

Class 5- Machine Learning concepts

Part II

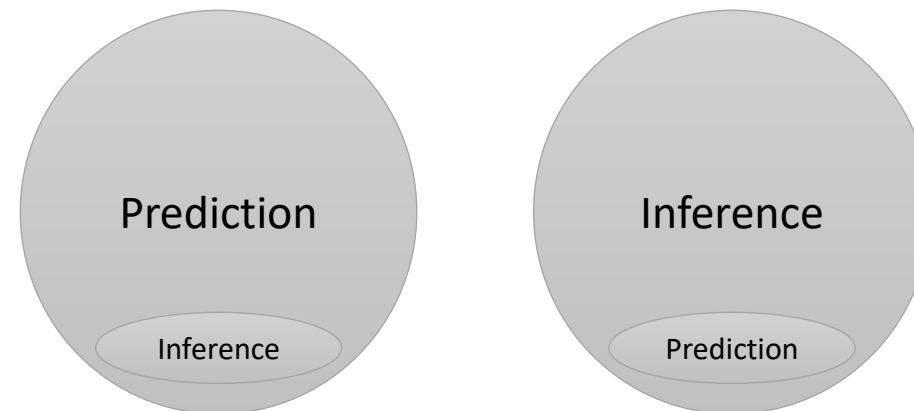




Motivation

Machine learning fundamental concepts:

- Inference and prediction
- Part I: The Model
 - Parameters and hyperparameters
 - Parametric vs nonparametric ML models
- Part II: Evaluation metrics
- Part III: Bias-Variance tradeoff
- Part IV: Resampling methods
- **Part V: Solvers/learners (GD, SGD, Adagrad, Adam, ...)**
- **Part VI: How do machines learn?**



Part V

Solvers (GD, SGD, Adagrad, Adam, ...)



Solvers (learners)!

A **Loss Function** tells us “how good” our model is at making predictions for a given set of parameters. The cost function has its own curve and its own gradients. The slope of this curve tells us how to update our parameters to make the model more accurate.

The two most frequently used optimization algorithms when the **loss function** is differentiable are:

- 1) Gradient Descent (GD)
- 2) Stochastic Gradient Descent (SGD)

Gradient Descent: is an iterative optimization algorithm for finding the minimum of a function. To find a local minimum of a function using gradient descent, one starts at some random point and **takes steps** proportional to the **negative of the gradient** of the function at the current point.

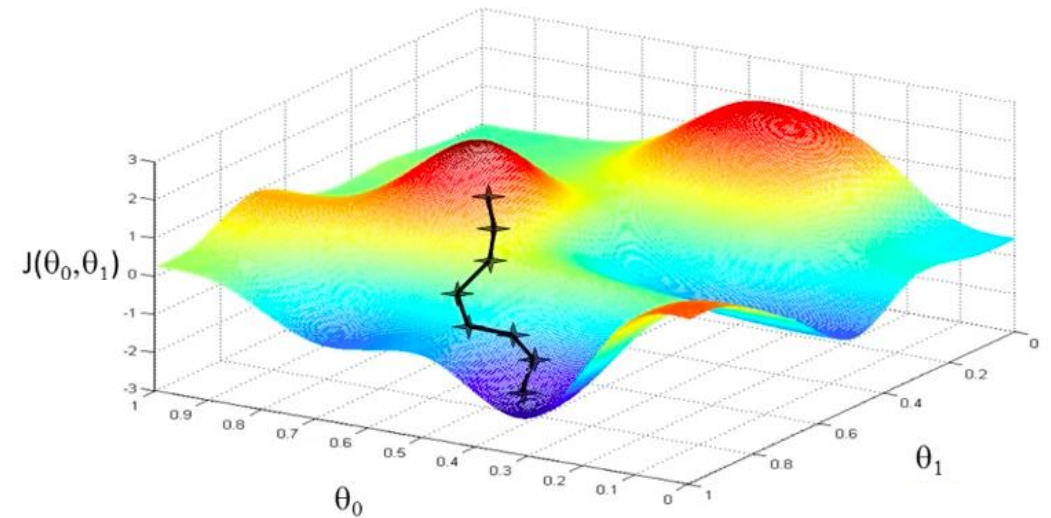
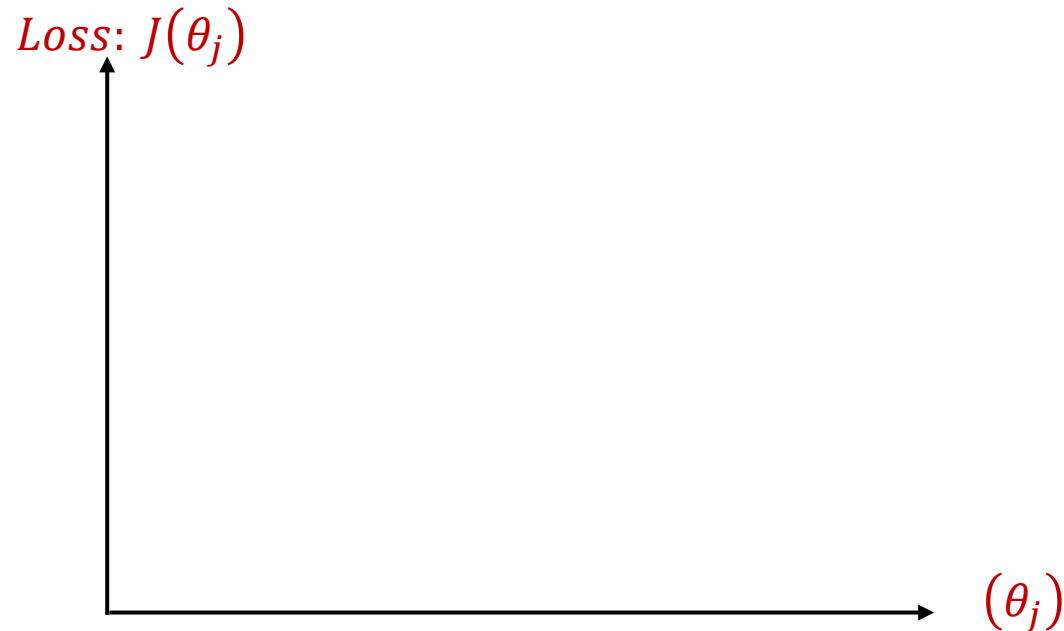
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

