Classical Data Analysis
Day 2

Master in Big Data & A.I. Solutions

BARCELONA TECHNOLOGY SCHOOL

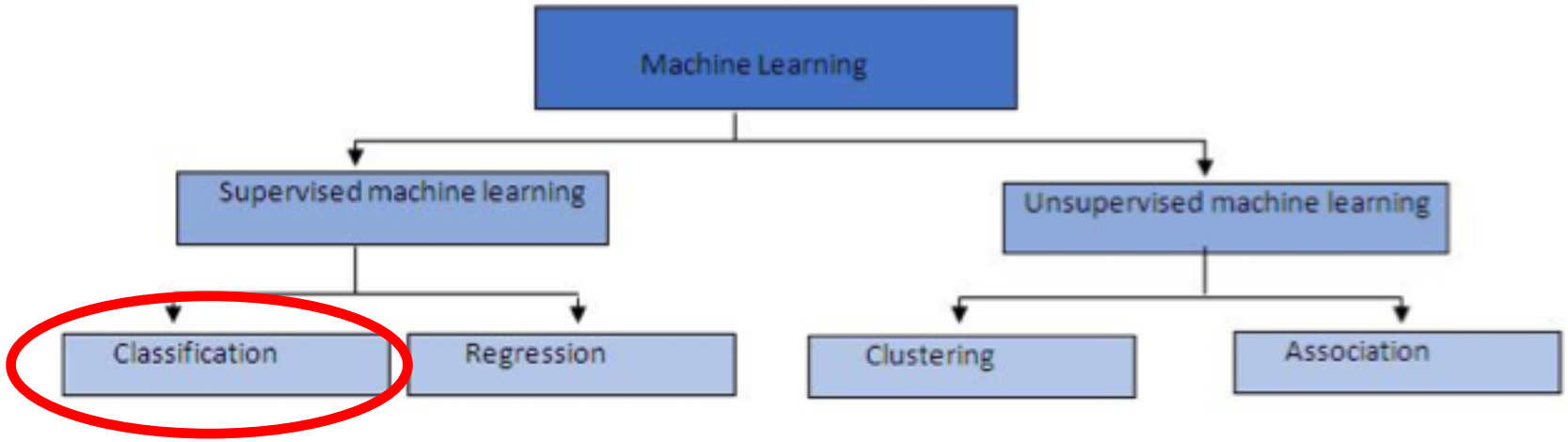
Trainer´s name: Miguel Sanz

BTS Email address: miguel.sanz@bts.tech

Today

- Introduction to data analysis
- Linear Regression
- **Logistic Regression**
- Regression analysis (Polynomial, Ridge, Lasso, etc.)
- Neural Networks (MLPs in depth)
- Support Vector Machines (SVM)
- Decision Trees
- Ensemble Methods (Random Forests, Bagging, etc.)
- Other Classifier (KNN, Naïve Bayes)
- K-Means
- PCA
- Hierarchical Clustering

Supervised learning: Regression



Logistic Regression

The main goal of logistic regression is to perform binary classification

Examples of applications:

- In medical research, in the field of predictive food microbiology, to describe bacterial growth/no growth interface, in the 0–1 format.
- Predict the risk of developing a given disease based on observed characteristics of the patient.
- In credit risk modelling in identifying potential loan defaulters.

All the above problems are binary classification problems.
Possible results:

