UNIVERSITY of HOUSTON *Downtown*

CS 4328 Parallel Computing

Instructor: Pablo Guillen-Rondon, Ph.D.
Adjunct Faculty

MPI

Applications: Parallel Machine Learning

Part 3

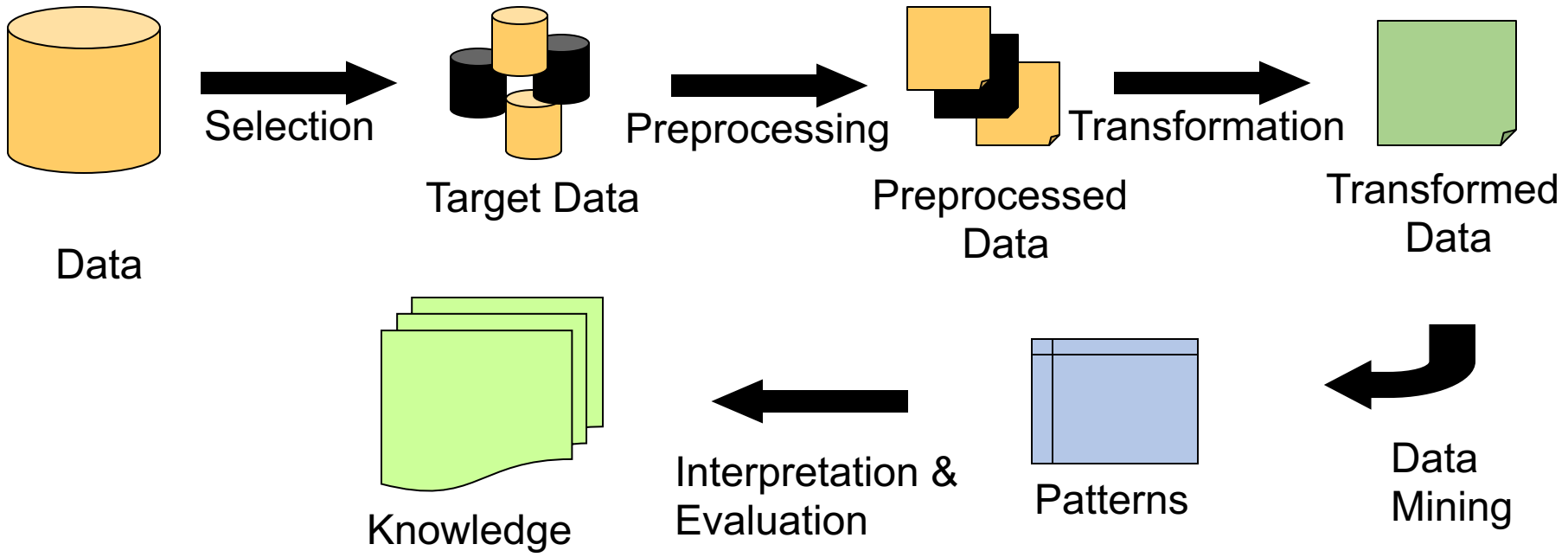
INTRODUCTION

In order to better model complex world real-data from different fields, including medical sciences, one approach is to develop:

- Pattern recognition techniques
- Robust features
- Data Mining
- Machine learning
- Deep learning methods
- Predictive analytics

