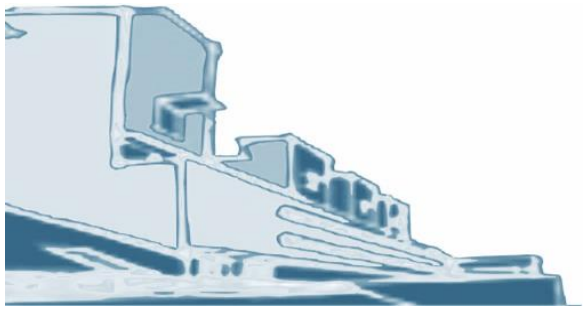
AC – Aprendizagem Computacional / Machine Learning

P1 – Introduction

Jorge Henriques
jh@dei.uc.pt

Departamento de Engenharia Informática
Faculdade de Ciências e Tecnologia



1 2 9 0
UNIVERSIDADE D COIMBRA

dei engenharia informática

Contents

1| Objectives

- Languages / Tools
  - Python 
  - Matlab 

Numpy / Pandas / Matplotlib

- **Python**
  - It is assumed that students are familiar with Python and libraries
- **NumPy**
  - Functionalities for scientific computing
  - Arrays, Vectors and Matrices, Matrix operations
  - Numerical calculus
- **Pandas**
  - Data Access
  - Data manipulation, operations
- **Matplotlib**
  - Visualization

Scikit-Learning

- **Traditional Machine Learning models**
  - Uses NumPy, SciPy, Matplotlib, Pandas
  - Data-science related tasks
  - Training/test split
  - Greater level of abstraction for ML algorithms
  - Learning without hardware (GPU) acceleration
  - Works well work relatively small datasets

- Keras/TensorFlow
  - **Deep learning libraries**
  - **Keras**
    - High-level deep learning library
    - Runs on top of Tensorflow
  - **TensorFlow**
    - Low-level deep-learning
    - Development of deep learning methos with hardware acceleration

- **Practical works**
  - One work – two classes

  - 1. Introduction to ML
  - 2. Clustering techniques
  - 3. Decision Trees

  Part A

  - 4. Neural Networks
  - 5. Advanced NN (Deep)
  - 6. Fuzzy systems

  Part B

Contents

Review Numpy / Pandas

- Load a data set (tabular)
- Perform some operations on the data (pre-processing)
- Build a simple model (classification)
- Evaluate the performance of the model (classifier)

Goal

- To develop a risk model, applicable to artery coronary syndrome (ACS) patients
- Patients have been admitted to the emergency unit with an episode of myocardial infarction (MI)

- The model should be able to predict if a new event rehospitalization will occur in the next 30 days

{0, 1}={No,Yes}

Contents

- Risk assessment of a new event

Gender
Age

Heart rate
Blood pressure
ST segment deviation

Creatinine
Guidelines

Classification

New event ?

{ 0, 1 } = { No, Yes }

2| Dataset

- Risk assessment of a new event

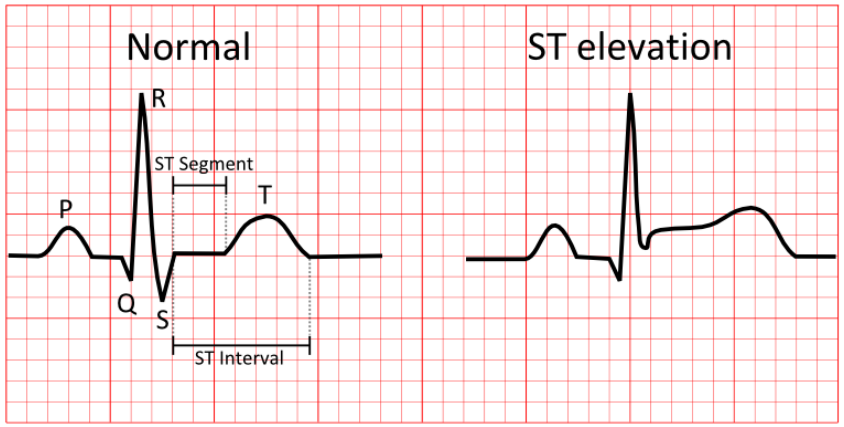
X - INPUTS

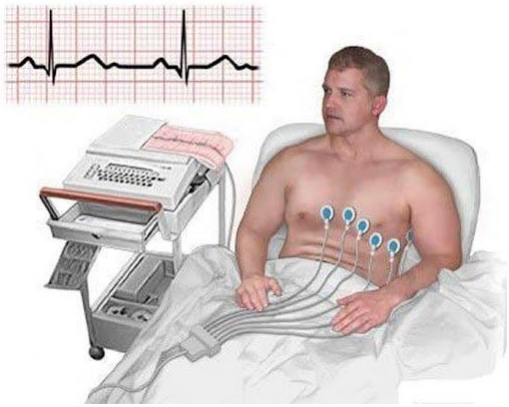
- 1| Historical
  - AG |Age | [33 .. 90]
  - GD |Gender | { female, male } = { 0, 1 }

