



MACHINE LEARNING

MEI/1

University of Beira Interior,
Department of Informatics

Hugo Pedro Proença,
hugomcp@di.ubi.pt, 2024/2025



Machine Learning

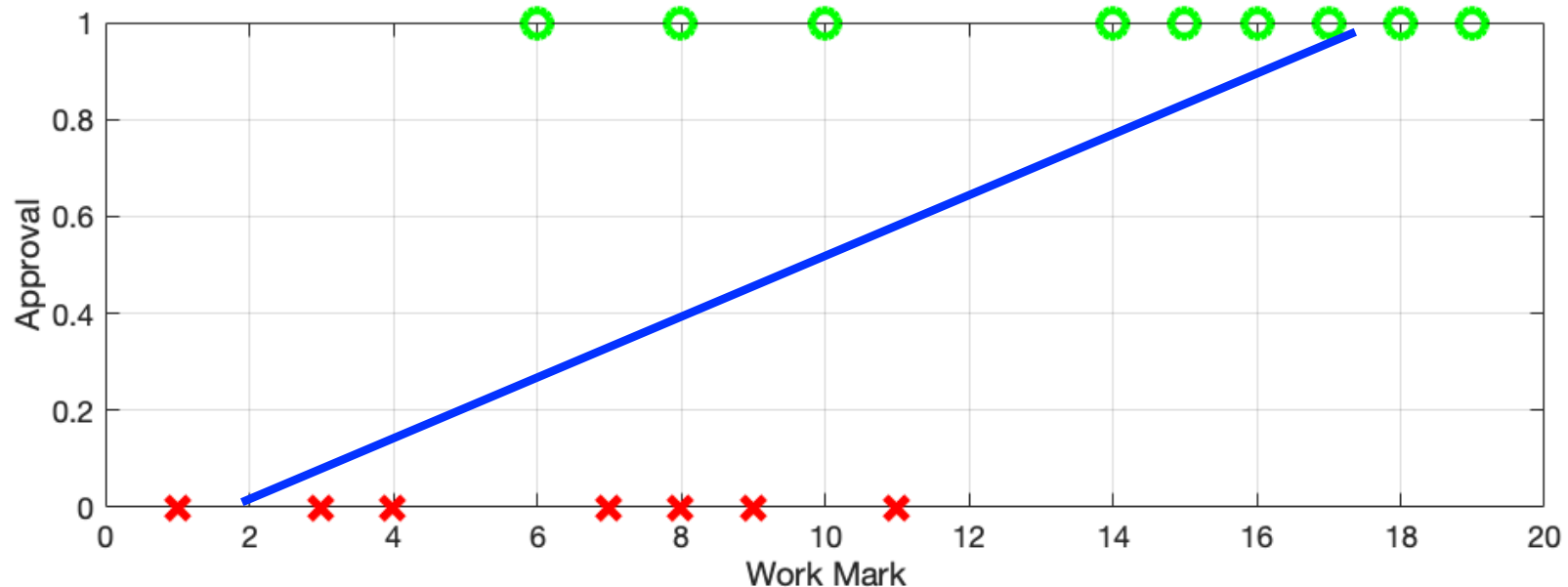
[03]

Syllabus

- Supervised Learning
- Logistic Regression

Students Performance

- Suppose that we are interested in predicting the **approval rate** of a class, based on the **students marks in the first practical work**.
 - Typically, students that get good marks in the first work, got approved at the course.
 - Students with very low marks at the first work tend to fail in the final examination.
- Hence, our machine learning model is expected to predict a **binary outcome** (1: pass vs. 0: fail)



