Machine Learning

Classification

Fernando Rodríguez Sánchez

ferjorosa@gmail.com

Universidad Politécnica de Madrid

16/10/2020

Table of contents

Table of contents

Supervised learning

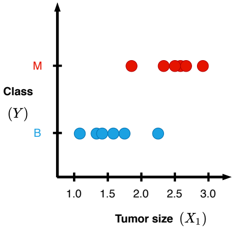
	X_1	$\dots$	X_n	Y
$(\mathbf{x}^{(1)}, y^{(1)})$	$x_1^{(1)}$	$\dots$	$x_n^{(1)}$	$y^{(1)}$
$(\mathbf{x}^{(2)}, y^{(2)})$	$x_1^{(2)}$	$\dots$	$x_n^{(2)}$	$y^{(2)}$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$(\mathbf{x}^{(m)}, y^{(m)})$	$x_1^{(m)}$	$\dots$	$x_n^{(m)}$	$y^{(m)}$

Classification

- X_i is discrete/continuous
- Y is discrete (the **class**)

Introduction

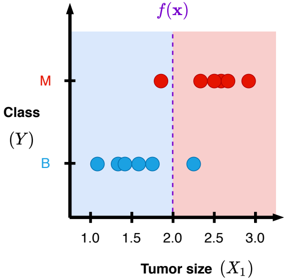
- Given $(\mathbf{x}^{(1)}, y^{(1)})$ learn a function $f(\mathbf{x})$ to predict y given $\mathbf{x}$
- y is discrete (the **class**)



One-dimensional

Introduction

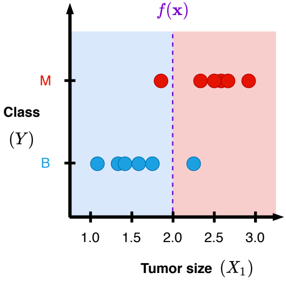
- Given $(\mathbf{x}^{(1)}, y^{(1)})$ learn a function $f(\mathbf{x})$ to predict y given $\mathbf{x}$
- y is discrete (the **class**)



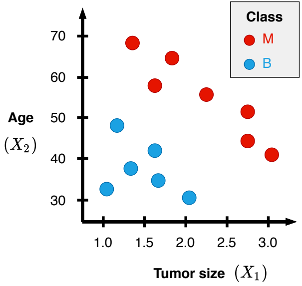
One-dimensional

Introduction

- Given $(\mathbf{x}^{(1)}, y^{(1)})$ learn a function $f(\mathbf{x})$ to predict y given $\mathbf{x}$
- y is discrete (the **class**)



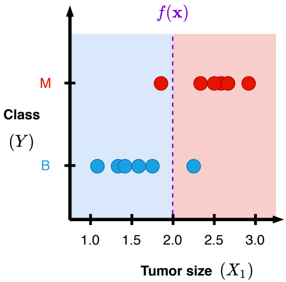
One-dimensional



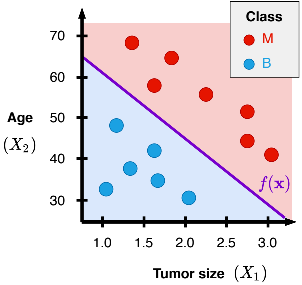
Multi-dimensional

Introduction

- Given $(\mathbf{x}^{(1)}, y^{(1)})$ learn a function $f(\mathbf{x})$ to predict y given $\mathbf{x}$
- y is discrete (the **class**)



One-dimensional



Multi-dimensional

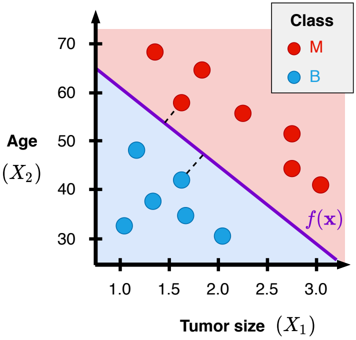
Table of contents

Support Vector Machines

Support Vector Machines try to find the linear function $f(x)$ that best separate **two** classes

