Week 1-4: Introduction to Machine learning

Dr. Rajesh Kumar Tripathy
Assistant Professor, EEE
BITS Pilani, Hyderabad Campus

Week 1-3 (Lecture 1-8)

Syllabus: Introduction to machine learning, Supervised, unsupervised and semi-supervised learning, Classification and regression problems, Linear regression, gradient descent (Batch gradient descent and stochastic gradient descent), Logistic regression, multiclass extension of logistic regression, Performance Measures for Classifiers (binary class and multiclass), Likelihood ratio test, Bayesian Multiclass classifier with ML and MAP Criteria.

Evaluation: **Assignment 1 (Please submit the report for the assignment 1 along-with the pseudo-code)**

Textbooks:

T1. Simon Haykin, “Neural Networks – A comprehensive Foundation”, Pearson Education, 1999.
T2. H. J. Zimmermann, “Fuzzy Set Theory and its Applications”,3rd Edition, Kluwer Academic, 1996.

Reference books/Materials

R1: CS229 Lecture notes: Stanford University
R2: CS231 Convolutional neural networks for visual recognition: Stanford University
R3: http://gyan.iitg.ernet.in/handle/123456789/833
R4: https://www.sciencedirect.com/science/article/pii/S0925231206000385
R5: https://www.springer.com/cda/content/document/cda_downloaddocument/9783319284354-c2.pdf?SGWID=0-0-45-1545215-p177863021

Introduction to Pattern Recognition

- ❑Pattern recognition stems from the need for automated machine recognition of objects, signals or images, or the need for automated decision-making based on a given set of parameters or features.

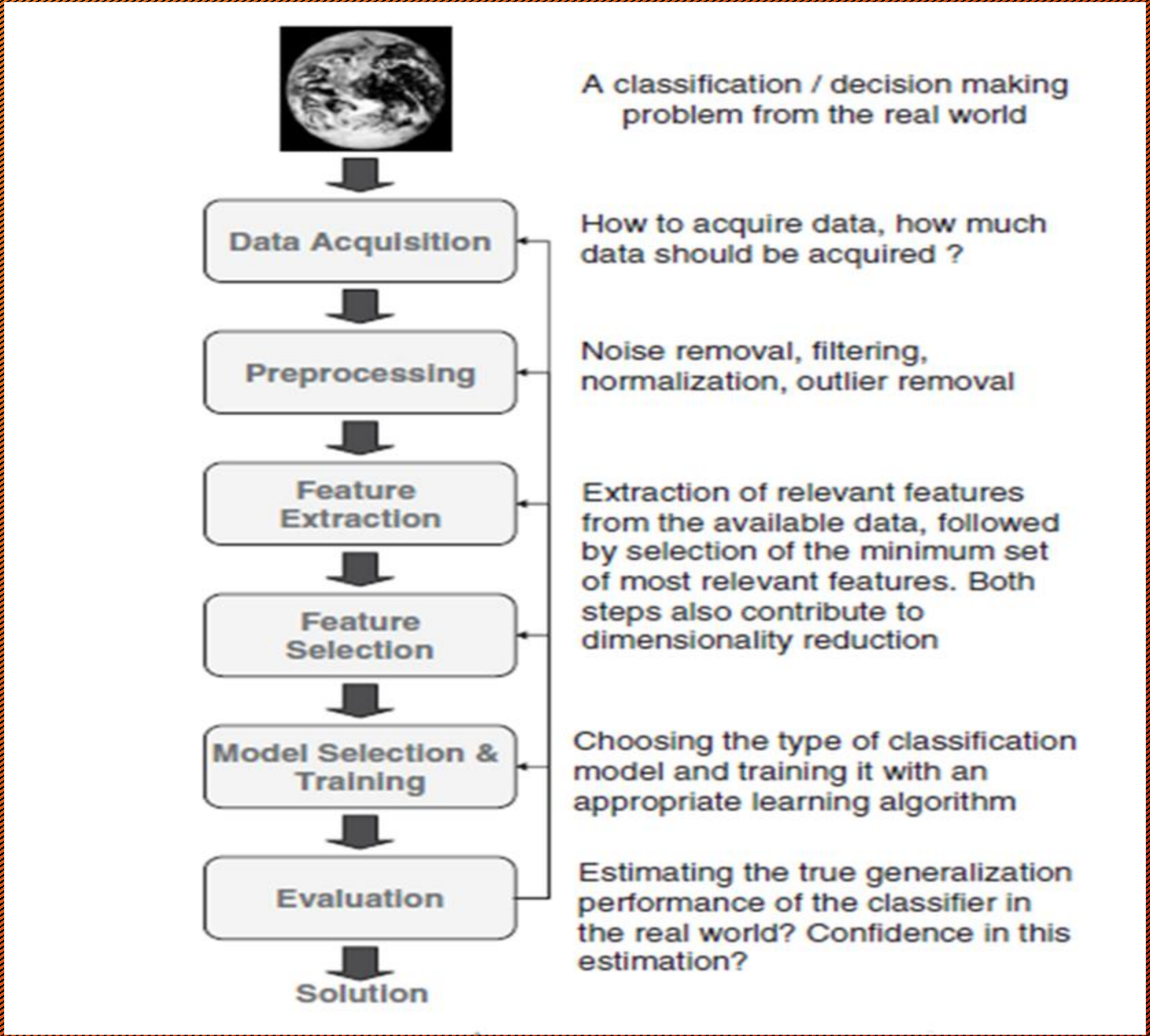
Applications:

- ❑ Speech recognition (e.g., automated voice-activated customer service)
- ❑ Speaker identification (Forensic applications)
- ❑ Handwritten character recognition (such as the one used by the postal system to automatically read the addresses on envelopes)
- ❑ Identification of a system malfunction based on sensor data

Applications:

- Loan or credit card application decision based on an individual’s credit report data
- Automated digital mammography analysis for early detection of cancer
- Automated electrocardiogram (ECG) or electroencephalogram (EEG) analysis for cardiovascular or neurological disorder diagnosis
- Biometrics (personal identification based on biological data such as iris scan, fingerprint, Heart sound, ECG, etc.).

Components of Pattern Recognition System


A classification / decision making problem from the real world
Data Acquisition
How to acquire data, how much data should be acquired ?
Preprocessing
Noise removal, filtering, normalization, outlier removal
Feature Extraction
Extraction of relevant features from the available data, followed by selection of the minimum set of most relevant features. Both steps also contribute to dimensionality reduction
Feature Selection
Model Selection & Training
Choosing the type of classification model and training it with an appropriate learning algorithm
Evaluation
Estimating the true generalization performance of the classifier in the real world? Confidence in this estimation?
Solution

What is Machine Learning?

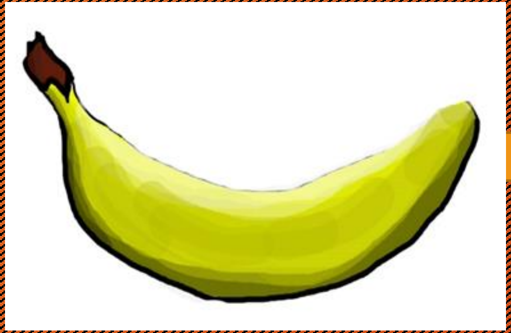
- Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

- Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.

- Machine learning is broadly categorized into three types as supervised learning, unsupervised learning and semi supervised learning.