Introduction

Knowledge Discovery and Data Mining



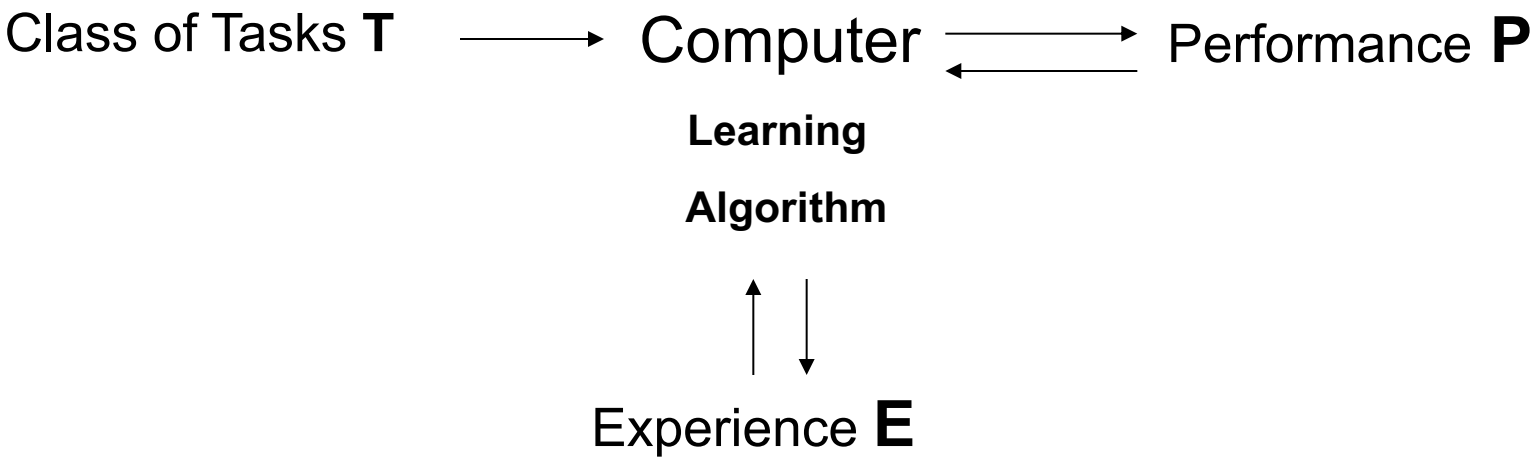
INTRODUCTION

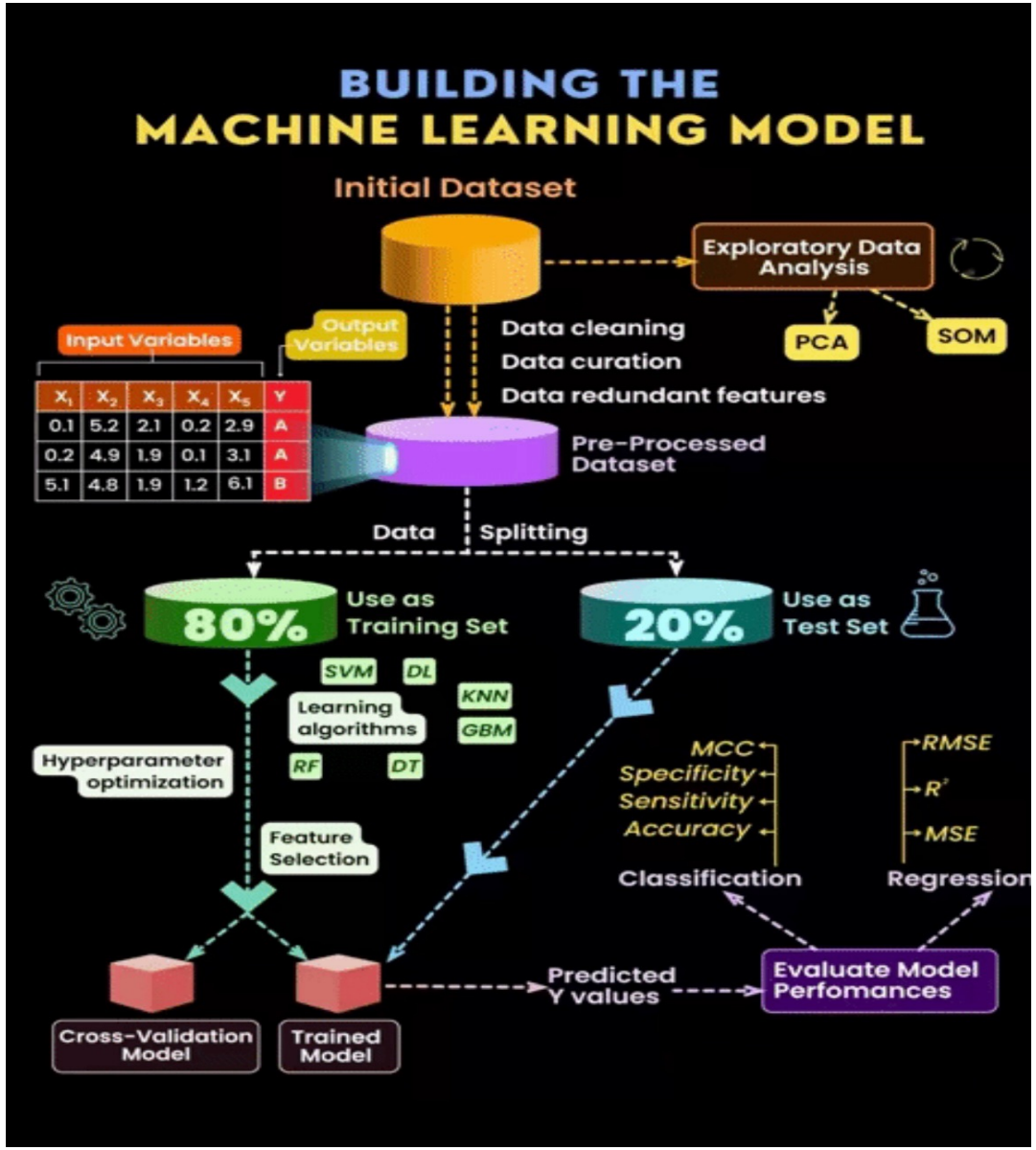
Machine learning

Machine learning is the scientific discipline that focuses on how computers learn from data. It arises at the intersection of statistics, which seeks to learn relationships from data, and computer science, with its emphasis on efficient computing algorithms.

Introduction

Machine learning is the study of how to make computers learn or adapt; the goal is to make computers improve their performance through experience.





X_1	X_2	X_3	X_4	X_5	Y
0.1	5.2	2.1	0.2	2.9	A
0.2	4.9	1.9	0.1	3.1	A
5.1	4.8	1.9	1.2	6.1	B

INTRODUCTION

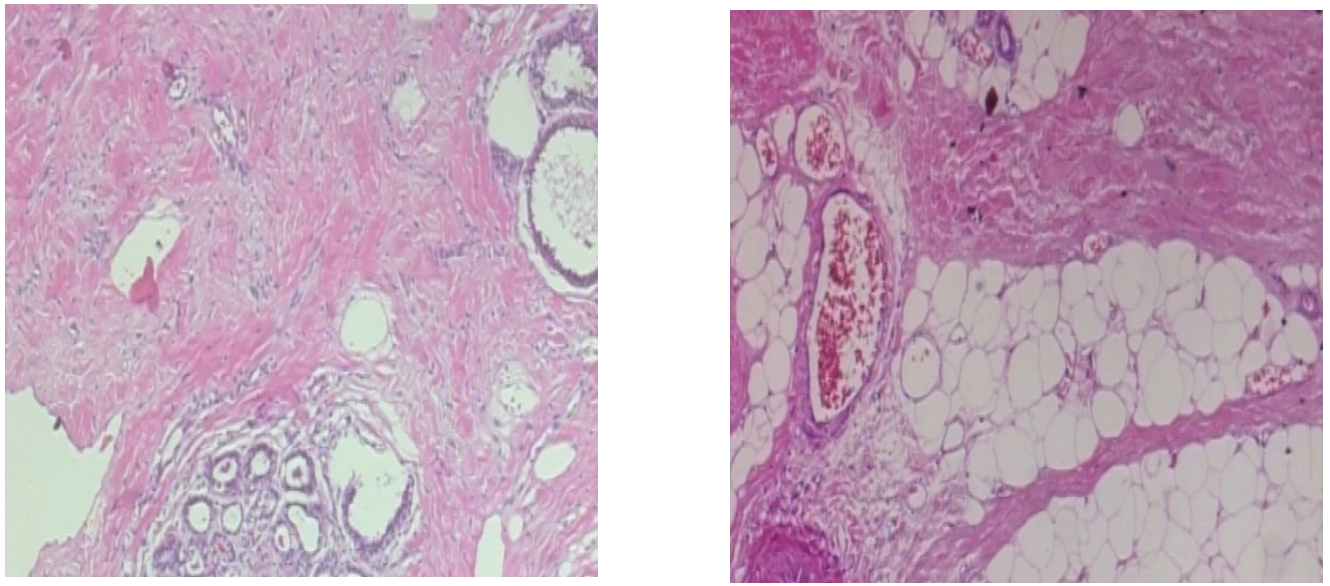
Data analytics methodologies include exploratory data analysis, data mining, predictive analytics, and machine learning

Predictive Analytics: The process of identifying patterns in real-time and historical data to detect trends and determine future outcomes.

Where can machine learning be applied?
Application

Deep Learning

Breast Cancer Classification: A Deep Learning Approach for Digital Pathology



(a) (b)

Fig. 1. (a) A slide of breast benign tumor, and (b) a slide of breast malignant tumor.

