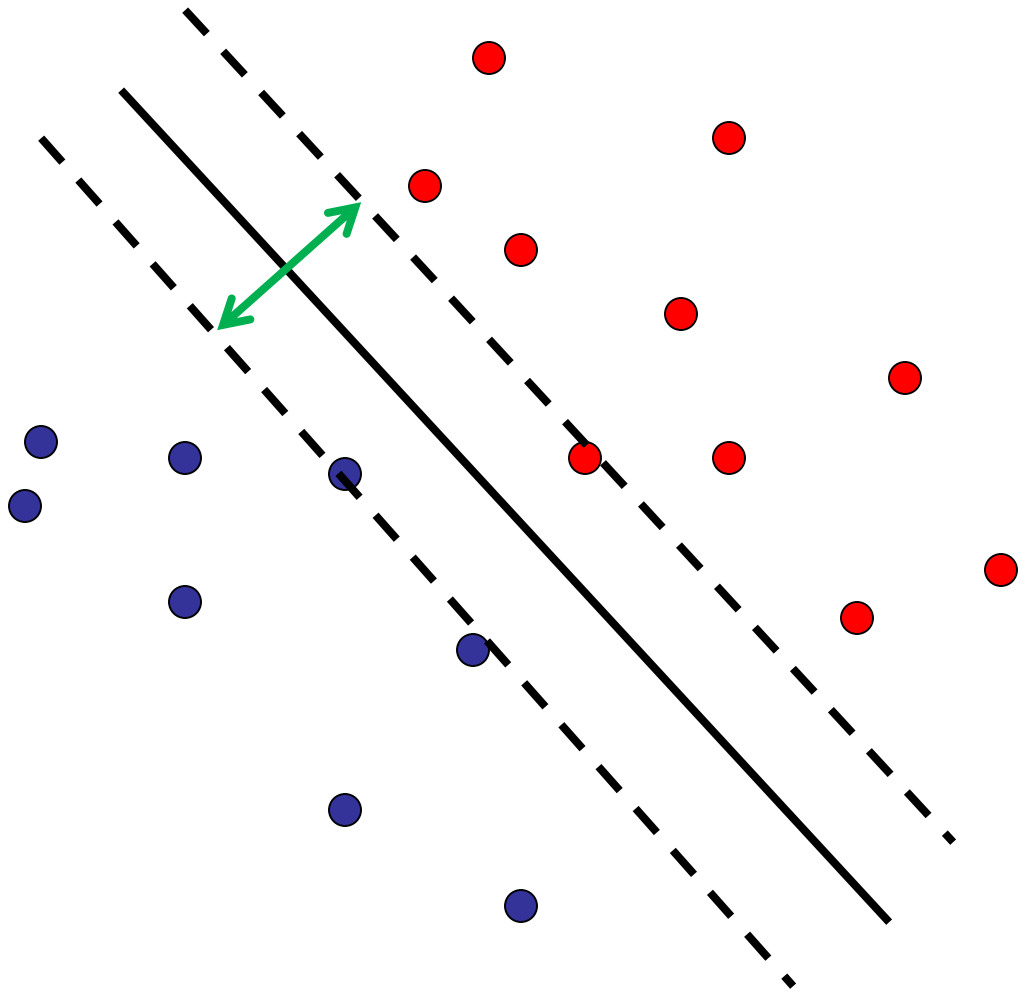


$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + c \geq 0$$

$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + c < 0$$

Which line is best?

