LOGISTIC REGRESSION, PERCEPTRON AND SOFTMAX REGRESSION

Activation Functions

Ahmed Hani

July 19, 2017

FCIS'17 Machine Learning Course

Today's Objectives

Learning

- Logistic Regression - A powerful binary classifier
- Softmax Regression - A Multi-class classification
- Activation Functions for Decision-making

To Do

- Implement Logistic and Softmax Regression using Numpy
- Introducing Plant-Iris and Titanic-survival Datasets

Classification

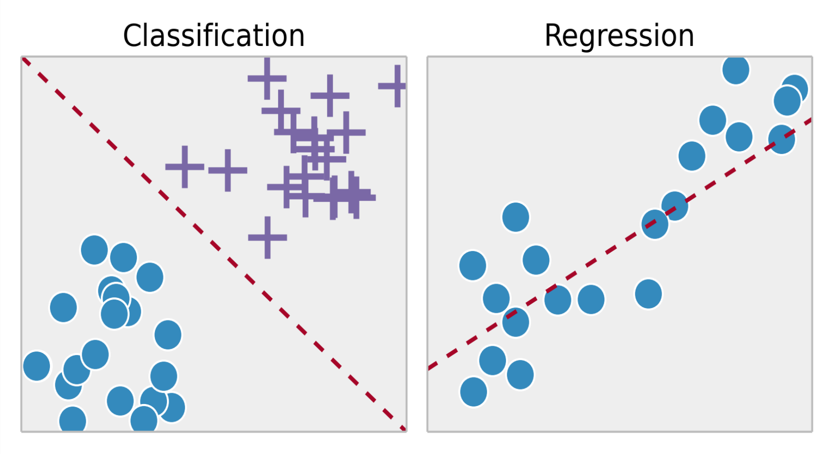
- Given a features vector *X = [x0, x1, ..., xn]*, the target is to get *Y*, where *Y* is a category, class or label which is the entity representation to the given features
- In probabilistic term, the target is to maximize the conditional probability *P(Y | X)*
- Binary Classification:

$$Y \in \{0, 1\}$$

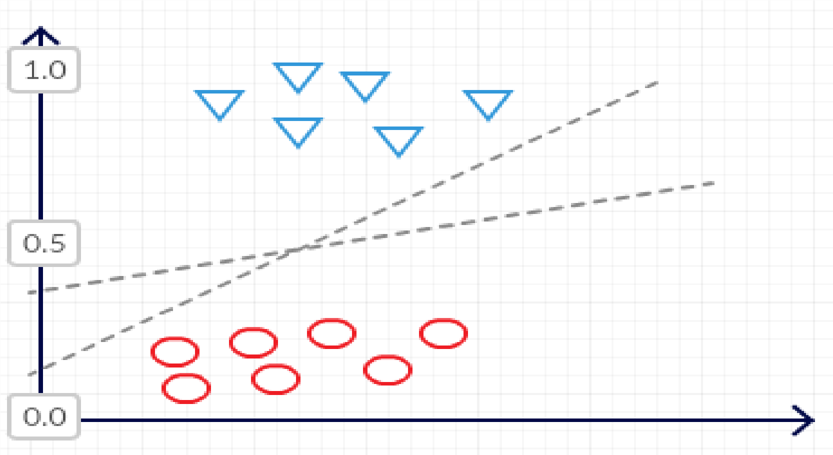
- Multi-class Classification:

