Supervised Learning – I

Linear Regression

Example

Advertising (X)	Sale (y)
1	1
2	1
3	2
4	2
5	4

We want to analyze the relationship between Advertising and Sales using regression analysis.

Regression analysis is a form of predictive modelling technique which investigates the relationship between a **dependent** (target) and **independent variable (s)** (predictor).

Regression analysis is an important tool for modelling and analyzing data.

Here, we fit a curve / line to the data points, in such a manner that the differences between the distances of data points from the curve or line is minimized.

Linear Regression

It is one of the most widely known modeling technique. Linear regression is usually among the first few topics which people pick while learning predictive modeling.

In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear.

Linear Regression establishes a relationship between **dependent variable (Y)** and one or more **independent variables (X)** using a **best fit straight line** (also known as regression line).

For example, in a simple regression problem (a single x and a single y), the form of the model would be:

y = B0 + B1*x

can also be written as,

$$y = \theta_0 + \theta_1 x$$

B0 is the intercept

B1 is the slope

In higher dimensions when we have more than one input (x), the line is called a plane or a hyper-plane. The representation therefore is the form of the equation and the specific values used for the coefficients (e.g. B0 and B1 in the above example).

We can add error term $\epsilon(epsilon)$ to this. So the equation for Simple Linear Regression (SLR) Model will be,

$$y = \beta_0 + \beta_1 X + \epsilon$$

Regression is a statistical tool for investigating the relationship between a dependent variable (y) and one or more independent variable (X).

X is also called as regression variable (is not a Random variable)

Y is also called as response variable (a random variable).

Conceptual Model: A change in X will make $\beta_1 X$ amount of change in Y.

Objective

The parameters β_0 & β_1 are unknown and need to be estimated for the given sample dataset,

(x1, y1), (x2, y2), …, (xn, yn)

This is also called as fitting a liner model.

Plot the lines (Illustrate the importance of error),

Least square estimation

It is one of the method for fitting a line for a sample dataset

The error term is usually defined as Mean squared error (MSE)

$$E = \frac{1}{N}\sum_{p=1}^{N}\left\|y_p - \hat{y}_p\right\|^2$$

Basic assumptions and derivation of Linear Regression Model coefficients (Refer notes)

Logistic Regression

First we start with the definition of classification

Then we will link this problem to that of regression

Classification: It is a technique concerned with separating distinct set of objects (or observations) to previously defined group (labeled class).

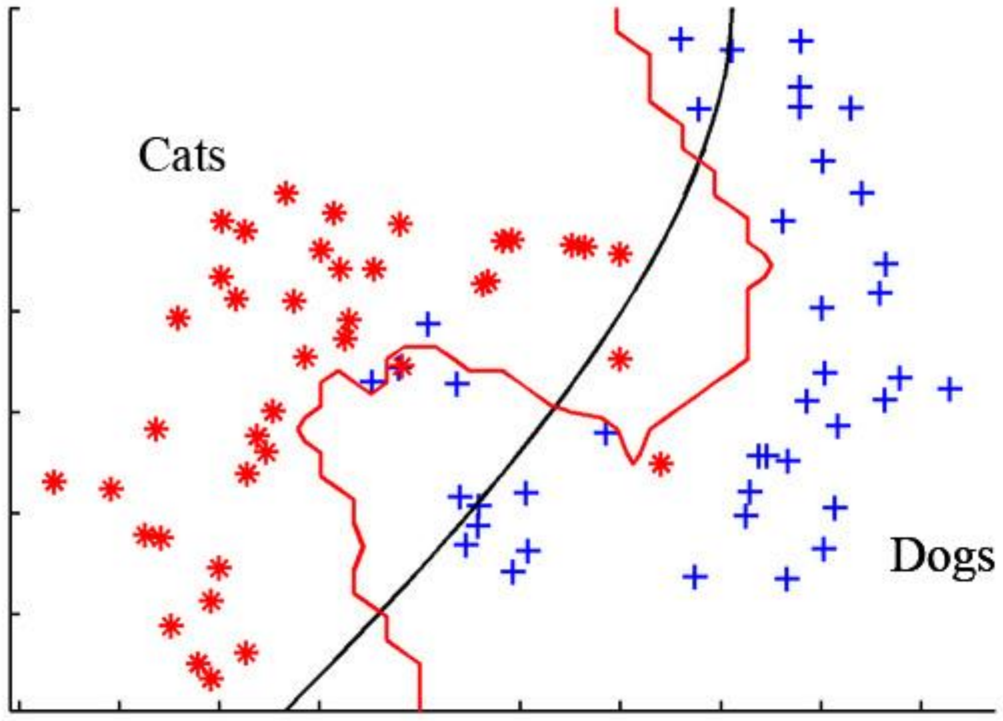
In case of Linear regression , the response variable is quantitative (continuous)