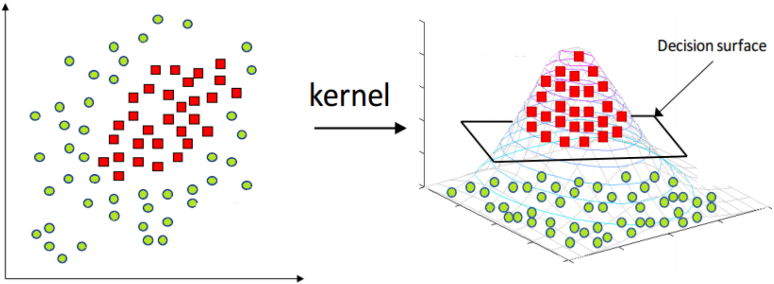
Tries to make the separation **as wide as possible**

Support vectors $\rightarrow$ closest points to the line



Kernel trick

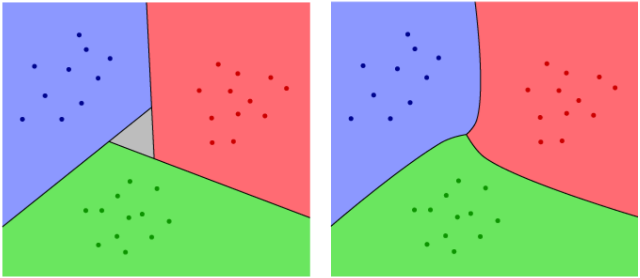
What happens when classes are not **linearly** separable?



The training points are mapped to a 3-dimensional space where a separating hyperplane can be easily found

$$(A, B) \rightarrow (A, B, A^2 + B^2)$$

Multi-class classification



Multi-class classification via **All vs. All**

What happens on **ties** (grey area)?

- Depends on implementation
- *Scikit-learn* assigns a class probability via K-fold cross validation

Strengths and weaknesses

Strengths

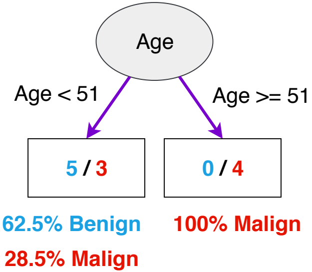
- Memory efficient (only need to store the support vectors)
- Can represent many decision boundaries via kernels
- Effective in high dimensional spaces

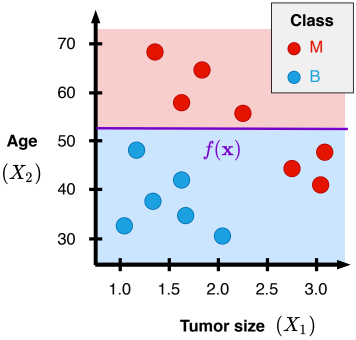
Weaknesses

- Performance is sometimes kernel-dependent
- Doesn't scale well to large datasets
- Doesn't work well with mixed data

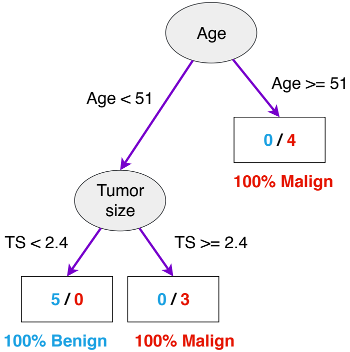
Table of contents

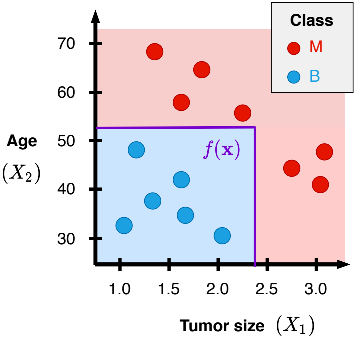
Decision trees





Decision trees





Overfitting?

Strengths and weaknesses

Strengths

- Easy to understand
- Works with mixed data
- **Very good when done in ensembles**

Weaknesses

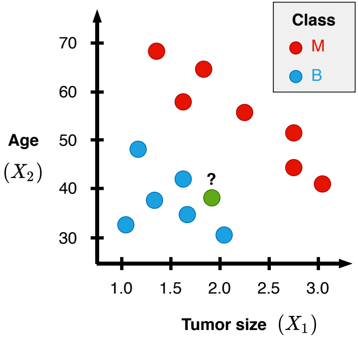
- Individual trees are prone to **overfitting**
- Pruning is usually necessary (when/how to **prune**?)