- Risk assessment of a new event

X - INPUTS

- 2| Measurements
  - **SBP** | Systolic blood pressure | [60 .. 221]
  - **HR** | Heart rate | [41 .. 151]
  - **ST** | ST segment deviation (ECG) | {0,1}





- Risk assessment of a new event

X - INPUTS

- 3| Exams/diagnosis

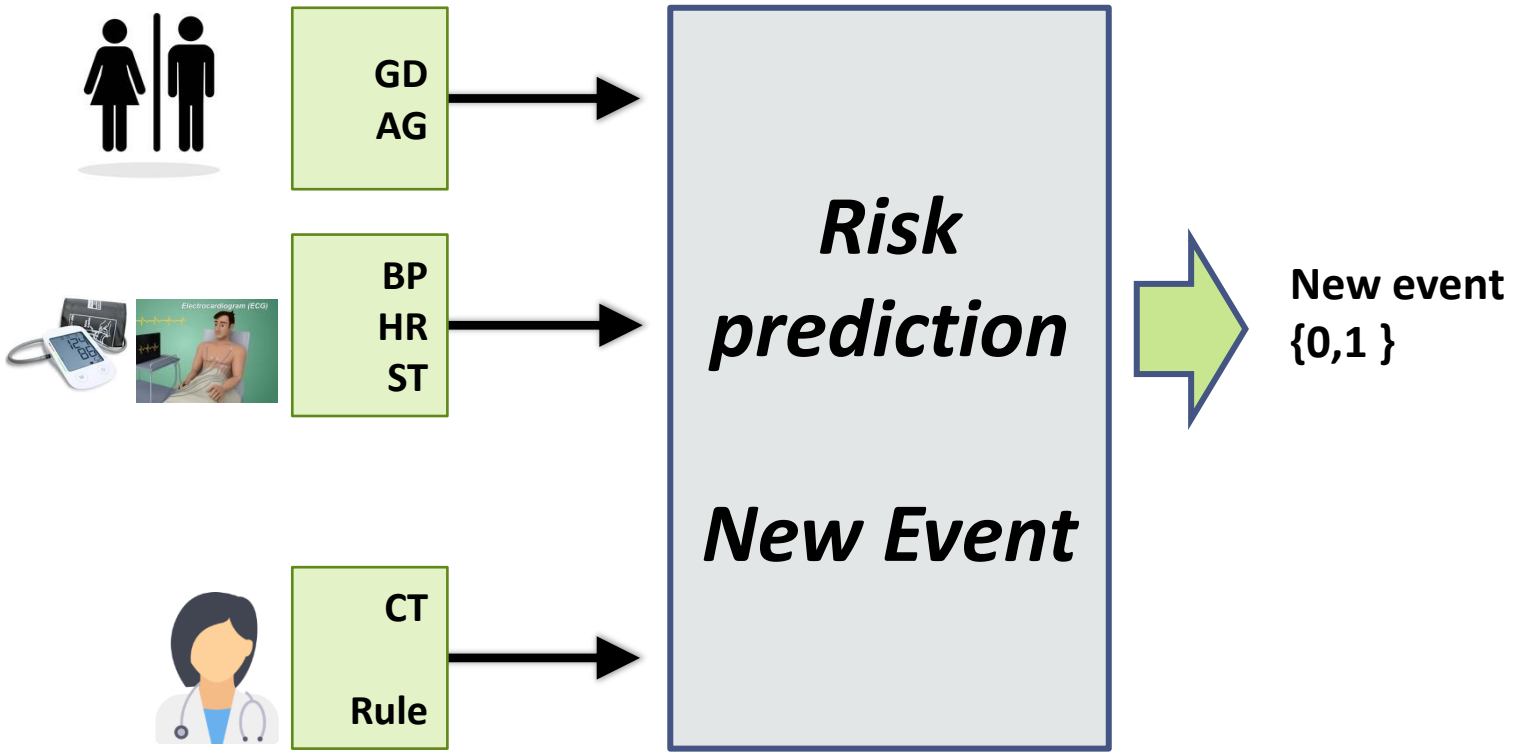
  - CT |Creatinine | [0.6 .. 11.5]
  - CT - Blood test - measure how well your kidneys are working.