Supervised Learning

Test data



$$\begin{tikzcd} \text{Test data} \arrow[r] & \boxed{\text{Trained Model}} \arrow[r] & \text{Test label (Banana)} \end{tikzcd}$$

Unsupervised Learning

- Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance.

Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

Semi-supervised Learning

- This kind of learning fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data.

- The systems that use this method are able to considerably improve learning accuracy.

- Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it.

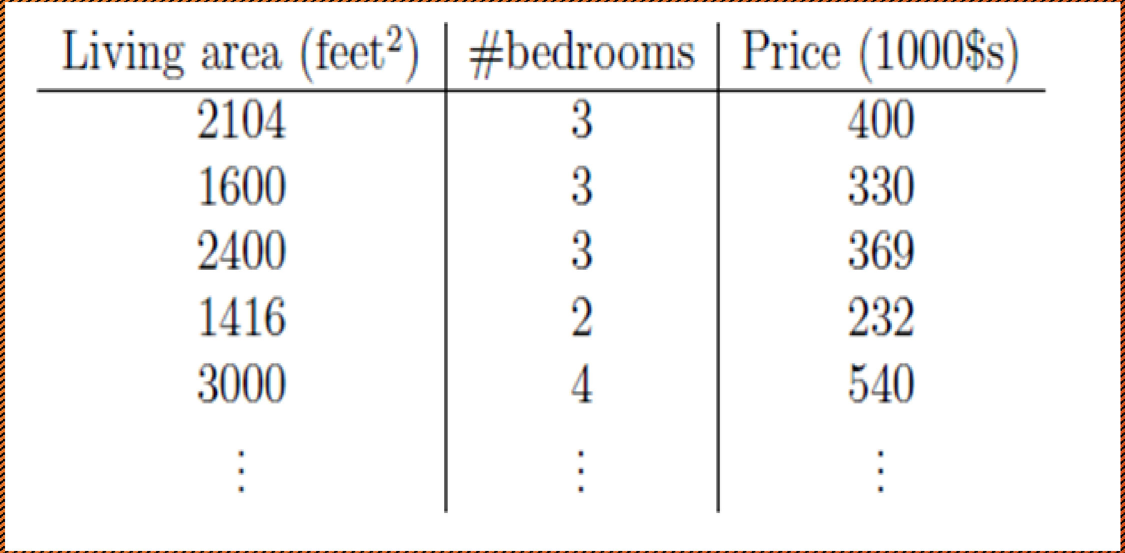
Category of Supervised Learning

- ❑ Supervised learning classified into two categories of algorithms:

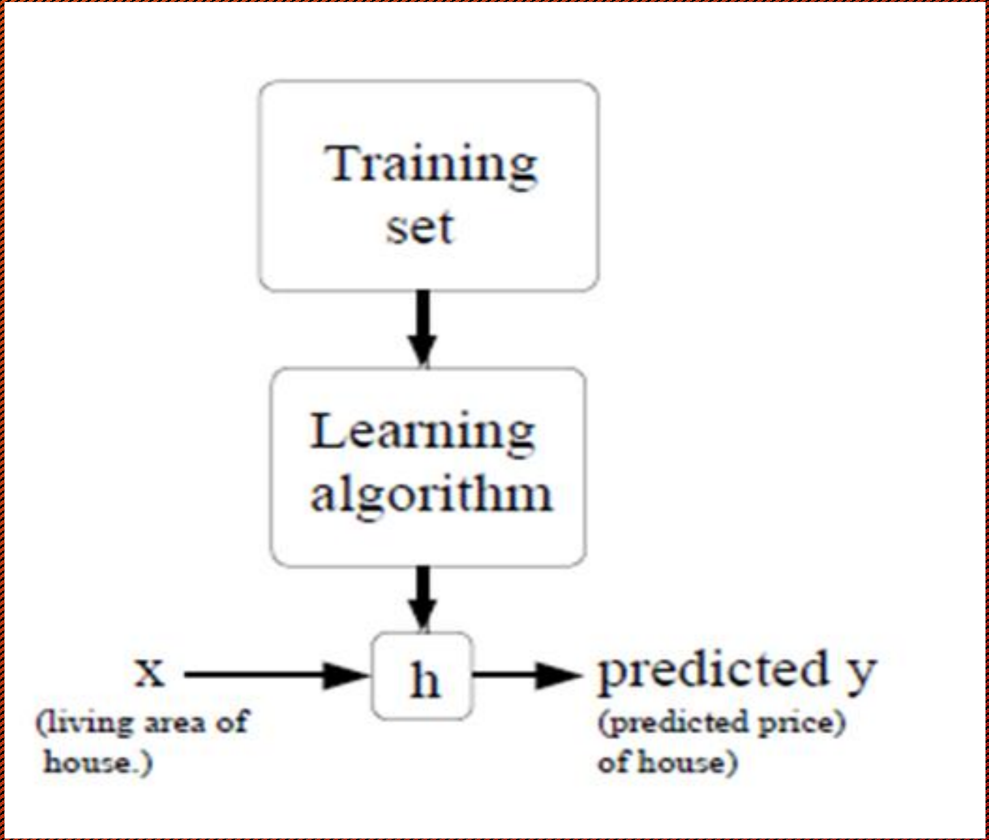
- ❑ **Classification**: A classification problem is when the output variable is a category, such as “Red” or “blue” or “disease” and “no disease”.

- ❑ **Regression**: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Linear Regression



Living area ($feet^2$)	#bedrooms	Price (1000$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮



- ❑ Living area, number of bedrooms: Features or attributes.
- ❑ Price of the house: Output (for regression problems) and class labels (classification problems)
- ❑ x i is the feature vector for ith instance and it is given by x i = [x1, x2]; where x1 is the Living area and x2 is the number of bedrooms. y i is the output or price of the house for i th instance.

Initially, the hypothesis is given as

$$h_w(\mathrm{x}) = w_0 + w_1x_1 + w_2x_2$$

We can also write in the intercept form as

$$h_w(\mathrm{x}) = \sum_{j=0}^{n} w_jx_j = w^T\mathrm{x}$$

Where ‘n’ is the number of input variables or features. Here, for house price prediction problem, n is given as 2.

The cost function can be defined as

$$J(w) = \frac{1}{2}\sum_{i=1}^{m}(h_w(\mathrm{x}^{(i)}) - y^{(i)})^2$$

Our objective is to evaluate the parameter ‘w’ so as to minimize the above cost function. The popular Least mean square (LMS) algorithm is widely used to estimate the parameter ‘w’.

LMS algorithm considers the gradient descent, which start with some initial theta and repeatedly perform the update as

$$w_j^{t+1} := w_j^{t} - \alpha \frac{\delta J(w)}{\delta w_j}$$

Where α is the learning rate and its value varies from 0 to 1.