- θ_j is the model's j^{th} parameter
- α is the learning rate
- $J(\theta)$ is the loss function (which is differentiable)

Gradient Descent Visualization

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Gradient descent proceeds in **epochs**. An epoch consists of using the training set entirely to update each parameter. The learning rate α controls the size of an update



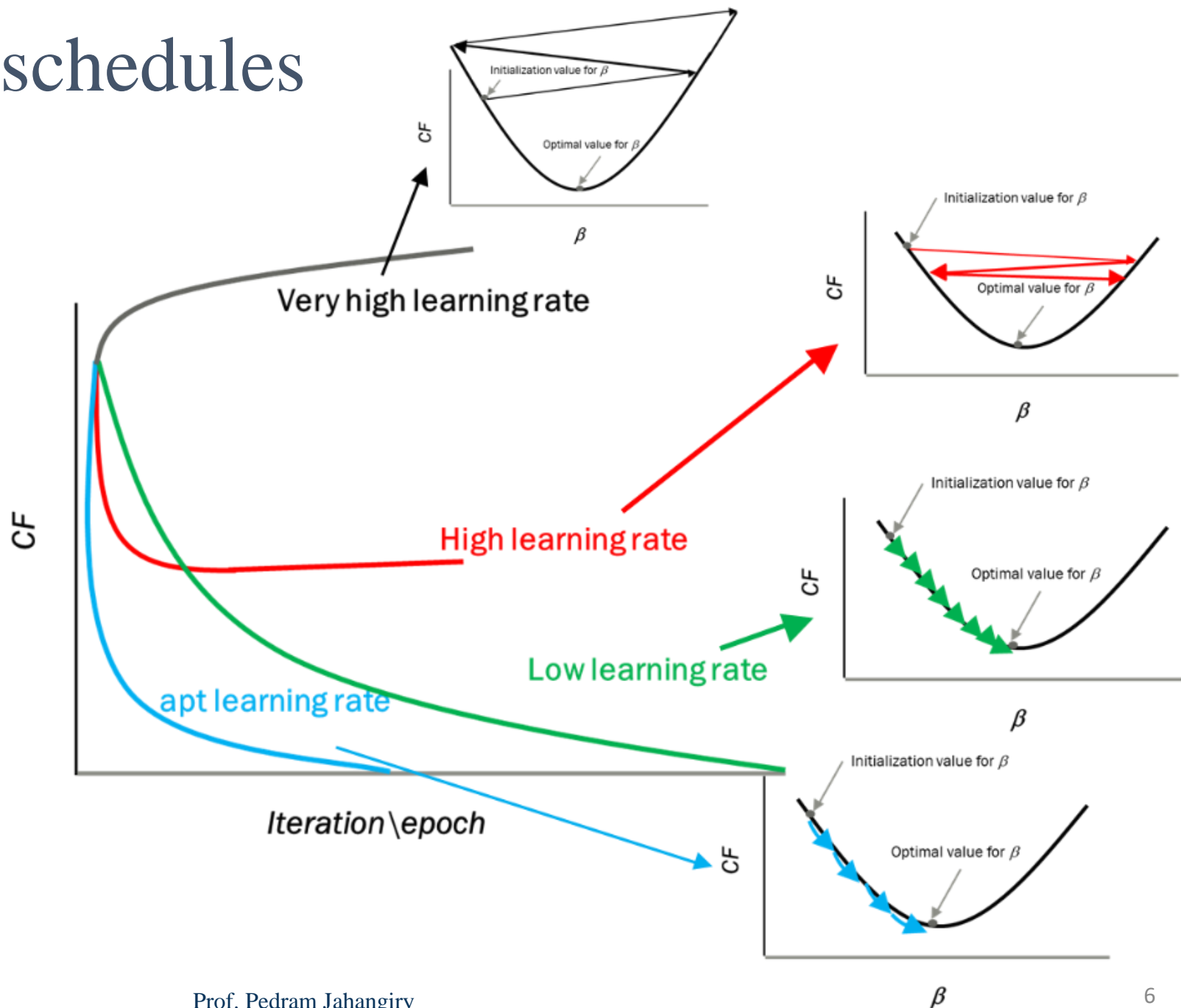
repeat until convergence {
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

(for $j = 1$ and $j = 0$)
}

Learning rate schedules

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

- If α is **too small**, gradient descent can be **slow**
- If α is **too large**, gradient descent can **overshoot** the minimum. It may fail to converge, or even **diverge**.