- 4| Guidelines

Clinical guidelines

If **CT>=1.3 AND ST=1 event=1**

- Model
  - Historical, measurements, clinical knowledge (guidelines)



- **DATASET : cardiacRisk.csv**
  {'Gender', 'Age', 'SBP', 'HR', 'ST', 'CRT', 'EVENT' }

X1	***Gender***	{0,1} = { Female, Male}
X2	***Age***	[31, 91]
X3	***Systolic Blood pressure***	[60, 221]
X4	***Heart rate***	[41, 151]
X5	***ST deviation***	{0, 1} = { No, YES}
X6	***Creatinine***	[0.6, 11.5]
T	***Target=event***	{0,1} = { No event, Event }

Contents

3| Tasks

- Tasks

  - 1| Load the data file
  - 2| Data pre-processing
  - 3| Data analysis / visualization
  - 4| Build a classification model
  - 5| Evaluate the performance of the classifier
  - 6| ??other improvements

- 1| Load the data file

```
df  = pd.read_csv('cardiacRisk.csv')
D   = df.values
X   = D[:,0:6]          # inputs
T   = D[:,6]            # Target
N   = X.shape[0]        # number of patients
```

- 2| Pre-processing data
  - For some patients, the **value of Age is missing**
  - The value (-1) is used to represent a missing value
  - Replace missing values by the average of the others (with a valid Age)

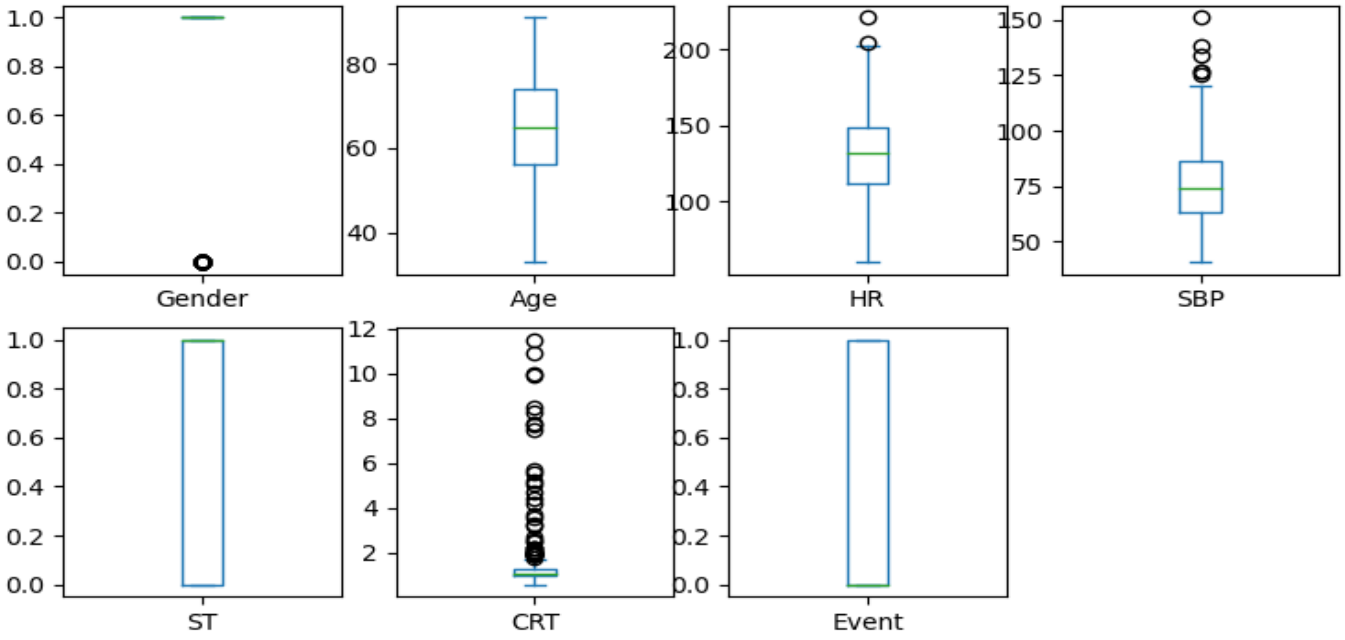
- 3| Analyse data / visualization
  - Data characteristics

```
print(df.describe())
```