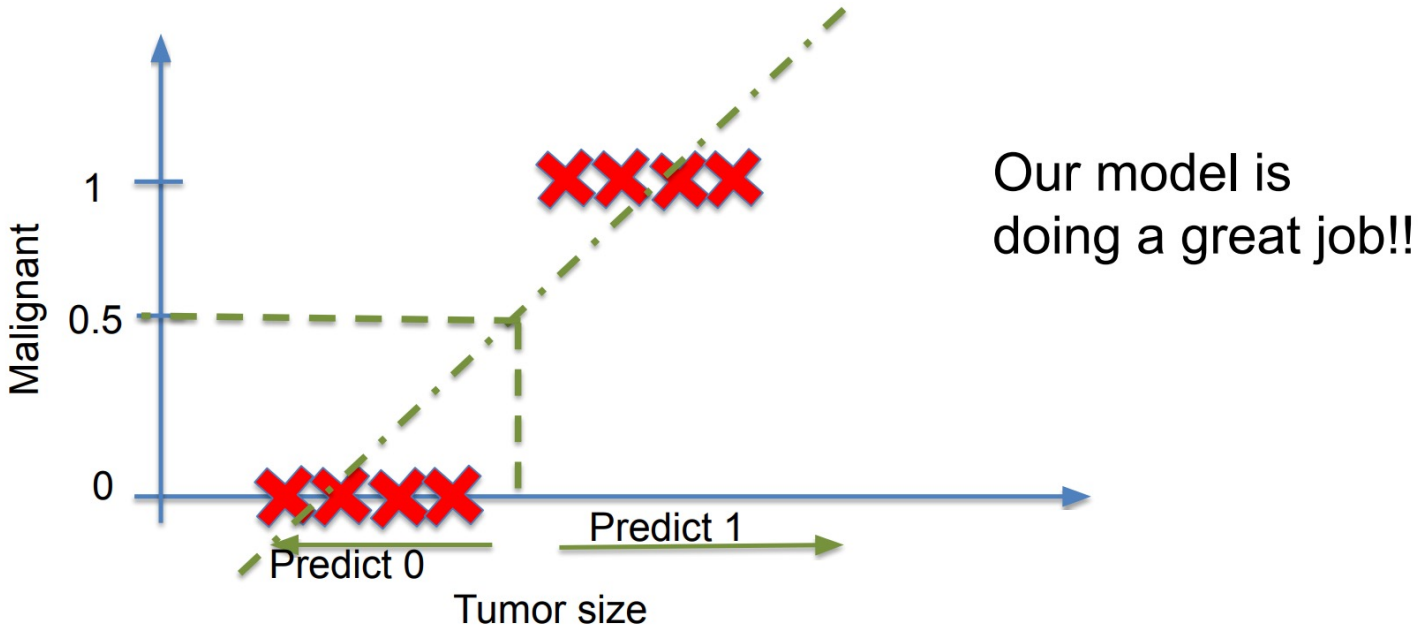
- y = 1
- y = 0

Feature Engineering plays an important role in regards to the performance of Logistic

Logistic Regression

How to develop a classification algorithm?

Let’s try applying a linear regression model to classify a tumor as malignant or not!

