Connectionists - Neural Networks (Deep Learning)

Connectionists are a class of machine learning algorithms that learn from examples by building a general model of the problem domain, and then apply the model to make predictions. Connectionists are also known as **neural networks** or **deep learning**.

They are inspired by the way biological nervous systems, such as the brain, process information.

- To simulate the way the brain works, connectionists use **artificial neural networks** (ANNs), which are composed of **neurons** - computational units that **receive** n **inputs** and **produce a single output** σ;
- Each input is **weighted** by a **synaptic weight** w_i;
- The neuron **sums** the weighted inputs and **applies an activation function** f to the result - f is usually a **sigmoid function**;
  - $\sigma = f(\sum_{i=1}^{n} w_i x_i)$
- **Learning** is achieved by **adjusting the synaptic weights**.

Summary:

- Records represented as **vectors** of **features**;
- **Training** consists of **adjusting the weights** of the **synapses** between neurons, in order to **minimize the generalization error**;
- **Classification** consists of **propagating** the input vector through the network, **activating** the neurons and **selecting** the class with the **highest activation**.

Perceptron

- **Perceptron** is a single-layer neural network that can be used for **classification** - it is a **linear classifier**;
- It can represent **hyperplanes** in the feature space;

  - It can represent **conjunctions** and **disjunctions**, but **not exclusive disjunctions** - it cannot represent datasets that are not linearly separable;

- A perceptron is a function $f : \mathbb{R}^d \to \mathbb{R}$ defined as:
  - $f(x) = \sigma(\sum_{i=1}^{d} w_i x_i + b)$
  - In binary classification, the range of f is $\{-1, 1\}$;
  - Usually, the **activation function** is a **logistic function** ($\sigma(x) = \frac{1}{1+e^{-x}}$) or a **hyperbolic tangent** ($\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$);
- The **error** of a perceptron is the **number of misclassified records**: $E = \frac{1}{2} \sum_{i=1}^{n} (y_i - f(x_i))^2$;

Gradient Descent

- **Gradient descent** is an **optimization algorithm** used to **minimize a function** by **iteratively moving in the direction of the steepest descent** as defined by the **negative of the gradient**;
- Used to **train perceptrons**;
- In order to **minimize the error**, we need to compute its **derivative** with respect to the **weights**: $\frac{\partial E}{\partial w_i}$;
- After computing the **gradient**, we **update the weights** by **subtracting** the **gradient** multiplied by a **learning rate** η:
  - $w_i \leftarrow w_i - \eta \frac{\partial E}{\partial w_i}$
  - The **learning rate** is a **hyperparameter** that controls the **step size**;
  - A common practice, is to use a **learning rate** that **decreases** over time.

Multi-Layer Perceptron

- **Multi-layer perceptron** (MLP) is a **feedforward neural network** that consists of **multiple layers of perceptrons**;
- By default, MLPs are **fully connected** - each neuron in a layer is connected to all neurons in the next layer;
- Another common type of MLP, is one where all variables are input to all neurons.

SVMs and Kernels

...