This is a natural algorithm which takes a step in the direction of steepest decrease of the cost function ‘J’. Now, the partial derivative is evaluated as

$$\frac{\delta J(w)}{\delta w_j} = \frac{\delta}{\delta w_j}[\frac{1}{2}\sum_{i=1}^{m}(h_w(\mathrm{x}) - y)^2]$$

For one instance, the partial derivative is evaluated as

$$\begin{aligned}\frac{\delta J(w)}{\delta w_j} \\ &= 2.\frac{1}{2}(h_w(\mathrm{x}) - y).\frac{\delta}{\delta w_j}(h_w(\mathrm{x}) - y) \\ &= (h_w(\mathrm{x}) - y).\frac{\delta}{\delta w_j}(\sum_{j=0}^{n} w_j x_j - y) \\ &= (h_w(\mathrm{x}) - y).x_j\end{aligned}$$

Thus, for a single training instance (i th instance), the LMS update rule is given by

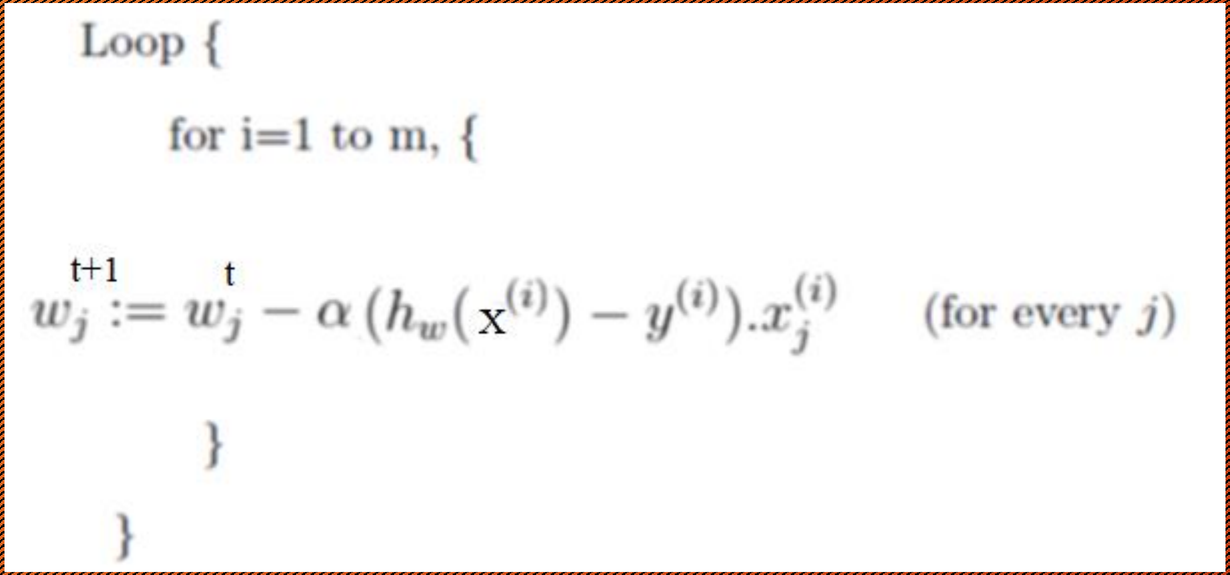
$$w_j^{t+1} := w_j^{t} - \alpha(h_w(\mathrm{x}^{(i)}) - y^{(i)}).x_j^{(i)}$$

For entire training set, the LMS update rule is given as

$$\text{Repeat until convergence } \{$$
$$w_j^{t+1} := w_j^{t} - \alpha \sum_{i=1}^{m} (h_w(\mathrm{x}^{(i)}) - y^{(i)}).x_j^{(i)} \quad (\text{for every } j)$$
$$\}$$

Where ‘j’ is the number of attributes or features with j=0, 1, 2…., n and the training instances varies from i=1, 2, …, m. This method looks at every example in the entire training set on every step, and is called batch gradient descent.

Stochastic gradient descent



```
Loop {
    for i=1 to m, {
        $w_j^{t+1} := w_j^{t} - \alpha \left(h_w(x^{(i)}) - y^{(i)}\right).x_j^{(i)}$    (for every $j$)
    }
}
```

- In this algorithm, we repeatedly run through the training set, and each time we encounter a training example, we update the parameters according to the gradient of the error with respect to that single training example only. This algorithm is called stochastic gradient descent (also incremental gradient descent).

- Often, stochastic gradient descent gets w “close” to the minimum much faster than batch gradient descent.

Regularized Linear Regression (Ridge Regression)

- ❑ Regularization helps to deal with the bias-variance problem of model development. When small changes are made to data, such as switching from the training to testing data, there can be wild changes in the estimates. Regularization can often smooth this problem out substantially.

- ❑ For highly correlated features, the regularization is helpful for smoother minimization of the cost function.

The cost function for ridge regression is given by

$$J(w) = \frac{1}{2}[\sum_{i=1}^{m}(h_w(\mathrm{x}^{i}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} w_j^2]$$

The partial derivative with respect to w_j is given by

$$\frac{\delta J(w)}{\delta w_j} = \frac{\delta}{\delta w_j}\frac{1}{2}[\sum_{i=1}^{m}(h_w(\mathrm{x}) - y)^2 + \lambda\sum_{j=1}^{n} w_j^2]$$

For one instance, the partial derivative is evaluated as

$$\frac{\delta J(w)}{\delta w_j}$$
$$= 2.\frac{1}{2}(h_w(\mathrm{x}) - y)\frac{\delta}{\delta w_j}(h_w(\mathrm{x}) - y) + 2.\frac{1}{2}\lambda w_j$$
$$= (h_w(\mathrm{x}) - y).\frac{\delta}{\delta w_j}(\sum_{j=0}^{n} w_j x_j - y) + \lambda w_j$$
$$= (h_w(\mathrm{x}) - y).x_j + \lambda w_j$$

The gradient descent for ridge regression is given by

$$\overset{t+1}{w_j} := \overset{t}{w_j} - \alpha[\sum_{i=1}^{m}(h_w(\mathrm{x}^{(i)}) - y^{(i)}).x_j^{(i)} + \lambda w_j]$$
$$= (1 - \alpha\lambda)w_j - \alpha[\sum_{i=1}^{m}(h_w(\mathrm{x}^{(i)}) - y^{(i)}).x_j^{(i)}]$$

Implementation of Linear Regression

```
%%%%%X is the feature matrix X=[x0, x1] and y is the output.
%%%%%%%%Weight vector is w= [ w0, w1]%%%%%

For i = 1 to K %%%%assign number of iteration initially (K)

                    T1 = w(1) - alpha * ((X * w ) - y)' * X(:, 1);
                    T2 = w(2) - alpha * ((X * w ) - y)' * X(:, 2);
                    w (1) = T1;
                    w (2) = T2;
                   J (i) = evaluatecostfunction (X, y, w);

End of the for loop
```

Vectorization based Linear regression

$$J(\mathbf{w}) = \frac{1}{2}\left\|\mathbf{y} - \mathbf{w}\mathbf{X}\right\|_2^2$$

$$= \frac{1}{2}(\mathbf{y} - \mathbf{w}\mathbf{X})^T(\mathbf{y} - \mathbf{w}\mathbf{X})$$

$$= \frac{1}{2}[\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{w}\mathbf{X} - \mathbf{X}^T\mathbf{w}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}]$$