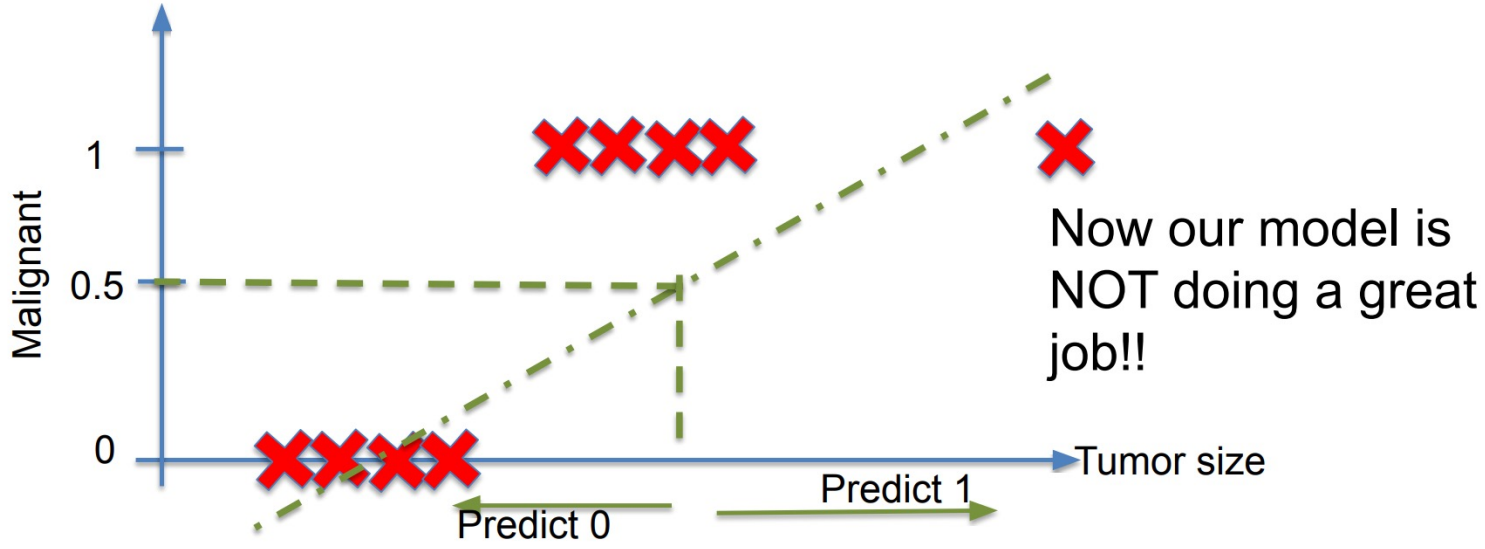


Logistic Regression

How to develop a classification algorithm?

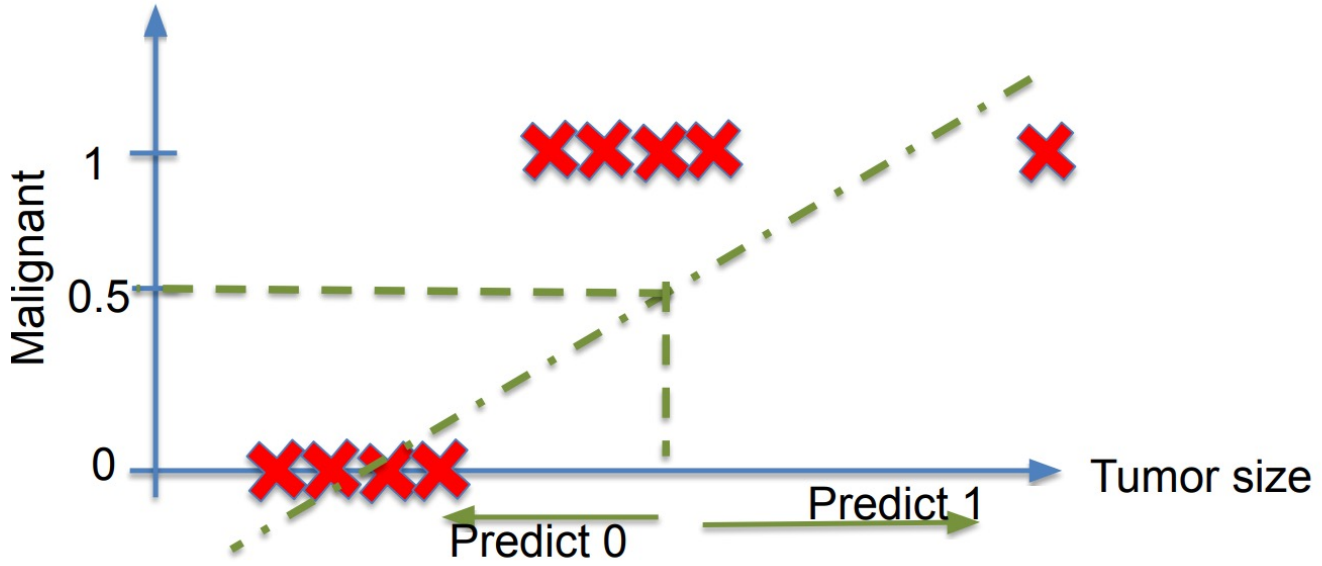
Let's try applying a linear regression model to classify a tumor as malignant or not!

Let'a add an additional example



Many positive examples are now classified as negative.

Logistic Regression



- Many positive examples are now classified as negative.
- It also doesn't make sense for hθ (x) to take values larger than 1 or smaller than 0 when we know that y ∈ {0, 1}.

