CSC421 Artificial Intelligence

Supervised Learning III

George Tzanetakis

University of Victoria

2021

Table of Contents I

1. Linear Classifiers
2. Clustering
3. Regression

Decision Hyperplanes

Main Idea

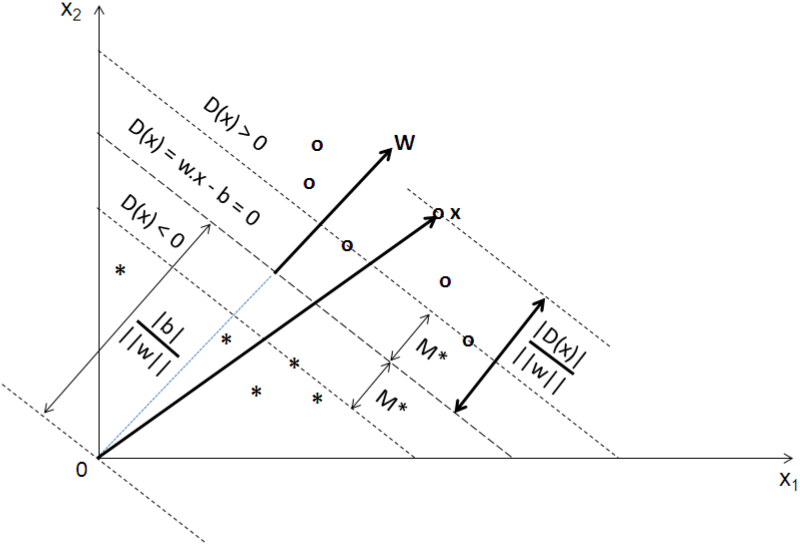
Instead of trying to model all the training data using probability density function just consider the linear decision boundary and focus on finding an optimal one. This directly solves the classification problem (at least for the binary case) rather than transforming it to the potentially more complex problem of fitting a distribution to a set of samples.

Hyperplane

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = 0, \tag{1}$$

where $\mathbf{w} = [w_1, w_2, \ldots, w_l]^T$ is the weight vector and w_0 is the threshold.

Geometry Decision Hyperplane



The perceptron algorithm

Problem

How can the unknown parameters w_i, $i = 0 \ldots l$ defining the decision hyperplane be determined based on a training set ?

Linear Separability

$$\mathbf{w}^{*T}\mathbf{x} > 0, \quad \forall \mathbf{x} \in \omega_1 \tag{2}$$

$$\mathbf{w}^{*T}\mathbf{x} < 0, \quad \forall \mathbf{x} \in \omega_2 \tag{3}$$

Perceptron Iterative Scheme

Cost Function

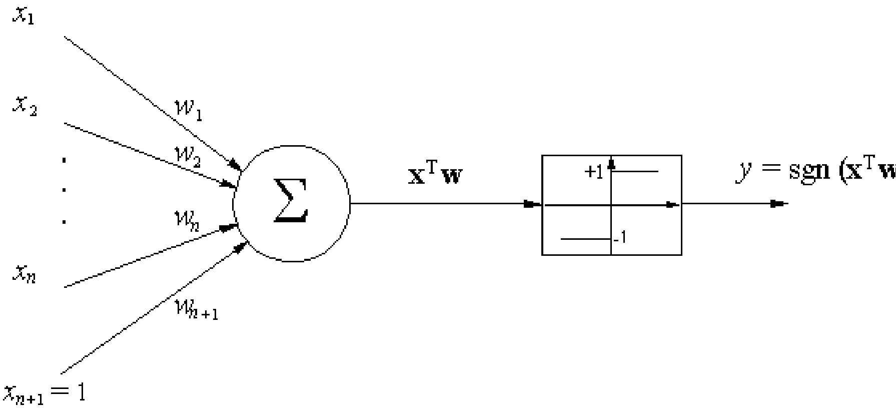
Cost function: $J(\mathbf{w}) = \sum_{x \in M}(\delta_x \mathbf{w}^T \mathbf{x})$ where M is the subset of training vectors, which are misclassified by the hyperplane defined by w. The variable δ_x is -1 if $x \in \omega_1$ and $+1$ if $\mathbf{x} \in \omega_2$. Notice that $J(\mathbf{w})$ will always be positive and will be zero when there are no misclassified examples in the training set. ρ_t is a sequence of decreasing positive numbers.