In case of classification, the response variable is qualitative (discrete)

Predicting a qualitative response for an observation is referred as classification.

Logistic regression is one of the techniques used to predict the qualitative response.

Consider an example of classifying an image as cats or dogs



Here the line that we see acts as a separator of the points of two classes rather than fitting the points. The line or the surface or the curve is called as decision surface.

The data points which lie on the decision surface are the points for which probability of it being either class1 (cat) or class2 (dog) is same.

$P(c1| x) = P(C2 | x)$

Hence, for the data points that lie on right side of the decision surface says that,

The probability that the data point belongs to class2 (dog) is high compared to the probability that the data point belongs to class1 (cat)

$P(c2|x) >= P(c1|x)$

And similarly, for the data points that lie on left side of the decision surface says that,

The probability that the data point belongs to class1 (cat) is high compared to the probability that the data point belongs to class2 (dog)

$P(c1|x) >= P(c2|x)$

In classification, we would like to compute the probability that a data point belongs to a particular class $P(c|x)$.

How regression can be used to solve this problem?

(x1, y1) x1=<26, 30> y1=cat

(x2, y2) x2=<17, 28> y2=dog

Can be converted as

(x1, y1) x1=<26, 30> y1=0

(x2, y2) x2=<17, 28> y2=1

Using linear regression,

$f(x) = P(Y=1|x)$

If the value for f(x) is > 0.5 we can say the data point belongs to class1

If the value for f(x) < 0.5 , we can say the data point belongs to class2

Problems

Linear regression is not limited in range

Output can be negative (So will be difficult to interpret as probability)

Solution

Use Linear regression for a transformed function.

The transformed function is called as Logistic (Logit) function,

Let p(x) denotes the p(y=1|x), then logit transformation is given by,

$$log\left[\frac{p(x)}{1-p(x)}\right]$$

Also called as **log-odds**

Now, we apply Logistic regression by trying to fit the following model,

$$log\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + x.\beta_1$$

So,

$$\left[\frac{p(x)}{1-p(x)}\right] = e^{\beta_0+x.\beta_1}$$

$$p(x) = e^{\beta_0+x.\beta_1} - p(x).e^{\beta_0+x.\beta_1}$$

$$p(x)\left(1+e^{\beta_0+x.\beta_1}\right) = e^{\beta_0+x.\beta_1}$$

$$p(x) = \frac{\left(e^{\beta_0+x.\beta_1}\right)}{\left(1+e^{\beta_0+x.\beta_1}\right)}$$

$$p(x) = \frac{1}{\left(1 + e^{-(\beta_0 + x.\beta_1)}\right)} \qquad (1)$$

Also called as Sigmoid function

During training time, we know the value for y and x and we estimate the parameters for β_0, β_1 by trying to fit with the above equation.

During testing/evaluation, we estimate the probability for given x and known β_0, β_1. If p(x)>0.5, then we estimate the x as class1 and class2 otherwise.

Decision boundary is, (linear classifier)

$$\beta_0 + x.\beta_1 = 0$$

Estimating the regression coefficients

The coefficients β_0, β_1 are unknown, and must be estimated based on the available training data.

Instead of using Least square estimation, (where the objective is to minimize the error), here maximum likelihood is preferred. So the objective is to maximize the probability.