Logistic Regression

Linear regression hypothesis function:

$$h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1 + \ \ldots + \theta_n x_n$$

In Logistic regression we want the output of the hypothesis function to be between 0 and 1. We apply the sigmoid function (aka logistic function) to the output of the linear regression, obtaining the hypothesis function for the logistic regression:

$$h_\theta(x) = g(\theta^T x) = g(\theta_0 + \theta_1 x_1 + \ \ldots + \theta_n x_n)$$

Logistic Regression is just like a linear regression model, it computes a weighted sum of the input features (plus a bias), but instead of outputting the linear result directly, it outputs the logistic of that result

Logistic Regression

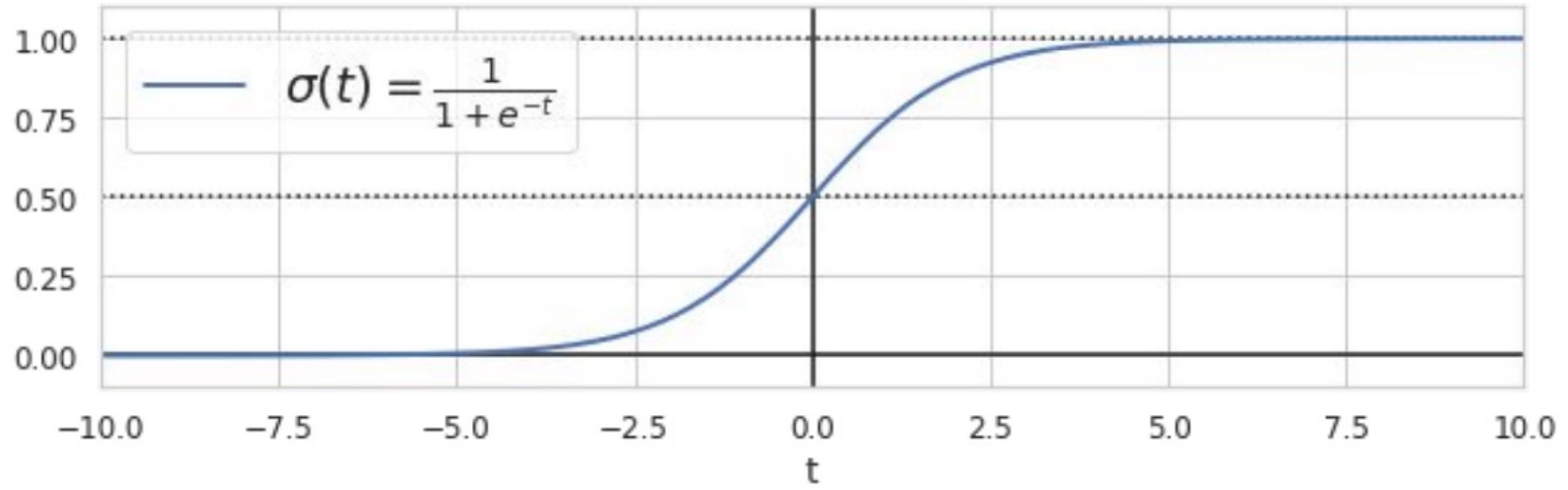
- Sigmoid function:

$$g(z) = \frac{1}{1 + e - ^{z}} \quad \Rightarrow \quad \frac{1}{1 + e^{-(\Theta^{T} X)}}$$

Logistic Regression

Sigmoid function:

$$g(z) = \frac{1}{1 + e - ^{z}} \quad \Rightarrow \quad \frac{1}{1 + e^{-(\Theta^T X)}}$$

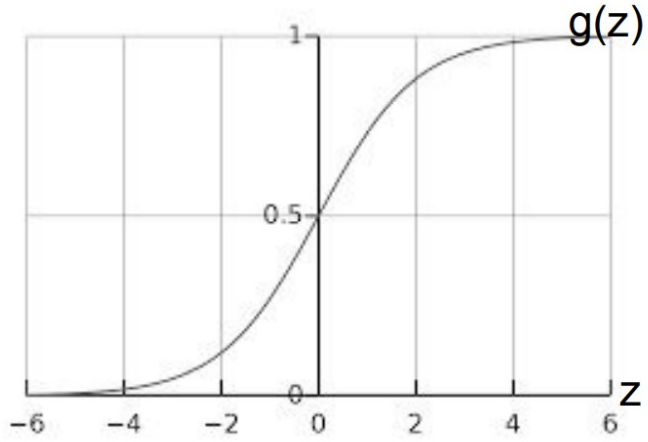


The logistic is a sigmoid function that outputs a number between 0 and 1

Logistic Regression

Logistic Regression hypothesis function:

$$g(z) = \frac{1}{1 + e - ^{z}}$$



When our hypothesis hθ (x) outputs a number, we treat that value as the estimated probability that y=1 on input x

Notice that g(z) tends towards 1 as z → ∞, and g(z) tends towards 0 as z → −∞. Moreover, g(z), and hence also h(x), is always bounded between 0 and 1.