Iterative Algorithm

Start with random weights and iterate until convergence. Multiple Local Minima (more than one hyperplane can satisfy the condition). Similar to gradient descent.

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \rho_t \sum_{x \in M} \delta_x \mathbf{x} \tag{4}$$

Neuron View


x_1
x_2
w_1
w_2
w_n
Σ
x_n
w_{n+1}
$x_{n+1} = 1$
$\mathbf{x}^\mathrm{T}\mathbf{w}$
+1
-1
$y = \mathrm{sgn}\,(\mathbf{x}^\mathrm{T}\mathbf{w}$

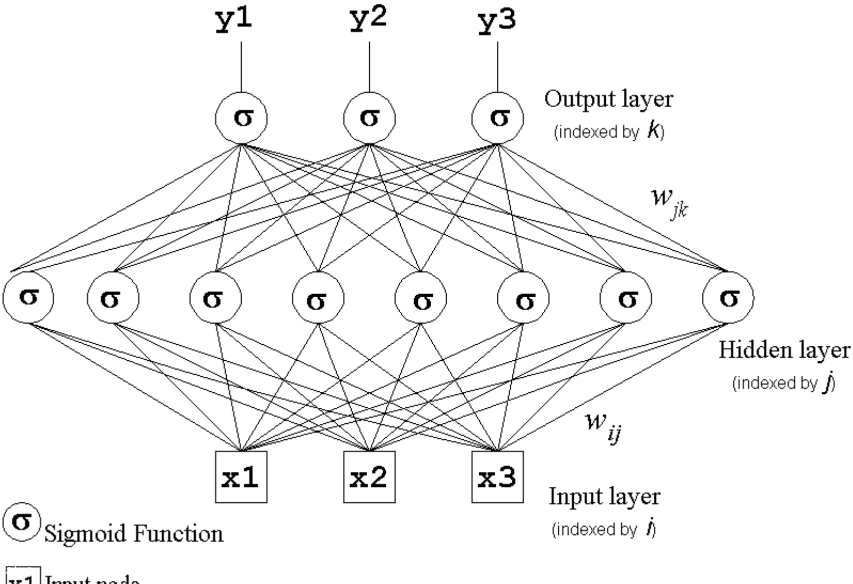
Characteristics of the basic Perceptron

- Initial optimism replaced by scepticism. The field of ANNs was killed for ten years in the 70s.
- Well suited for online processing - easy to implement
- Only work for linearly separable datasets
- Prone to overfitting - no consideration of margin

Multilayer Perceptron Networks

Also known as Artificial Neural Networks (ANNs). Resurgence in the 1980s when the *backpropagation* training algorithm became popular. *Backpropagation* requires that the activation function is differentiable (replacing the step function used in the basic Perceptron, for example, with a sigmoid function). Allow arbitrary modeling of input to output supporting naturally classification as well as regression and multilabel classification. Fell out of fashion around 2000 with the rise of the support vector machines but have recently resurfaced with deep learning.

Multilayer Perceptron Network



Characteristics of ANNs

- Slow convergence - training
- Prone to overfitting due to the large number of parameters
- Can solve complex problems in addition to classification
- Unclear how to find the optimal architercture for the hidden layer(s)
- Easy to parallelize - deep learning resurgence has been enabled among other things by GPU processing