➔ Beyond Gradient Descent?

Disadvantages of gradient descent:

- Single batch: use the entire training set to update a parameter!
- Sensitive to the choice of the learning rate
- Slow for large datasets

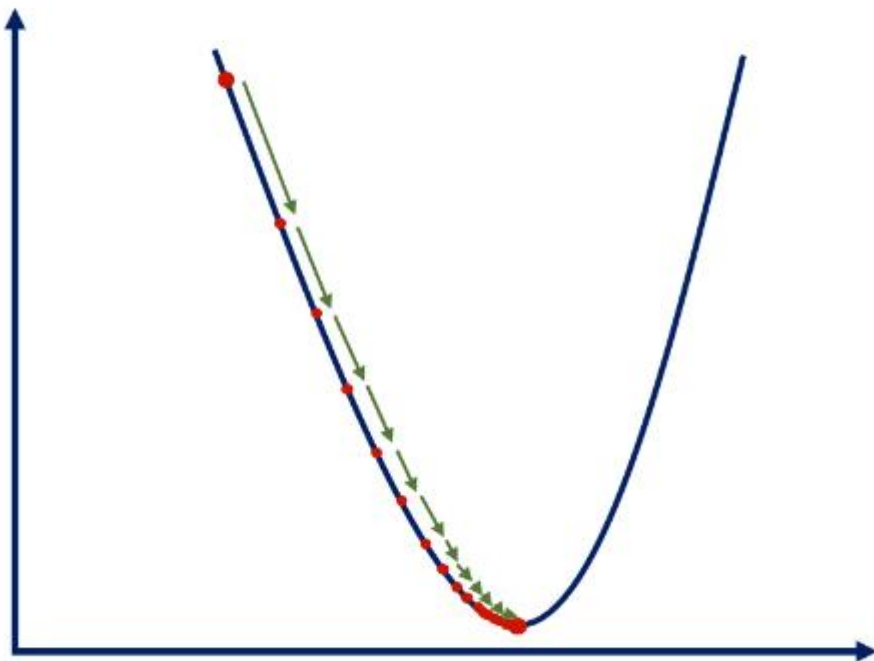
(Minibatch) Stochastic Gradient Descent: is a version of the algorithm that speeds up the computation by approximating the gradient using **smaller batches** (subsets) of the training data. SGD itself has various “upgrades”.

- 1) Adagrad
- 2) Adam

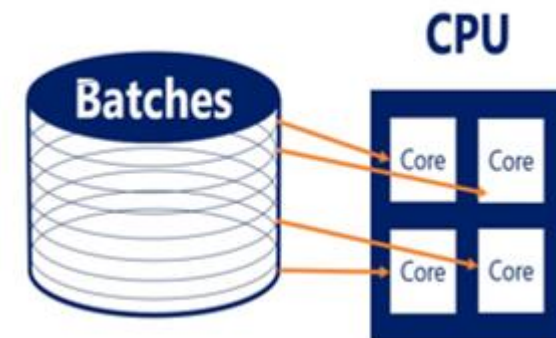
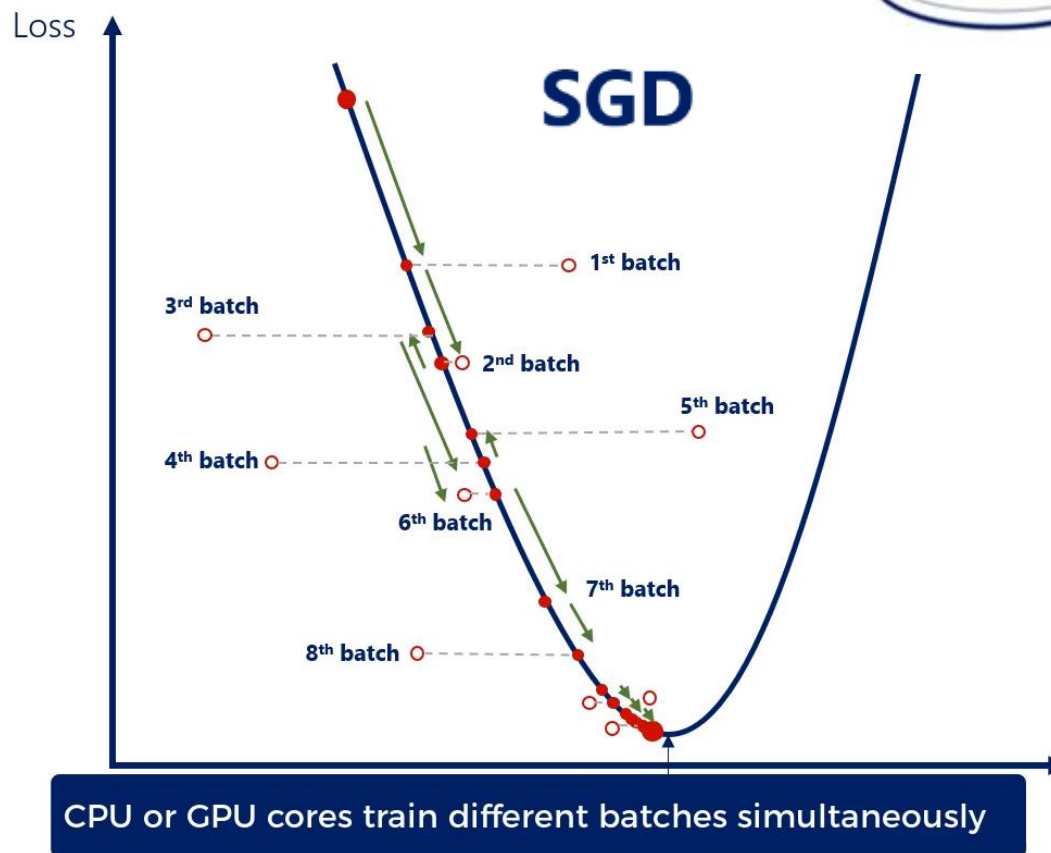