$$Y \in \{0, 1, 2, .., M\}$$

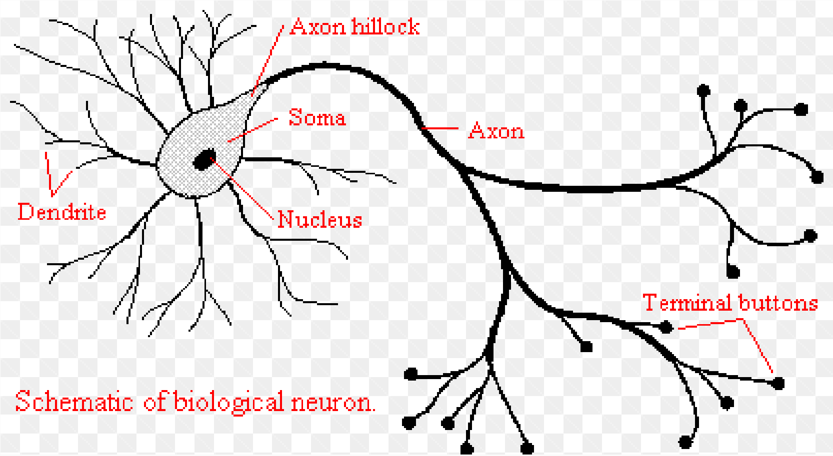
Classification vs Regression



Binary Classification



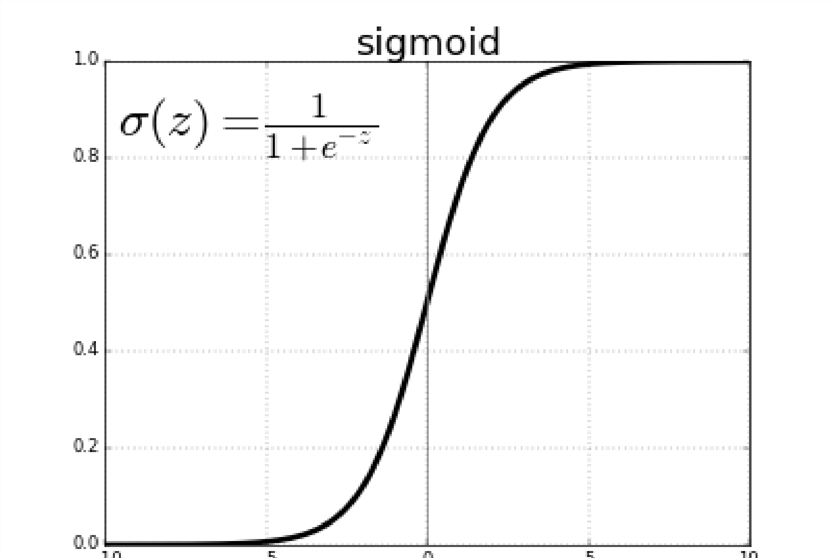
Activation Functions - Biological Inspiration



Activation Functions Properties

- Produces non-linear values
- Differentiable functions
- Has maximum and minimum value (interval)
- Examples: ***Sigmoid***, Signum, ***Tanh***, Rectified Linear Unit (***ReLU***), ***Softmax***

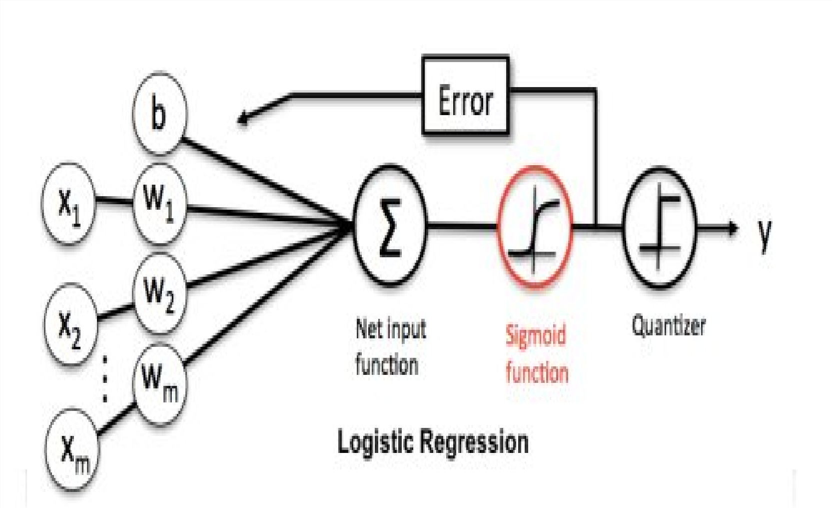
Sigmoid Function



Sigmoid Function Derivative

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] \\
&= \frac{d}{dx}\left(1+\mathrm{e}^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{\left(1+e^{-x}\right)^2} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \left(1-\frac{1}{1+e^{-x}}\right)
\end{aligned}
$$

Logistic Regression



Logistic Regression (Cont.)