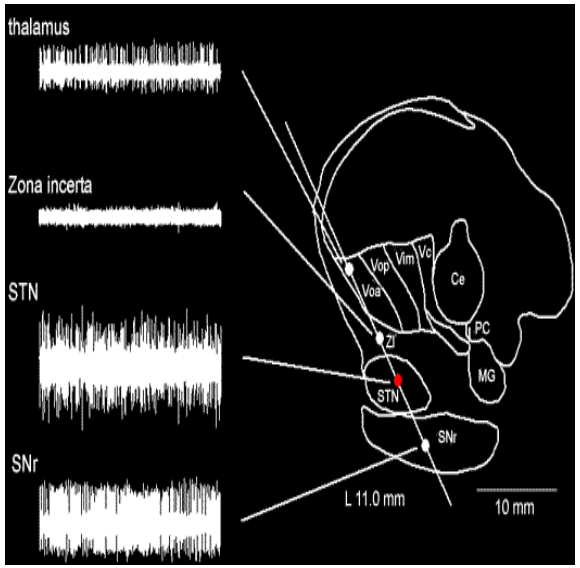
Accuracy of 91 %

Where can machine learning be applied?
Application

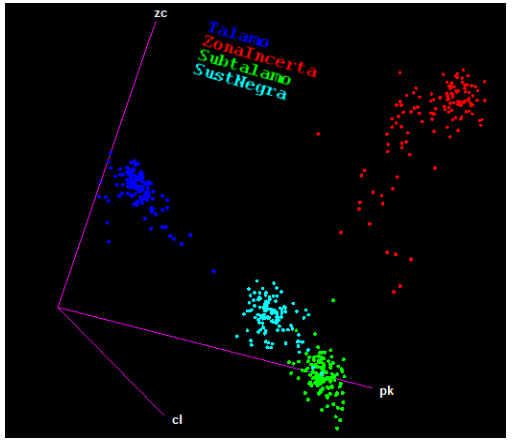
Deep brain stimulation for Parkinson's disease

Learning to classify subcorticals structures for DBS

- Features



Machine learning



Features

$$L = \sum_{i=1}^{N-1} |x_{i+1} - x_i| \qquad (6)$$

$$\gamma = \frac{3}{N-1}\sqrt{\sum_{i=1}^{N}(x_i - \bar{X})^2} \qquad (7)$$

$$\kappa = \frac{1}{2}\sum_{i=1}^{N-2} max\{0, |sgn[x_{i+1} - x_i] - sgn[x_{i+2} - x_{i+1}]|\} \qquad (8)$$

$$max(a,b) = \begin{cases} a & si \quad a > b \\ b & si \quad a < b \\ a \ o \ b & si \quad a = b \end{cases}$$

$$sgn(x) = \begin{cases} 1, & si \quad x > 0 \\ 0, & si \quad x = 0 \\ -1, & si \quad x < 0 \end{cases}$$

$$\delta = \sqrt{\frac{\sum_{i=1}^{N} x_i^2}{N}} \qquad (9)$$

$$\Psi = \frac{1}{N-2}\sum_{i=2}^{N-1} x_i^2 - x_{i-1}x_{i+1} \qquad (10)$$

$$\kappa = \frac{1}{2}\sum_{i=1}^{N-1} |\ sgn(x_{i+1}) - sgn(x_i)| \qquad (11)$$

Artificial Intelligence

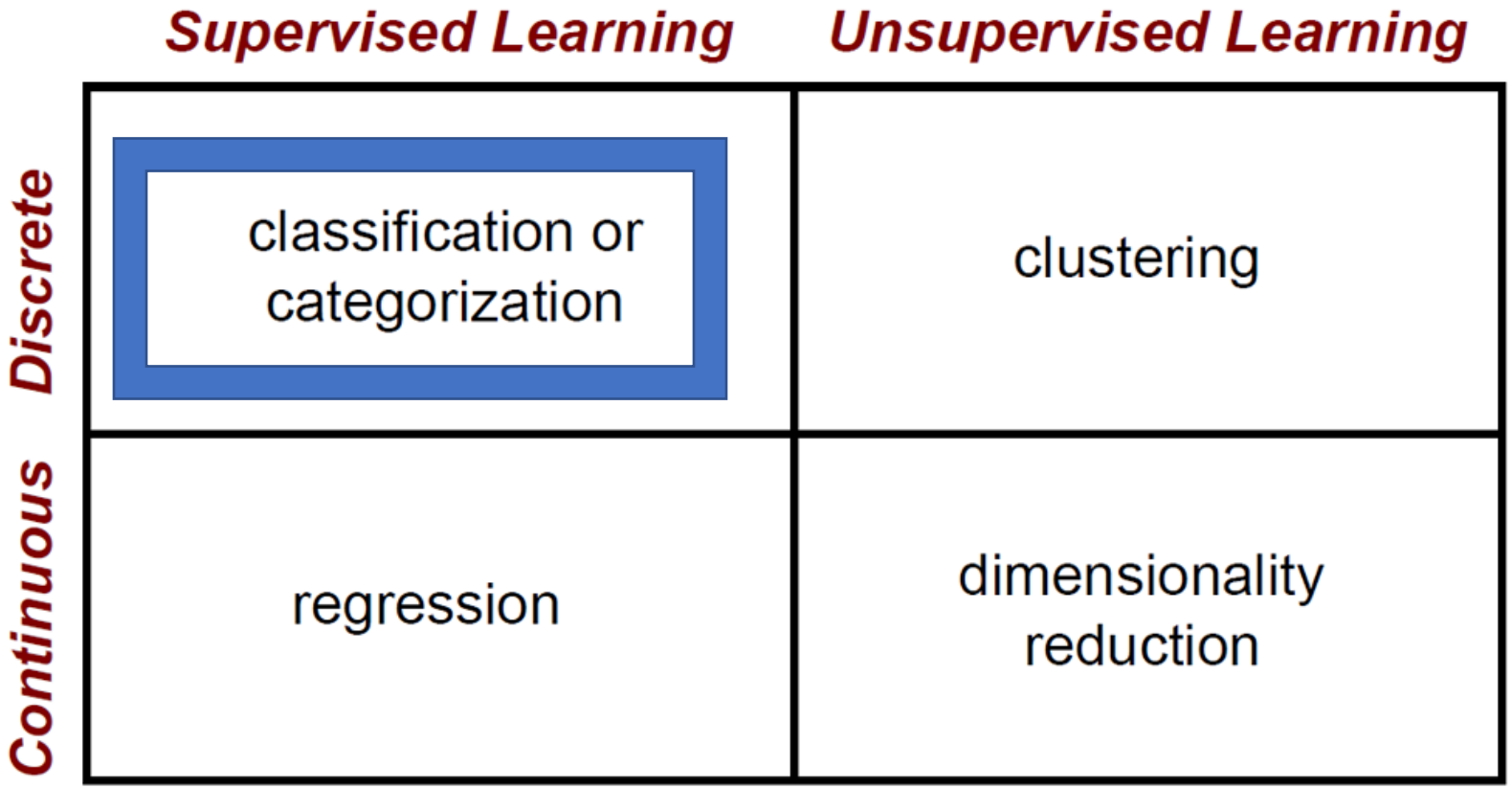
Artificial intelligence has been defined as the study of algorithms that give machines the ability to reason and perform functions such as problem solving, object and word recognition, inference of world states, and decision-making.

