LECTURE 03 17/10/2023

APPLICATION OF INFORMATION THEORY TO CLASSIFIERS

FEATURE, INSTANCE, CLASS
TOTAL SET, TRAINING SET
CLASSIFIER
DIFFERENT TYPES, DECISION TREE
HOW TO BUILD A CLASSIFIER
INFORMATION GAIN RATIO
ID3 - ITERATIVE DICHOTOMISER 3
EXAMPLE
STOPPING CRITERIA

NUMERICAL VALUES : C4.5
EXERCISE 1.5

EXAMPLES ON REAL FUNCTIONS
HYPER. RECTANGLES

TREE ENSEMBLES AND RANDOM FORESTS

ASSIGNMENT 1

CLASSIFIERS : IDEA

$$\underline{V} = (x_1, \ldots, x_i, \ldots, x_N)$$

$\underline{V}$ — VECTOR, INSTANCE (OBSERVATION, EXPERIMENT)

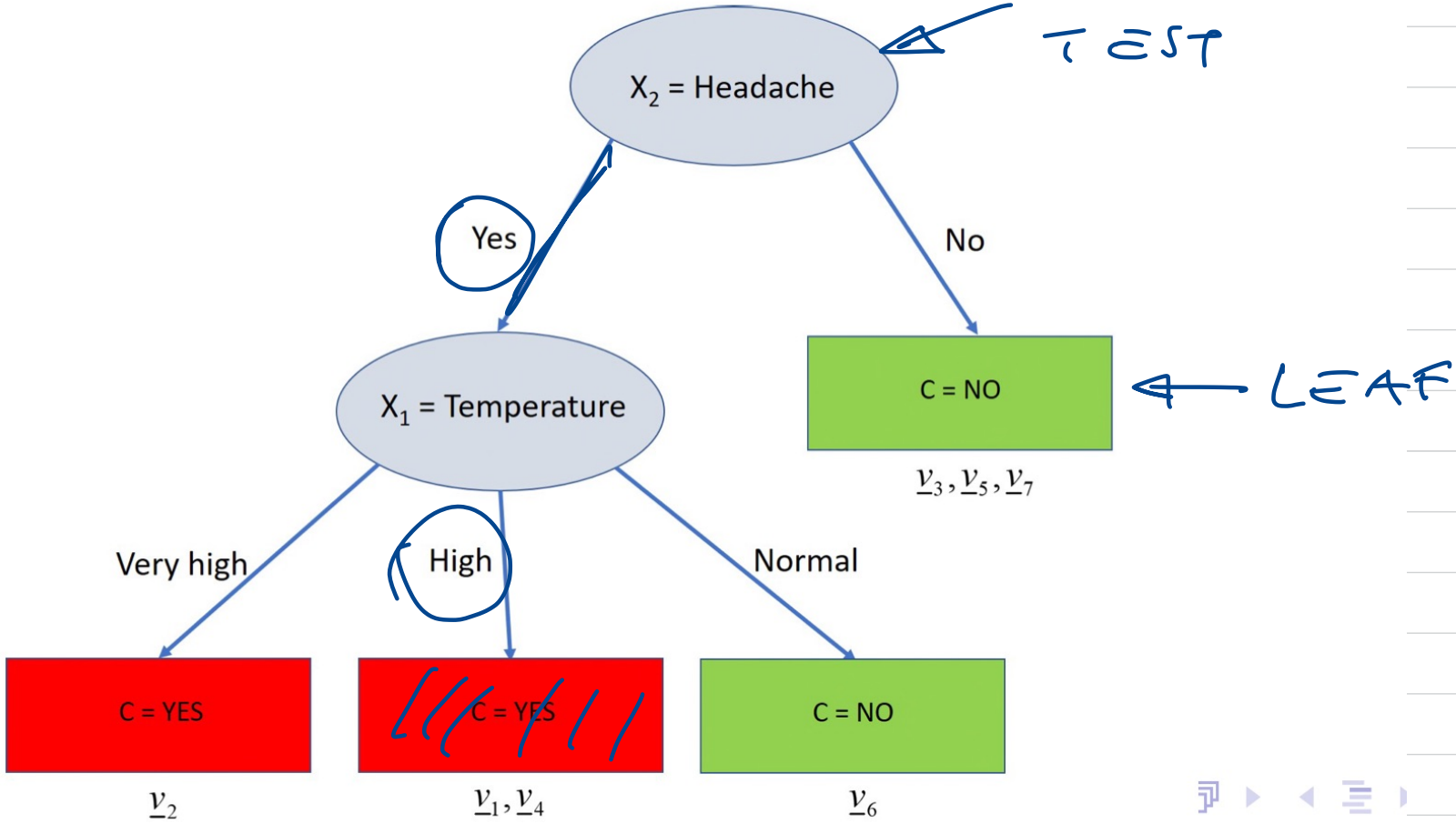
x_i — VARIABLES, FEATURES

$$F(\underline{V}) = C$$

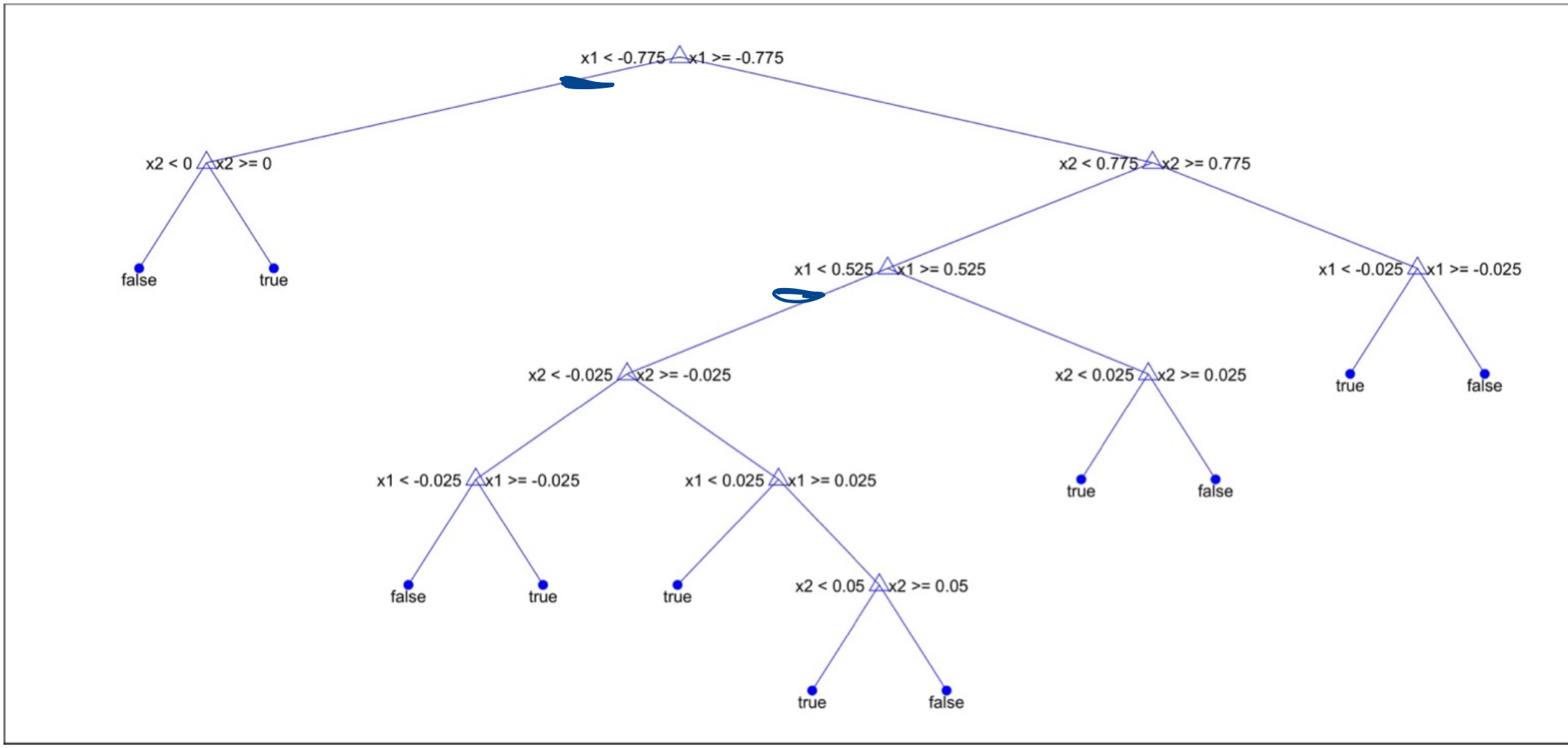
C — CLASS (CATEGORY)