Logistic Regression Model

$$0 \leq h(\theta^T x) \leq 1$$

$$h_\theta(x) = g(\theta^T x)$$

$$g(\theta^T x) = \frac{1}{1 + e^- \theta^T x}$$

Hypothesis Probabilistic Interpretation

$$P(y = 0|x; \theta) = h_\theta(x)$$

$$P(y = 1|x; \theta) = 1 - h_\theta(x)$$

$$P(y = 0|x; \theta) + P(y = 1|x; \theta) = 1$$

$$P(y|x; \theta) = (h_\theta(x))^y (1 - (h_\theta(x))^{y-1}$$

Cost Function:

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

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

Parameters Update: Cost Function (Cont.)

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

Parameters Update: Maximum Likelihood Estimation

Assume that we have *M* training examples that were generated independently, we can then write down the likelihood of the parameters as

$$
\begin{aligned}
L(\theta) &= p(\vec{y} \mid X; \theta) \\
&= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta) \\
&= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1 - h_\theta(x^{(i)})\right)^{1-y^{(i)}}
\end{aligned}
$$

Maximum Likelihood Estimation (Cont.)

It will be easier to maximize the ***log*** likelihood (Summation instead of Product)

$$\begin{aligned}\ell(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))\end{aligned}$$

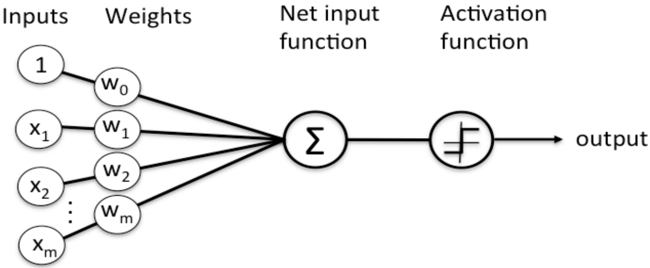
Maximum Likelihood Estimation (Cont.)

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j}\ell(\theta) &= \left(y\frac{1}{g(\theta^T x)} - (1-y)\frac{1}{1-g(\theta^T x)}\right)\frac{\partial}{\partial \theta_j} g(\theta^T x) \\
&= \left(y\frac{1}{g(\theta^T x)} - (1-y)\frac{1}{1-g(\theta^T x)}\right) g(\theta^T x)(1-g(\theta^T x)\frac{\partial}{\partial \theta_j}\theta^T x \\
&= \left(y(1-g(\theta^T x)) - (1-y)g(\theta^T x)\right) x_j \\
&= \left(y - h_\theta(x)\right) x_j
\end{aligned}
$$

Stochastic Gradient Descent Update Rule

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}$$

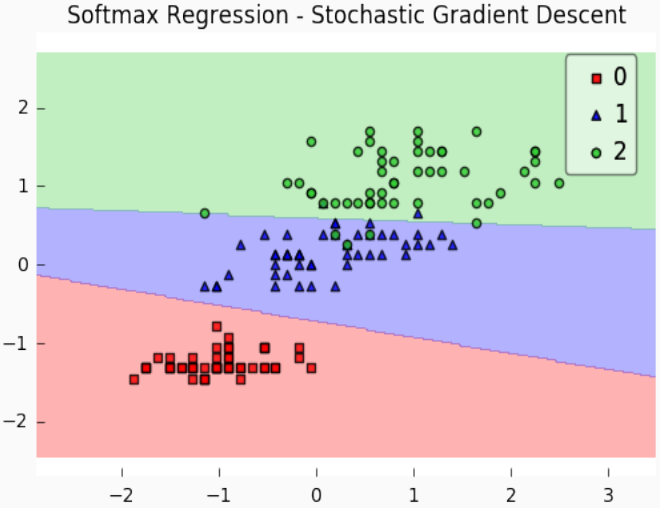
Perceptron Learning



Schematic of Rosenblatt's perceptron.

Perceptron Learning (Cont.)