Major advances in computer science, such as hardware-based improvements in processing and storage, have enabled the base technologies required for the advent of artificial intelligence.

Artificial intelligence has been applied to various aspects of medicine, ranging from largely diagnostic applications in radiology, and pathology, to more therapeutic and interventional applications in cardiology, and surgery.

As the development and application of artificial intelligence technologies in medicine continues to grow, it is important for clinicians in every field to understand what these technologies are and how they can be leveraged to deliver safer, more efficient, more cost-effective care.

Machine Learning Problems



Instance based learning

- Approximating real, valued or discrete-valued target functions
- Learning in this algorithm consists of storing the presented training data
- When a new query instance is encountered, a set of similar related instances is retrieved from memory and used to classify the new query instance
- Construct only local approximation to the target function that applies in the neighborhood of the new query instance
- Never construct an approximation designed to perform well over the entire instance space
- Appropriate definition of “neighboring” instances
- Disadvantage of instance-based methods is that the costs of classifying new instances can be high
- Nearly all computation takes place at classification time rather than learning time

K-Nearest Neighbor algorithm

- Most basic instance-based method
- Data are represented in a vector space
- Supervised learning

Feature space

$$\left\{ < \vec{x}^{(1)}, f(\vec{x}^{(1)}) >, < \vec{x}^{(2)} f(\vec{x}^{(2)}) >, \ldots, < \vec{x}^{(n)}, f(\vec{x}^{(n)}) > \right\}$$

$$\vec{x} = \begin{cases} x_1 \\ x_2 \\ .. \\ .. \\ x_d \end{cases} \in \Re^d \qquad \left\| \vec{x} - \vec{y} \right\| = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

K-Nearest Neighbor algorithm

- In nearest-neighbor learning the target function may be either discrete-valued or real valued
- Learning a discrete valued function
- $f : \Re^d \rightarrow V$, V is the finite set $\{v_1,.....,v_n\}$
- For discrete-valued, the *k*-NN returns the most common value among the k training examples nearest to *xq*.

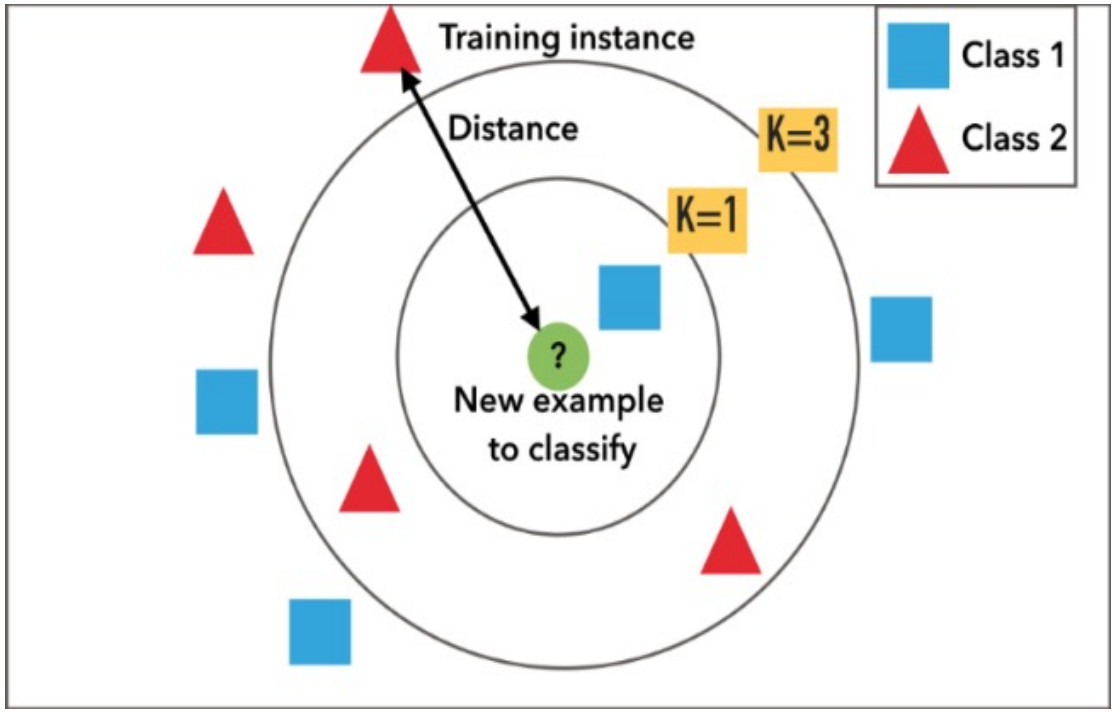
K-Nearest Neighbor algorithm

- Training algorithm
  - For each training example <*x, f(x)*> add the example to the list
- Classification algorithm
  - Given a query instance x_q to be classified
    - Let $x_1,..,x_k$ k instances which are nearest to x_q

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\arg\max} \sum_{i=1}^{k} \delta(v, f(x_i))$$

    - Where $\delta(a,b)=1$ if $a=b$, else $\delta(a,b)=0$ (Kronecker function)

K-Nearest Neighbor algorithm



Example of k-NN classification. The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles. If k = 3 (solid line circle) it is assigned to the second class because there are 2 triangles and only 1 square inside the inner circle. If k = 5 it is assigned to the first class (3 squares vs. 2 triangles inside the outer circle).

Logistic Regression

- Form of regression that allows the prediction of discrete variables by a mix of continuous and discrete predictors.

- Addresses the same questions that discriminant function analysis and multiple regression do but with no distributional assumptions on the predictors (the predictors do not have to be normally distributed, linearly related or have equal variance in each group)

- Logistic regression is often used because the relationship between the DV (a discrete variable) and a predictor is non-linear

π = Proportion of "Success"

In ordinary regression the model predicts the *mean* Y for any combination of predictors.

What's the "mean" of a 0/1 indicator variable?

$$\bar{y} = \frac{\sum y_i}{n} = \frac{\#\text{ of } 1's}{\#\text{ of trials}} = \text{Proportion of "success"}$$

Goal of logistic regression: Predict the "true" proportion of success, π, at any value of the predictor.