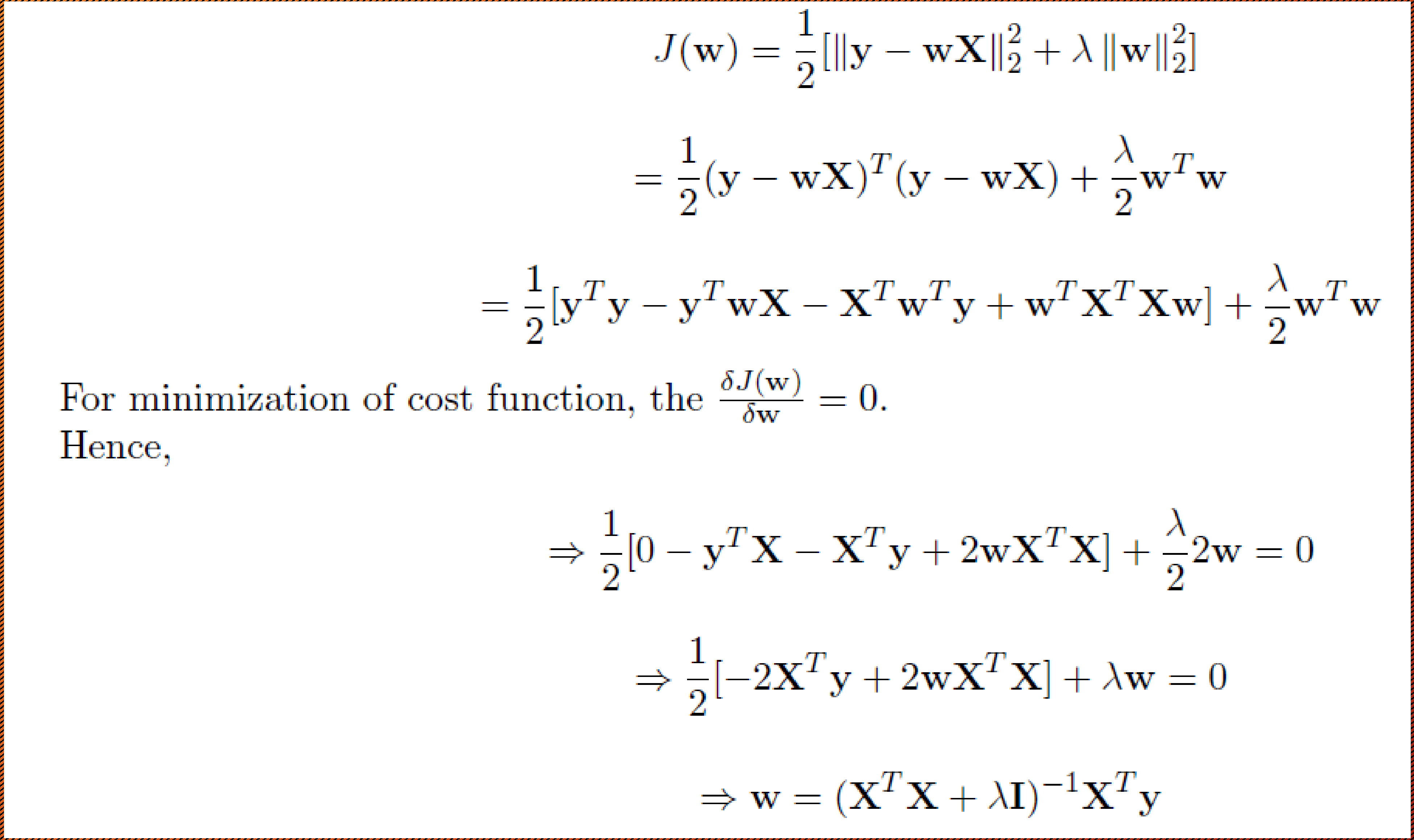
For minimization of cost function, the $\frac{\delta J(\mathbf{w})}{\delta \mathbf{w}} = 0$.
Hence,

$$\Rightarrow \frac{1}{2}[0 - \mathbf{y}^T\mathbf{X} - \mathbf{X}^T\mathbf{y} + 2\mathbf{w}\mathbf{X}^T\mathbf{X}] = 0$$

$$\Rightarrow \frac{1}{2}[-2\mathbf{X}^T\mathbf{y} + 2\mathbf{w}\mathbf{X}^T\mathbf{X}] = 0$$

$$\Rightarrow \mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Vectorization based Ridge regression



$$J(\mathbf{w}) = \frac{1}{2}[\|\mathbf{y} - \mathbf{w}\mathbf{X}\|_2^2 + \lambda\,\|\mathbf{w}\|_2^2]$$

$$= \frac{1}{2}(\mathbf{y} - \mathbf{w}\mathbf{X})^T(\mathbf{y} - \mathbf{w}\mathbf{X}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

$$= \frac{1}{2}[\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{w}\mathbf{X} - \mathbf{X}^T\mathbf{w}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}] + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

For minimization of cost function, the $\frac{\delta J(\mathbf{w})}{\delta \mathbf{w}} = 0$.
Hence,

$$\Rightarrow \frac{1}{2}[0 - \mathbf{y}^T\mathbf{X} - \mathbf{X}^T\mathbf{y} + 2\mathbf{w}\mathbf{X}^T\mathbf{X}] + \frac{\lambda}{2}2\mathbf{w} = 0$$

$$\Rightarrow \frac{1}{2}[-2\mathbf{X}^T\mathbf{y} + 2\mathbf{w}\mathbf{X}^T\mathbf{X}] + \lambda\mathbf{w} = 0$$

$$\Rightarrow \mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

Vectorization based Least Angle regression

$$J(\mathbf{w}) = \frac{1}{2}[\|\mathbf{y} - \mathbf{w}\mathbf{X}\|_2^2 + \lambda \|\mathbf{w}\|_1]$$

$$= \frac{1}{2}(\mathbf{y} - \mathbf{w}\mathbf{X})^T(\mathbf{y} - \mathbf{w}\mathbf{X}) + \frac{\lambda}{2}\|\mathbf{w}\|_1$$

$$= \frac{1}{2}[\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{w}\mathbf{X} - \mathbf{X}^T\mathbf{w}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}] + \frac{\lambda}{2}\|\mathbf{w}\|_1$$

For minimization of cost function, the $\frac{\delta J(\mathbf{w})}{\delta \mathbf{w}} = 0$.
Hence,

$$\Rightarrow \frac{1}{2}[0 - \mathbf{y}^T\mathbf{X} - \mathbf{X}^T\mathbf{y} + 2\mathbf{w}\mathbf{X}^T\mathbf{X}] + \frac{\lambda}{2}sgn(\mathbf{w}) = 0$$

$$\Rightarrow \frac{1}{2}[-2\mathbf{X}^T\mathbf{y} + 2\mathbf{w}\mathbf{X}^T\mathbf{X}] + \frac{\lambda}{2}sgn(\mathbf{w}) = 0$$

$$\Rightarrow \mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{y} - \frac{\lambda}{2}sgn(\mathbf{w}))$$

Linear Regression from Probabilistic Prospective

Let us assume that the target variables and the inputs are related via the equation as

$$y^{(i)} = w^T \mathbf{x}^i + \epsilon^{(i)}$$

$\epsilon^{(i)}$ are independently and identically distributed according to a Gaussian distribution with zero mean and some variance σ^2.