Table of contents

K-nearest neighbours

Procedure to classify a new $\mathbf{x}$:

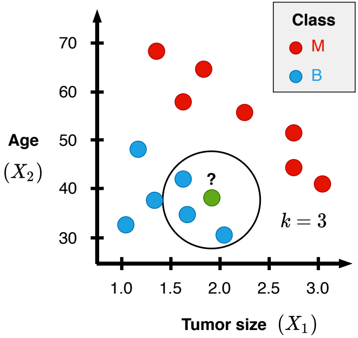
- Measure distance to all the other instances
- Select k closest ones
- Assigns the most frequent class of those k instances



K-nearest neighbours

Procedure to classify a new $\mathbf{x}$:

- Measure distance to all the other instance
- Select k closest ones
- Assigns the most frequent class of those k instances

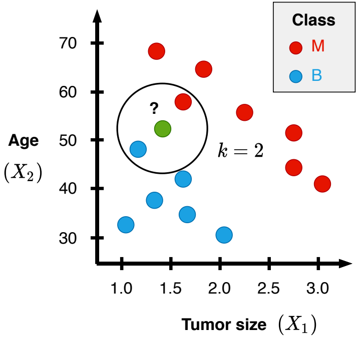


K-nearest neighbours

What happens if there is a **tie** (even k value)?

- Depends on implementation
- *Scikit-learn* chooses the first ordered instance of the k and assigns its class to x

We can also use an **uneven** k value



Strengths and weaknesses

Strengths

- Easy to understand
- Can represent any function with enough data

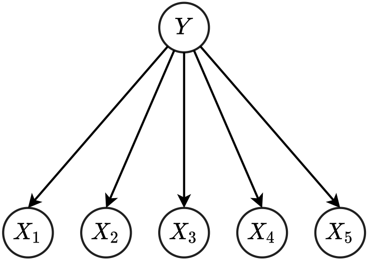
Weaknesses

- Memory intensive
- Problems on high dimensional data (distances)
- Doesn't work well with mixed data

Table of contents

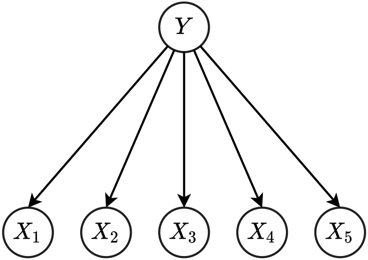
Naïve Bayes

- Probabilistic model
- Models the joint probability distribution of data
- Predictor variables are independent given Y
- Uses statistical inference to predict the value of Y given $\mathbf{X}$



Naïve Bayes

- **Gaussian Naïve Bayes**
- Categorical Naïve Bayes
- Multinomial Naïve Bayes
- etc.

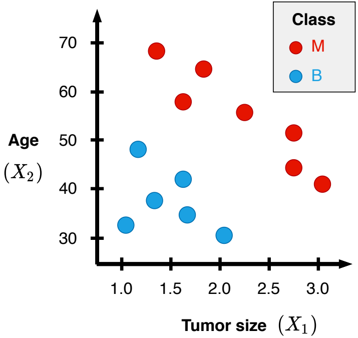


Gaussian Naïve Bayes

$p(Y)$
$p(Y = M) = (7/13) = 0.54$
$p(Y = B) = (6/13) = 0.46$

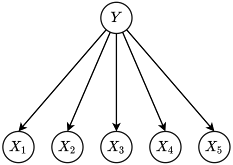
$p(X_1 \mid Y)$
$p(X_1 \mid Y = M) = \mathcal{N}(2.2, 0.39)$
$p(X_1 \mid Y = B) = \mathcal{N}(1.5, 0.15)$

$p(X_2 \mid Y)$
$p(X_2 \mid Y = M) = \mathcal{N}(55.7, 88.2)$
$p(X_2 \mid Y = B) = \mathcal{N}(37.8, 45.4)$

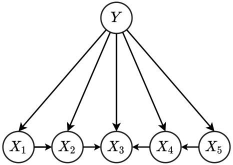


$p(Y|\mathbf{X}) \rightarrow$ Bayes' Theorem

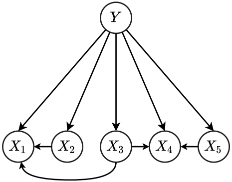
Naïve Bayes extensions



Naïve Bayes



TAN



K-DB

Strengths and weaknesses

Strengths

- Allows uncertainty in the predictions
- Only requires a small number of data instances to work
- Can handle high dimensional data
- Rarely overfits the data

Weaknesses

- Usually underfits the data (Generative vs discriminative)
- It is not implemented in Scikit-learn for mixed data

Machine Learning
Classification

Fernando Rodríguez Sánchez

ferjorosa@gmail.com

Universidad Politécnica de Madrid

16/10/2020