Students Performance

- In this kind of problems, the dependent variable assumes a reduced set of labels:
 - Emails: “is this a **spam** or **no spam** email”? $y \in \{0, 1\}$
 - Medical diagnosis: “is the patient **ill** or **healthy**” $y \in \{0, 1\}$
 - How will be the weather tomorrow?: “will it be **sunny**, **cloudy** or **rainy**”? $y \in \{0, 1, 2\}$
- In this case, a best fitting line is not enough
 - Even though this line will be the basis of our **classification model**

$$h_{\theta}(x) = \theta_1 \cdot x + \theta_2$$

- In this kind of problems, the dependent variable assumes a reduced set of labels:
 - Emails: “is this a **spam** or **no spam** email”? $y \in \{0, 1\}$
 - Medical diagnosis: “is the patient **ill** or **healthy**” $y \in \{0, 1\}$
 - How will be the weather tomorrow?: “will it be **sunny**, **cloudy** or **rainy**”? $y \in \{0, 1, 2\}$
- In this case, a best fitting line is not enough
 - Even though this line will be the basis of our **classification model**

Logistic Regression: Classification

- The obvious idea will be to define a threshold at the classifier output $h_\theta(x)$, that binarizes the system response:
 - Typically, “0.5” would be the choice, for “*equal classification risks*”
 - It might be more dangerous to predict erroneously one class instead of other one.
 - For example, in a machine learning-based systems for medical diagnosis, classes have different risk.
 - Predict a “**malignant cancer**” on a “**healthy**” subject represents a unnecessary concern for the patient and would probably imply to perform additional (an unnecessary) exams.
 - However, provide a “**healthy**” response for a patient suffering of a “**malignant cancer**” might represent the patient dead sentence.
 - $f(x) = \begin{cases} 0, & h_\theta(x) < 0 \\ 1, & h_\theta(x) \geq 0 \end{cases}$
- Hence, the response of our classification system can be seen as a composition of two functions: $f = g \circ h$

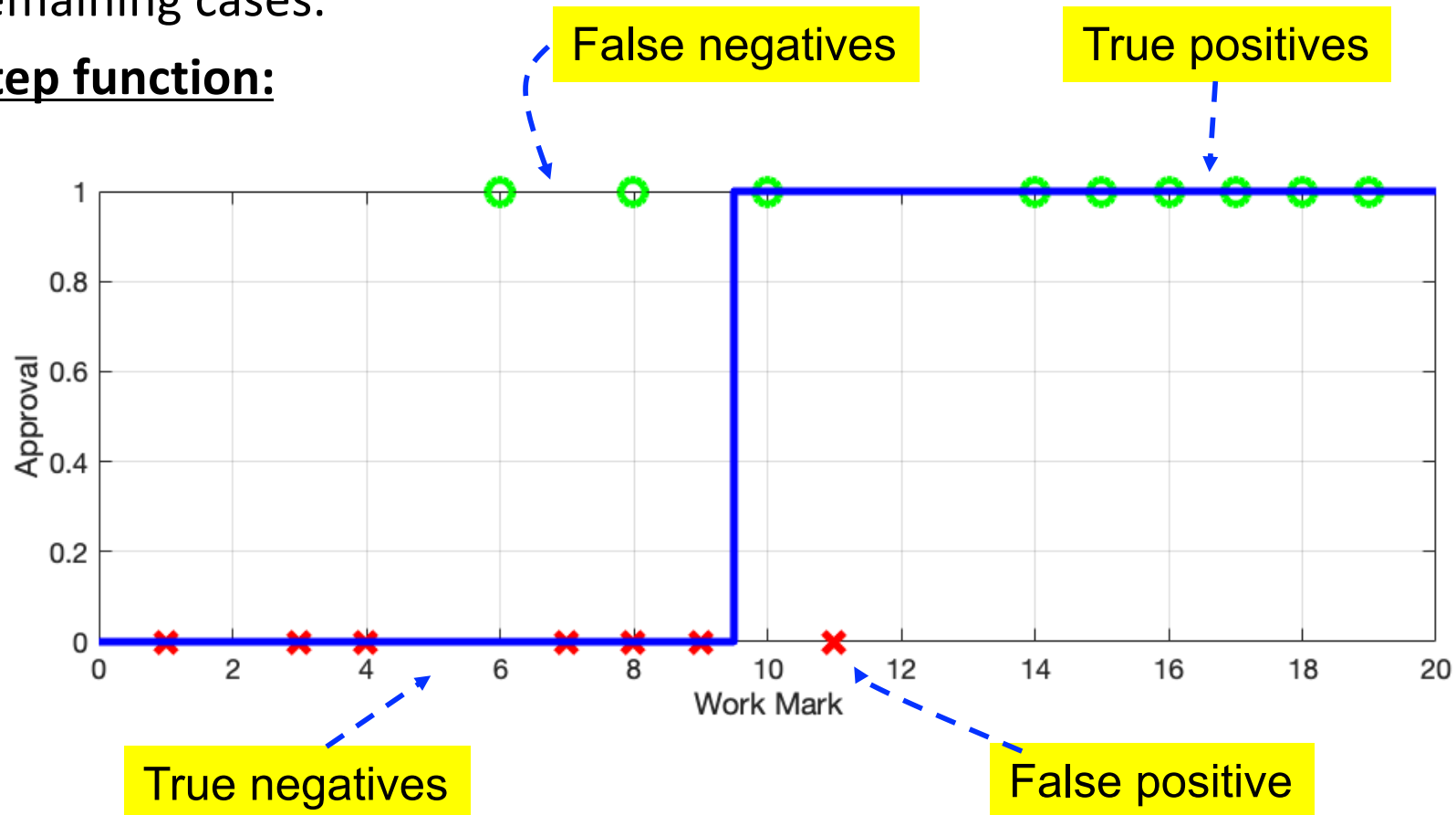
“f” is “g” after “h”