Why SGD?

GD

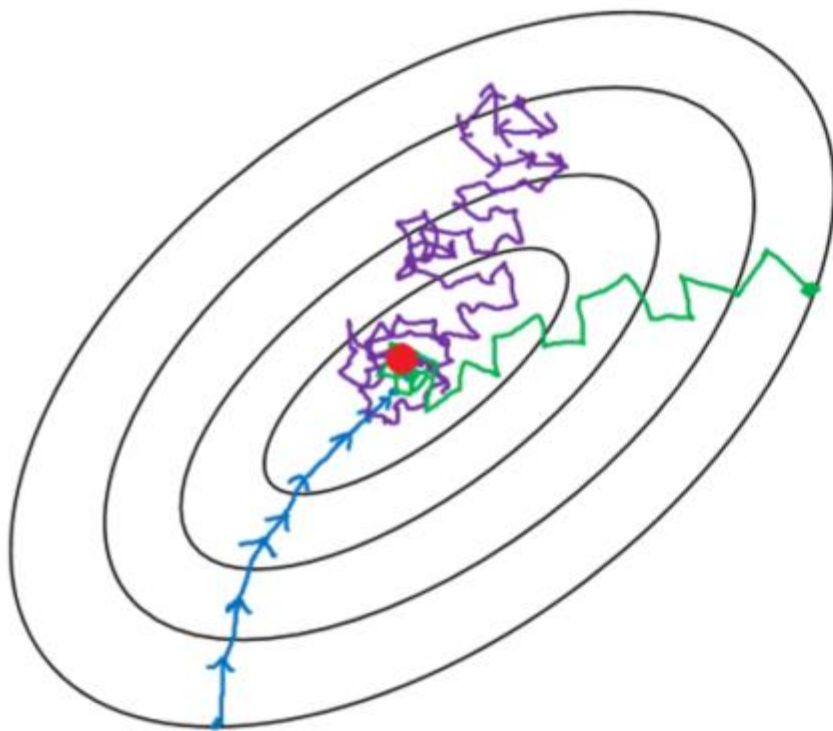


SGD

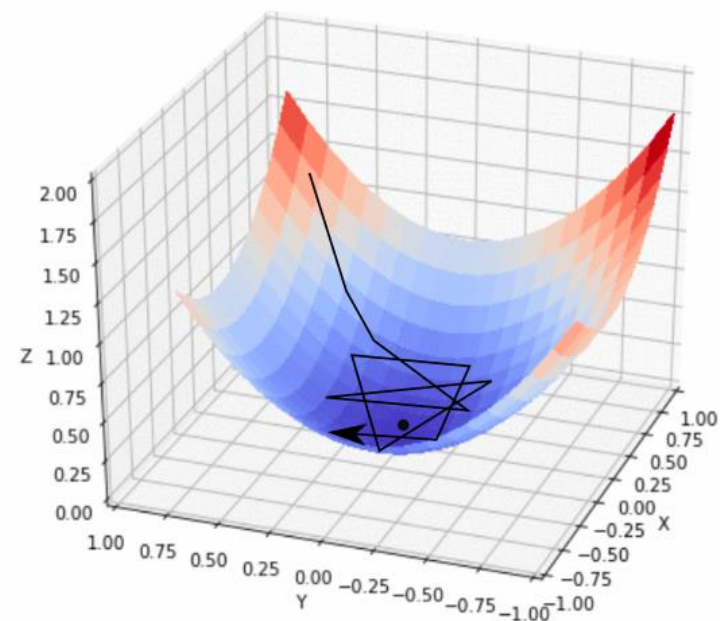




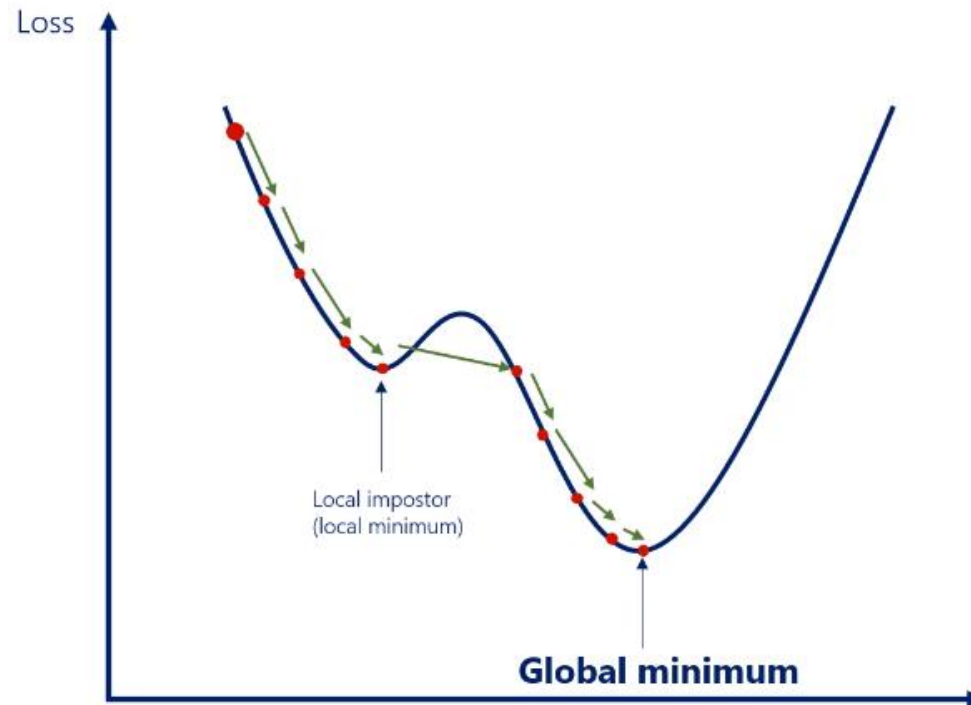
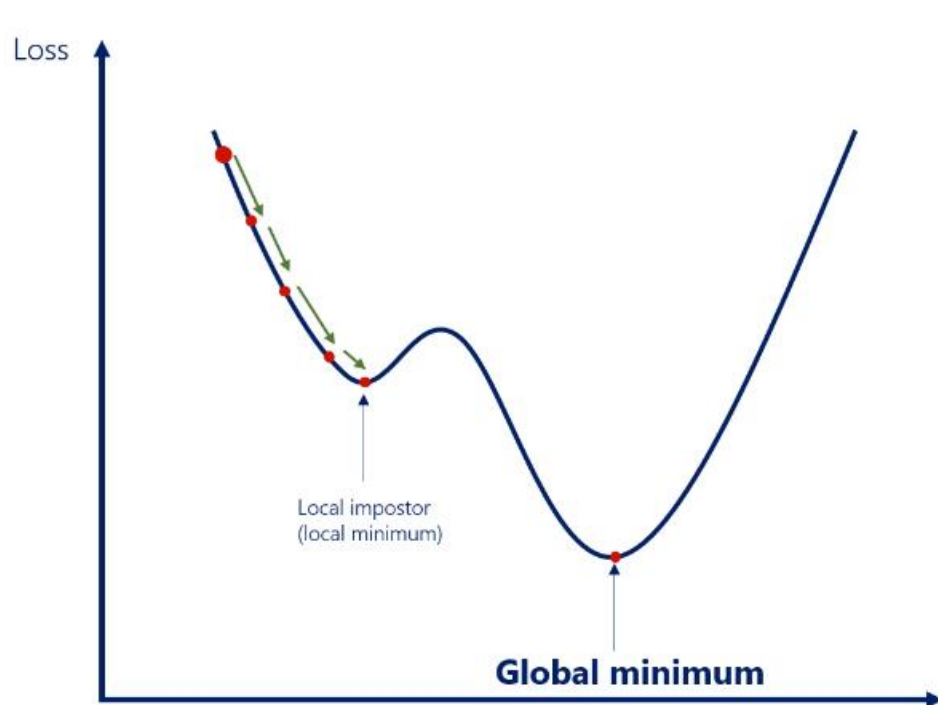
SGD vs GD

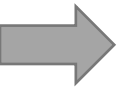


- Batch gradient descent
- Mini-batch gradient Descent
- Stochastic gradient descent



➔ Why upgrade SGD?





