Machine Learning

- Gradient Descent Algorithm
- Linear Regression
- Non-Linear Regression
- Logistic Regression
- Decision Trees
  - Regression Trees
  - Classification Trees
- Clustering Algorithms
  - K-Means
  - Hierarchical clustering
  - DB-Scan
  - Mean Shift
  - GMM
- Support Vector Machine

Deep Learning

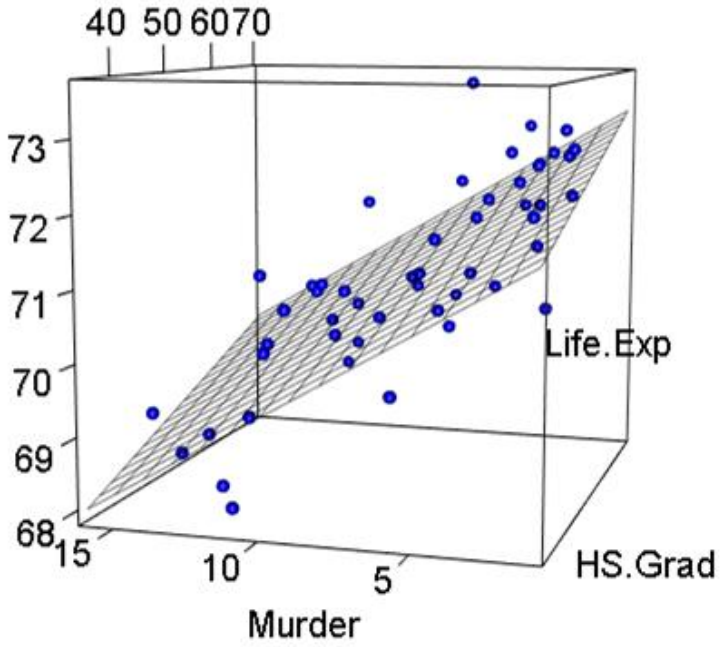
- MLP
- CNN

Datasets

- Breast Cancer Wisconsin
- MIMIC-III
- Framingham Heart Study
- Alzheimer’s Disease Neuroimaging Initiative
- Drug discovery
- Microbiome

C2 - Multivariable Linear Regression

Multiple Regression → $y = m_1x_1 + m_2x_2 + b$



Univariable Linear Regression

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

gradient descent

(Vanilla) Gradient Descent

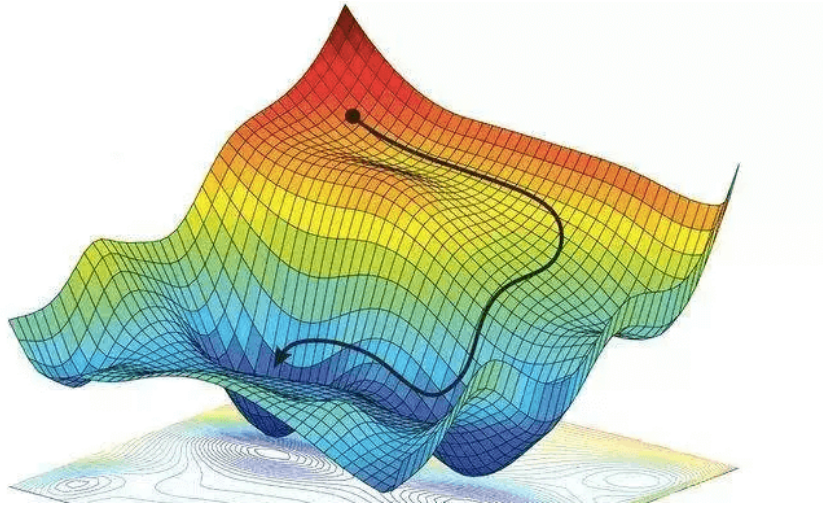
First-order optimization algorithm to find a local minimum/maximum of a given function.
Only two specific requirements (differentiable, convex).

$$p_{n+1} = p_n - \eta \nabla f(p_n)$$

Repeat until convergence {

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

}



Convergence is determined by two ways: **max_iter**, **tol**.
Tradeoff smaller/greater learning rate.