- *We have seen “h” before, but what is “g”?*

Logistic Regression: Classification

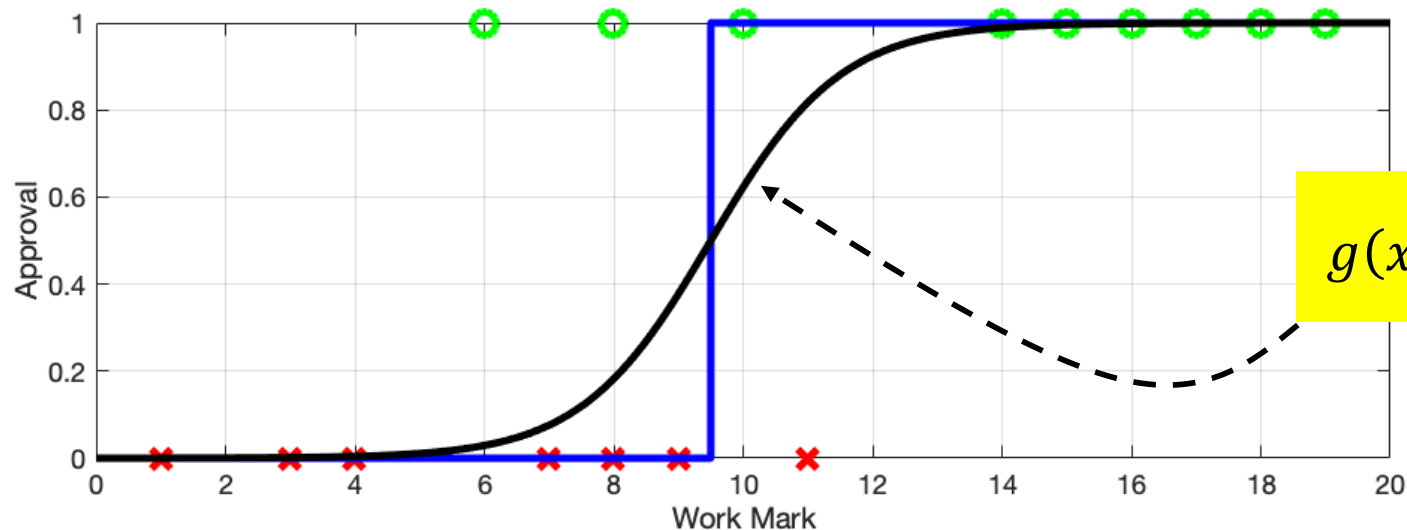
- Essentially, “g” performs a binarization of its input, and produces “1” responses when the input is higher than some threshold, and “0” in the remaining cases.