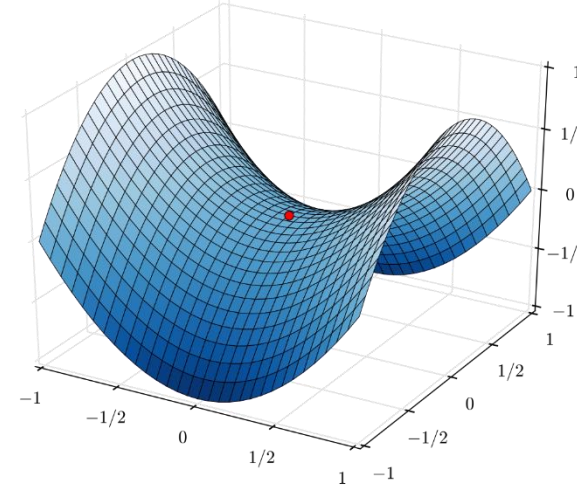
Beyond Stochastic Gradient Descent?

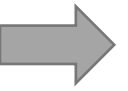
Disadvantages of **Stochastic gradient descent**:

- Get trapped in suboptimal local minima (for non-convex loss functions)
- The same learning rate applies to all parameter updates

SGD upgrades:

- 1) Momentum
- 2) Adagrad
- 3) Adam

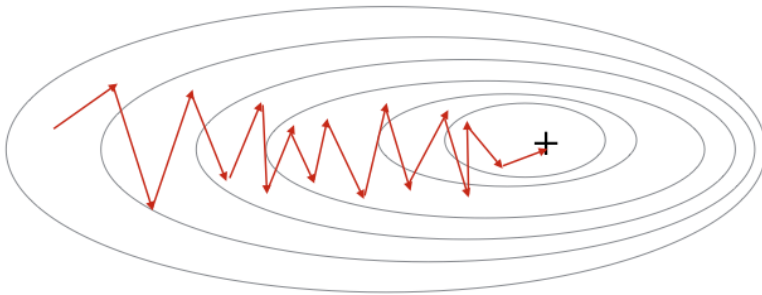




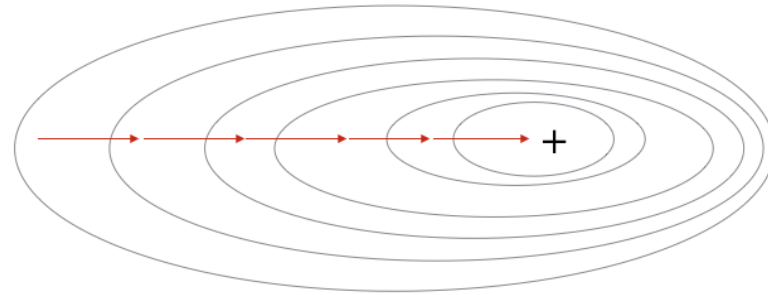
Momentum

- Momentum is a method that helps **accelerate SGD** in the **relevant direction** and **dampens oscillations**.
- Essentially, when using momentum, we push a ball down a hill. The ball accumulates momentum as it rolls downhill, becoming faster and faster on the way.

Stochastic Gradient Descent



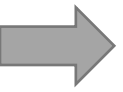
Gradient Descent





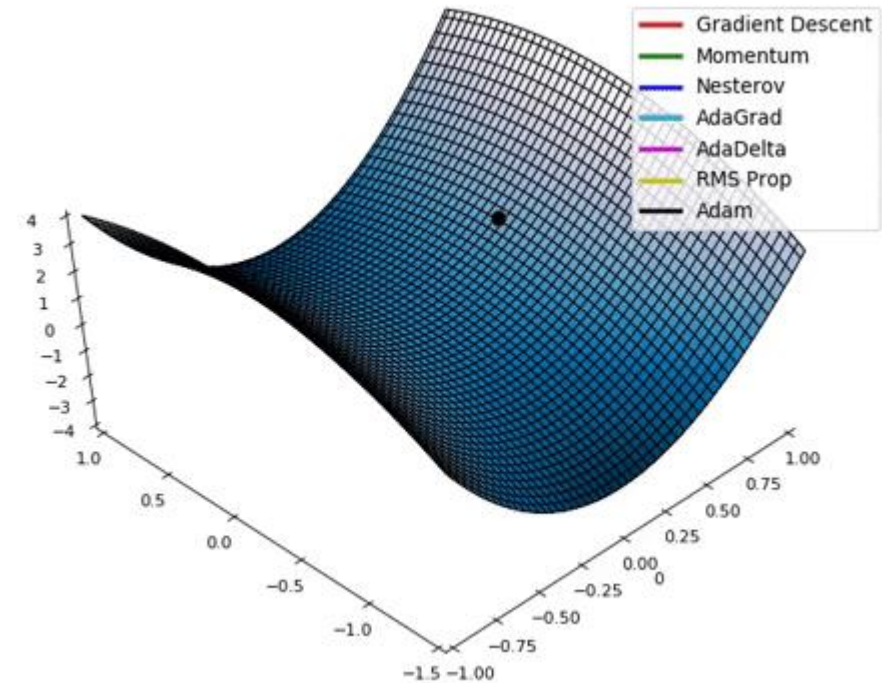
Adagrad

- **Adaptive** Gradient Algorithm is a version of SGD that **scales the learning rate** for each parameter according to the history of gradients. As a result, the learning rate is reduced for very large gradients and vice-versa.
- It adapts the learning rate to the parameters, performing **smaller updates** (low learning rates) for parameters associated with **frequently occurring features**, and **larger updates** (high learning rates) for parameters associated with **infrequent features**. For this reason, it is **well-suited for dealing with sparse data**.