The probability density function is given by

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi\sigma}} exp(-\frac{(\epsilon^{(i)})^2}{2\sigma^2})$$

This can also be written as

$$p(y^{(i)}/\mathbf{x}^{(i)}; w) = \frac{1}{\sqrt{2\pi\sigma}} exp(-\frac{(y^{(i)} - w^T \mathbf{x}^i)^2}{2\sigma^2})$$

The likelihood function by considering all the training examples is given by

$$L(w) = \prod_{i=1}^{m} p(y^{(i)}/\mathbf{x}^{(i)}; w)$$

$$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y^{(i)} - w^T\mathbf{x}^i)^2}{2\sigma^2})$$

The minimization of error is same as the maximization of the log likelihood and it is given by

$$l(w) = log(L(w))$$

$$= log(\prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y^{(i)} - w^T\mathbf{x}^i)^2}{2\sigma^2}))$$

$$= \sum_{i=1}^{m} log(\frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y^{(i)} - w^T\mathbf{x}^i)^2}{2\sigma^2}))$$

$$= m\ log(\frac{1}{\sqrt{2\pi}\sigma}) - \frac{1}{\sigma^2} \cdot \frac{1}{2} [\sum_{i=1}^{m} (h_w(\mathbf{x}^i) - y^{(i)})^2]$$

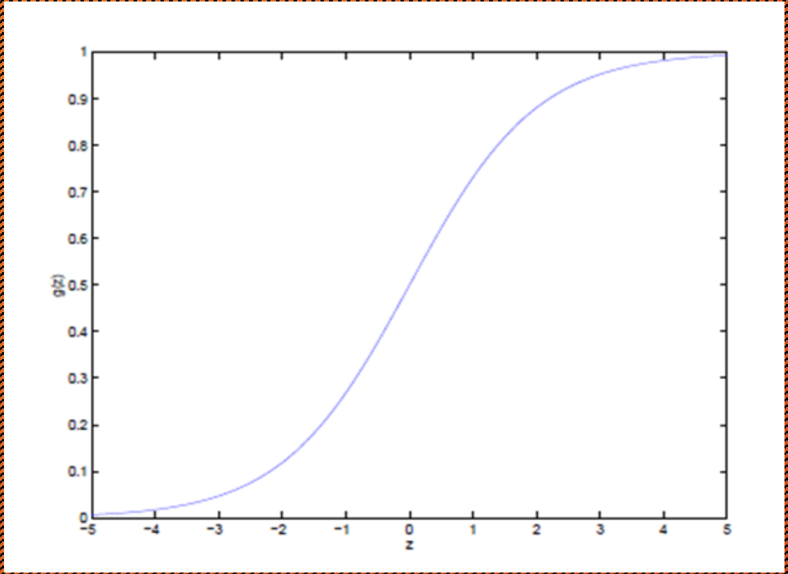
Logistic regression (Binary Classification)

The hypothesis for logistic regression is given as

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

$$g(z) = \frac{1}{1+e^{-z}}$$

logistic function or the sigmoid function.



g(z) tends towards 1 as z → ∞, and g(z) tends towards 0 as z → −∞.

$$\begin{aligned} g'(z) &= \frac{d}{dz}\frac{1}{1+e^{-z}} \\ &= \frac{1}{(1+e^{-z})^2}\left(e^{-z}\right) \\ &= \frac{1}{(1+e^{-z})}\cdot\left(1-\frac{1}{(1+e^{-z})}\right) \\ &= g(z)(1-g(z)) \end{aligned}$$

The hypothesis for the classification task is given by

$$P(y = 1/\mathrm{x}; w) = h_w(\mathrm{x})$$

$$P(y = 0/\mathrm{x}; w) = 1 - h_w(\mathrm{x})$$

More compactly, it can be written as

$$P(y/\mathrm{x}; w) = (h_w(\mathrm{x}))^y (1 - h_w(\mathrm{x}))^{1-y}$$

The likelihood function by considering 'm' training examples is given by

$$\begin{aligned} L(w) &= P(y/\mathrm{x}; w) \\ &= \prod_{i=1}^{m} p(y^{(i)}/\mathbf{x}^{(i)}; w) \\ &= \prod_{i=1}^{m} [h_w(\mathrm{x}^{(i)})]^{y^{(i)}} [(1 - h_w(\mathrm{x}^{(i)}))]^{1-y^{(i)}} \end{aligned}$$

The maximization of the cost function is given by

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

The partial derivative of the cost function for single training instance is given by

$$
\begin{aligned}
\frac{\delta l(w)}{\delta w_j} &= [y \frac{1}{g(w^T \mathrm{x})} - (1 - y) \frac{1}{1 - g(w^T \mathrm{x})}] \frac{\delta (g(w^T \mathrm{x}))}{\delta w_j} \\
&= [y \frac{1}{g(w^T \mathrm{x})} - (1 - y) \frac{1}{1 - g(w^T \mathrm{x})}] g(w^T \mathrm{x})(1 - g(w^T \mathrm{x})) \frac{\delta (w^T \mathrm{x})}{\delta w_j} \\
&= [y(1 - g(w^T \mathrm{x})) - (1 - y) g(w^T \mathrm{x})] x_j \; = [y - g(w^T \mathrm{x})] x_j
\end{aligned}
$$

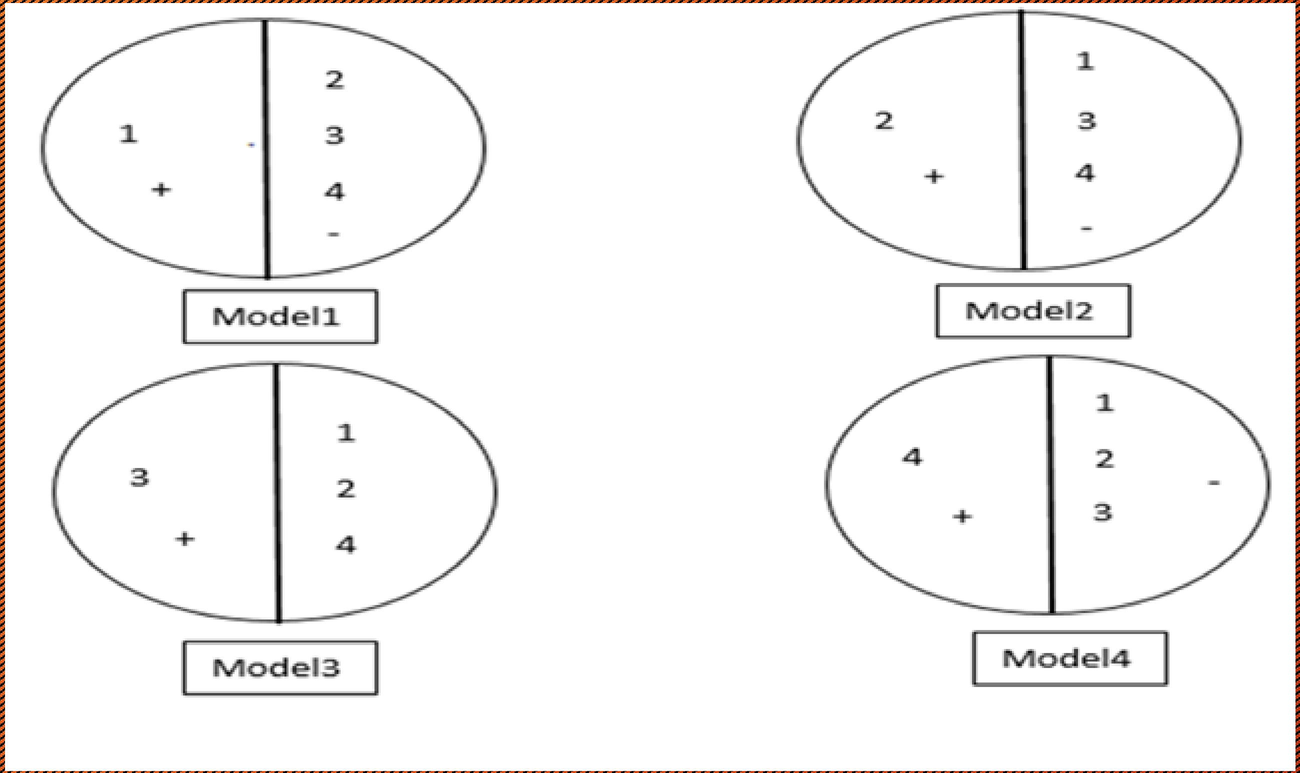
Hence using the LMS rule, the weight values for logistic regression is estimated as

$$
\overset{t+1}{w_j} := \overset{t}{w_j} - \alpha [\sum_{i=1}^{m} (y^{(i)} (1 - g(w^T \mathrm{x}^{(i)})) - (1 - y^{(i)}) g(w^T \mathrm{x}^{(i)})) x_j^{(i)}]
$$