The logistic function

$$\widehat{Y}_i = \frac{e^u}{1+e^u}$$

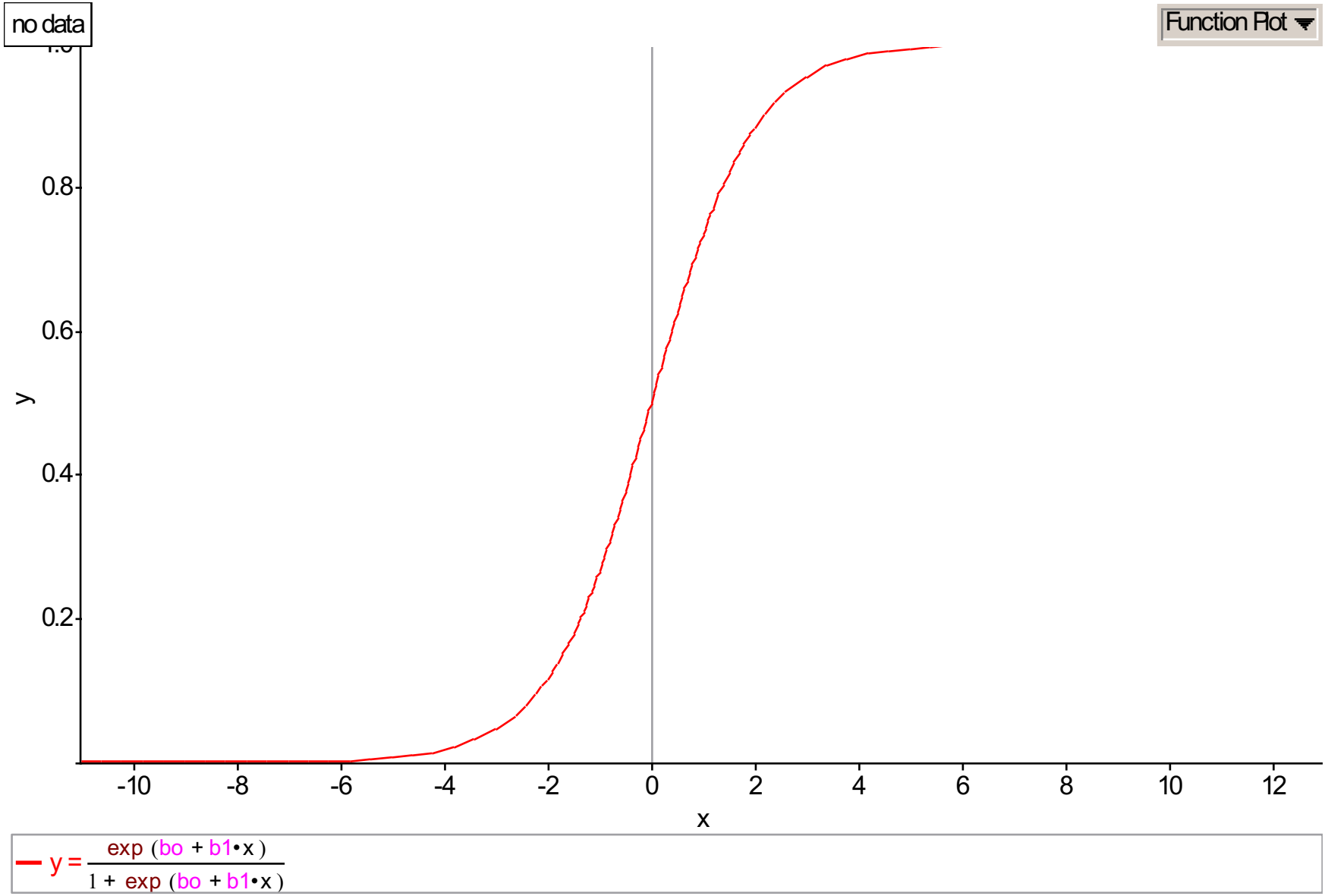
- Where Y-hat is the estimated probability that the i-th case is in a category and u is the regular linear regression equation:

$$u = A + B_1X_1 + B_2X_2 + \cdots + B_KX_K$$

The logistic function

$$\hat{\pi}_i = \frac{e^{b_0 + b_1 X_1}}{1 + e^{b_0 + b_1 X_1}}$$

Logit Function



Discriminant Analysis

Classification is an important component of all scientific research. Statistical techniques concerned with classification are essentially of two types.

The first (cluster analysis) aims to uncover groups of observations from initially unclassified data.

The second (**discriminant analysis**) works with data that is already classified into groups to derive rules for classifying new (and as yet unclassified) individuals on the basis of their observed variable values.

Discriminant analysis is used to determine which variables discriminate between two or more naturally occurring groups.

Linear Discriminant Functions

A discriminant function is a linear combination of the components of x:

$$g(x) = w^t x + w_0$$

where

- w^t is the weight vector
- w_0 is the bias or threshold weight

For the two-class problem we can use the following decision rule:

Decide c1 if $g(x) > 0$ and c2 if $g(x) < 0$.

For the general case we will have one discriminant function for each class.

The Problem with Multiple Classes

How do we use a linear discriminant when we have more than two classes?

There are two approaches:

1. Learn one discriminant function for each class
2. Learn a discriminant function for all pairs of classes

If c is the number of classes, in the first case we have c functions and in the second case we have c(c-1) / 2 functions.

In both cases we are left with ambiguous regions.

Figure 5.3

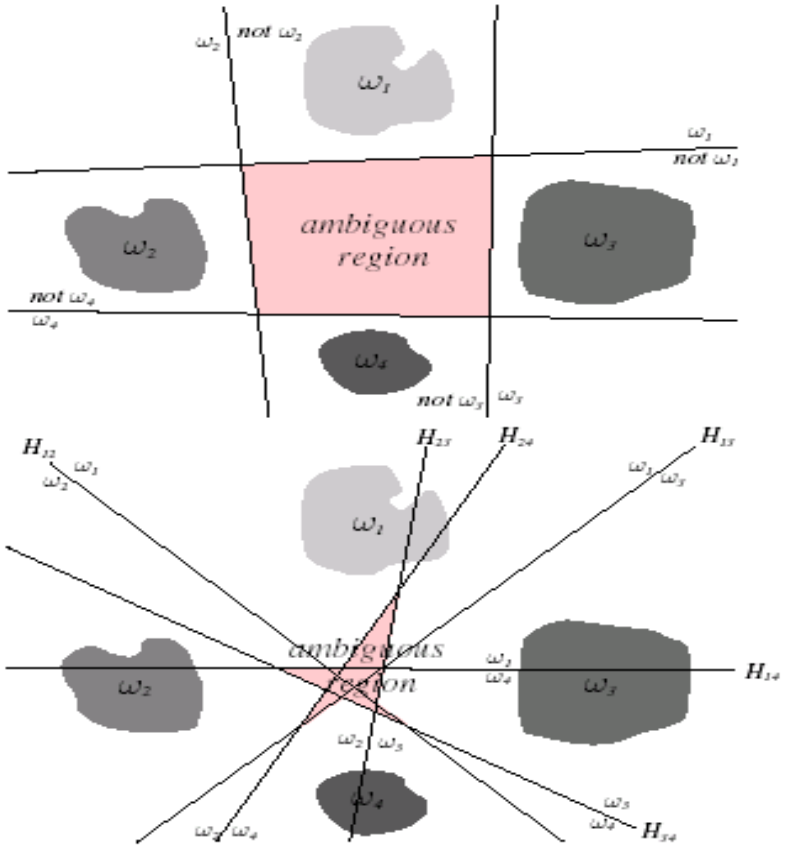


FIGURE 5.3. Linear decision boundaries for a four-class problem. The top figure shows ω_i/not ω_i dichotomies while the bottom figure shows ω_i/ω_j dichotomies and the corresponding decision boundaries H_{ij}. The pink regions have ambiguous category assignments. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Linear Machines

To avoid the problem of ambiguous regions we can use linear machines:

- We define c linear discriminant functions and choose the one with highest value for a given x.

$$g_k(x) = w_k^t x + w_{k0} \quad k = 1, \ldots, c$$

- In this case the decision regions are convex (a set which contains the entire line segment joining any pair of its points) and thus are limited in flexibility and accuracy.