Adam

- Adaptive Moment Estimation takes both momentum and adaptive learning rate (RMSprop) and putting them together.
- Whereas momentum can be seen as a ball running down a slope, Adam behaves like a heavy ball with friction, which thus prefers flat minima in the error surface





Final message!

Notice that gradient descent and its variants **are not machine learning algorithms**. They are **solvers** of minimization problems in which the function to minimize has a gradient (in most points of its domain).

➔ Question of the day!

Me optimizing linear regression
using gradient descent

Least Squares:

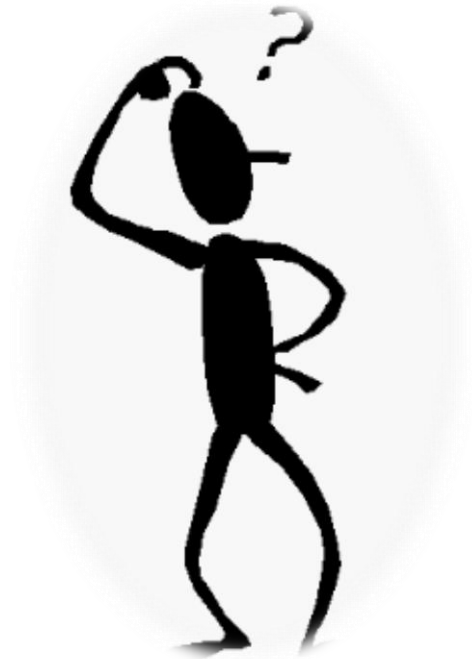


Part VI

How do machines learn?

➔ What is Machine Learning?

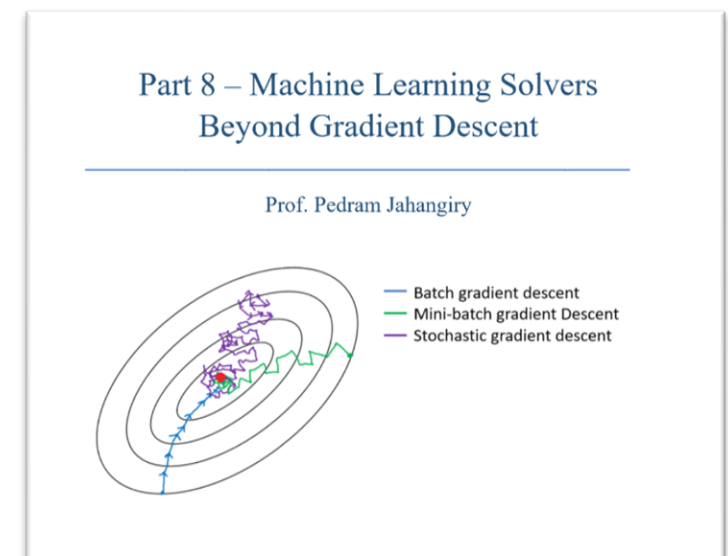
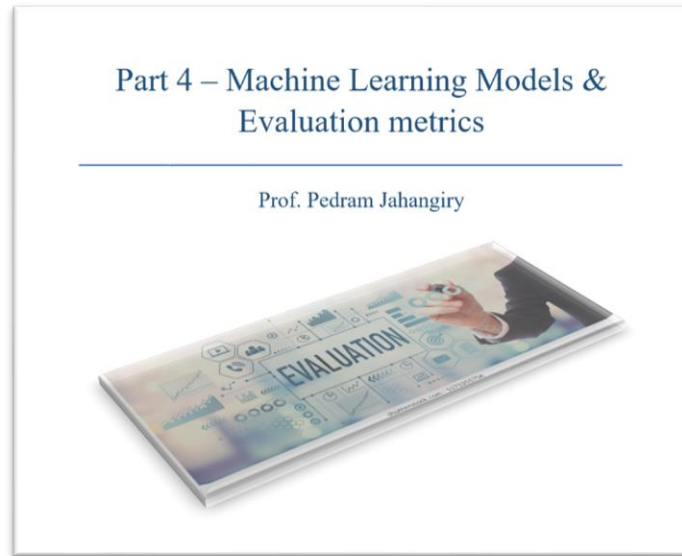
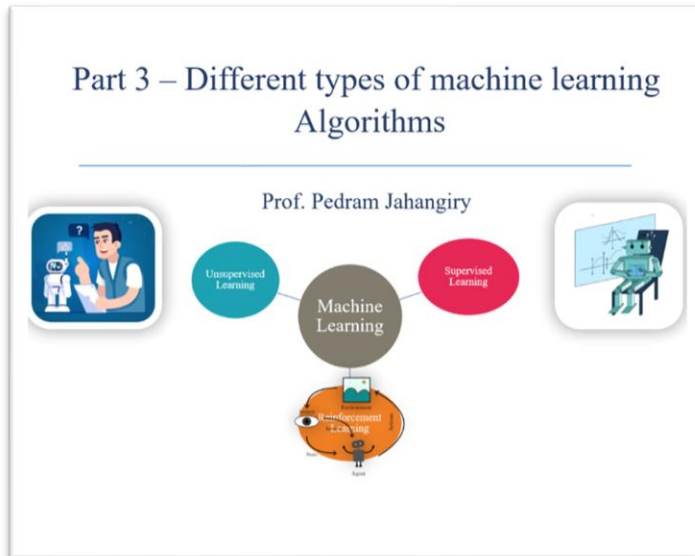
“A ML algorithm learns **complex patterns** in a **high dimensional** space **without being specifically directed**”