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$


Softmax Regression - Stochastic Gradient Descent
0
1
2
2
1
0
−1
−2
−2
−1
0
1
2
3

Softmax Activation Function

SoftMax

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

Given

$$Z = \{z_0, z_1, z_2, .., z_k\}$$

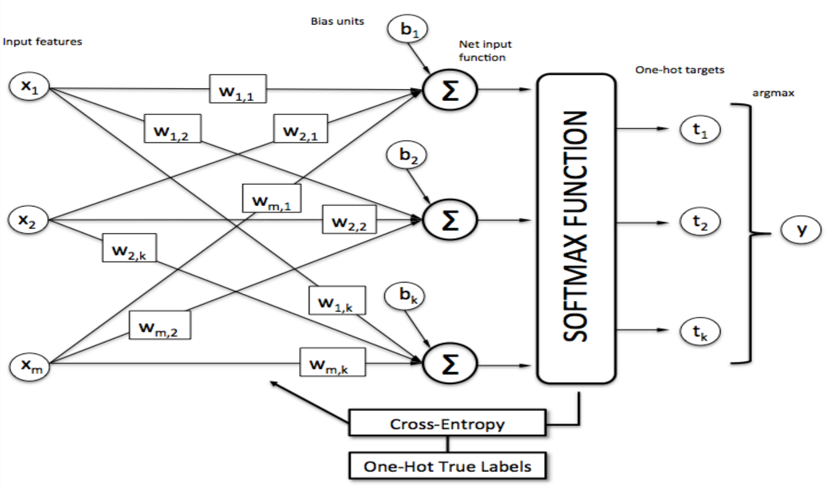
the function output

$$Z' = \{z'_0, z'_1, z'_2, .., z'_k\}$$

where the sum of the new elements equals to 1.0 (probability vector)

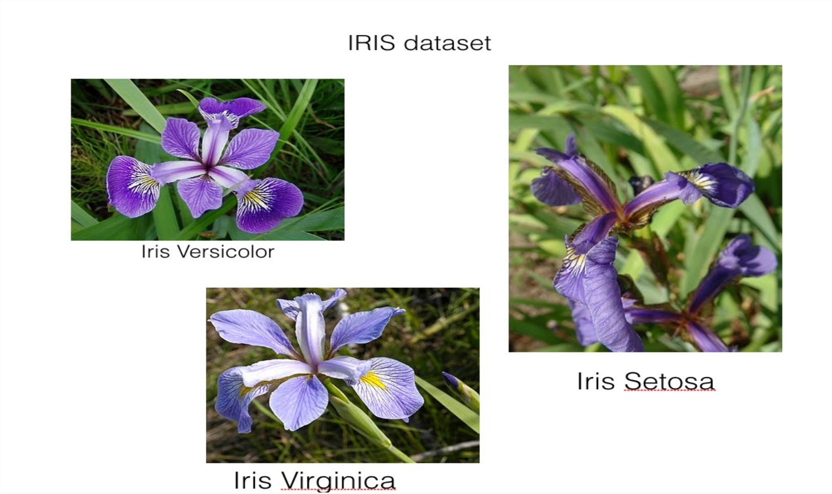
Softmax Derivative: Your Turn!

Softmax Regression Model

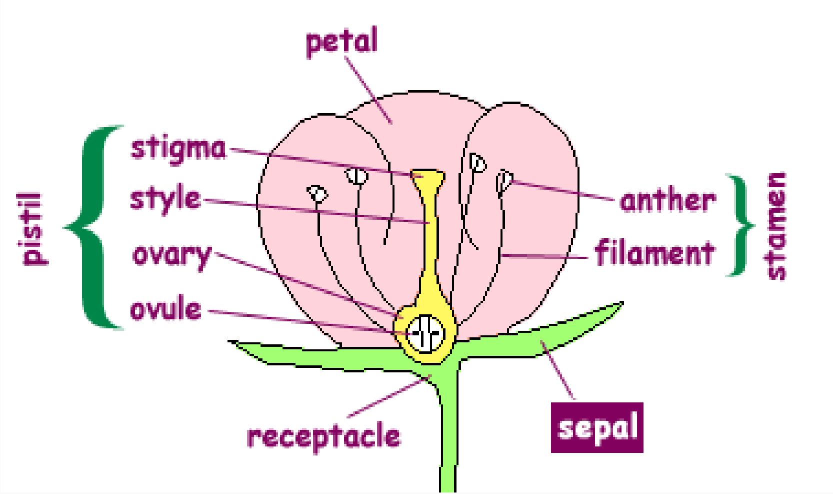


Practical

Iris Dataset



Iris Dataset (Cont.)



Iris Dataset (Cont.)

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Titanic Survival Dataset

Titanic Survival Dataset (Cont.)

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Thanks!