Figure 5.4

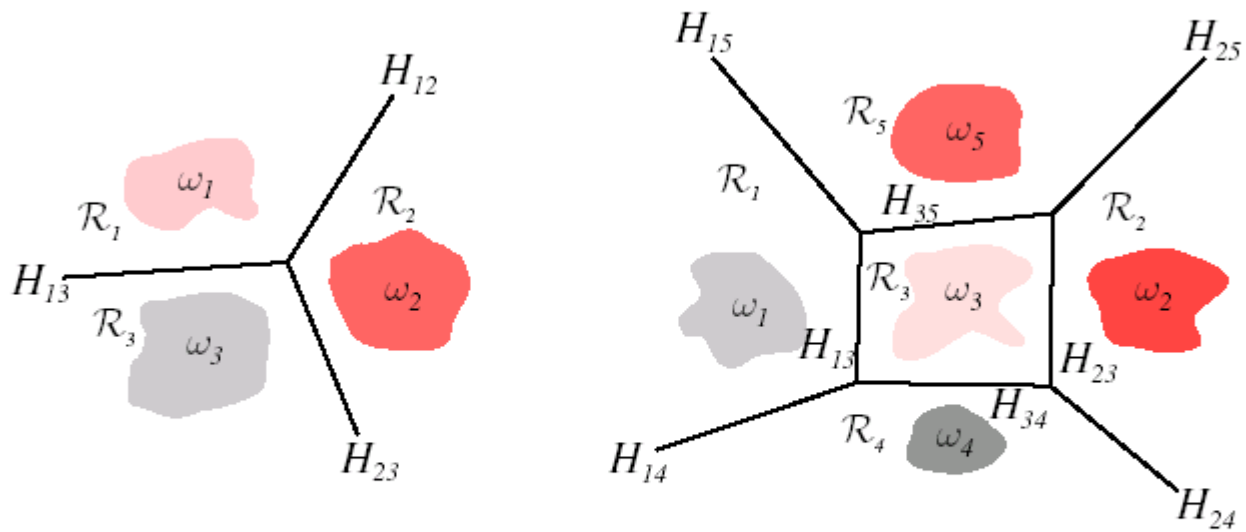


FIGURE 5.4. Decision boundaries produced by a linear machine for a three-class problem and a five-class problem. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Support Vector Machines

- Support Vector Machines (SVMs) were invented by Vladimir Vapnik for classification based on prior knowledge [Vapnik98, Burges98].

- Create classification functions from a set of labeled training data.

- SVMs find a hypersurface in the space of possible inputs.
  - Split the positive examples from the negative examples.
  - The split is chosen to have the largest distance from the hypersurface to the nearest of the positive and negative examples.

- SVMs can be used to multiclass classification

Support Vector Machine

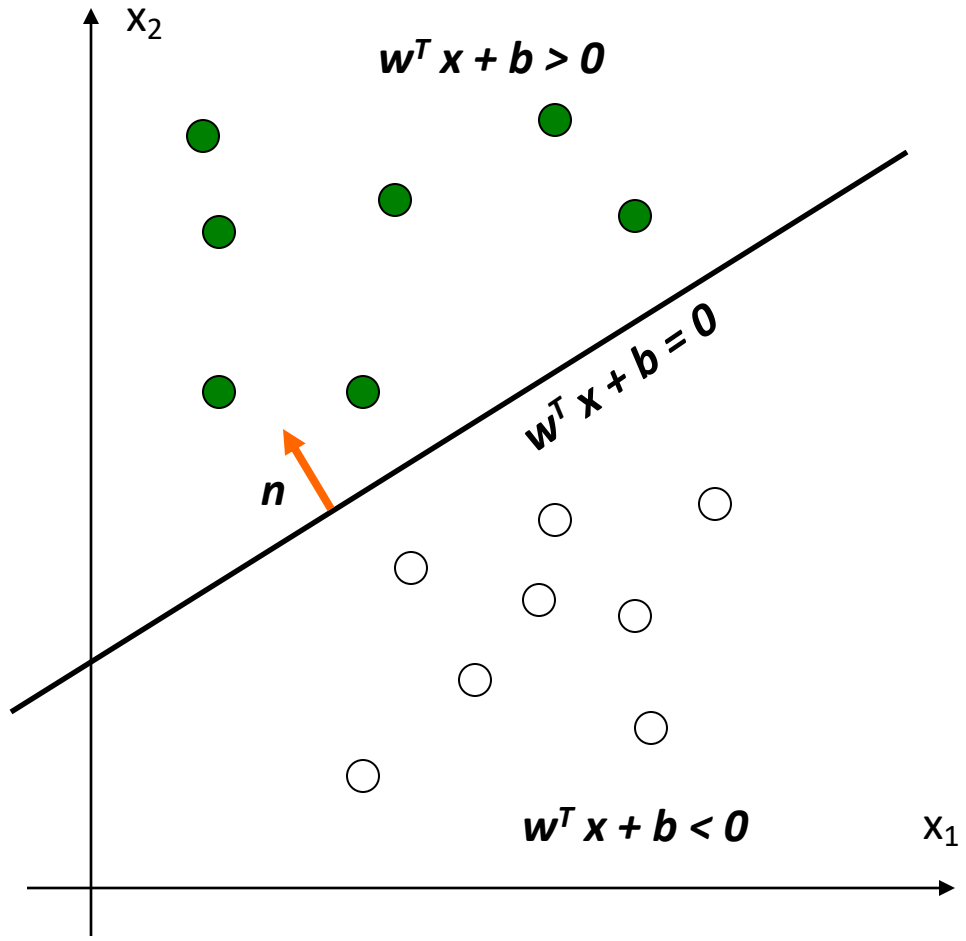
- g(x) is a linear function:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