Logistic Regression (aka Logit Regression)

- It is commonly used to estimate the probability that an instance belongs to a particular class
    - What is the probability that an email is spam?
    - What is the probability that I will pass this class?
- If the estimated probability is above 50%, the model is predicting that the instance belongs to that class (positive class, labelled 1)
- If the estimated probability is below 50%, the model predicts that the instance belongs to the other class (negative class, labelled 0)

Making predictions is easy…

$$\hat{y} = \begin{cases} 0 \rightarrow \hat{p} < 0.5 \\ 1 \rightarrow \hat{p} \geq 0.5 \end{cases}$$

Logistic Regression

Cost function

Alright, we know now that Logistic Regression estimates probabilities and makes predictions out of it but…

How is it trained?

The objective of the training is to set the parameter vector hθ() so that:

- Estimates high probabilities for positive instances (y=1)
- Estimates low probabilities for negative instances (y=0)

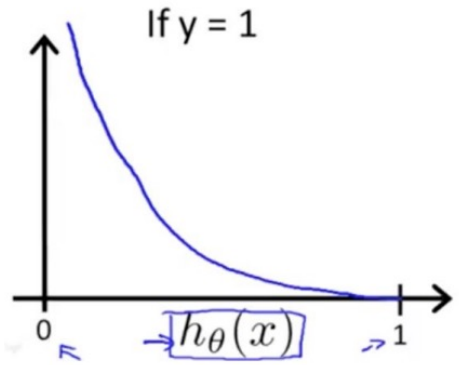
$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

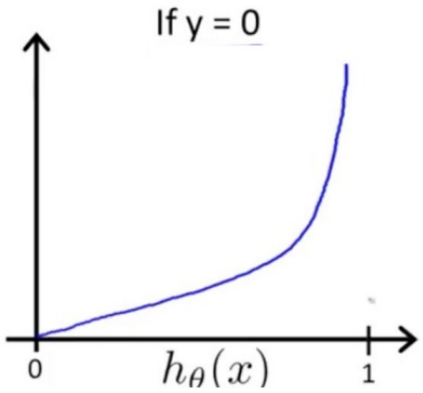
Logistic Regression

Cost function

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

- In this example for a single instance (x), -log(hθ(x)), grows very large when hθ(x) approaches 0
  - ✓ So the cost will be very large if the model predicts a probability close to 0 for a positive instance
- -log(1- hθ(x)), grows very large when hθ(x) approaches 1
  - ✓ So the cost will be very large if the model predicts a probability close to 1 for a negative instance



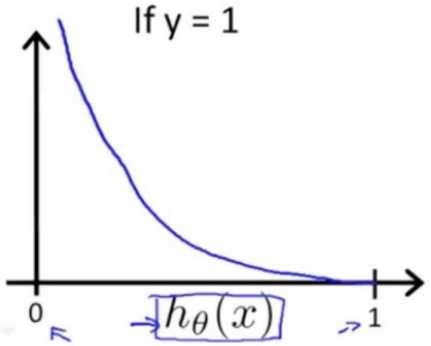


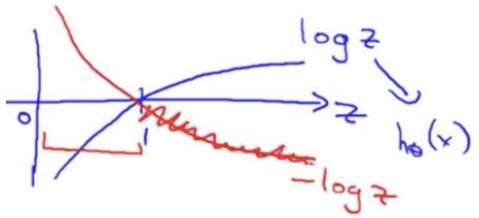
Logistic Regression

Cost function

- Given the logistic regression model, how do we fit the θparameters?

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





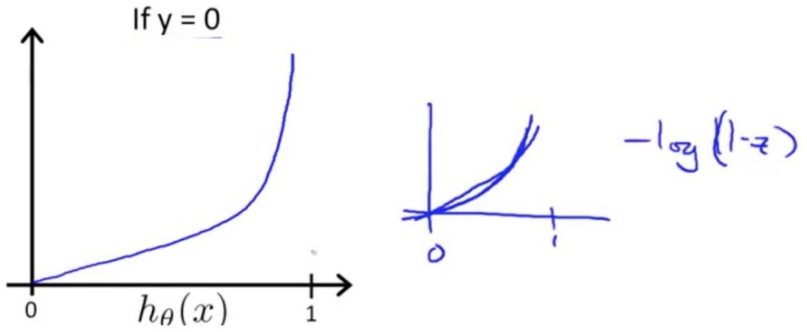
- if the hypothesis predicts that y = 1 and y = 1 cost = 0.
- if the hypothesis predicts that y = 0 and y = 1 cost -> inf.