The intuition is as follows,

We seek estimates for $\widehat{\beta_0}$ and $\widehat{\beta_1}$ such that plugging these estimates such that the predicted probability $\widehat{p(x_\iota)}$ of default for individual (1), corresponds as closely as possible to the individual's observed default status.

In other words, we try find $\widehat{\beta_0}$ and $\widehat{\beta_1}$ such that plugging these estimates into the model for $p(x)$ yields a number close to one for all defaulted, and a number close to zero for all individuals who did not.

It is described with the following equation,

$$\ell(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i^{'}:y_i{'}=0} 1 - p(x_{i^{'}})$$

x_i and y_i input / output pair for positive examples

$x_{i'}$ and $y_i{'}$ input/output pair for negative examples

The estimates $\widehat{\beta_0}$ and $\widehat{\beta_1}$ are chosen such that to maximize this likelihood function.

Making Predictions

Once the coefficients are computed, then the predicted y can be computed using the equation in 1

Multiple Logistic Regressions

Here we have more than one predictor variables and one response variable. The equation (1) can be extended as,

$$log\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + x1.\beta_1 + x2.\beta_2 + \cdots + xp.\beta_p$$

$$p(x) = \frac{1}{\left(1+e^{-(\beta_0+x1.\beta_1+x2.\beta_2+\cdots+xp.\beta_p)}\right)}$$

Artificial Neural Network

Reference : Ethem Alpaydin

Neural Network derived its origin from Human brain consisting of parallel interconnection of neurons that achieves various perception tasks in very small amount of time.

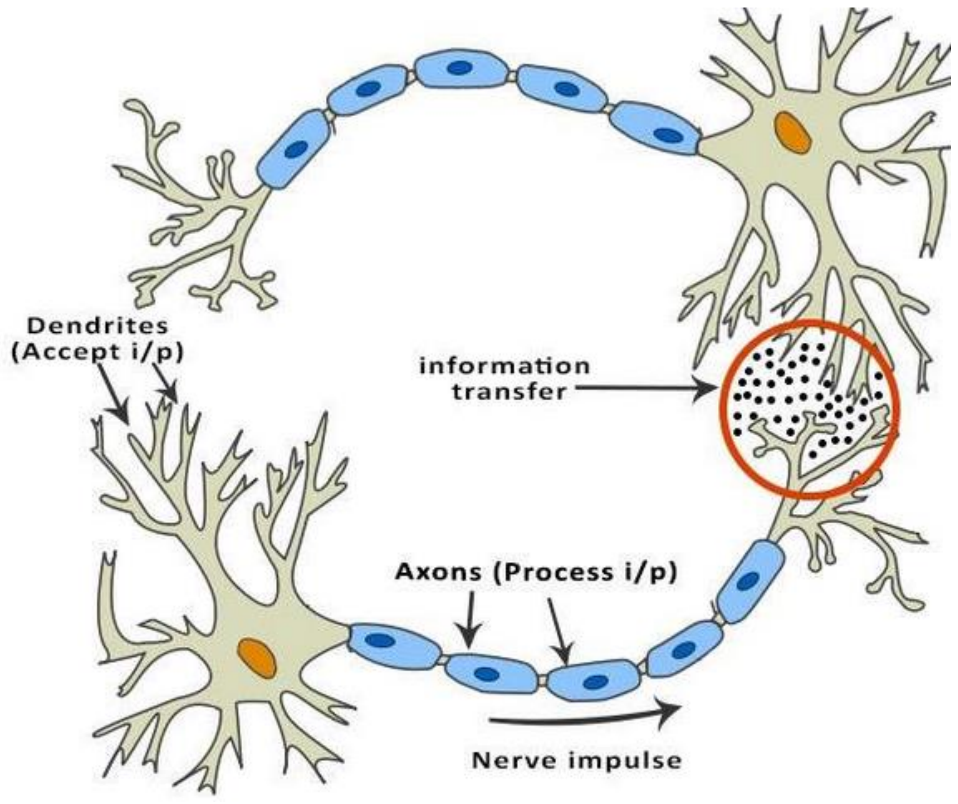
Whether computer can mimic the similar structure?