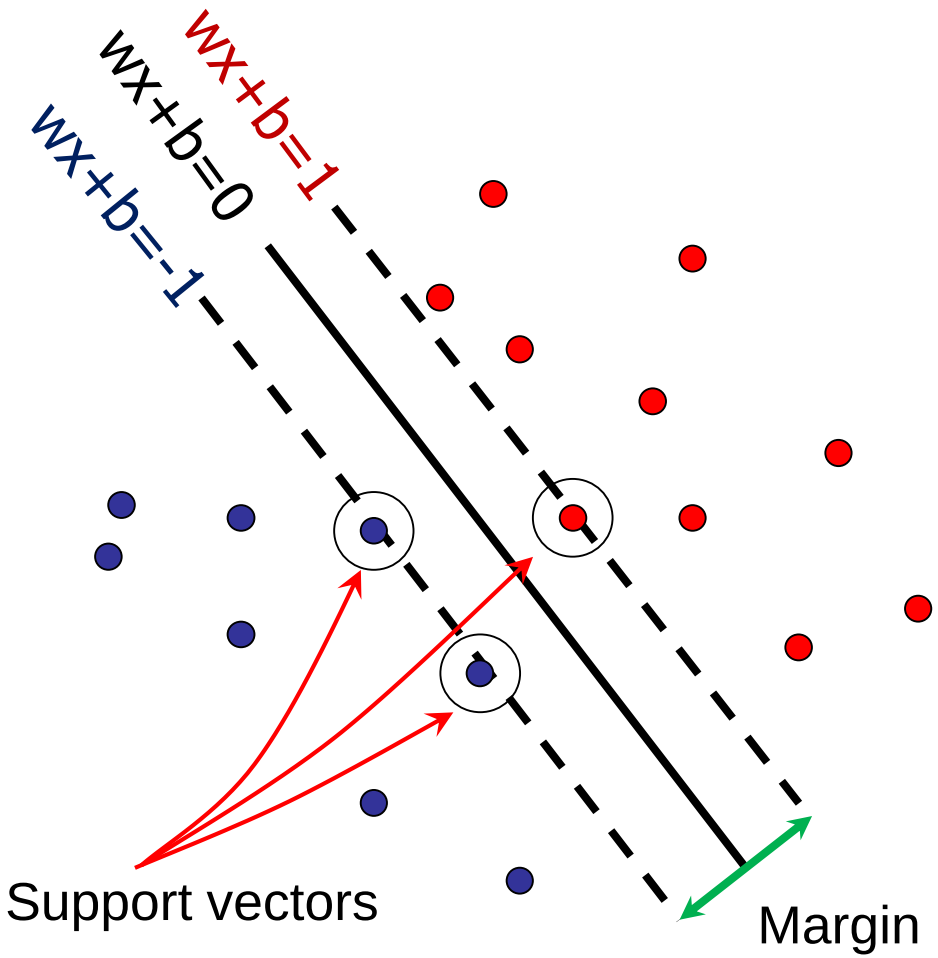
Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating line (for 2d case)*
- Maximize the *margin* between the positive and negative training examples

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + c \geq 1$

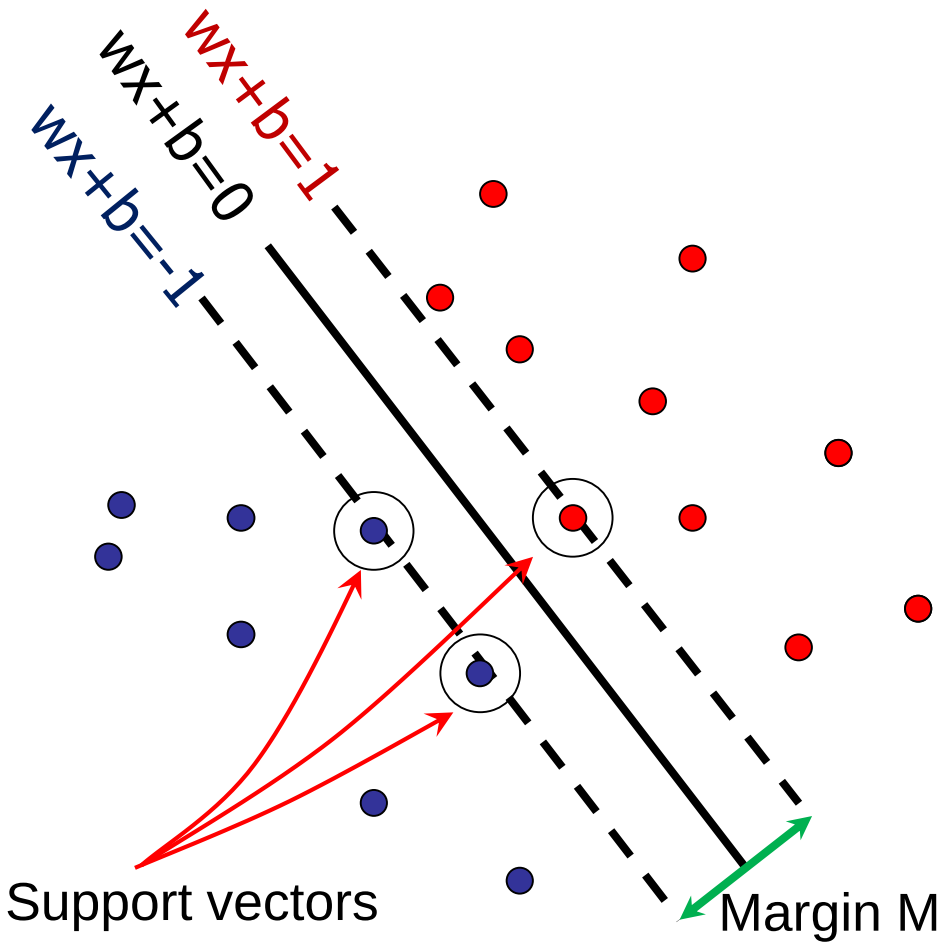
$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + c \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + c = \pm 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + c \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + c \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + c = \pm 1$