For test data (x_t), the output is evaluated as $y_p = g(w^T \mathrm{x}_t)$

Multiclass Extension of Logistic Regression (One Vs All)

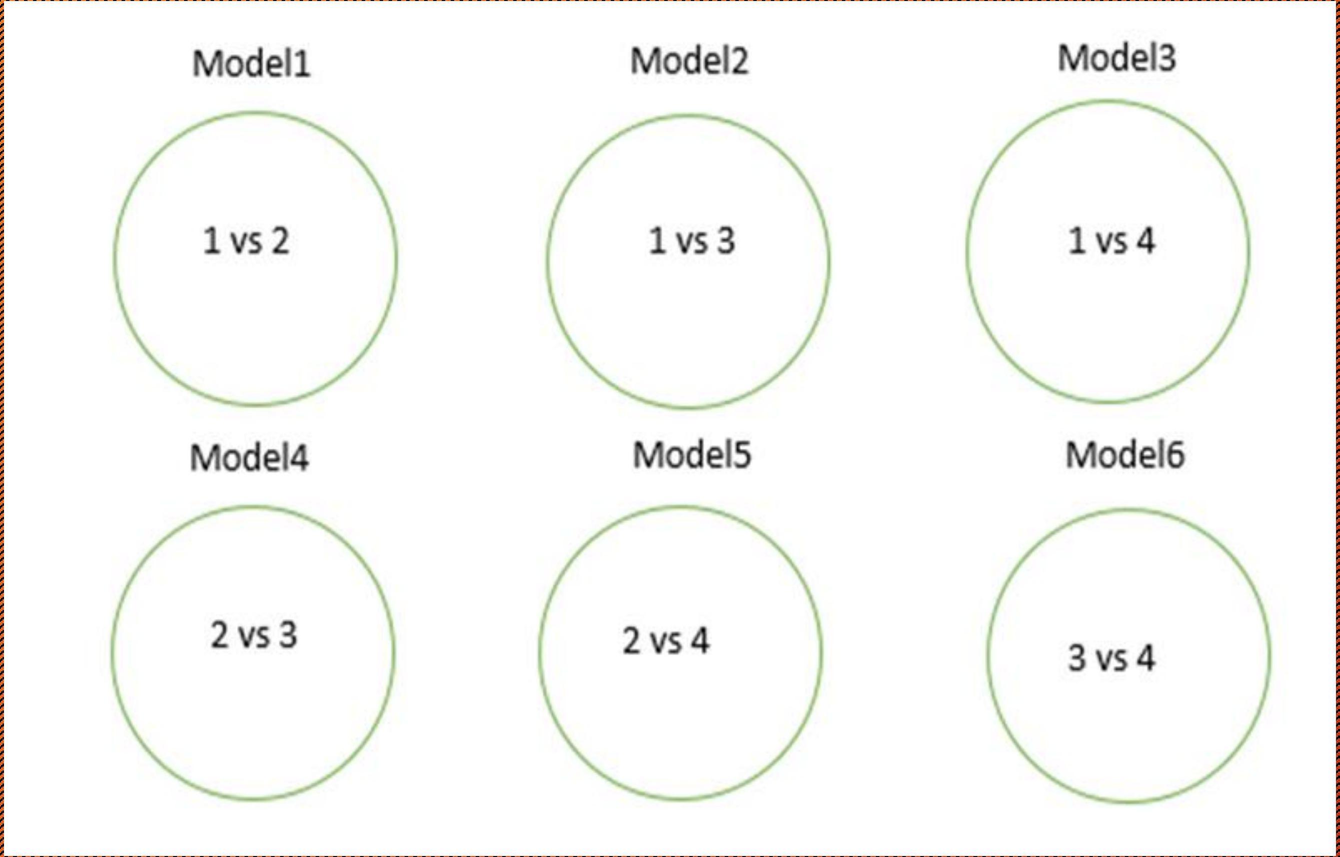
- Let’s consider a four class classification problem with class labels as 1, 2, 3 and 4.
- In One Vs All algorithm, 4 binary class models are created (‘n’ classes so ‘n’ number of models). The following block diagram explains the procedure for One Vs All based multiclass coding algorithm.



yt=max (y1, y2, y3, y4), where y1, y2, y3 and y4 are the predicted class labels using model1, model2, model3 and model4, respectively.

Multiclass Extension of Logistic Regression (One Vs One)

- In One Vs One algorithm, 6 binary class models are created for 4 class classification task (for 'n' classes 'n(n-1)/2' number of models). The following block diagram explains the procedure for One Vs All based multiclass coding algorithm.



yt= mode (y1, y2, y3, y4, y5, y6), where y1, y2, y3, y4, y5 and y6 are the predicted class labels using model1, model2, model3, model4, model5 and model6, respectively.

Logistic Regression with L2-Norm Regularization

The cost function or likelihood function for logistic regression with regularization is given by

$$
\begin{aligned}
l(w) &= log(L(w)) \\
&= \sum_{i=1}^{m}[y^{(i)}log(h_w(\mathbf{x}^{(i)})) + (1-y^{(i)})log(1-h_w(\mathbf{x}^{(i)}))] + \frac{\lambda}{2}\sum_{j=1}^{n} w_j^2
\end{aligned}
$$

Now, the gradient of the cost function for single training instance is given by

$$\frac{\delta l(w)}{\delta w_j} = [y(1-g(w^T\mathbf{x})) - (1-y)g(w^T\mathbf{x})]\mathbf{x}_j \ + \lambda w_j$$

Hence using the LMS rule, the weight values for the cost function is estimated as

$$w_j := (1-\alpha\lambda)w_j - \alpha[\sum_{i=1}^{m}(y^{(i)}(1-g(w^T\mathbf{x}^{(i)})) - (1-y^{(i)})g(w^T\mathbf{x}^{(i)}))\,\mathbf{x}_j^{(i)}]$$

Logistic Regression with L1-Norm Regularization

The cost function or likelihood function for logistic regression with L1-Norm regularization is given by

$$
\begin{aligned}
l(w) &= log(L(w)) \\
&= \sum_{i=1}^{m}[y^{(i)}log(h_w(\mathbf{x}^{(i)})) + (1-y^{(i)})log(1-h_w(\mathbf{x}^{(i)}))] + \frac{\lambda}{2}\sum_{j=1}^{n}|w_j|
\end{aligned}
$$

Now, the gradient of the cost function for a single training instance is given by

$$\frac{\delta l(w)}{\delta w_j} = [y(1-g(w^T\mathbf{x})) - (1-y)g(w^T\mathbf{x})]\mathbf{x}_j + \frac{\lambda}{2}sgn(w_j)$$

Hence using the LMS rule, the weight values for the cost function is estimated as

$$w_j := w_j - \alpha[\sum_{i=1}^{m}(y^{(i)}(1-g(w^T\mathbf{x}^{(i)})) - (1-\overset{(i)}{y})g(w^T\mathbf{x}^{(i)}))\ \mathbf{x}_j^{(i)}] - \alpha\frac{\lambda}{2}sgn(w_j)$$

Unsupervised Learning (k-means clustering)

Let X = {x1,x2,x3,……..,xm} be the set of data points and
V = {v1,v2,…….,vc} be the set of centers.