Logistic Regression

Cost function

- Given the logistic regression model, how do we fit the θparameters?

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



- if the hypothesis predicts that y = 1 and y = 1 cost = 0.
- if the hypothesis predicts that y = 0 and y = 1 cost -> inf.

Logistic Regression

Cost function

- Given the logistic regression model, how do we fit the θparameters?

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

- This cost function can be written as follows:

$$-\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

- We can apply the gradient descent algorithm to find the parameter that minimize the cost function.

Logistic Regression

Guaranteed success! (almost)

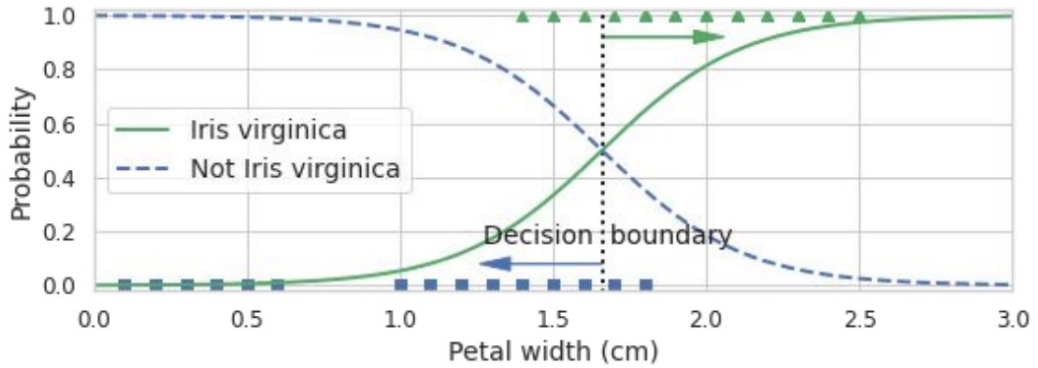
$$-\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

- The cost function for the logistic regression is **convex**, so that Gradient Descent or any other optimization algorithm, will guarantee to find the global minimum
  - ✓ That is… if the learning rate is not too large and one has enough time to wait
  - ✓ This can be controlled with the tolerance criteria and the number of iterations defined in Sklearn