Neural Network (Natural)

Brain: is a highly complex, nonlinear & parallel computer (massively parallel)

Its main structural components are **“Neurons”**

Neurons are interconnected in a very complex manner.

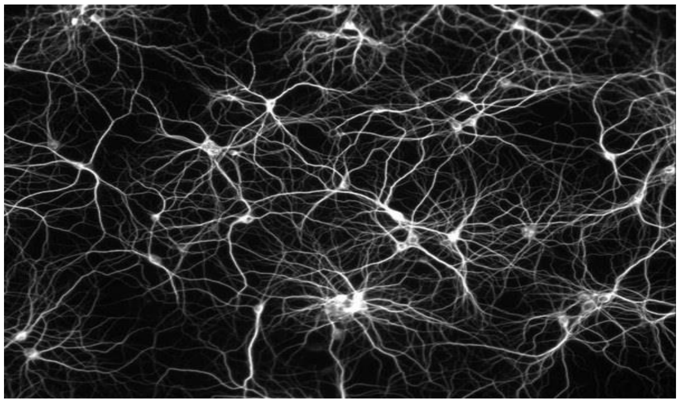


The human brain is composed of 100 billion nerve cells called **neurons.** They are connected to other thousand cells by **Axons.** Stimuli from external environment or inputs from sensory organs are accepted by dendrites. These inputs create electric impulses, which quickly travel through the neural network. A neuron can then send the message to other neuron to handle the issue or does not send it forward.

Billions of nerve cells

Trillions of interconnections

The structure of Neural Network in brain is not completely known yet



Artificial Neural Network

ANN are computers whose architecture is modeled after the brain and they typically consists of hundreds of simple processing units which are wired together in a complex communication network.

ANN cannot mimic the human brain 100%. But can be designed to perform a particular task.

Each unit or node in ANN is a simplified model of a real neuron which fires if it receives a sufficient input

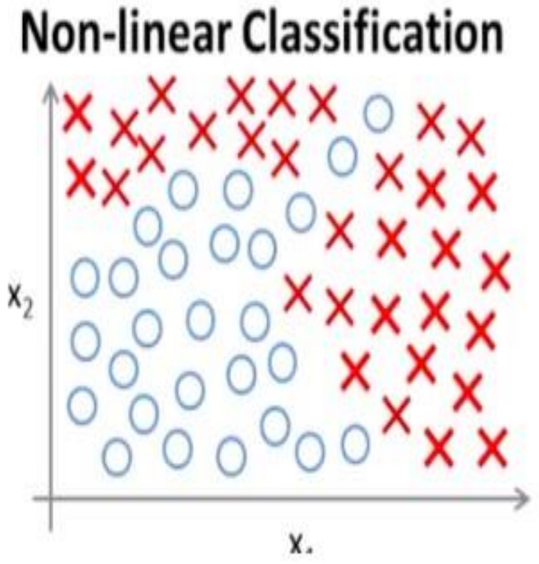


Figure 1: Example of a non-linear classification problem

In an Artificial Neural Network implemented in a computer, we use a very simple model of what a neuron does. We are going to model a neuron as a simple logistic unit

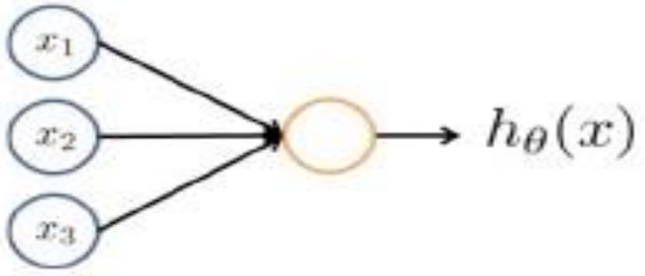
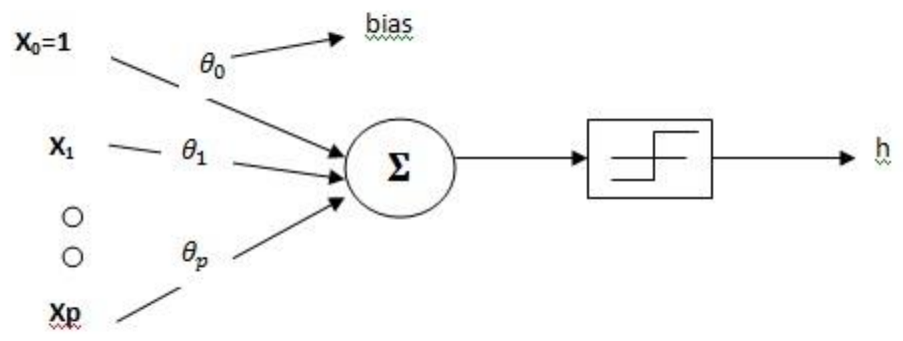


