Artificial Intelligence I: Introduction to Data Science and Machine Learning

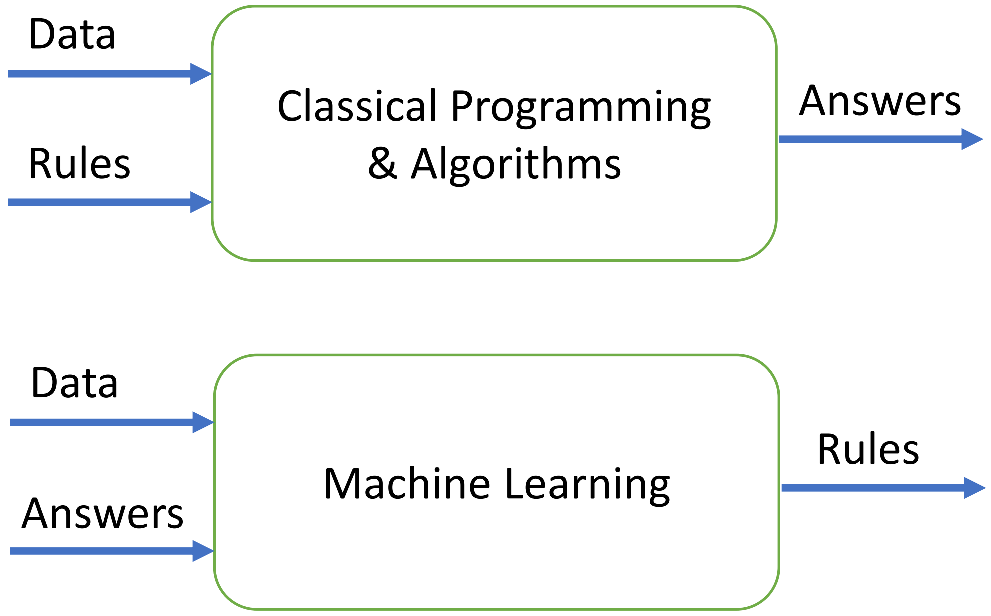
Assoc. Prof. Dr. Taner Arsan

H. Fuat Alsan, PhD(c)

Sena Kılınç, PhD(c)

Machine Learning Introduction

- Subfield of artificial intelligence
- Pattern recognition on data instead of hard coded rules



Learning Paradigms

- Supervised learning
  - Dataset with labels are required
  - Easier to evaluate
- Unsupervised learning
  - Doesn’t require labeled dataset
  - Harder to evaluate
- Also:
  - Semi-supervised learning
  - Reinforcement learning
  - Self-supervised learning
  - Weakly-supervised learning
  - Etc.

Machine Learning Tasks

- Regression:
  - Predicting a continuous value
  - Ex: predicting house prices
- Classification:
  - Given the features, decide which object class the data sample belongs to among fixed number of object classes
  - Ex: given an image, decide whether it is a car or truck
- Clustering
  - Grouping similar data (using a similarity metric)
  - Unsupervised (doesn’t require labeled dataset)
  - Ex: Customer segmentation