Logistic Regression

A note on Decision Boundaries

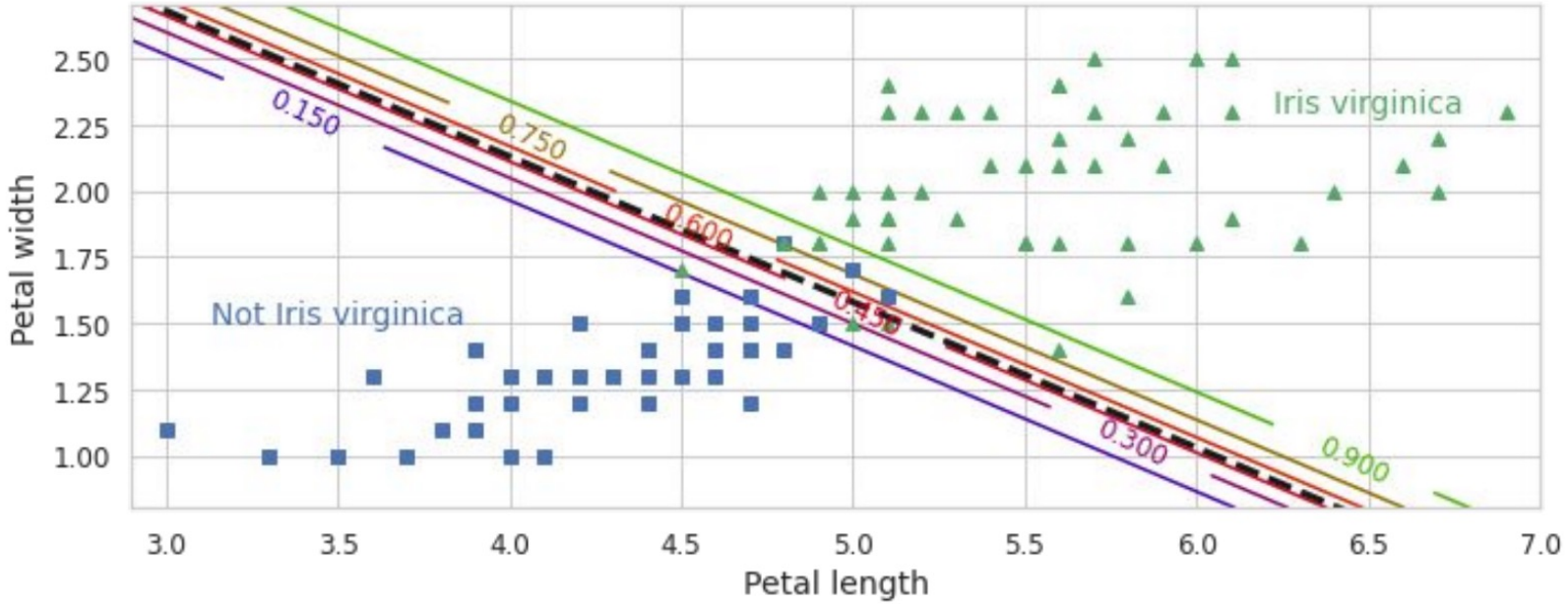
- Decision boundaries in LogReg are usually set when negative and positive probabilities cross at y=0.5
- If you want to get probabilistic results rather than classes you can use sklearns predict_proba() instead of predict()
- That can be used for further refinement of the decision boundary (in case that one wants to classify something as positive if it goes below 0.5)



Logistic Regression

A note on Decision Boundaries

- predict_proba() returns the probabilities of theclass to be the class that represents in its vector space