$\underline{v} = (x_1\ x_2\ x_3)$

FEATURE TEST



CATEGORICAL FEATURES



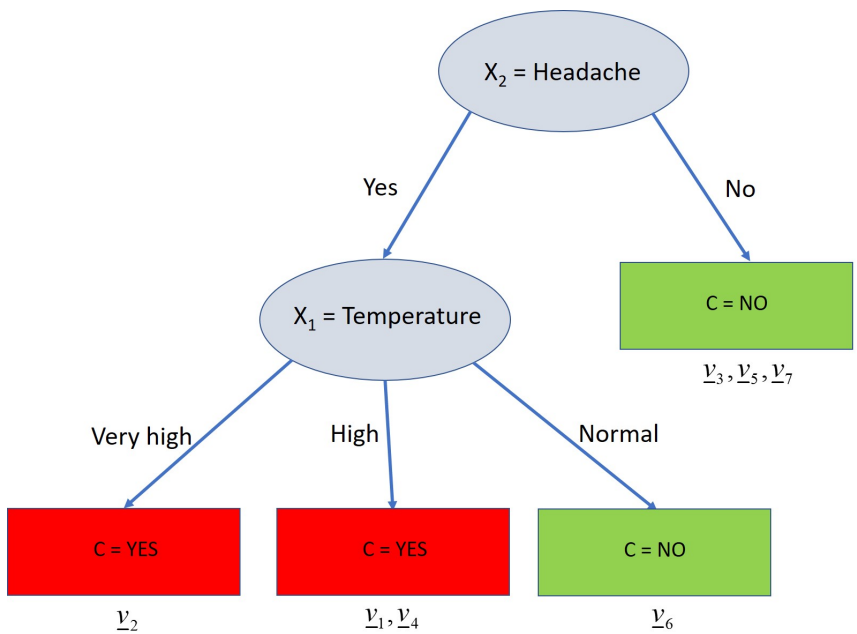
NUMERICAL FEATURES

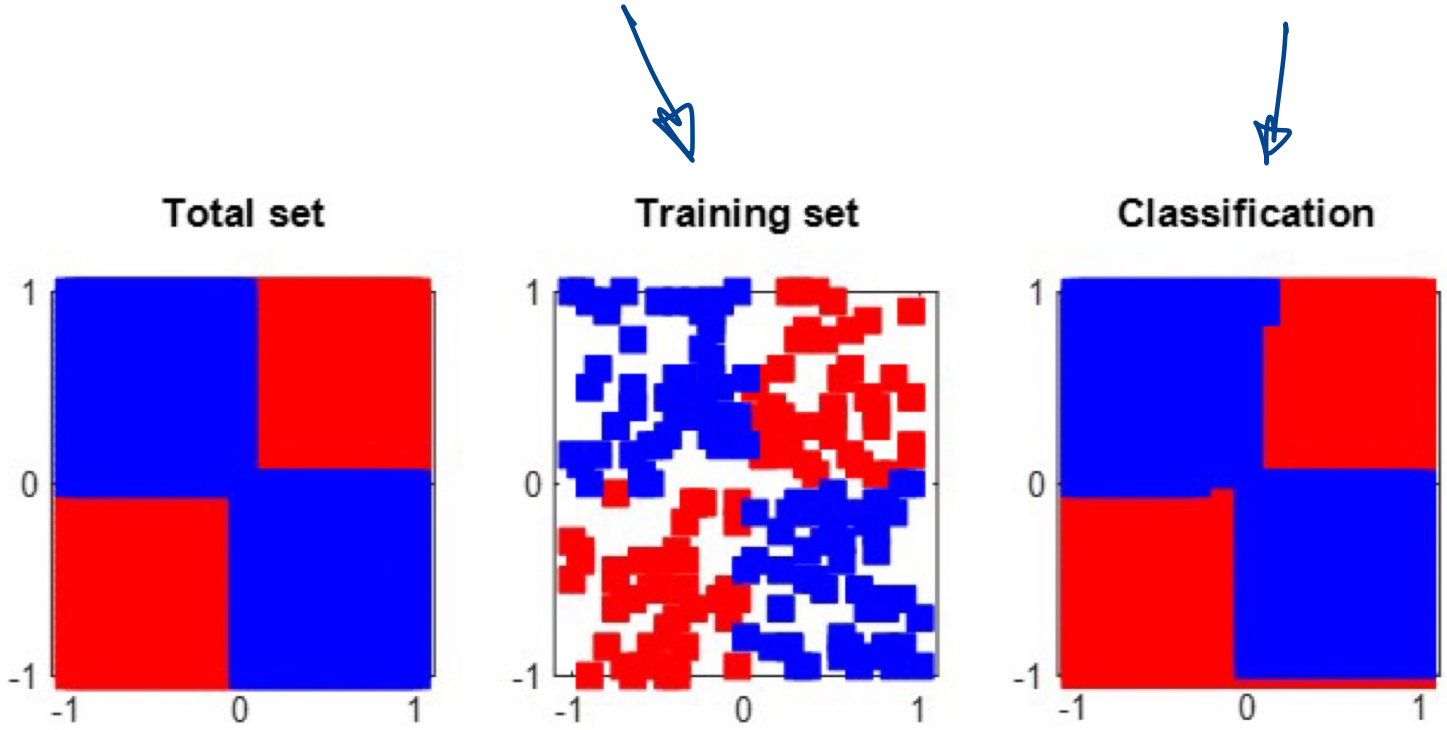
Example

TRAINING SET

Given a record of data a doctor must understand if a patient is healthy or ill

	Attributes			*Decision*
	Temperature	**Headache**	**Nausea**	**Flu**
1	high	yes	no	yes
2	very_high	yes	yes	yes
3	normal	no	no	no
4	high	yes	yes	yes
5	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no




Total set
Training set
Classification
1
0
-1
-1
0
1

Statistical classification has many different approaches:

- Linear Discriminant Analysis
- Naive Bayes Classifiers
- Nearest neighbor
- Support Vector Machine
- **Decision Trees**
- Neural Networks

FEATURE , INSTANCE , CLASS

INSTANCE

FEATURE

$$\underline{v} = (x_1, \ldots, x_i, \ldots, x_n)$$

$x_i \in \Omega_i$ ← ALPHABET OF FEATURE i

$$\underline{v} \in \Omega_V \subseteq \Omega_1 \times \cdots \times \Omega_i \times \cdots \times \Omega_n$$

↑ ALPHABET OF INSTANCE $\underline{v}$

$$C \in \Omega_C$$

CLASS ↑ ↑ CLASS ALPHABET

$$f: \Omega_V \to \Omega_C$$
$$\underline{v} \to f(\underline{v}) = C$$

TRAINING SET

$TS \subseteq \Omega_V$

FOR ALL $\underline{w} \in TS$ WE KNOW

THE EXACT CLASS $f(\underline{w}) = c$

CLASSIFIER g

GIVEN THE TS WE BUILD A FUNCTION g THAT SHOULD MIMIC THE FUNCTION f

$$g(\underline{v}) = c'$$

IDEAL GOAL 1

$\forall \underline{w} \in TS \quad g(\underline{w}) = f(\underline{w})$ NOT ALWAYS POSSIBLE

IDEAL GOAL 2

$\forall \underline{v} \notin TS \quad g(\underline{v}) = f(\underline{v})$ NOT ALWAYS VERIFIED

TREE CLASSIFIER

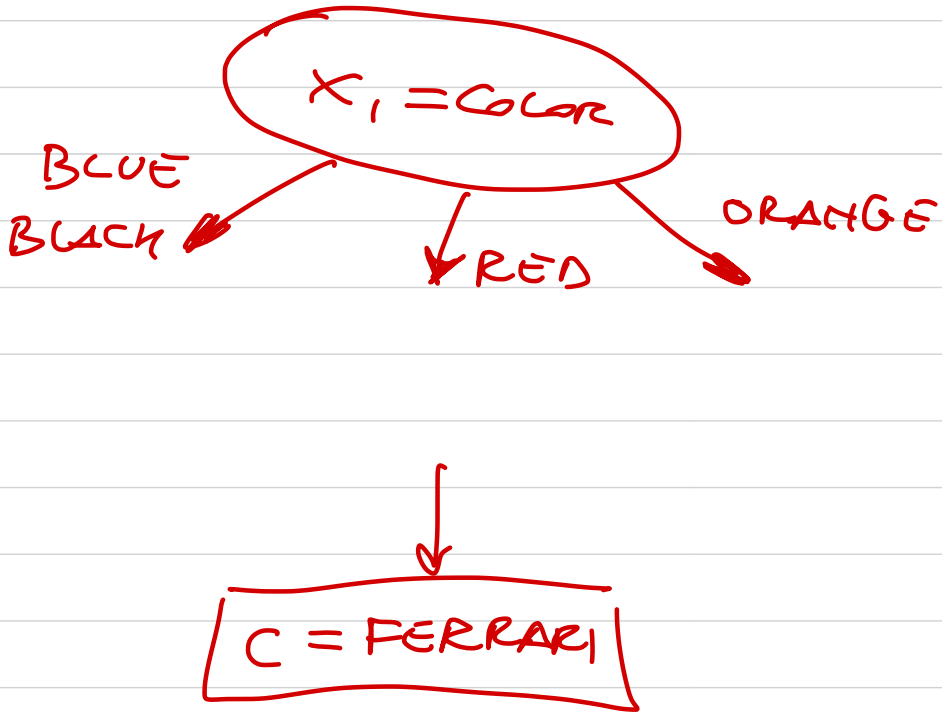
IT IS MADE BY TWO KIND OF BLOCKS

- FEATURE TEST

  ANALYSIS OF A SINGLE FEATURE

- LEAF

  CLASS ASSOCIATION



BASIC IDEA: INFORMATION GAIN

GIVEN THE TS WE WANT TO BUILD THE TREE CLASSIFIER.