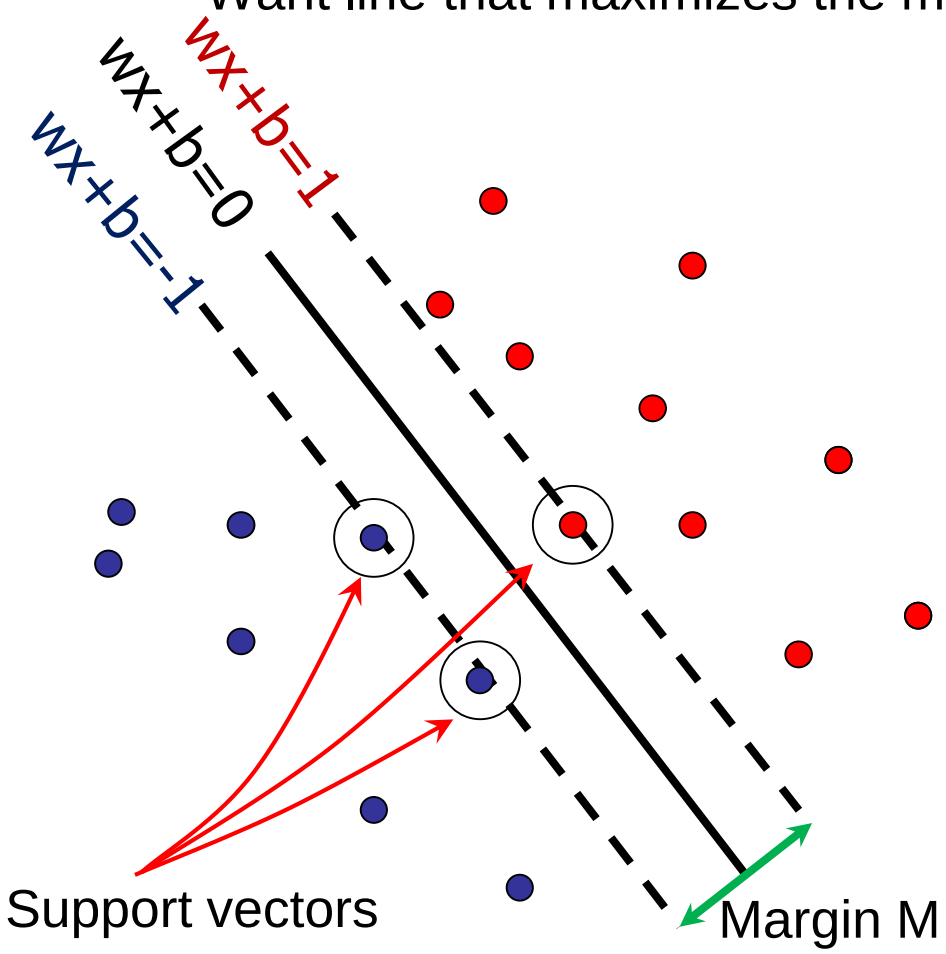
Distance between point and line: $\dfrac{|\mathbf{x}_i \cdot \mathbf{w} + c|}{\|\mathbf{w}\|}$

For support vectors:

$$\frac{\mathbf{w}^T\mathbf{x} + c}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + c \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + c \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + c = \pm 1$

Distance between point and line: $\dfrac{|\mathbf{x}_i \cdot \mathbf{w} + c|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

Finding the maximum margin line

1. Maximize margin 2/||w||
2. Correctly classify all training data points:

$$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

3. Quadratic optimization problem:

4. Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

   Subject to $\mathbf{y}_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery,

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

learned weight

Support vector

- The weights α_i are non-zero only at support vectors.