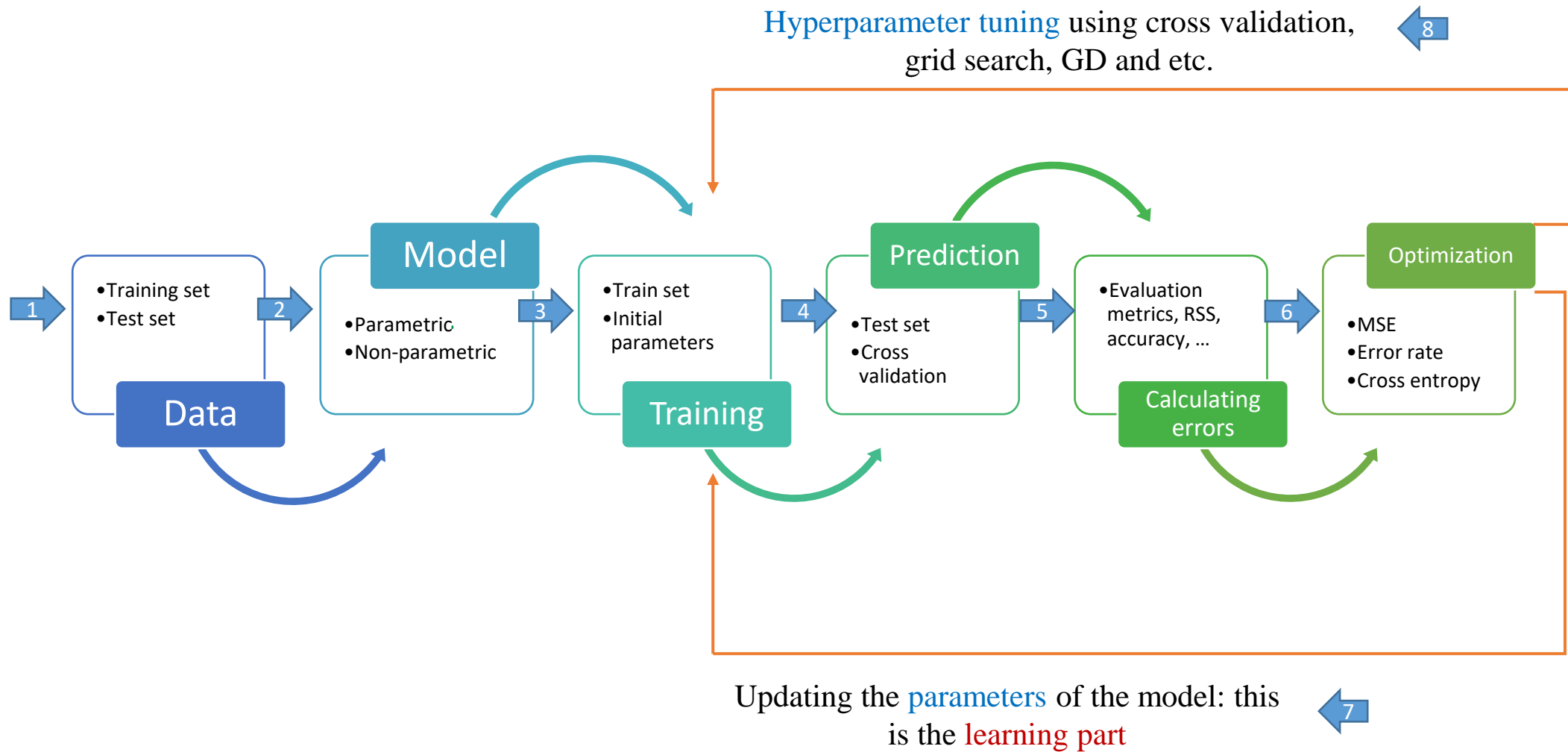
➔ How do machines learn?

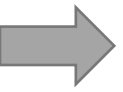
The short answer: by **Algorithms**!

- **Algorithm**: a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.
- Generally, the more data a machine learning algorithm is provided with, the more accurate it becomes.



How do machines learn?





Question of the day!





What's next?