- ❑ Randomly select ‘c’ cluster centers.
- ❑ Calculate the distance between each data point and cluster centers.
- ❑ Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
- ❑ Recalculate the new cluster center using:

$$v_k = \frac{1}{c_k} \sum_{i=1}^{c_k} \mathbf{x}_i$$

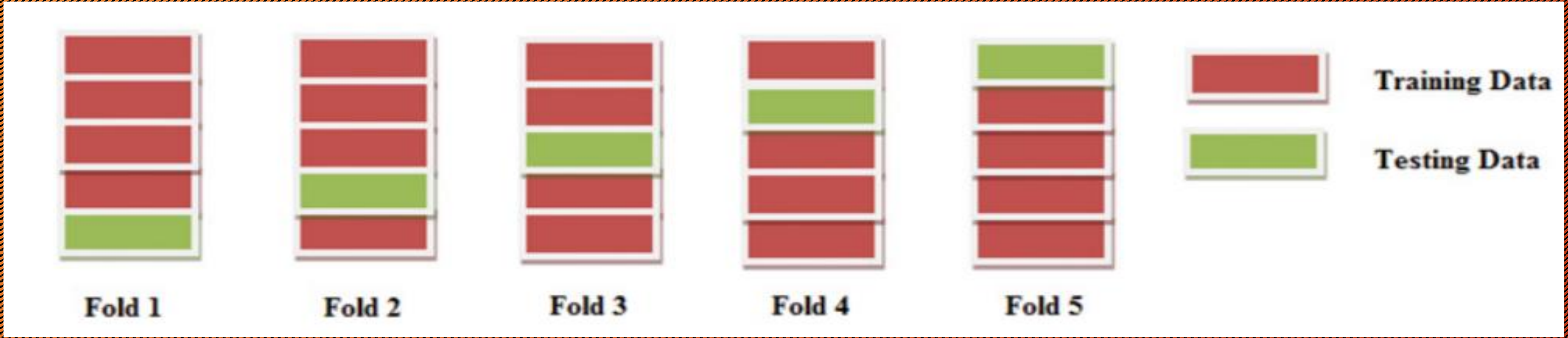
where, ‘c_k’ represents the number of data points in k^{th} cluster.

How to Select training and Test instances?

Hold-out cross validation:

- ❑ The holdout method is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set.
- ❑ The function approximator fits a function using the training set only. Then the function approximator is asked to predict the output values for the data in the testing set (it has never seen these output values before).
- ❑ Either 60/40, 70/30 or 80/20 based hold-out cross-validation approaches are followed.

K-fold cross-validation



- K-fold cross validation is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times.
- Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set.
- Then the average error across all k trials is computed. The advantage of this method is that it matters less how the data gets divided.
- The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation.

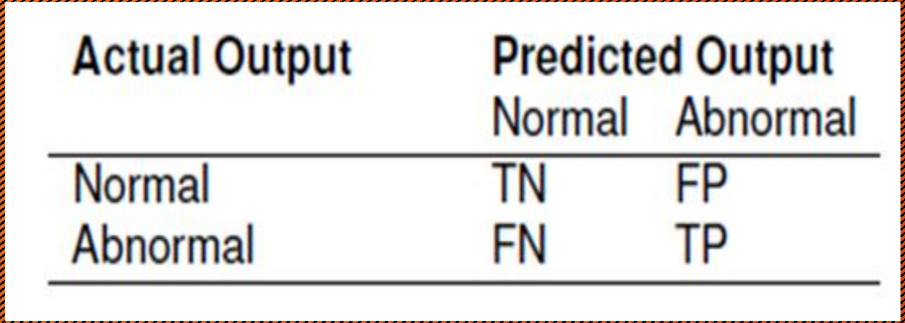
Leave-one-out cross validation

- Leave-one-out cross validation is K-fold cross validation taken to its logical extreme, with K equal to N, the number of data points in the set.

- That means that N separate times, the function approximator is trained on all the data except for one point and a prediction is made for that point.

Performance Measures for Binary Classifier

- The performance of the binary classifier is evaluated using the confusion matrix. This matrix is given by



Actual Output	Predicted Output	
	Normal	Abnormal
Normal	TN	FP
Abnormal	FN	TP

- The TP, the TN, the FP and the FN are number of true positives, number of true negatives, number of false positives and number of false negatives, respectively.
  - The sensitivity (SE) is defined as proportion of abnormal episodes that are accurately classified as abnormal and it is given by

$$SE = \frac{TP}{TP + FN}$$

- The specificity (SP) is defined as proportion of normal episodes that are accurately classified as normal and it is given by

$$SP = \frac{TN}{TN + FP}$$

- The accuracy (Acc) is defined as the proportion of episodes that are classified as normal and abnormal. It is given by

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Performance Measures for Multiclass Classifier

- The performance measures for multiclass classifier are individual class accuracy and the overall accuracy. These measures are evaluated from the multiclass confusion matrix. The confusion matrix is given as

		Predicted Output				
		Class1	Class2	Class3	Class4	Class5
Actual Output	Class1	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
	Class2	u_{21}	u_{22}	u_{23}	u_{24}	u_{25}
	Class3	u_{31}	u_{32}	u_{33}	u_{34}	u_{35}
	Class4	u_{41}	u_{42}	u_{43}	u_{44}	u_{45}
	Class5	u_{51}	u_{52}	u_{53}	u_{54}	u_{55}

$$IA_i = \frac{u_{ii}}{\sum_{j=1}^{5} u_{ij}}$$

- The individual class accuracy (IA) value of i th class is given by

- where i, j = 1, 2, 3, 4, 5. The overall accuracy (OA) of the multiclass classifier is evaluated as

$$OA = \frac{\sum_{i=j=1}^{5} u_{ij}}{\sum_{i=1}^{5} \sum_{j=1}^{5} u_{ij}}$$

Bayesian Decision Theory

The feature matrix and the class label vector are denoted by

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & - & - & x_{1n} \\ - & - & - & - & - \\ - & - & - & - & - \\ - & - & - & - & - \\ x_{m1} & x_{m2} & - & - & x_{mn} \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ - \\ - \\ y_m \end{bmatrix}$$

The feature matrix consists of ‘m’ feature vectors.

$$\mathbf{x}_i \in R^n \text{ with } \mathbf{x}_i = [x_1 \ x_2 \ ... \ x_n]$$ i=1,2,…m

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ - \\ - \\ \mathbf{x}_m \end{bmatrix}$$

The posterior probability evaluated using Bayes's theorem is given by

$$P(y_k/\mathbf{x}_i) = \frac{P(\mathbf{x}_i/y_k)P(y_k)}{\sum_{k=1}^{c} P(\mathbf{x}_i/y_k)P(y_k)}$$

$P(y_k)$ is the prior or probability of class 'k'.

$P(y_k/\mathbf{x}_i)$ is the posterior probability of class 'k' given i^{th} feature vector $\mathbf{x}_i$.

$P(\mathbf{x}_i/y_k)$ is the likelihood probability of i^{th} feature vector $\mathbf{x}_i$ given k^{th} class.