Figure 7: A simple logistic unit



Structure of simple neuron with sign function as activation function
θ is also called as the weight associated with each input.

Two functionality of a neuron

1. Summation : weighted sum of all the input along with bias

2. Activation function can be Linear or Non-linear

    a. Linear : f(x)=x;
    b. Threshold (sign function) -- Nonlinear
    f(x)={0 if x<=0 ; 1 if x>0}

    c. Sigmoid function --- Nonlinear
    f(x)=$\frac{1}{1+e^{-x}}$

The expression for a single neuron with sigmoid function as activation function can be written as,

z=$\theta_0 x_0$+$\theta_1 x_1$+…+$\theta_d x_d$

or in ANN we θ is written as w. ($z = \sum_{j=1}^{d} w_j x_j$)

h=$\frac{1}{1+e^{-z}}$

Equation 2 Sigmoid function

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

A neuron is also called as Perceptron

Derivative of sigmoid function is

$$h = \frac{1}{1+e^{-z}}$$

$$\frac{d}{dz}\left(\frac{1}{1+e^{-z}}\right) = \left(\frac{1}{1+e^{-z}}\right)(1 - \frac{1}{1+e^{-z}})$$

Example

Training a Neuron

Stochastic Online Gradient Descent Algorithm

1. Initialize the weights randomly
2. For each example in the training set
    a. Compute the predicted value $\hat{y}$ using the function h(x)
    b. Compute the error/cost function between the predicted response $\hat{y}$ and desired response (y)
    c. Update the weights according to the gradient descent
        i. Compute the derivative the cost function
        ii. For each of the weight w_j, update the weights

Elaboration of each step.

1. Initializing the weights randomly

```
For j=1:p
        Wj=rand();
```

2. **Computing error/cost function**

Minimizing the error in logistic

c regression is equivalent to maximizing the likelihood probability.

Rewriting the above equation according to neural network notation, we get,

$$L(w) = \prod_{i=1}^{m} h(x^i)^{y^i}\,(1 - h(x^i))^{1-y^i}$$

3. Update the weights

Gradient Descent: (Refer: Ethem Alpaydin)

The **gradient** is a fancy word for derivative, or the rate of change of a function. It's a vector (a direction to move) that (https://betterexplained.com/articles/vector-calculus-understanding-the-gradient/)

- Points in the direction of greatest increase of a function
- Is zero at a local maximum or local minimum (because there is no single direction of increase)

Through differential calculus, one can calculate the slope of the tangent line to a curve f(x) at a point x0. At each point x0, the derivative is the slope of a line that is tangent to the curve

Differentiation is a method to compute the rate at which a dependent output y changes with respect to the change in the independent input x

.

(Refer class notes)

We need to maximize the above function and so we take the differentiation. To make the computation simpler, before computing the differentiation, we take the logarithm (log likelihood) of the above equation.

$$l(w) = Log\left(L(w)\right)$$

$$= \sum_{i=1}^{m} y^i \log h(x^i) + \left(1 - y^i\right) \log(1 - h(x^i))$$

To maximize the above log likelihood function, compute the derivative. If we consider one training example,

$$\frac{\partial}{\partial w_j} l(w) = (y - h(x))x_j$$

Weight is updated according to the following formula

$$w_j = w_j + \alpha(y^i - h(x^i))\, x_j$$

α is called as the learning rate and is initialized to small value like, 0.01, 0.001 etc.