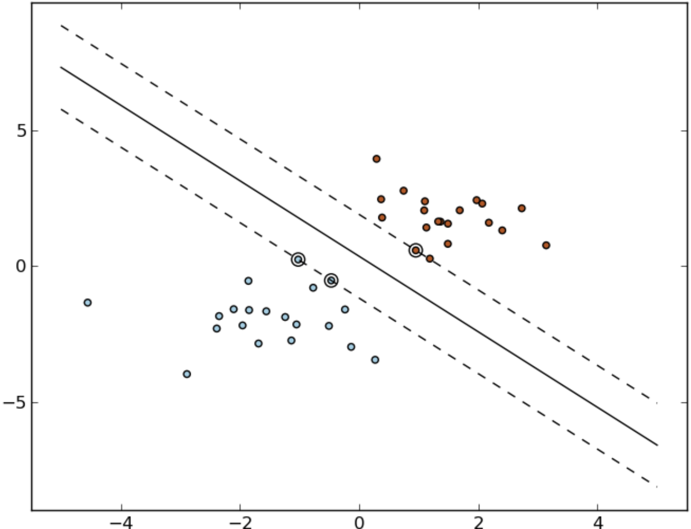
Support Vector Machines

When a problem is linearly separable there are several hyperplanes that satisfy perfect separation. If we could choose among them is there any reason to prefer one over another ? Hyperplane that leaves more “room” on either side should be chosen so that data from both classes can “move” more freely with less risk of causing an error. The formal term for this “room” is called the *margin* and our goal is to have the same distance to the nearest points in ω_1 and ω_2. So now instead of simply searching for a hyperplane that perfectly separates the two classes we also have the additional requirement of a maximum margin. This can be cast as non-linear (quadratic) optimization problem subject to a set of linear inequality constraints.

Support Vector Machines

When a problem is linearly separable there are several hyperplanes that satisfy perfect separation. If we could choose among them is there any reason to prefer one over another ? Hyperplane that leaves more “room” on either side should be chosen so that data from both classes can “move” more freely with less risk of causing an error. The formal term for this “room” is called the *margin* and our goal is to have the same distance to the nearest points in ω_1 and ω_2. So now instead of simply searching for a hyperplane that perfectly separates the two classes we also have the additional requirement of a maximum margin. This can be cast as non-linear (quadratic) optimization problem subject to a set of linear inequality constraints.

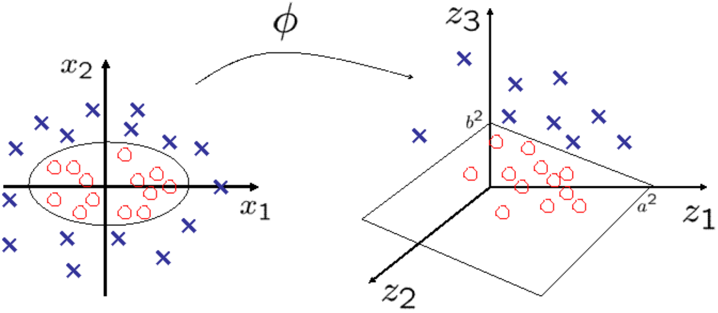
Maximum Margin Separating Hyperplane



Characteristics of SVMs

- Effective in high dimensional spaces
- Uses only a subset (the support vectors) of training points
- All operations involve inner products (Kernel trick)
- Do not directly provide probability estimates but can be extended to do so
- Require normalization of the training data (min/max)
- Binary classifier needs to be extended for multi-class using one-vs-all or all-pairs.
- Prediction can be extremely fast
- Generalization is backed by elegant theory
- Hard to implement optimization but a lot of good implementations exist

Kernel Trick



$$\phi : (x_1, x_2) \longrightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

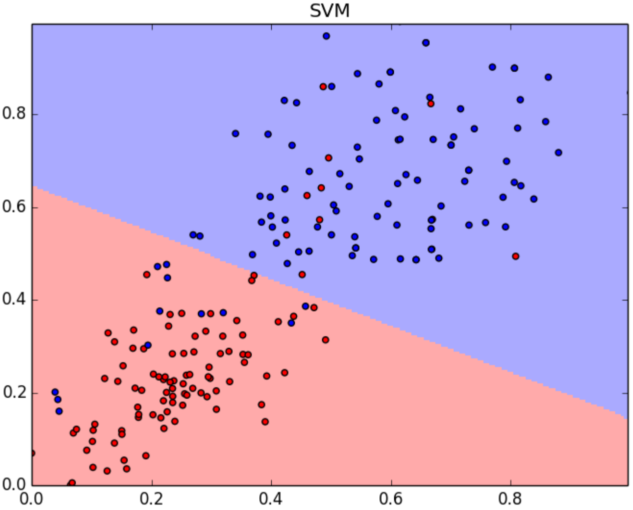
$$\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2 = 1 \longrightarrow \frac{z_1}{a^2} + \frac{z_3}{b^2} = 1$$

Decision Trees

Definition

Decision trees are models that predict the value of a target variables based on several input variable. Each interior node corresponds to one of the input variables and there are edges to children for each possible value of the input variable. Each leaf represents a value of the target variable given the values of the input variables corresponding to the path from the root to the leaf. The classification process is more transparent i.e easy to interpret than other algorithms.