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

$b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)

$$\mathbf{w} \cdot \mathbf{x} + c = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + c$$

- Classification function:

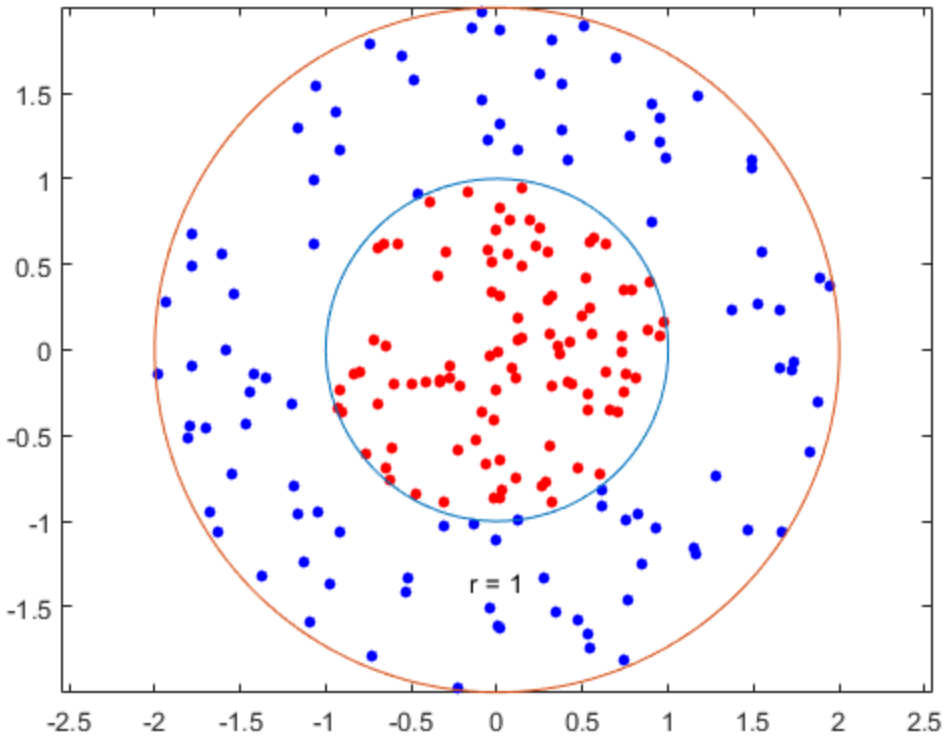
$$f(x) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + \text{c})$$

$$= \text{sign}\left(\sum_{\text{i}} \alpha_{\text{i}} \mathbf{x}_i \cdot \mathbf{x} + c\right)$$

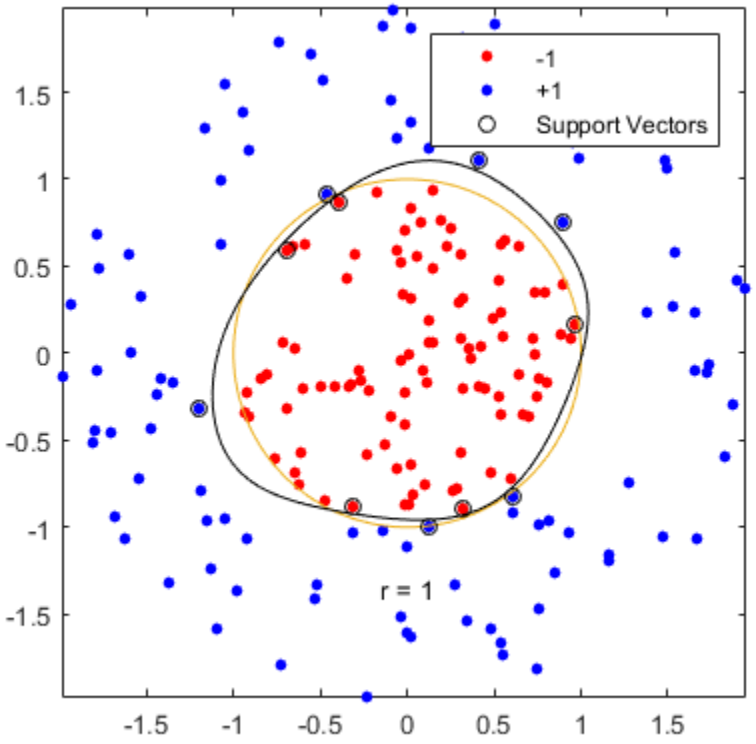
If f(x) < 0, classify as negative,

if f(x) > 0, classify as positive

Data Set



Support Vectors



Questions?