AT ANY LEVEL WE MUST DECIDE THE FEATURE TO BE ANALYZED

$$\underline{V} = (x_1\ x_2\ x_3\ x_4) \rightarrow C$$

$$I(C, x_1) = H(C) - H(C|x_1)$$

$\equiv$ REDUCTION OF UNCERTAINTY ABOUT C WHEN THE VALUE OF x_1 IS REVEALED

IDEA

$$\underline{v} = (x_1 \; x_2 \; x_3 \; x_4) \qquad G$$

WE COMPUTE

$I(C, x_1)$ $I(C, x_2)$ $I(C x_3)$ $I(C x_4)$

AND WE CHOOSE THE LARGEST

THIS FEATURE MAX. THE

UNCERTAINTY REDUCTION

ABOUT THE CLASS

ID3 - ITERATIVE DICHOTOMISER (Ross Quinlan)

- WE START FROM THE ENTIRE TS
- WE LOOK FOR THE FEATURE THAT MAXIMIZES INFORMATION GAIN

ACTUALLY, WE USE

INF. GAIN RATIO (NORMALIZED INF. GAIN)

$$IGR(a, x_i) = \frac{I(a, x_i)}{H(x_i)}$$

- WE USE THIS FEATURE TO IMPLEMENT A FEATURE TEST
- WE ITERATE OVER THE NEW SET OF VECTORS

- IF THE FEATURES ARE CATEGORICAL AT THE NEXT LEVEL WE ANALYZE THE REMAINING FEATURES (WE CANNOT TEST A GIVEN FEATURE TWICE)

- IF THE FEATURES ARE NUMERICAL
  - WE CONSIDER A THRESHOLD VARYING BETWEEN THE MINIMUM AND THE MAXIMUM VALUES
  - FOR EACH THRESHOLD WE COMPUTE IGR
  - WE SELECT THE THRESHOLD THAT MAXIMIZES IGR FOR THAT FEATURE
  - WE SELECT THE FEATURE WITH MAXIMUM IGR (MAX OF MAX)

Example

$IGR(C, x_1)$

$U = (x_1 \quad x_2 \quad x_3) \quad C$

$= \dfrac{I(C, x_1)}{H(x_1)}$

	Attributes			Decision
	Temperature	Headache	Nausea	Flu
1	high	yes	no	yes
2	very_high	yes	yes	yes
3	normal	no	no	no
4	high	yes	yes	yes
5	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no

$I(C, x_1) = H(C) - H(C|x_1)$

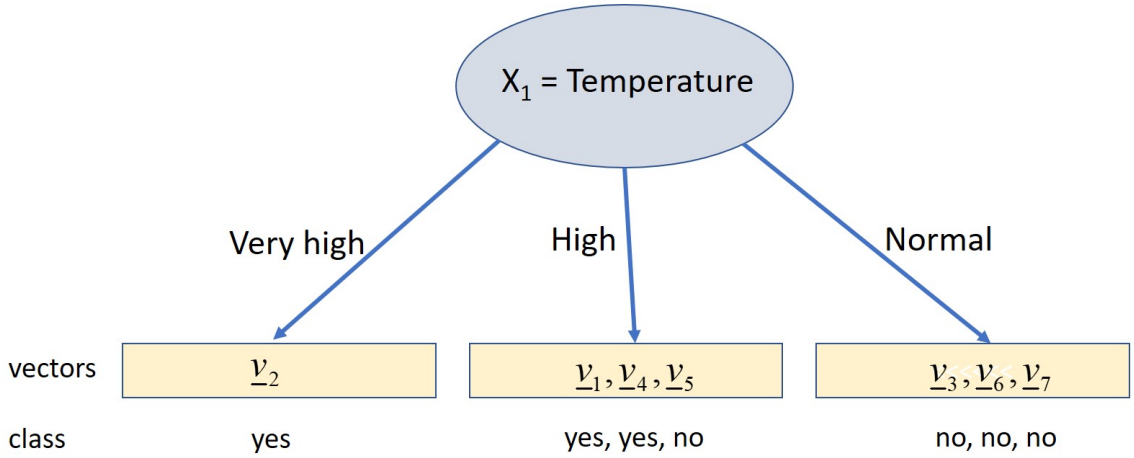
Let us first compute the entropy of the class:

$$P(c = \text{yes}) = \frac{3}{7} \quad P(c = \text{no}) = \frac{4}{7}$$

$$H(C) = 0.9852$$

Example

Let us consider the first feature:



$$H(C) = 0.9852$$

$$H(C|X_1 = \text{Very High}) = 0$$

$$H(C|X_1 = \text{High}) = 0.9183$$

$$H(C|X_1 = \text{Normal}) = 0$$

$$H(C|X_1) = 0.3936$$

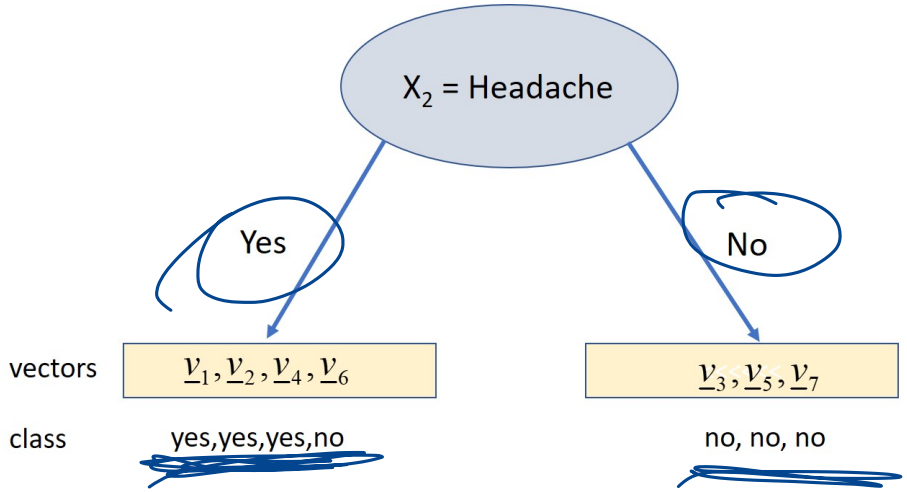
$$I(C; X_1) = 0.5916 \quad = H(C) - H(C|X_1)$$

$$H(X_1) = 1.4488$$

$$IGR(C, X_1) = 0.4083 \quad = I(C, X_1) / H(X_1)$$

Example

Let us consider the second feature:



$$H(C) = 0.9852$$

$$H(C|X_2 = \text{Yes}) = 0.8113$$

$$H(C|X_2 = \text{No}) = 0$$

$$H(C|X_2) = 0.4636$$

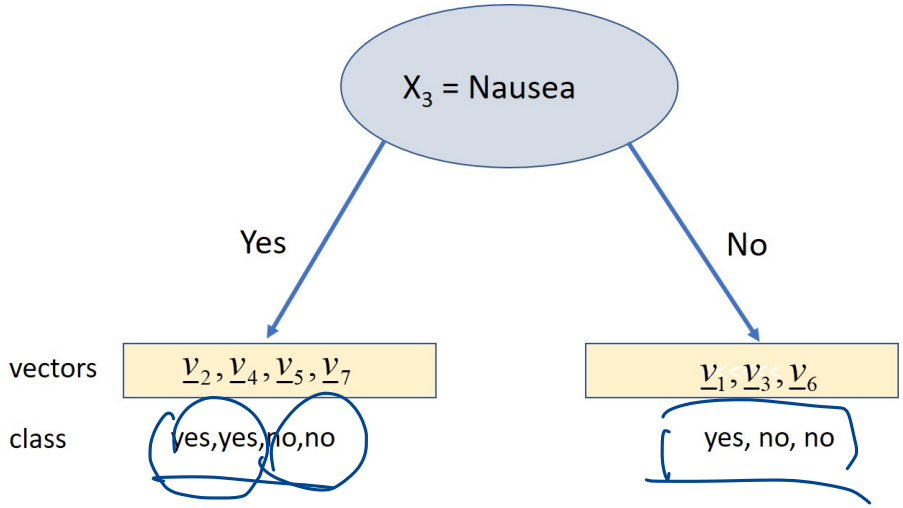
$$I(C; X_2) = 0.5216$$

$$H(X_2) = 0.9852$$

$$IGR(C, X_2) = 0.5294$$

Example

Let us consider the third feature:



$$H(C) = 0.9852$$

$$H(C|X_3 = \text{Yes}) = 1$$

$$H(C|X_3 = \text{No}) = 0.9183$$

$$H(C|X_3) = 0.9650$$

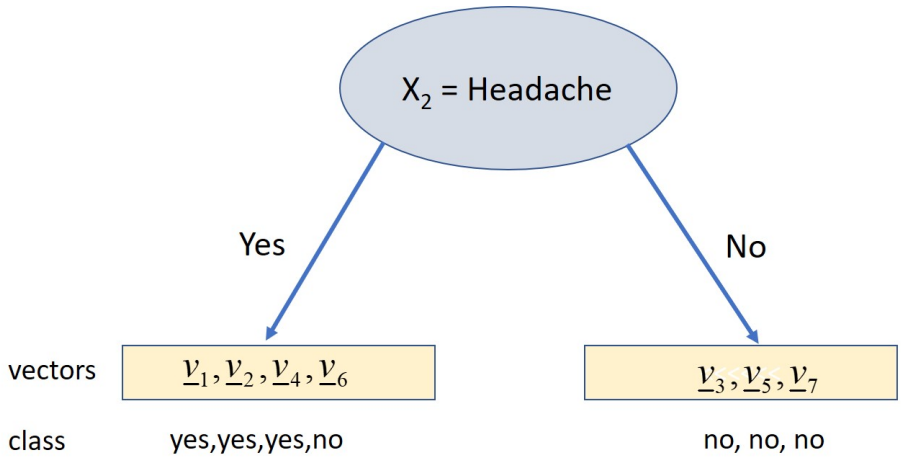
$$I(C; X_3) = 0.0202$$

$$H(X_3) = 0.9852$$

$$IGR(C, X_3) = 0.0205$$

Example

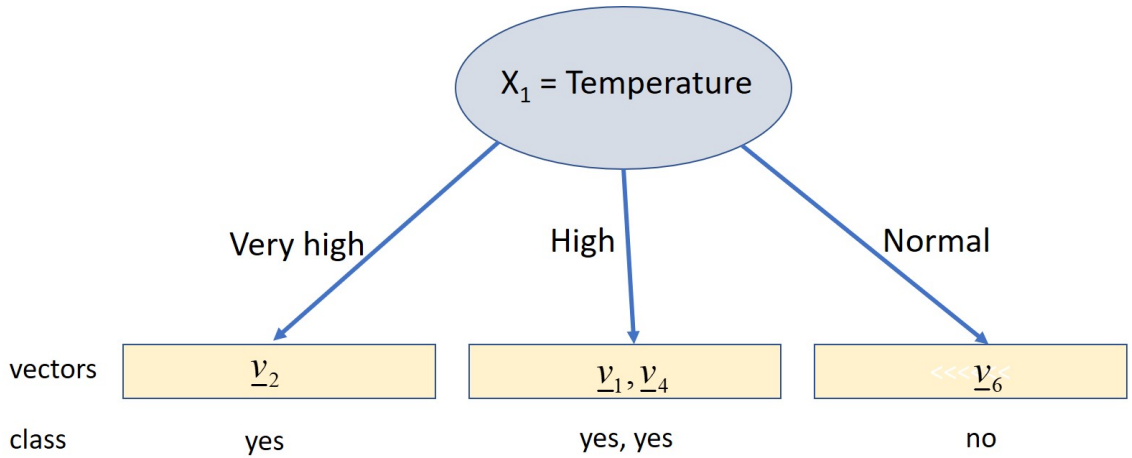
Based on the best Information Gain Ratio, we choose the second feature:



Now we apply the same procedure to the subset $\underline{v_1}, \underline{v_2}, \underline{v_4}, \underline{v_6}$. We consider the remaining features: x_1 =Temperature and x_3 = Nausea.