The hoops dataset (Naive Bayes Gaussian)



Decision Tree Learning

Typically done by recursive partitioning of the training set.

- Select which input variable to split based on some measure of which split is best.
- Several split quality measures have been proposed (for example Gini impurity, information gain). They are applied to each candidate subset and then averaged to determine the overall quality of the split.
- Initially only delt with categorical attributes but have been extended to deal with ordinal and contrinuous attributes (requiring a step of discretization).

Split quality measures

Definition

Gini impurity

$$I_G(f) = \sum_{i=1}^{m} f_i(1 - f_i) = \sum_{i=1}^{m}(f_i - f_i^2) = 1 - \sum_{i=1}^{m} f_i^2, \qquad (5)$$

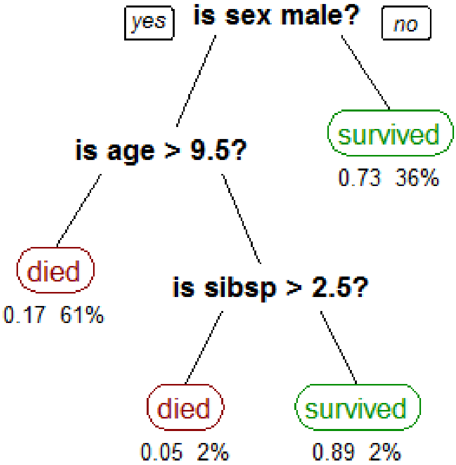
where f_i is the fraction of items labeled with value i in the data.

Definition

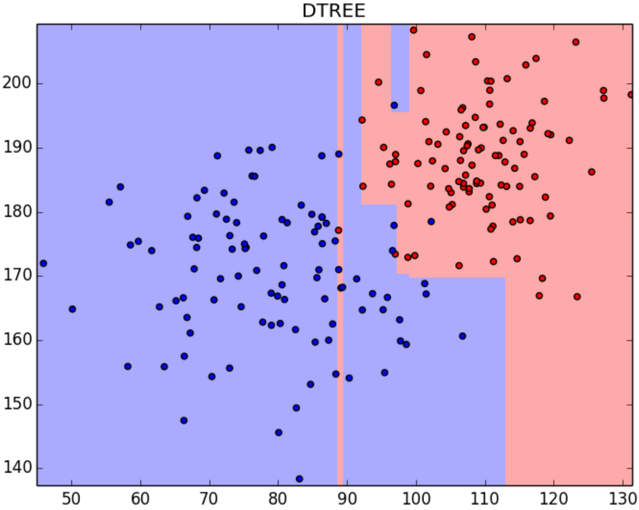
Information Gain

$$I_E(f) = -\sum_{i=1}^{m} f_i \log_2 f_i \qquad (6)$$

Decision Tree Example



Decision Tree with Hoops dataset



Characteristics of Decision Trees

- Simple to understand and interpret - white box compared to black box SVMs/ANNs (typically white box learning algorithms tend to perform worse than black box ones)
- Can handle categorical and numerical data
- Robust and can handle large datasets
- Parallel implementation are possible
- Require little data preparation
- NP-complete problem - heuristics such as greedy search do not guarantee “optimum” tree.
- Prone to overfitting - require pruning
- Some problems can result in very large trees

Table of Contents I

1 Linear Classifiers

2 Clustering

3 Regression

Clustering

Definition

Clustering The goal is to reveal the organization of a set of unlabeled data points by assigning them to clusters and thus effectively labeling them. It is also known as unsupervised learning. The detected groups should be "sensible" meaning that they should allow us to discover and observe similarities and differences among the patterns.