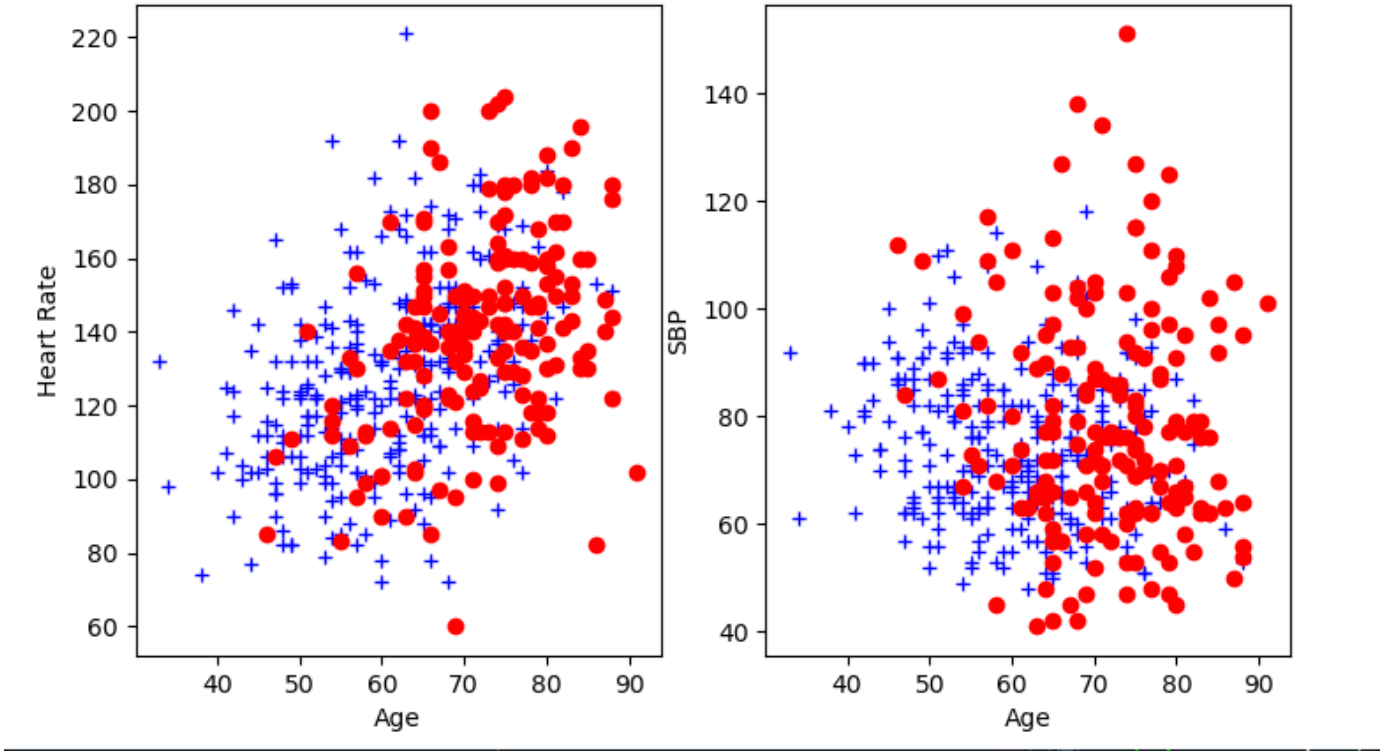
	Gender	Age	HR	...	ST	CRT	Event
count	457.000000	457.000000	457.000000	...	457.000000	457.000000	457.000000
mean	0.787746	64.693654	131.783370	...	0.531729	1.379431	0.391685
std	0.409352	11.198284	26.692102	...	0.499539	1.266251	0.488662
min	0.000000	33.000000	60.000000	...	0.000000	0.600000	0.000000
25%	1.000000	56.000000	112.000000	...	0.000000	1.000000	0.000000
50%	1.000000	65.000000	132.000000	...	1.000000	1.100000	0.000000
75%	1.000000	74.000000	148.000000	...	1.000000	1.300000	1.000000
max	1.000000	91.000000	221.000000	...	1.000000	11.500000	1.000000