Example

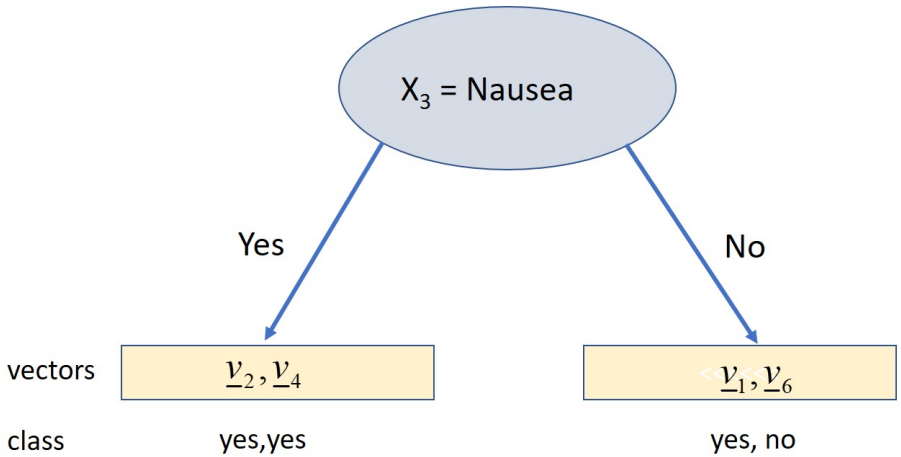
Let us consider the first feature x_1 =Temperature:



$$H(C) = 0.8113$$
$$H(C|X_1 = \text{Very High}) = 0$$
$$H(C|X_1 = \text{High}) = 0$$
$$H(C|X_1 = \text{Normal}) = 0$$
$$H(C|X_1) = 0$$
$$I(C; X_1) = 0.8113$$
$$H(X_1) = 1.5$$
$$IGR(C, X_1) = 0.5408$$

Example

Let us consider the third feature x_3 =Nausea:



$$H(C) = 0.8113$$

$$H(C|X_3 = \text{Yes}) = 0$$

$$H(C|X_3 = \text{No}) = 1$$

$$H(C|X_3) = 0.5$$

$$I(C; X_3) = 0.3113$$

$$H(X_3) = 0.9852$$

$$IGR(C, X_3) = 0.3160$$

Example

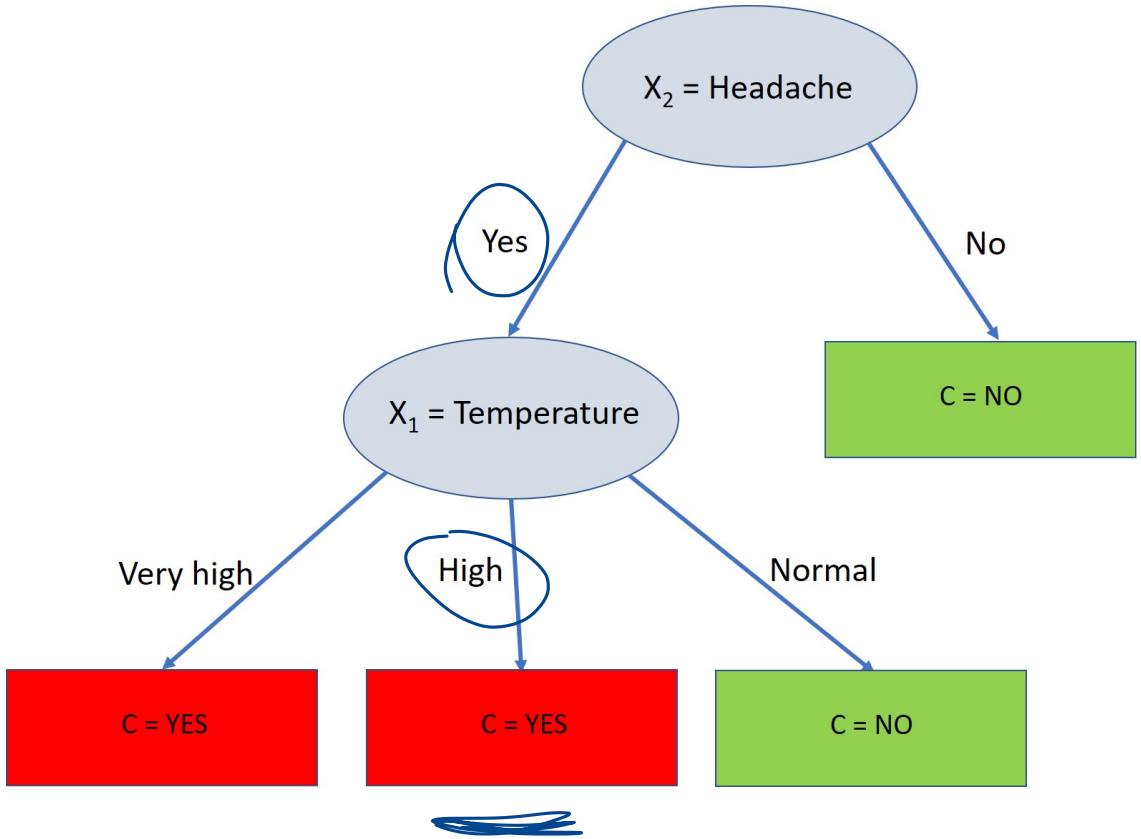
For the other subset it is useless to make a choice because all the vectors already belong to the same class:

$\underline{v}_3, \underline{v}_5, \underline{v}_7$

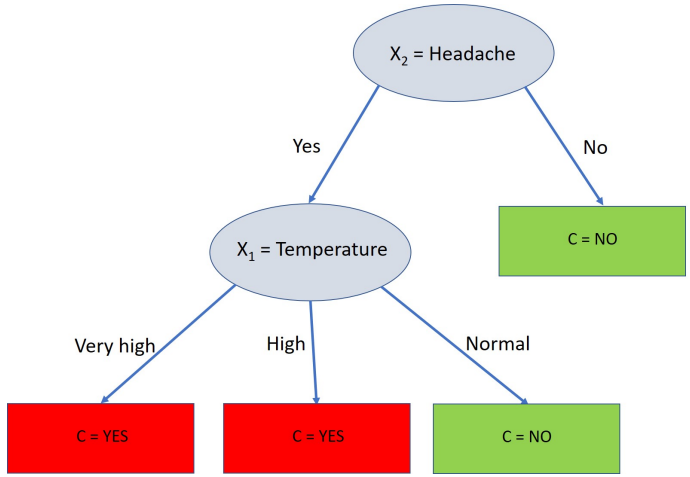
no, no, no

Example

Finally we have our decision tree:



Example



Now we can use the decision tree as a classifier. If we observe a new vector, we apply the rule to compute the class it belongs to. As an example

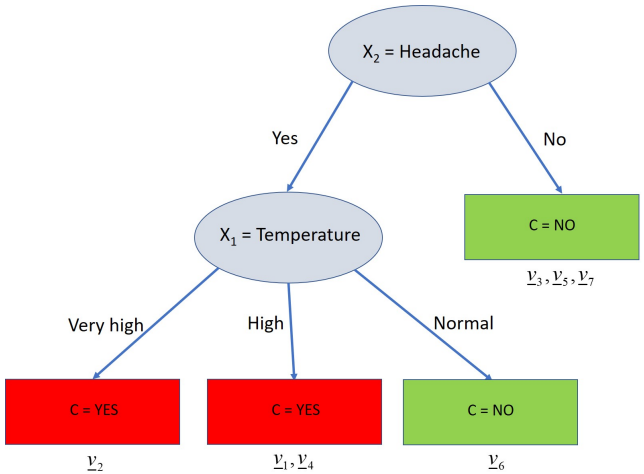
$$\underline{v} = (\text{normal}, \text{yes}, \text{yes})$$

is mapped into class NO

$$\underline{v} = (\text{veryhigh}, \text{yes}, \text{no})$$

is mapped into class YES

Example



Note that we can translate the tree classifier into these logical rules:

$$(\text{Headache} = \text{yes}) \quad \text{AND}\, (\text{Temperature} = \text{veryhigh}) \longrightarrow (\text{Class} = \text{YES})$$

$$(\text{Headache} = \text{yes}) \quad \text{AND}\, (\text{Temperature} = \text{high}) \longrightarrow (\text{Class} = \text{YES})$$