- Step function:**



Logistic Regression: Classification

- Assuming the step function as “g”, and $f = g \circ h$, obtaining the automatic optimal parameterization of “f” with respect to our data (i.e., machine learning) yields two problems:
 - **Problem 1:** “g” is **not differentiable**
 - It has not a continuous derivative at a single point
 - **Problem 2:** in every other points “g” **has derivative 0**
- The solution is to use a function is close to the step function, without suffering of the above described problems.
 - **Sigmoid Function**



Logistic Regression: Classification

- Using this composition of functions, our classification system is given by:

$$f_{\theta}(x) = \frac{1}{1 + e^{h_{\theta}(x)}}$$

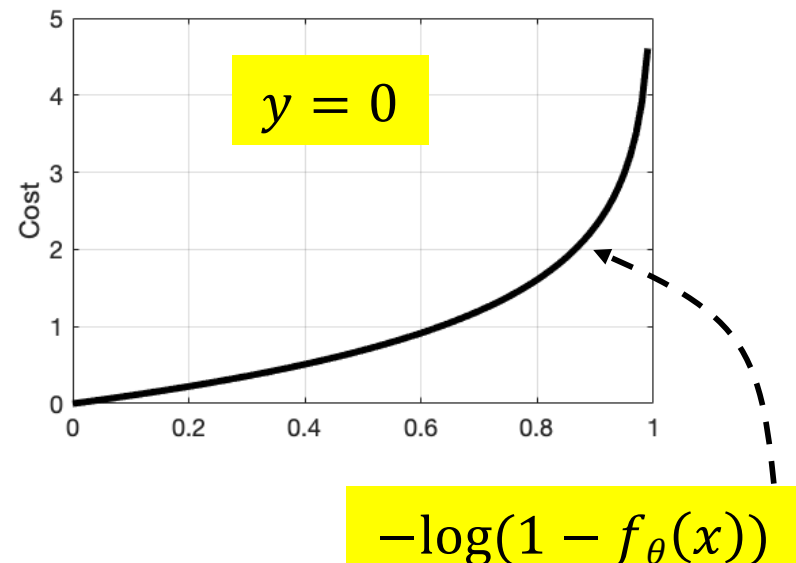
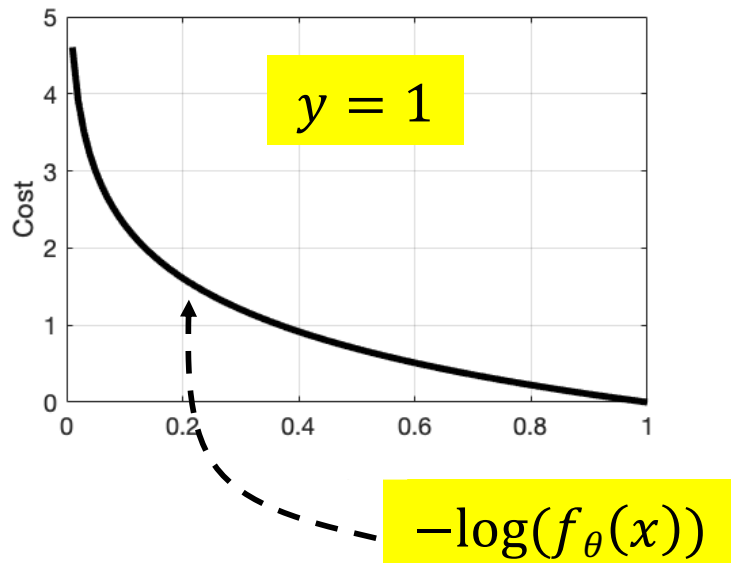
i.e.,

$$f_{\theta}(x) = \frac{1}{1 + e^{\theta_1 x + \theta_2}}$$

- The remaining problem is the same as in linear regression:
 - How to find the θ optimal parameterization?
- According to the basic principles of Machine Learning, up to now we've only defined our model.
 - It is also required to define a "Cost Function" (Loss function) that measures how good it is na hypothesis.
 - ...And a systematic optimization process.