- A hyper-plane in the feature space

- (Unit-length) normal vector of the hyper-plane:

$$\mathbf{n} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$$



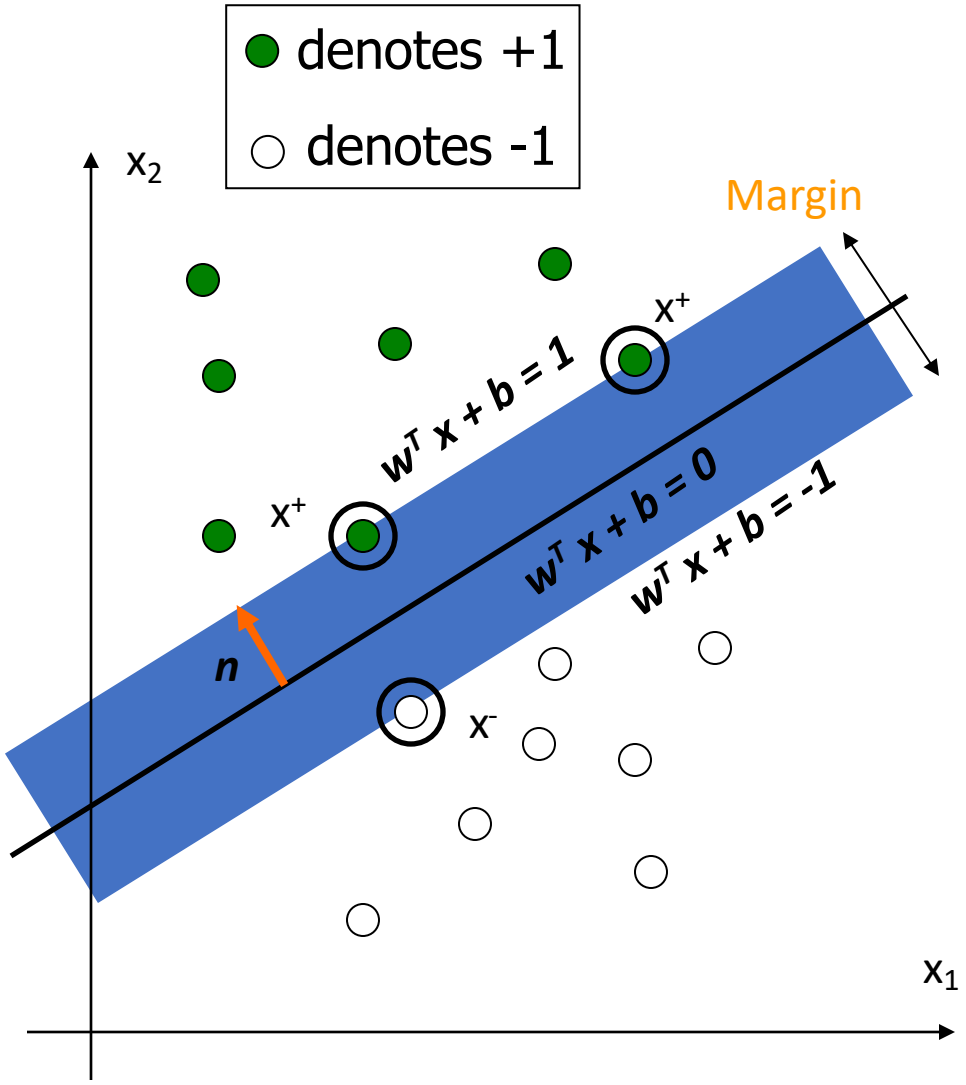
Large Margin Linear Classifier

- Formulation:

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

such that

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

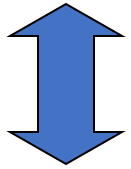


Solving the Optimization Problem

Quadratic programming with linear constraints

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$



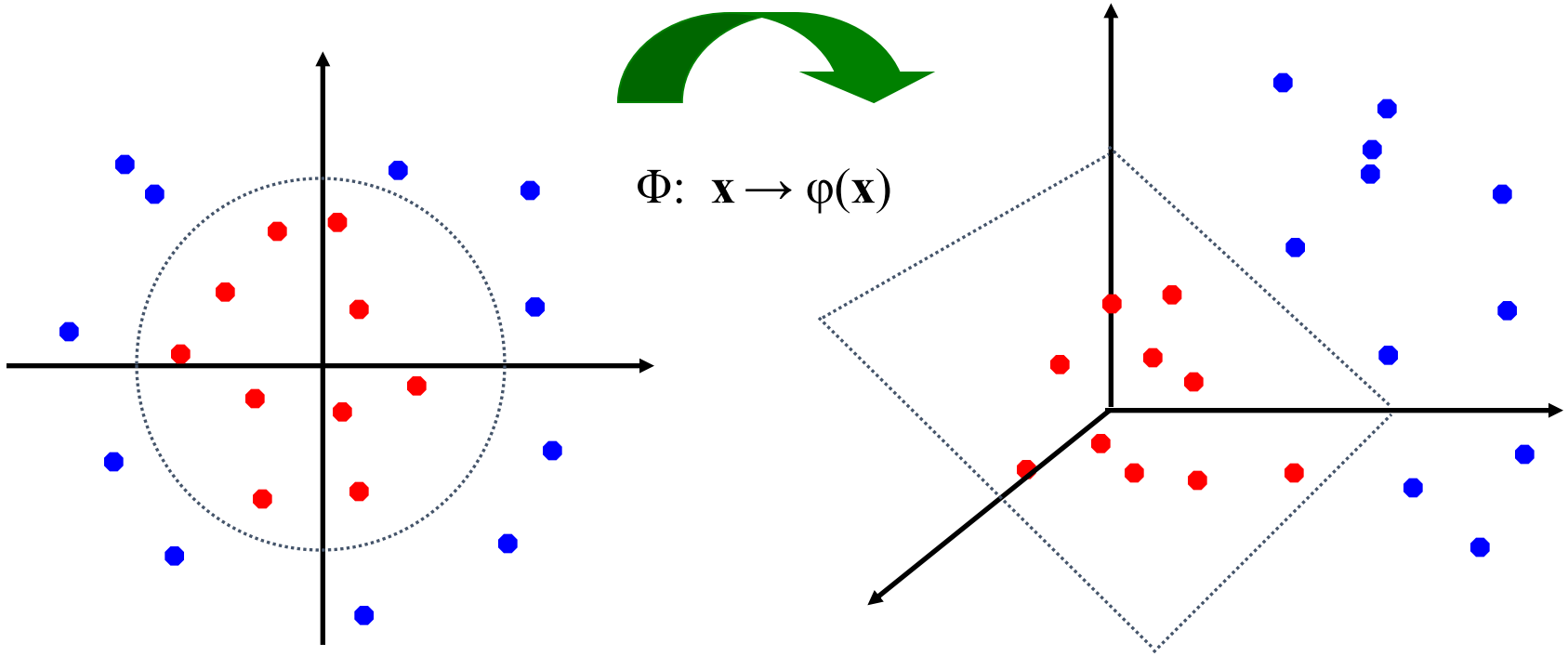
Lagrangian Function

$$\text{minimize} \quad L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

Non-linear SVMs: Feature Space

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:



Nonlinear SVMs: The Kernel Trick

- With this mapping, our discriminant function is now:

$$g(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{i \in \text{SV}} \alpha_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) + b$$

- No need to know this mapping explicitly, because we only use the dot product of feature vectors in both the training and test.