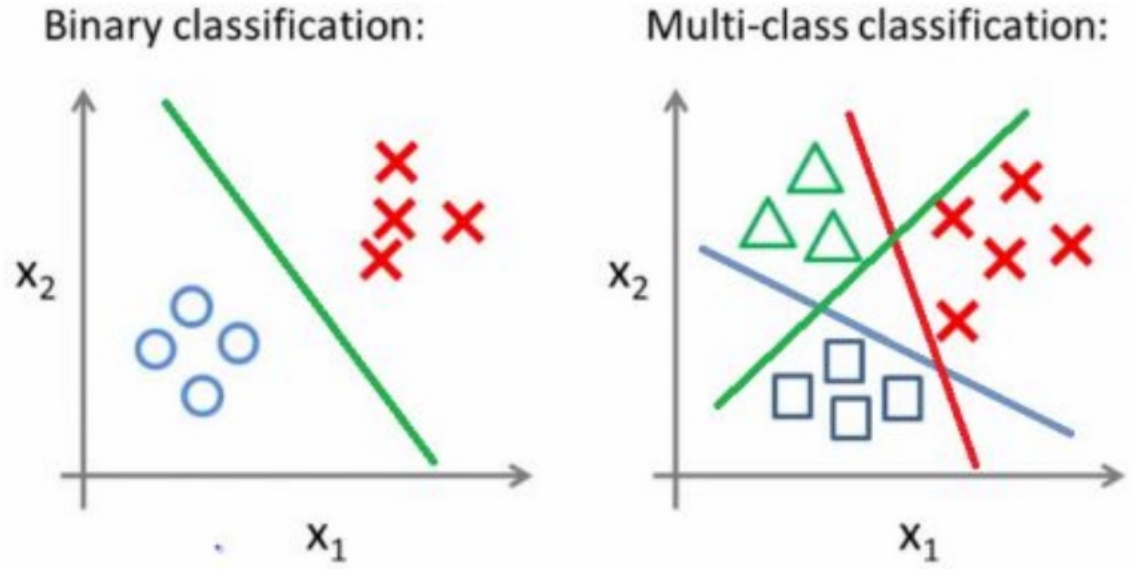


Logistic Regression

Multi-class Classification

One vs. all classification

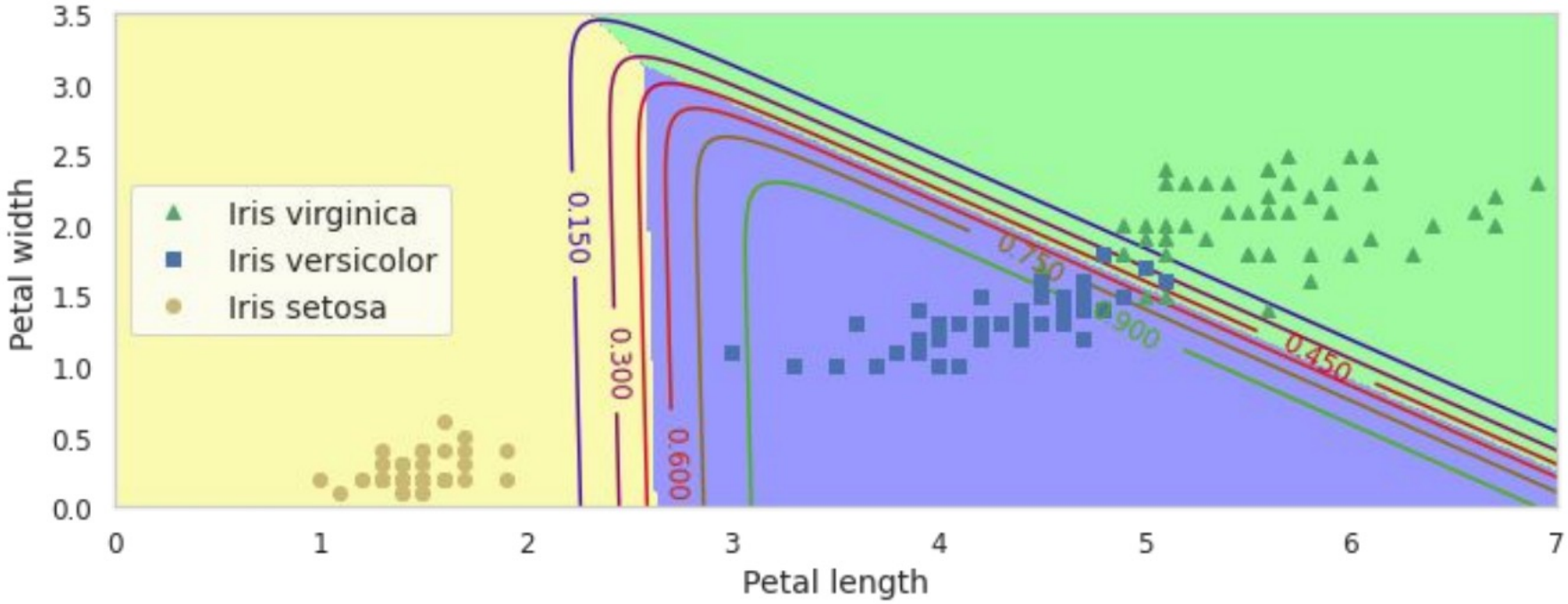
- ✓ classification problem with N distinct classes
- ✓ Train N distinct binary classifiers, each designed for recognizing a particular class



Logistic Regression

Multi-class Classification

- We can use multi class classification in Sklearn LogReg bypassing the multiclass parameter



Logistic Regression

Dealing with overfit and underfit

- A model generalization error can be expressed as:

  - ✓ BIAS: this part of the generalization error is due to wrong assumptions, such as assuming that the data is linear when it is actually quadratic. High-bias model is highly likely to underfit the training data

  - ✓ VARIANCE: this error is due to having a model that is too sensitive to small variations in training data. A model with a lot of degrees of freedom, will have high variance and it will overfit our training data

Logistic Regression

How do we regularize those errors?

- Lasso, L1 penalty
  - *L1 penalizes sum of absolute value of weights.*
- Ridge, L2 penalty (added by default by Sklearn)
  - *L2 regularization penalizes sum of square weights.*

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization

$$\text{Cost} = \sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2 + \lambda \sum_{j=0}^{M} W_j^2$$

Loss function

Regularization Term

In Sklearn, this value is represented by C, and it is the inverse of lambda. So, the higher the value of C, the less the model will be regularized

Thank you!

BARCELONA TECHNOLOGY SCHOOL