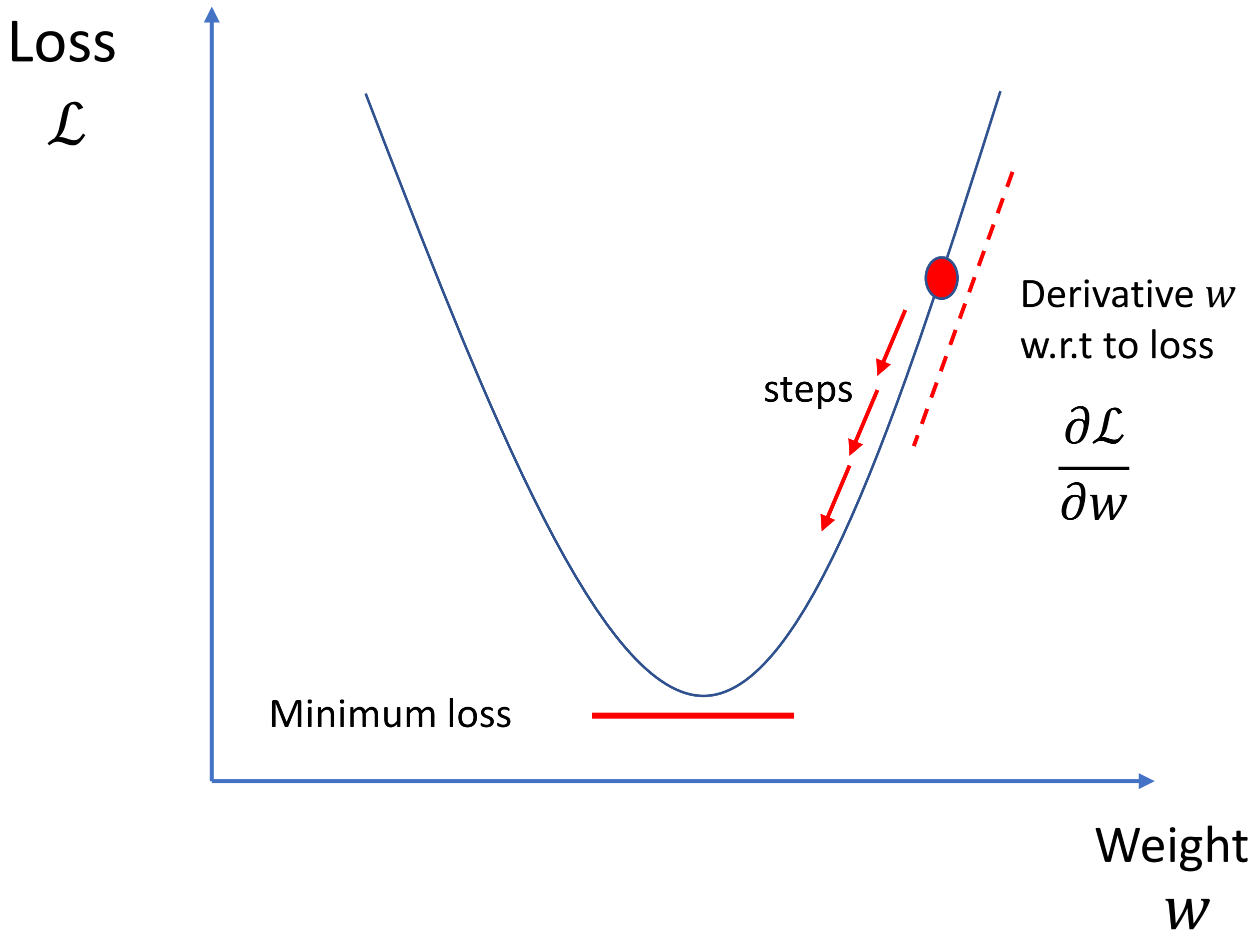
Data Generation

- Datasets have licences, legal issues for commercial applications
- Generate synthetic data, avoid legal issues
- For regression we can manually generate with function:
  - $y = x + \epsilon$
- We can also use sklearn functions:
  - (from sklearn import datasets)
  - datasets.make_regression()
  - datasets.make_classification()
  - datasets.make_blobs()

Gradient Descent

- In machine learning, we don’t have closed form analytical solutions for training the models
- Instead we use iterative solutions like gradient descent.
- **In each iteration, we take the derivative of the loss function with respect to parameters of the model and move towards to minimum loss step by step**
- After sufficient iterations, models are ready to make predictions

Gradient Descent (Visualized)



Example: Basic Linear Regression

- Consider basic linear model: $y = w_0 + w_1 x$
- Where w_0, w_1 are trainable parameters, x is the input feature and y is the prediction
- Loss (cost) function: $\mathcal{L}(\hat{y}, y) = \frac{1}{N}\sum_{i=1}^{N}(\widehat{y_i} - y_i)^2$
  - Where N total number of data samples, y_i is the target data, $\widehat{y_i}$ is the prediction by model
  - Mean squared error between model prediction and target data
  - Lower is better
- **Goal:** We wish to find such values of w_0, w_1 that loss function is minimized. This is called training (or fitting) the model

Calculation of Gradients

- Parameters Using power rule: $\frac{\partial}{\partial x} f(x)^n = nf(x)^{n-1}f'(x)$