- 3| Analyse data / visualization
  - Data analysis

```
df.plot(kind='box', subplots=True, layout=(2,4),
sharex=False,sharey=False)
```



- 3| Analyse data / visualization
  - Plot (age, heart rate), (age, sbp)
    - For patients NO event (+) / Event (●)



3| Tasks

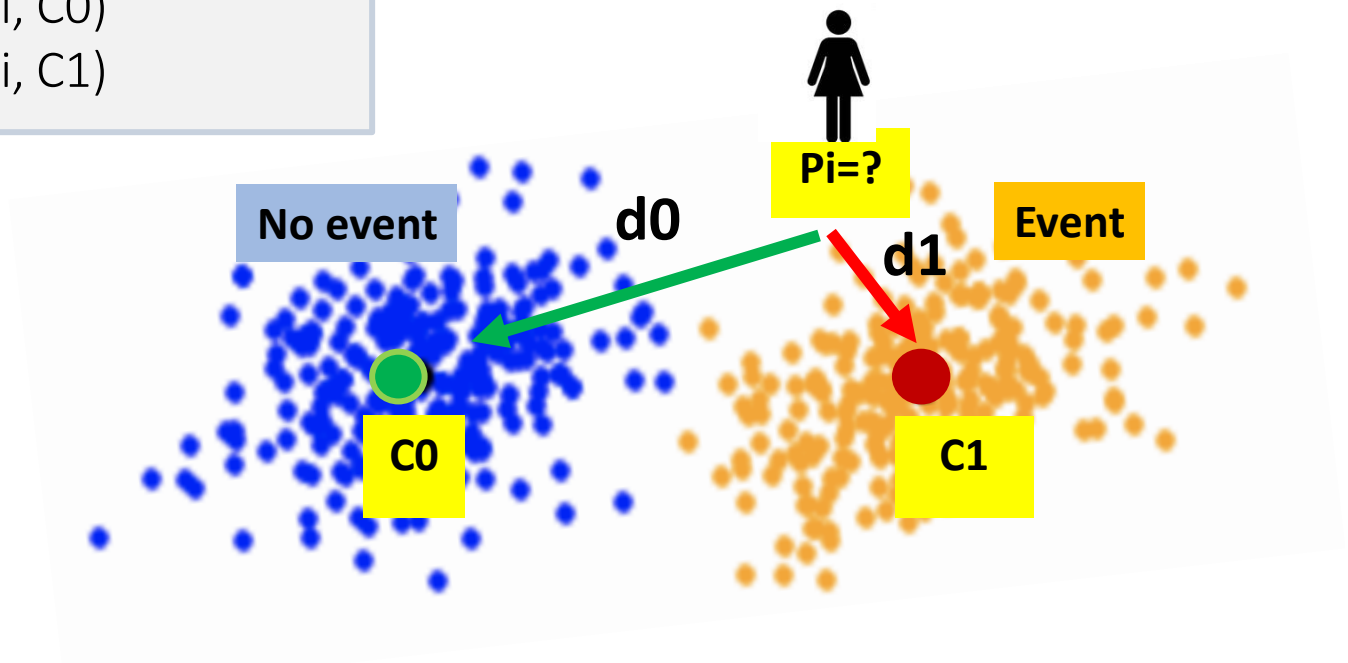
- 3| Analyse data / visualization
  - Any other idea ??

- 4| Build a classification model
  - C0 - Virtual patient (no event) – mean of patients with attributes {T=0}
  - C1 - Virtual patient (event) – mean of patients with attributes {T=1}

d0 < d1 then **NO event**

d0 > d1 then **event**

d0=distance(Pi, C0)
d1=distance(Pi, C1)



- 5| Evaluate the classifier's performance
  - Compute sensitivity (SE)
  - Compute specificity (SP)

		Target / Actual	
		Event = 0	**Event =1**
Model Estimates	Event = 0	**TN**	**FN**
	Event = 1	**FP**	**TP**

$$SE = \frac{TP}{TP + FN} \qquad SP = \frac{TN}{TN + FP}$$

- 6| Conclusions

  - Use of numPy and/or Pandas
    - Read a tabular data (*.csv)
    - Access to a specific data subset
    - Perform some data operations / visualization
    - Introduction to machine learning:
      - Pré-processing
      - Classification model
      - Evaluation

  - Improvements ?
    - ?? Any other idea ?