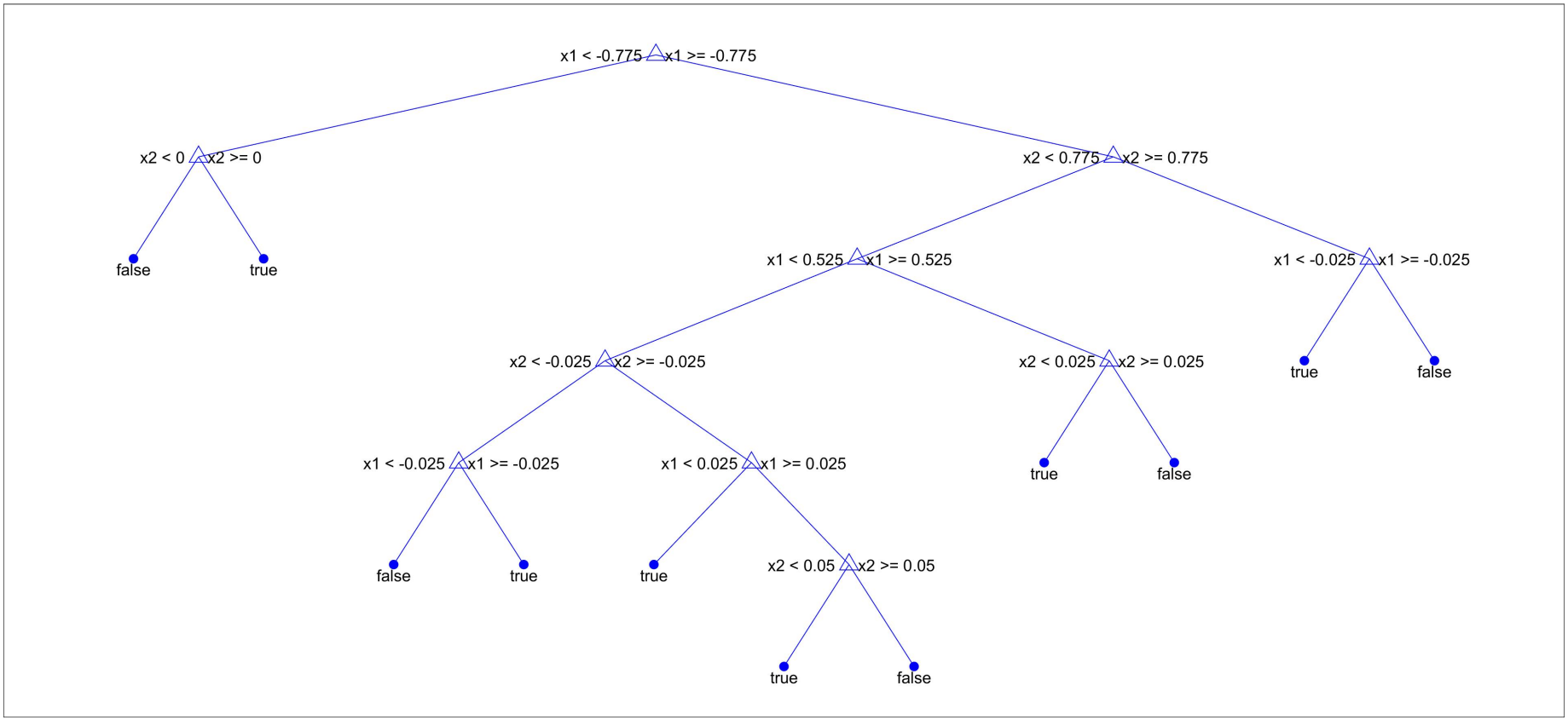
$$(\text{Headache} = \text{yes}) \quad \text{AND}\, (\text{Temperature} = \text{normal}) \longrightarrow (\text{Class} = \text{NO})$$

$$(\text{Headache} = \text{no}) \longrightarrow (\text{Class} = \text{NO})$$

Stopping criterion:

- When a subset contains only vectors of the same class: we create a leaf node labeled by this class.
- When a subset contains vectors of different classes, but all the features have already been considered: we create a leaf node labeled by the most common class among the subset vectors.

Example 1


x1 < -0.775
x1 >= -0.775
x2 < 0
x2 >= 0
x2 < 0.775
x2 >= 0.775
false
true
x1 < 0.525
x1 >= 0.525
x1 < -0.025
x1 >= -0.025
x2 < -0.025
x2 >= -0.025
x2 < 0.025
x2 >= 0.025
true
false
x1 < -0.025
x1 >= -0.025
x1 < 0.025
x1 >= 0.025
true
false
false
true
true
x2 < 0.05
x2 >= 0.05
true
false

Given this training set

	x_1	x_2	x_3	c
v_1	30	0	10	0
v_2	30	0	70	0
v_3	30	1	20	0
v_4	30	1	80	1
v_5	60	0	40	0
v_6	60	0	60	1
v_7	60	1	50	0
v_8	60	1	60	1

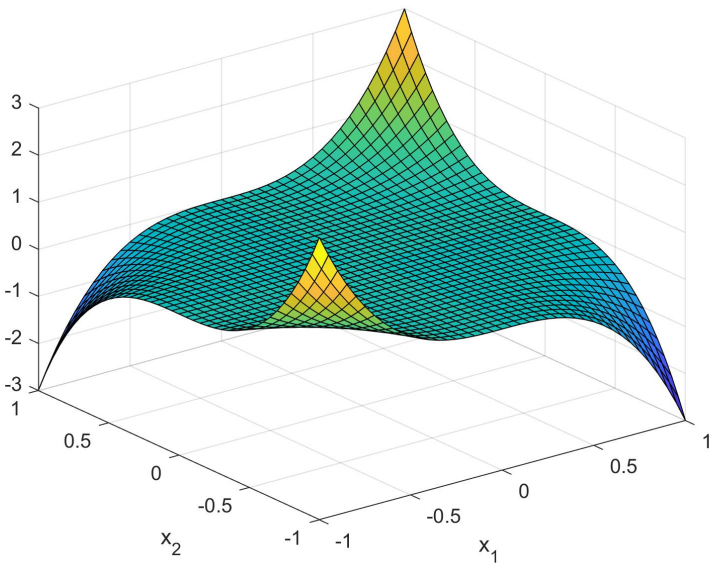
Write a Matlab program that:

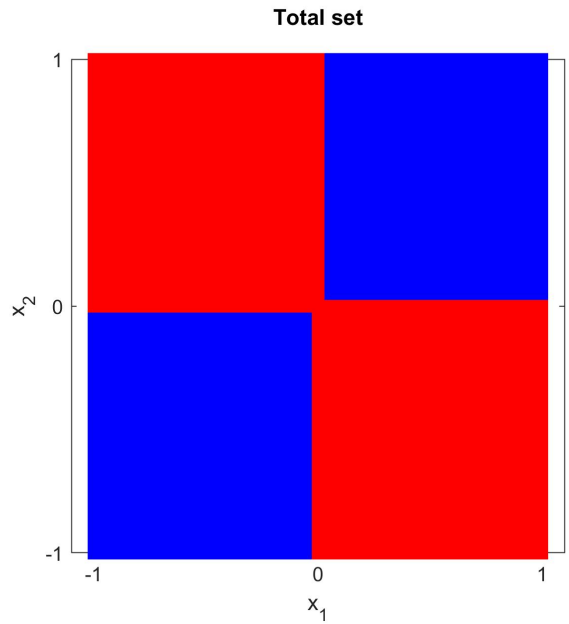
1. Builds a tree classifier based on the information gain ratio.
2. Given a vector $\underline{x} = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix}$, computes the corresponding class c.

Example 2

$$\underline{v} = (x_1, x_2)$$

$$c = f(\underline{v}) = \text{sign}\left(3x_1^3 x_2^3\right)$$



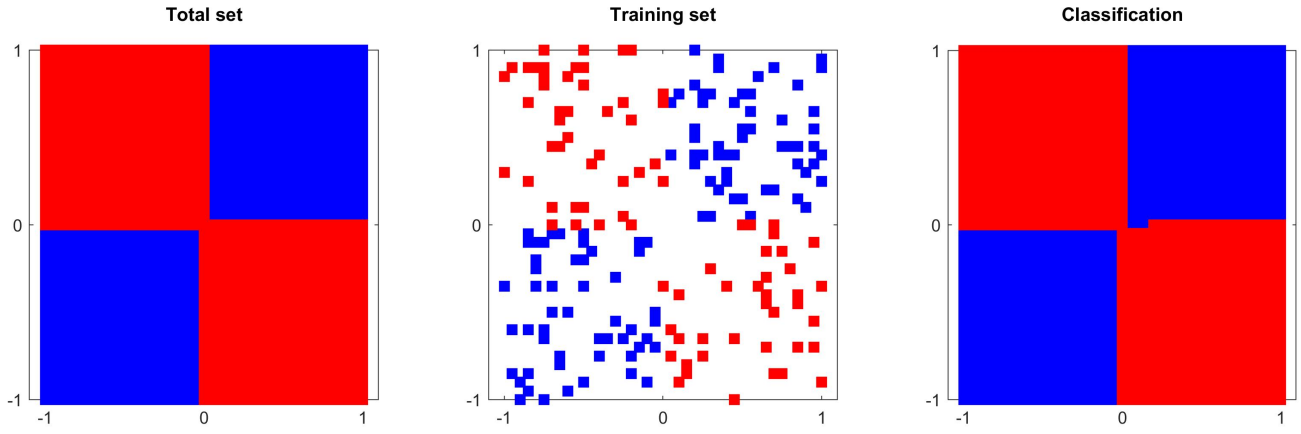


Example 2: training sets 1

$$x_1 = -1 : 0.05 : 1 \quad x_2 = -1 : 0.05 : 1;$$

$$N_v = 1681$$

$$N_{TS} = 300$$



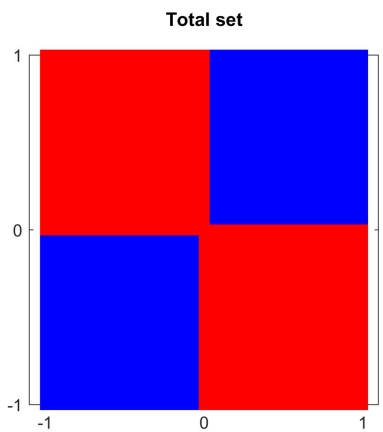
number of errors: 3/1681

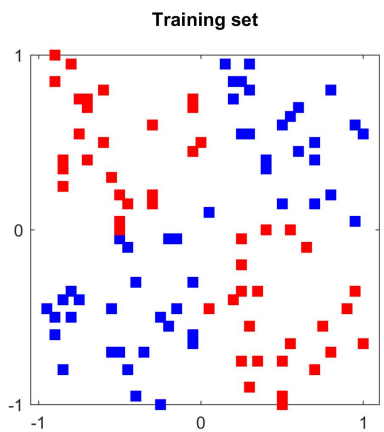
Example 2: training sets 2

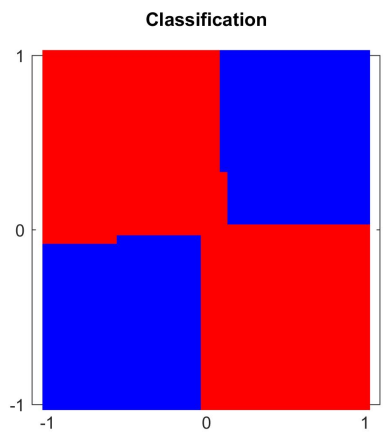
$$x_1 = -1 : 0.05 : 1 \quad x_2 = -1 : 0.05 : 1;$$