Artificial Neural Network Structure

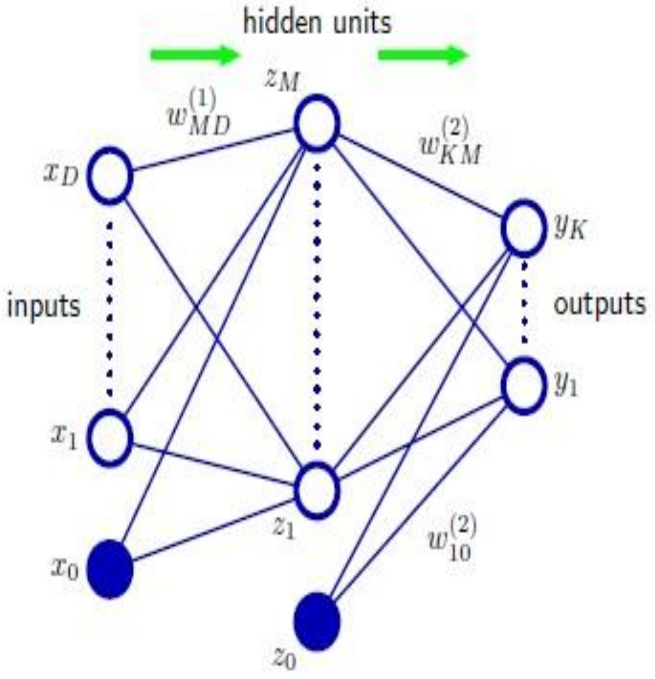
Basic unit of computing element is neuron.
ANN is built of two or more than layers
Each layer consists of many neurons.

Neural network has

1. Input layer (layer 1): in this layer we input the features x_1, x_2 … x_p (of our training set);
2. Hidden layer (layer 2): it contains values that we don't observe in the training set;
3. Output Layer (layer 3): it has a neuron / units (it can have more units) that output the final value computed by the hypothesis.

Figure Source: C. M. Bishop
\

Figure 5.1 Network diagram for the two-layer neural network corresponding to (5.7). The input, hidden, and output variables are represented by nodes, and the weight parameters are represented by links between the nodes, in which the bias parameters are denoted by links coming from additional input and hidden variables x_0 and z_0. Arrows denote the direction of information flow through the network during forward propagation.



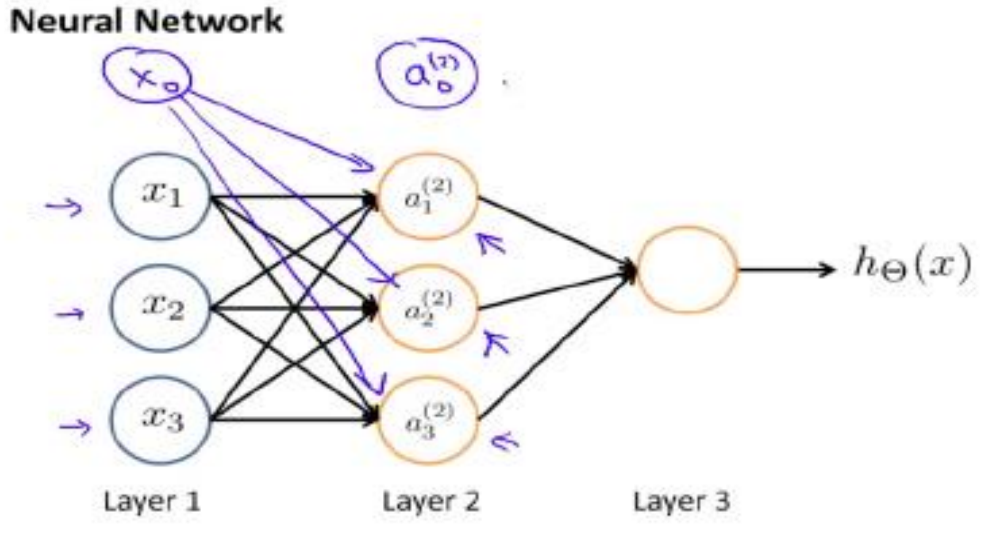


Figure 8: An example of neural network