- A *kernel function* is defined as a function that corresponds to a dot product of two feature vectors in some expanded feature space:

$$K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Nonlinear SVMs: The Kernel Trick

- Examples of commonly-used kernel functions:
  - Linear kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
  - Polynomial kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$
  - Gaussian (Radial-Basis Function (RBF)) kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{2\sigma^2}\right)$$

  - Sigmoid:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

Support Vector Machine: Algorithm

- 1. Choose a kernel function
- 2. Choose a value for C
- 3. Solve the quadratic programming problem
- 4. Construct the discriminant function from the support vectors

Decision Trees and Random Forest

Decision trees

A predictive model that uses a set of binary rules applied to calculate a target value

Can be used for classification (categorical variables) or regression (continuous variables) applications

Rules are developed using software available in many statistics packages

Different algorithms are used to determine the “best” split at a node

Decision trees

How do classification trees work? Uses training data to build model

Tree generator determines:

- Which variable to split at a node and the value of the split
- Decision to stop (make a terminal node) or split again
- Assign terminal nodes to a class

Decision trees

- Decision Trees are one of the most used and known classification algorithms in machine learning. They generate models formed by a set of conditions that are posed in a tree shape of easy understanding

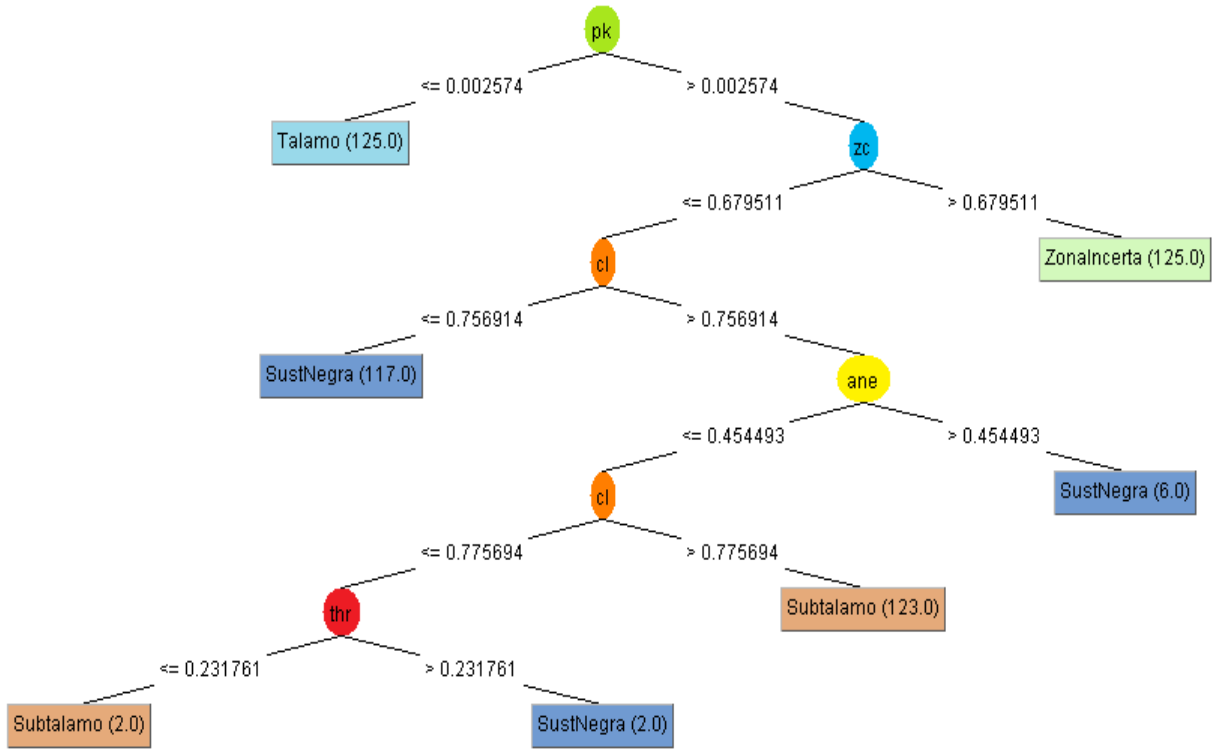
- The trees show the conditions organized in a hierarchic manner, in such a way that the final class or decision to be taken can be obtained following the conditions satisfied from the root of the tree

- In general, the decision trees perform a search in the space of instances of the variable (and its value) that better divide such space in disjoint classes. This process, known as “partition-based search”, is done in successive steps until the classes are separated adequately

Decision trees

- The most important current algorithms such as CART, ID3, ASSISTANT and C4.5, essentially differ in the criteria used for partition. One of the most efficient and robust algorithm is the C4.5 algorithm. This algorithm uses partition criteria based on the computation of entropy, with features that are more robust to noise and outliers or missing values. The fundamental advantages of the decision trees algorithm are its easy implementation, robustness, ability to work with many input variables, and the non requirement of a high number of samples for training.

Decision trees



What are ensemble models?

- Combines the results from different models
- Models can be a similar type or different
- The result from an ensemble model is usually better than the result from one of the individual models

Random Forest

- An ensemble classifier using many decision tree models
- Can be used for classification or regression
- Accuracy and variable importance information is provided with the results

Estimating Error

- Cross-validation (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set
  - Stratified cross-validation: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

The diagonal elements show the number of correct classifications for each class

The off-diagonal elements provides the misclassifications

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

More measures

- True Positive Rate (TPR) (Sensitivity)
  - a/a+b
- True Negative Rate (TNR) (Specificity)
  - d/c+d
- False Positive Rate (FPR)
  - c/c+d
- False Negative Rate (FNR)
  - b/a+b

Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value
- **Loss function**: measures the error betw. y_i and the predicted value y_i'
  - Absolute error: $| y_i - y_i' |$
  - Squared error: $(y_i - y_i')^2$
- Test error (generalization error): the average loss over the test set
  - Mean absolute error: $\frac{\sum_{i=1}^{d} | y_i - y_i' |}{d}$ Mean squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$
  - Relative absolute error: $\frac{\sum_{i=1}^{d} | y_i - y_i' |}{\sum_{i=1}^{d} | y_i - \bar{y} |}$ Relative squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$

Sklearn.metrics

Compute precision, recall, F-measure and support for each class

- The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier.

- The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

- The F-beta score can be interpreted as a weighted harmonic mean of the precision and recall, where an F-beta score reaches its best value at 1 and worst score at 0.

- The F-beta score weights recall more than precision by a factor of beta.
  beta == 1.0 means recall and precision are equally important.

- The support is the number of occurrences of each class in y_true.