$$N_v = 1681$$

$$N_{TS} = 100$$







number of errors: 35/1681

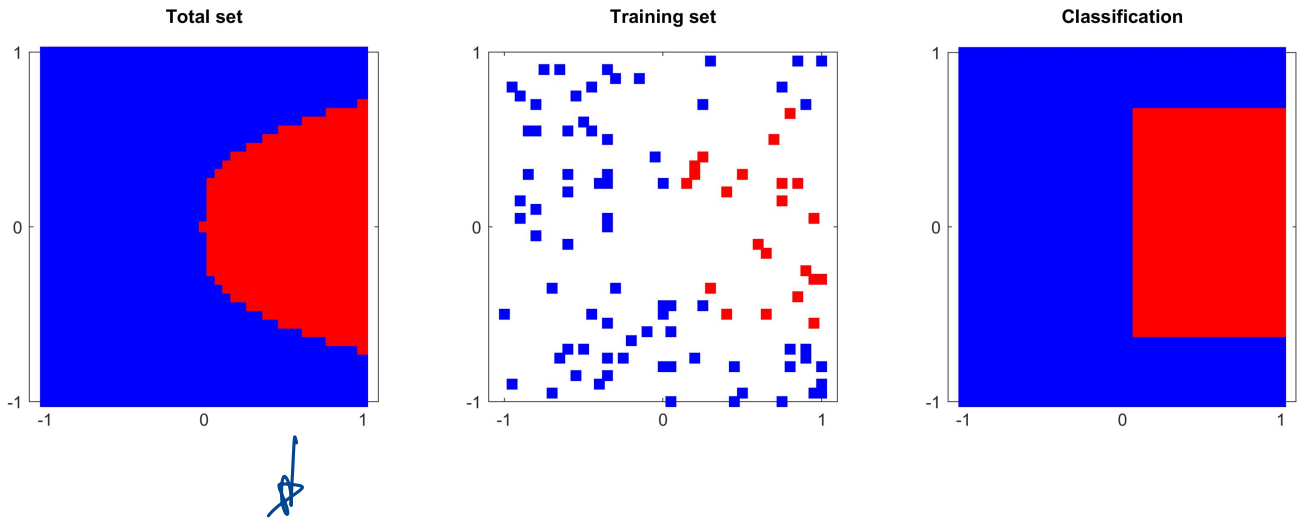
Example 3: training set 1

$$\underline{v} = (x_1, x_2)$$

$$c = f(\underline{v}) = \text{sign}\left(-2\sqrt[3]{x_1^2} + 4x_2^2\right)$$

$$x_1 = -1 : 0.05 : 1 \quad x_2 = -1 : 0.05 : 1;$$

$$N_v = 1681 \quad N_{TS} = 100$$



number of errors: 97/1681

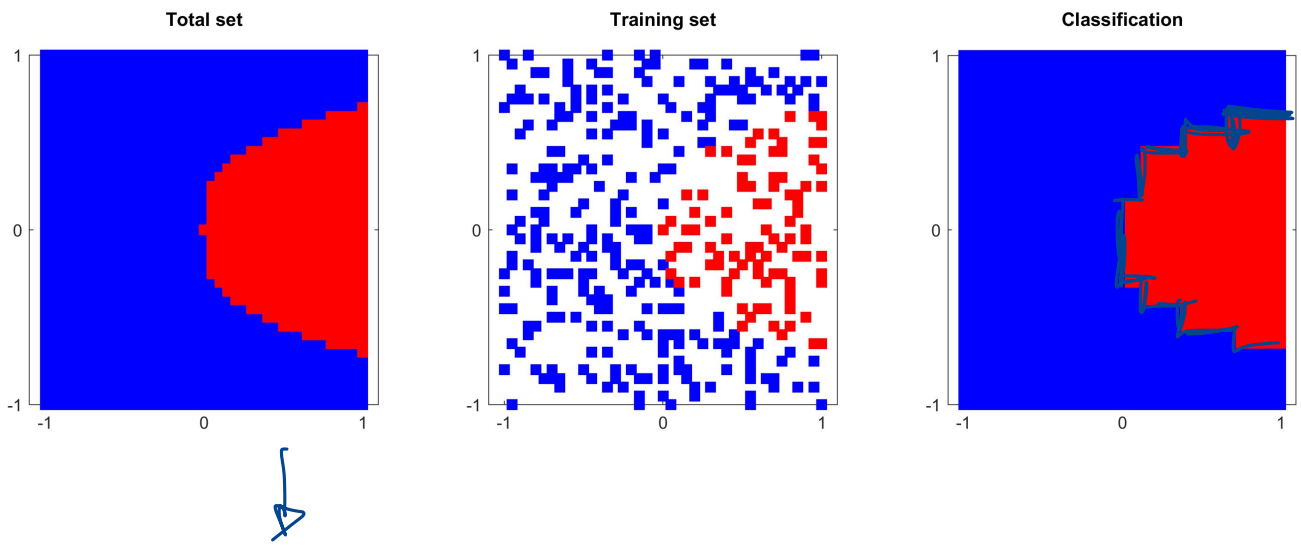
Example 3: training set 2

$$\underline{v} = (x_1, x_2)$$

$$c = f(\underline{v}) = \text{sign}\left(-2\sqrt[3]{x_1^2} + 4x_2^2\right)$$

$$x_1 = -1 : 0.05 : 1 \quad x_2 = -1 : 0.05 : 1;$$

$$N_v = 1681 \quad N_{TS} = 400$$



number of errors: 26/1681

Example 4: training set 1

$$\underline{v} = (x_1, x_2)$$

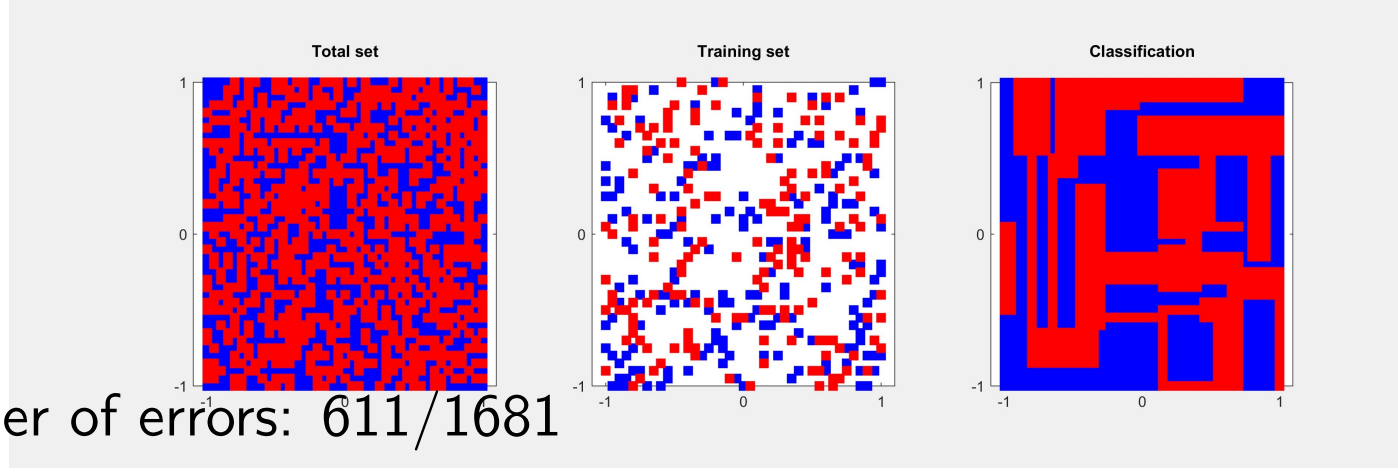
$$c = f\left(\underline{v}\right) = \operatorname{sign}\left(3x_1^3 x_2^3 h(x_1, x_2)\right)$$