- $\frac{\partial \mathcal{L}}{\partial w_0} = \frac{\partial \mathcal{L}}{\partial w_0}\frac{1}{N}\sum_{i=1}^{N}(w_0 + w_1 x_i - y_i)^2 = \frac{2}{N}\sum_{i=1}^{N}(w_0 + w_1 x_i - y_i)(1)$
- $\frac{\partial \mathcal{L}}{\partial w_0} = \frac{2}{N}\sum_{i=1}^{N}(\widehat{y_i} - y_i)$

- $\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial \mathcal{L}}{\partial w_1}\frac{1}{N}\sum_{i=1}^{N}(w_0 + w_1 x_i - y_i)^2 = \frac{2}{N}\sum_{i=1}^{N}(w_0 + w_1 x_i - y_i)(x_i)$
- $\frac{\partial \mathcal{L}}{\partial w_1} = \frac{2}{N}\sum_{i=1}^{N}(\widehat{y_i} - y_i)(x_i)$

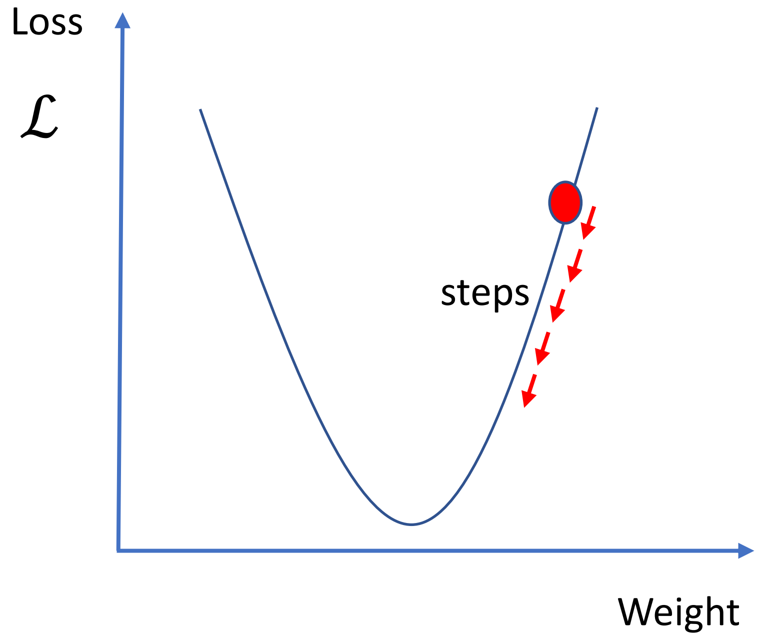
Model Optimization

- Parameters of model needs to be updated for reducing the loss
- $w_0 = w_0 - \alpha \frac{\partial \mathcal{L}}{\partial w_0}$
- $w_1 = w_1 - \alpha \frac{\partial \mathcal{L}}{\partial w_1}$

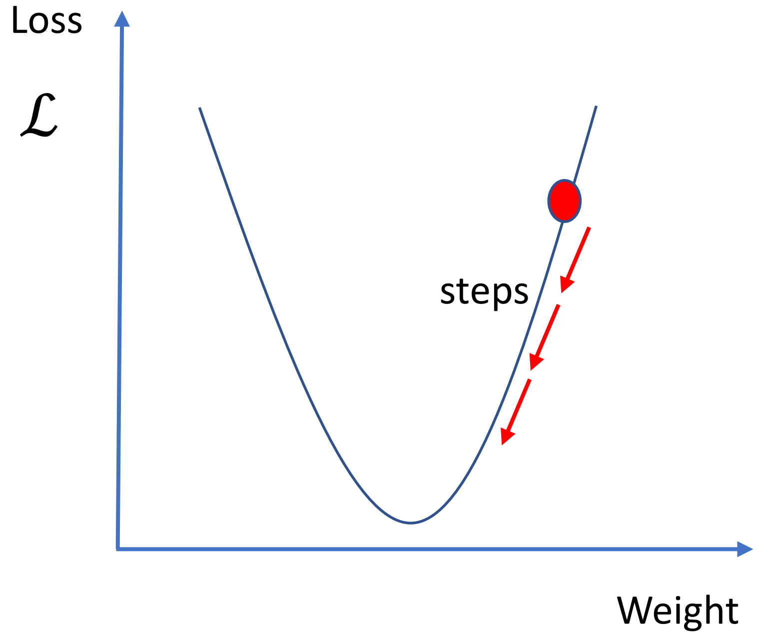
Loop until loss is minimized

- α is the learning rate, adjusts the speed of learning
  - Must be chosen carefully
  - If too small -> training takes very long time
  - If too large -> training goes unstable