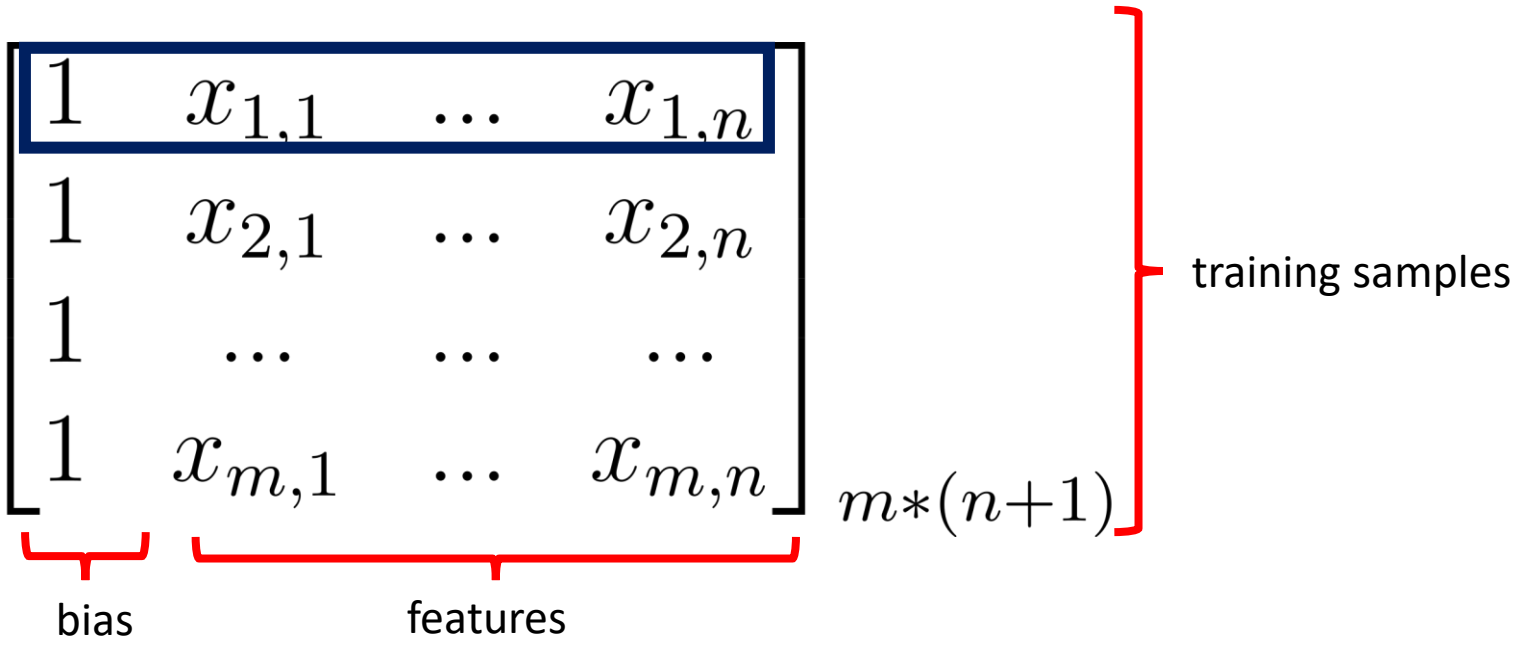
Multivariable Linear Regression

Hypothesis:

$$h(x^{(i)}) = w_0 * x_0^{(i)} + w_1 * x_1^{(i)} + w_2 * x_2^{(i)} + \ldots + w_n * x_n^{(i)}$$

Where $x^{(i)}$ represents the i-th training sample.
And $x_0^{(i)}$ is 1 (bias) $\forall\, i$.

Dataset:



Multivariable Linear Regression

Hypothesis:

$$h(x^{(i)}) = w_0 + w_1 * x_1^{(i)} + w_2 * x_2^{(i)} + ... + w_n * x_n^{(i)}$$

Parameters:

$$w = \begin{bmatrix} w_0 & w_1 & ... & w_n \end{bmatrix}_{1*(n+1)}$$

Prediction:

$$h(x_j) = x_j \times w^t$$

Multivariable Linear Regression

.

$$h(x_j) = x_j \times w^t$$

$$h(x_j) = \begin{bmatrix} 1 & x_{j,1} & \dots & x_{j,n} \end{bmatrix}_{1*(n+1)} \times \begin{bmatrix} w_0 \\ w_1 \\ \dots \\ w_n \end{bmatrix}_{(n+1)*1} = \text{model prediction}$$

$$h(x_j) = \begin{bmatrix} 1 & x_{j,1} & \dots & x_{j,n} \end{bmatrix}_{1*(n+1)} \times \begin{bmatrix} w_0 \\ w_1 \\ \dots \\ w_n \end{bmatrix}_{(n+1)*1} = y_j$$

Multivariable Linear Regression

Given m training samples.

$$\begin{bmatrix} 1 & x_{1,1} & \dots & x_{1,n} \\ 1 & x_{2,1} & \dots & x_{2,n} \\ 1 & \dots & \dots & \dots \\ 1 & x_{m,1} & \dots & x_{m,n} \end{bmatrix}_{m*(n+1)} \times \begin{bmatrix} w_0 \\ w_1 \\ \dots \\ w_n \end{bmatrix}_{(n+1)*1} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}_{m*1}$$

And update parameters via gradient-descent algorithm minimizing the loss function (MSE).

Multivariable Linear Regression

Update parameters via gradient-descent algorithm

$$\frac{\partial L}{\partial w_0} = \frac{1}{m}\sum_{i=0}^{m}(y_i - h(x^i))(-1)$$

$$\frac{\partial L}{\partial w_{j\neq 0}} = \frac{1}{m}\sum_{i=0}^{m}(y_i - h(x^i))(-x_j^i)$$

C2 - Non – Linear Regression

Bibliography

https://towardsdatascience.com/multiple-regression-as-a-machine-learning-algorithm-a98a6b9f307b

Code