Binary Classification: Cost Function

- As previously, the cost function will measure how well the model responses ($f_{\theta}(x)$) resemble the “ground-truth” (y)
 - Intuitively, in cases where the system is supposed to output a “1” and the model predicts a “1”, the cost should be “0”.
 - The same thing should hold for “0” responses.
 - However, the cost (loss) should grow in cases when the system response is far from the ground-truth.
 - The $\log()$ function is a good choice for representing the desired costs (losses)
 - It varies non-linearly with respect to the distance between the desired and actual responses
 - Attempts to avoid “**very wrong responses**”.



Binary Classification: Cost Function

- Hence, the cost function for one instance is given by:

$$\bullet \text{ Cost}(f_{\theta}(x), y) = \begin{cases} -\log(f_{\theta}(x)), & y = 1 \\ -\log(1 - f_{\theta}(x)), & y = 0 \end{cases}$$

- And the cost function for the whole dataset is given by the sum of the individual costs:

$$\mathbf{J}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^N (\text{Cost}(f_{\theta}(x^{(i)}), y^{(i)}))$$

- Considering that y can only assume 2 values (0 or 1), we have:

$$\mathbf{J}(\boldsymbol{\theta}) = -\frac{1}{N} \sum_{i=1}^N y^{(i)} \log(f_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - f_{\theta}(x^{(i)}))$$

Logistic Regression: Optimization

- The optimization can be done exactly as in the linear regression case.
- Using the gradient descent strategy, it is required to find the derivatives of the cost function $J()$ with respect to the θ parameters:

$$\frac{\partial}{\partial \theta} J(\theta)$$

- In matrix form, we have:

$$\theta = [\theta_0, \theta_1]^T$$

$$x^{(i)} = [x^{(i)}, 1]^T$$

- $f_{\theta}(x) = \frac{1}{1+e^{-\theta^T x}}$

- $\log(f_{\theta}(x)) = \log\left(\frac{1}{1+e^{-\theta^T x}}\right)$
 $= -\log\left(\frac{1+e^{-\theta^T x}}{1}\right)$

- $\log(1 - f_{\theta}(x)) = -\theta x - \log\left(\frac{1+e^{-\theta^T x}}{1}\right)$

Logistic Regression: Optimization

- Plugging the two simplified expressions in the original cost function, we obtain:

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N -y^{(i)} \log(1 + e^{-\theta x}) + (1 - y^{(i)}) (-\theta x - \log(1 + e^{-\theta x}))$$

- Which can be simplified to:

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N y^{(i)} \theta x - \log(1 + e^{-\theta x})$$

- Now, as

$$\frac{\partial}{\partial \theta_j} y^{(i)} \theta x = y^{(i)} \theta x$$

$$\frac{\partial}{\partial \theta_j} \log(1 + e^{\theta x}) = \frac{x_j e^{-\theta x}}{1 + e^{\theta x}} = x_j^i f_{\theta}(x)$$

- We have:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \sum_{i=1}^N x_j^i (f_{\theta}(x^{(i)}) - y^{(i)})$$

Logistic Regression: Multi-class

- Up to now, we've only considering binary classification problems.
- When the number of classes $c \geq 2$, the typical approach is to train "c" classifiers
 - In each classifier $f_{\theta}^{(i)}(x)$, instances of the i^{th} class are considered positive examples, whereas instances of all the remaining classes are regarded as negative instances.
- During classification, we pick the class that produces the maximum output response, i.e.:

$$\max_i f_{\theta}^{(i)}(x)$$