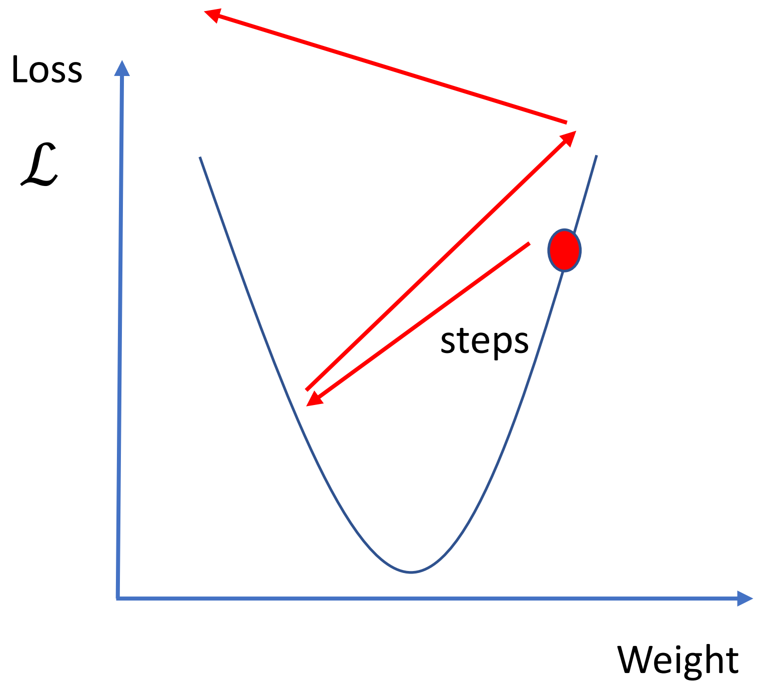
Choosing Learning Rate Properly



Low Learning Rate
Small steps, training takes too long



Just Right Learning Rate
The optimal learning rate



High Learning Rate
Steps are too large and causes instability in training

Second Example: Polynomial Regression

- Consider quadratic model: $y = \ w_0 + \ w_1 x + w_2 x^2$
- Computing the gradients same way before, we get:
  - *(computations are skipped but you should derive it on your own for exercise)*

- $\frac{\partial \mathcal{L}}{\partial w_0} = \frac{2}{N} \sum_{i=1}^{N} (\widehat{y_i} - y_i)$
- $\frac{\partial \mathcal{L}}{\partial w_1} = \frac{2}{N} \sum_{i=1}^{N} (\widehat{y_i} - y_i)(x_i)$
- $\frac{\partial \mathcal{L}}{\partial w_2} = \frac{2}{N} \sum_{i=1}^{N} (\widehat{y_i} - y_i)(x_i^2)$

Polynomial Regression Model Optimization

- The same but we have three learnable parameters

- $w_0 = w_0 - \alpha \dfrac{\partial \mathcal{L}}{\partial w_0}$
- $w_1 = w_1 - \alpha \dfrac{\partial \mathcal{L}}{\partial w_1}$
- $w_2 = w_2 - \alpha \dfrac{\partial \mathcal{L}}{\partial w_2}$

Loop until loss is minimized

Gradient Descent Important Notes

- Exploding gradients
  - Gradient values can get very large numbers (watch out for that)
- Vanishing gradients
  - Gradient values can get very small numbers (watch out for that)
- Some functions have limiting properties
  - Ex: log() function cannot take negative numbers (watch out for that)
- Learning rate should be chosen carefully

Other Training Methods

- Least Squares
  - Simple but can be only applied to basic models (ex: Linear Regression)
- Maximum Likelihood Estimation (MLE)
  - Useful for fitting probability distributions
- Maximum A Posteriori (MAP) Estimation
  - Similar to MLE but uses Bayes Rule