where $h(x_1, x_2)$ is a 2-dimensional normal random variable with pdf

$$f_h(x_1, x_2) = \frac{1}{2\pi} \exp\left(\frac{-x^2 - y^2}{2}\right)$$

$$x_1 = -1 : 0.05 : 1 \quad x_2 = -1 : 0.05 : 1;$$

$$N_v = 1681 \quad N_{TS} = 400$$



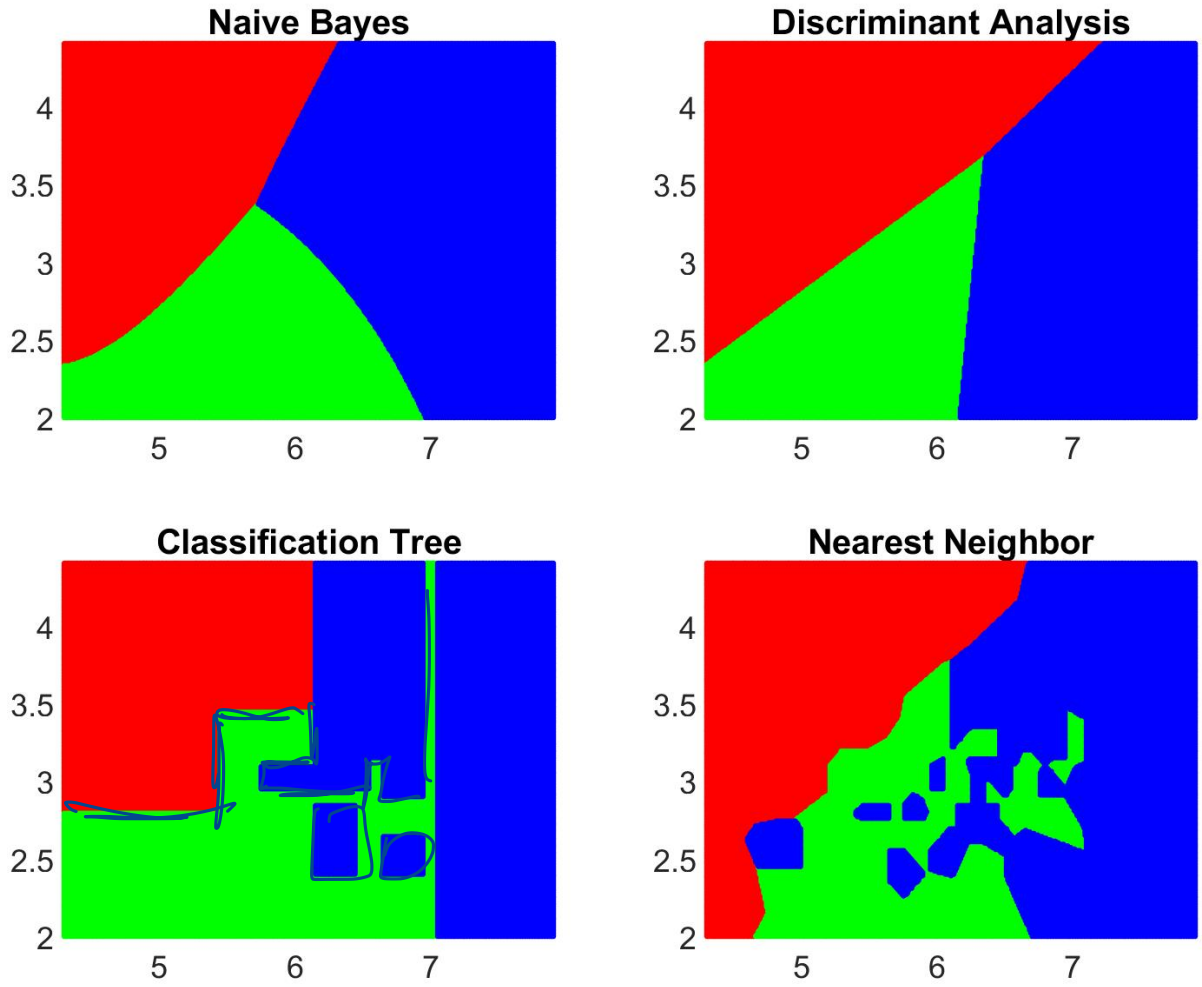
number of errors: 611/1681

Problem

For numerical features, this Classifier divides the space $A_v = \mathbb{R}^M$ into **hyper-rectangles** of dimension M defined as
$x_{1,inf} < x_1 < x_{1,sup}, x_{2,inf} < x_2 < x_{2,sup} \ldots x_{M,inf} < x_M < x_{M,sup}$.

For each of these hyperrectangles, there is one and only one class. As a consequence, the set of vectors which is mapped by the classification function g into a given class is always the union of hyper-rectangles. If the true set mapped by the original function f into the same class has a different shape we can only approximate it.

Comparison


Naive Bayes
Discriminant Analysis
Classification Tree
Nearest Neighbor
4
3.5
3
2.5
2
5
6
7

Decision Tree Ensembles

To improve the performance of tree classifiers: Tree Ensembles

Main idea: given the training data-set, instead of building a single classification tree we build many and we combine their decisions.

Random forest approach

Bagging: starting from the training data-set T_S we build N bootstrapped data-sets made by randomly extracted vectors of T_S.

Random feature selection: For each bootstrapped data-set, we build a classification tree. When doing this, each time we process a subset to build a node, we only use $M' \leq M$ randomly extracted features.

Decision: given a new vector, we process it with all the N tree classifiers and we assign the most popular class among the N results.

(To decide the number M' of features, we usually start from a value $M' \simeq \sqrt{M}$. Then we test the accuracy of the forest by using the out-of-bag vectors (the vectors which are outside the different bootstrapped data-sets). We repeat the procedure by increasing and decreasing the values of M' looking for the value with best accuracy.)

Information Theory for Data Science

Assignment 1

Introduction to Information Theory and application to Classifiers

Final version 1.1

Exercises:

1. Renyi Entropy of a binary random variable (pt. 6)
2. Application of the principle of maximum entropy (pt. 6)
3. Permutation Entropy for Time Series Anomaly Detection (pt. 6)
4. Kullback Leibler divergence from empirical distribution (pt. 6)
5. Information Gain Ratio and Tree Classifiers (pt. 6)

Exercise 1 - Entropy of a binary random variable

Exercise 1.A

1. Plot the entropy of a binary random variable
2. Discuss the result

Exercise 1.B – Renyi entropy

$$H_\alpha(X) = \frac{1}{1-\alpha}\log_2\left[\sum_i p_i^\alpha\right] \qquad \alpha \geq 0 \qquad \alpha \neq 1$$

3. Prove that the limit for $\alpha \to 1$ is the Shannon entropy.

4. Repeat point 1 with different values of α (smaller and bigger than 1)

5. Comment the results

Exercise 1.C

- Prove that this averaging operation $\{p, p, p_3\}$ $p = \frac{p_1 + p_2}{2}$ always increases the entropy

Hint: use the log inequality

Exercise 2 - Application of the principle of maximum entropy

Exercise 2

1. Invent an exercise where you have a random variable X with alphabet Ω_X where each outcome has a given “cost”.
2. Fix the mean value bigger than the arithmetic average of the costs, and apply the principle of maximum entropy to find the probability distribution P(X)
3. Plot P(X)
4. Repeat with a mean value equal to the arithmetic average and plot the result
5. Repeat with other values of the mean value and plot the results
6. Comment the results

You must numerically solve the equation generated by the Lagrange optimization.

As an example , for Matlab you can use

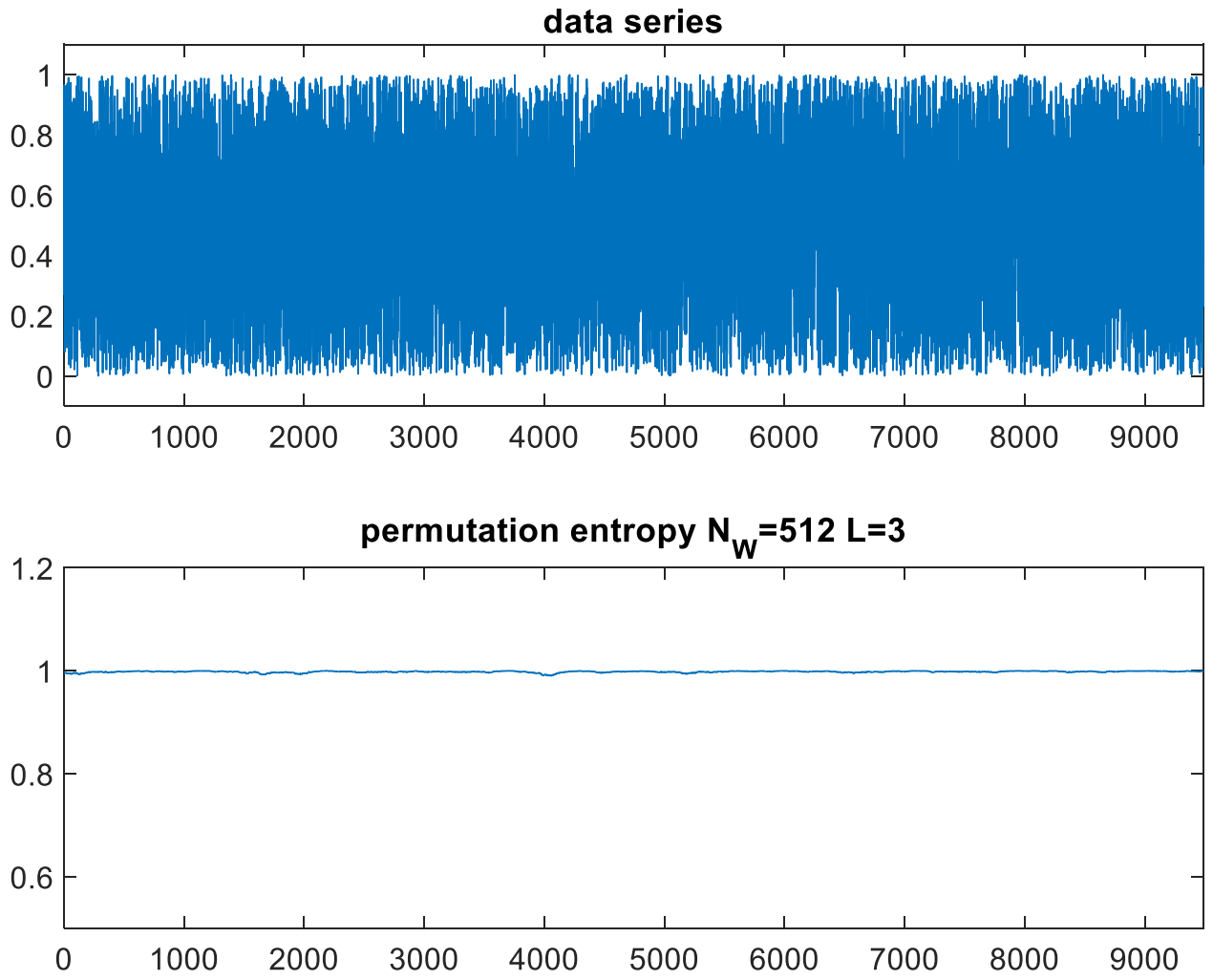
```
syms x
eqn = ( . . .  )*mu == ( . . . );
V = vpasolve(eqn,x,[0 10])
```