Types of clustering

- Hierarchical (bottom-up and top-down)
- Data-based (input is a nSamples by nAttributes feature matrix)
- Affinity-based (input is a nSamples by nSamples similarity matrix)

Clustering notation

Let X be the data set we are interested in clustering:

$$\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \tag{7}$$

Definition

A m-clustering of X is the partitioning of X into m sets (clusters) $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_m$ such that the following conditions hold:

$$\mathbf{C}_i \neq \emptyset, i = 1, \ldots, m \tag{8}$$

$$\cap_{i=1}^{m} \mathbf{C}_i = \mathbf{X} \tag{9}$$

$$\mathbf{C}_i \cup \mathbf{C}_j = \emptyset, i \neq j, i, j = 1, \ldots m \tag{10}$$

Evaluation of clustering

Evaluation of clustering is more challenging than classification and requires much more subjective analysis. Many criteria have been proposed for this purpose. They can be grouped into internal (no external information is required) and external (external partition information about the "correct" clustering is provided). Explaining the criteria in detail is beyond the scope of this tutorial. A simple example of an internal criterion would be preferring clustering schemes in which the average distance between points within a cluster is lower than the average distance between points among different clusters. A simple example of an external criterion would the information gain (measuring the class purity of a cluster) if associated class labels are provided for the data points.

K-Means

Basic Algorithm

The only input is K the number of clusters.

1. Randomly assign each data point to one of K clusters
2. For each cluster compute the mean value of the data points in it
3. Re-assign each data point to the cluster with the closest mean
4. Repeat the last two steps until means don't change

Similar in some ways to the EM-algorithm which can also be used for clustering. Applications in Vector Quantization.

Clustering algorithm comparison

University of Victoria
Computer Science

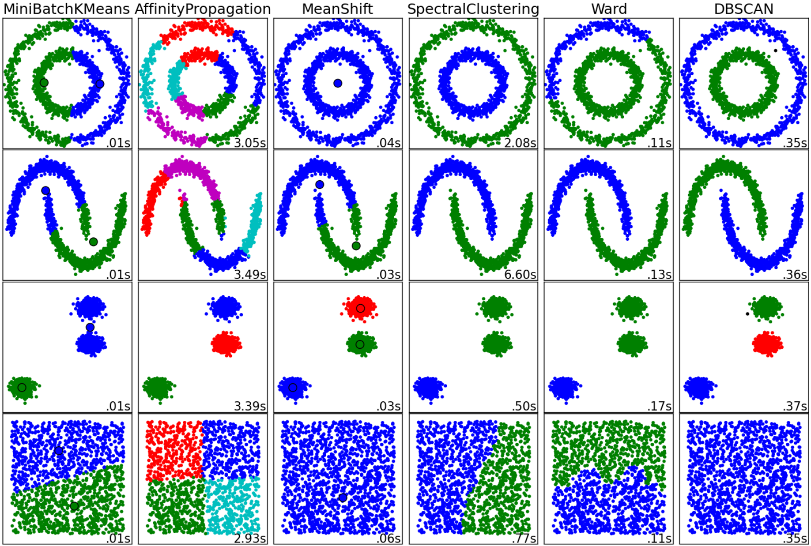


Table of Contents I

1 Linear Classifiers

2 Clustering

3 Regression

Regression

Definition

Regression is a predictive machine learning task in which the target value is continuous rather than discrete as in classification. Examples could be stock market index prediction, predicting sales, predicting weather etc. It can also be univariate or multivariate. In linear regression the prediction function is constrained to be a weighted linear combination of the input attributes.

Error functions

$$Absolute \quad Error = \sum_{i} |y_i - f(\mathbf{x}_i)| \tag{11}$$

$$Squarred \quad Error = \sum (y_i - f(\mathbf{x}_i))^2 \tag{12}$$

Linear Regression using Least Squares

Definition

Optimization problem of minimizing residual - can be solved analyitically resulting in set of linear equation.

$$min_w(||\mathbf{X}\mathbf{w} - \mathbf{y}||_2)^2 \tag{13}$$

There are many other variants of regression that can handle non-linearities, co-linearities, noise, favor sparsity and others.

Linear Regression Example