Exercise 3 - Permutation Entropy for Time Series Anomaly Detection

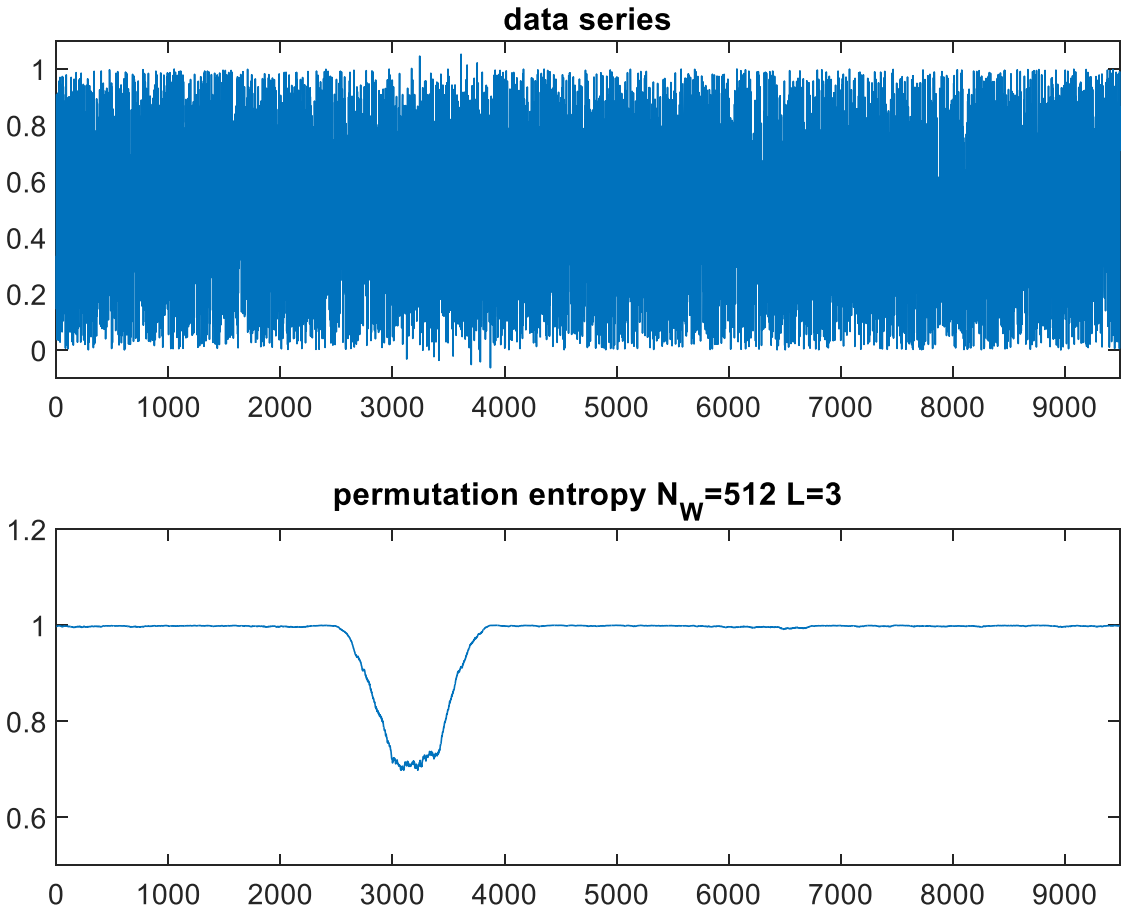
Exercise 3

Write a program that:

- Generates a data series made by 10,000 random symbols.
- Computes the permutation entropy with a sliding window of $N_W = 512$ symbols. (For example, choose an order m = 3.)
- Plot the data and the entropy.


data series
1
0.8
0.6
0.4
0.2
0
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
permutation entropy N_W=512 L=3
1.2
1
0.8
0.6
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

- Insert a pattern between 3000 and 4000 made by correlated data (with about the same mean value and variance of the original random data).
- Describe how you generated the correlated data
- Compute the permutation entropy with the same N_W and m used before.
- Plot the data and the entropy.
- Comment the results


data series
1
0.8
0.6
0.4
0.2
0
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
permutation entropy N_W=512 L=3
1.2
1
0.8
0.6
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

Some Matlab functions that might be useful

```
sort
perms
```

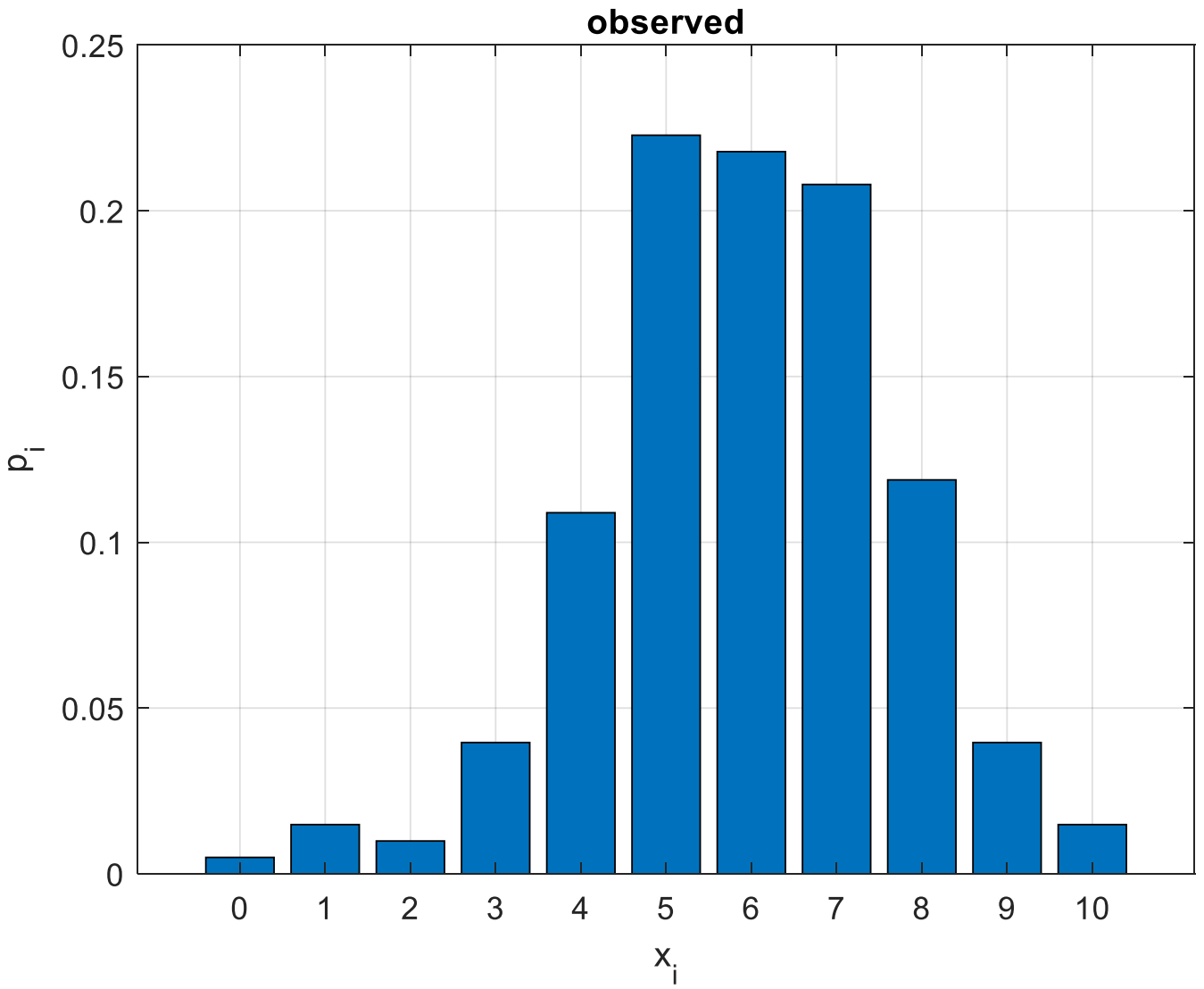
Exercise 4 - Kullback-Leibler distance from empirical distribution

Exercise 4

Consider the observed data

$$x_i = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{bmatrix}$$

$$\mathrm{NUM}\left(x_i\right) = \begin{bmatrix} 1 & 3 & 2 & 8 & 22 & 45 & 44 & 42 & 24 & 8 & 3 \end{bmatrix}$$


observed
0.25
0.2
0.15
0.1
0.05
0
p_i
0
1
2
3
4
5
6
7
8
9
10
x_i

1. Compare with uniform pmf
2. Write in the title the KL divergence value

Compare with binomial pmf with $0 < p < 1$ (step=0.001)

3. Identify the pmf at minimum KL divergence
4. Plot the pmf
5. In the title write the value of p and KL divergence

Exercise 5 – Information Gain and Tree Classifiers

Exercise 5

Given this Training Set

	x_1	x_2	x_3	c
v_1	30	0	10	0
v_2	30	0	70	0
v_3	30	1	20	0
v_4	30	1	80	1
v_5	60	0	40	0
v_6	60	0	60	1
v_7	60	1	50	0
v_8	60	1	60	1

Given this Training Set

Write a program that:

- Builds a tree classifier based on the information gain ratio according to the C4.5 rules for this Training Set (you do not need to write a program able to process any Training set, it is enough if it works for this TS with these three features).
- Accepts a vector $x= (x_1, x_2, x_3)$ as input (from keyboard or file or graphical interface) and outputs the corresponding class.

In the report, plot the tree classifier and present at least 4 examples of class computation.

Important

Assigned on 17/10/2023

Delivery by

- 01/11/2023, 11.59 PM: +2 points
- 08/11/2023, 11.59 PM: +1 point
- 15/11/2023, 11.59 PM: 0 points
- **Later: not accepted**

Version 1.1

Slide 13: N_R changed into m
Slide 18: figure changed