The posterior probability can also be written as

$$P(y_k/\mathbf{x}_i) = \frac{P(\mathbf{x}_i/y_k)P(y_k)}{\sum_{k=1}^{c} P(\mathbf{x}_i/y_k)P(y_k)} = \frac{P(\mathbf{x}_i/y_k)P(y_k)}{P(\mathbf{x}_i)}$$

$P(\mathbf{x}_i)$ is the evidence

The likelihood is modeled using Normal or Gaussian distribution. For single dimensional feature vector, the likelihood function is given by

$$P(x/y_k) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

For multi dimensional feature vector, the likelihood function is given by $\mathbf{x} = [x_1 \; x_2 \; \dots \; x_n]$

$$P(\mathbf{x}/y_k) = \frac{1}{(2\pi)^{n/2} \left|\sum\right|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \sum^{-1} (\mathbf{x}-\mu)}$$

Likelihood Ratio Test (LRT) for Two-class Bayesian Classifier

The decision rule for Two-class Bayesian Classifier using a posteriori is given by

$$P(y_1/\mathbf{x}) \underset{y_2}{\overset{y_1}{\gtrless}} P(y_2/\mathbf{x})$$

We can also write

$$\frac{P(\mathbf{x}/y_1)P(y_1)}{P(\mathbf{x})} \underset{y_2}{\overset{y_1}{\gtrless}} \frac{P(\mathbf{x}/y_2)P(y_2)}{P(\mathbf{x})}$$

As P(x) does not affect the decision rule, so it can be eliminated. The LRT is defined as

$$\Delta(\mathbf{x}) = \frac{P(\mathbf{x}/y_1)}{P(\mathbf{x}/y_2)} \underset{y_2}{\overset{y_1}{\gtrless}} \frac{P(y_2)}{P(y_1)}$$

Maximum a posteriori Probability (MAP) Decision rule

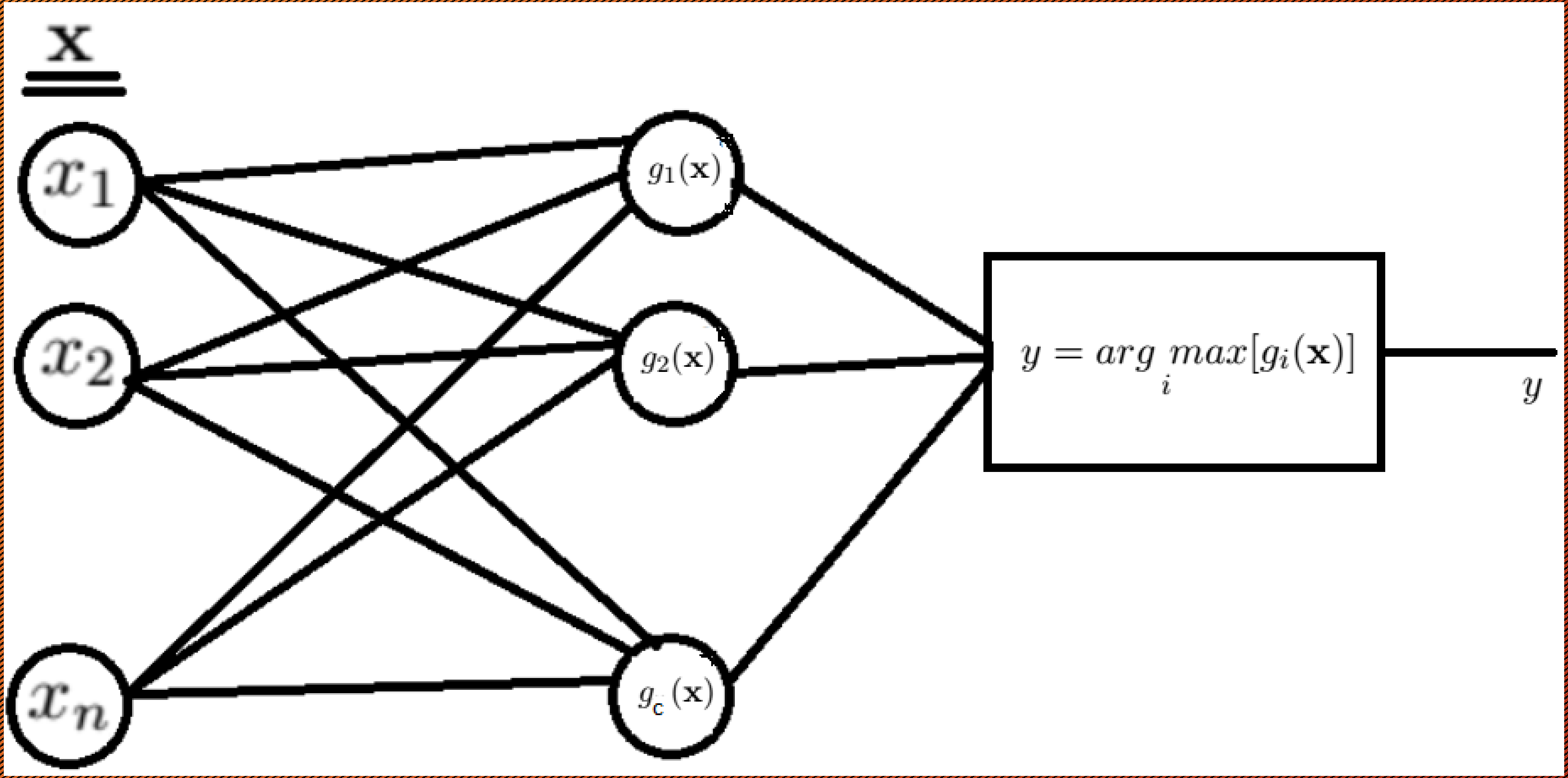
The LRT is given by $$\Delta(\mathbf{x}) = \frac{P(\mathbf{x}/y_1)}{P(\mathbf{x}/y_2)} \underset{y_2}{\overset{y_1}{\gtrless}} \frac{P(y_2)}{P(y_1)}$$

The LRT can also be written as $$\Delta(\mathbf{x}) = \frac{P(\mathbf{x}/y_1)}{P(\mathbf{x}/y_2)}\frac{P(y_1)}{P(y_2)} \underset{y_2}{\overset{y_1}{\gtrless}} 1$$

For binary class, the MAP decision rule is given by $$\Delta_{\text{MAP}}(\mathbf{x}) = \frac{P(y_1/\mathbf{x})}{P(y_2/\mathbf{x})} \underset{y_2}{\overset{y_1}{\gtrless}} 1$$

The MAP decision rule for multiclass classification is given by $$y = \arg\max_{k} P(y_k/\mathbf{x})$$

MAP decision rule for Multiclass Classification


$\underline{\underline{\mathbf{x}}}$
x_1
x_2
x_n
$g_1(\mathbf{x})$
$g_2(\mathbf{x})$
$g_c(\mathbf{x})$
$y = arg \max_i [g_i(\mathbf{x})]$
y

$$g_i(\mathbf{x}) = P(y_i/\mathbf{x}) = P(y_i/[x_1\ x_2,\ \ldots,\ x_n])$$

$$y = arg \max_i [g_i(\mathbf{x})]$$

Maximum Likelihood (ML) Decision rule

The LRT for two-class classifier is given by

$$\Delta(\mathbf{x}) = \frac{P(\mathbf{x}/y_1)}{P(\mathbf{x}/y_2)} \underset{y_2}{\overset{y_1}{\gtrless}} \frac{P(y_2)}{P(y_1)}$$

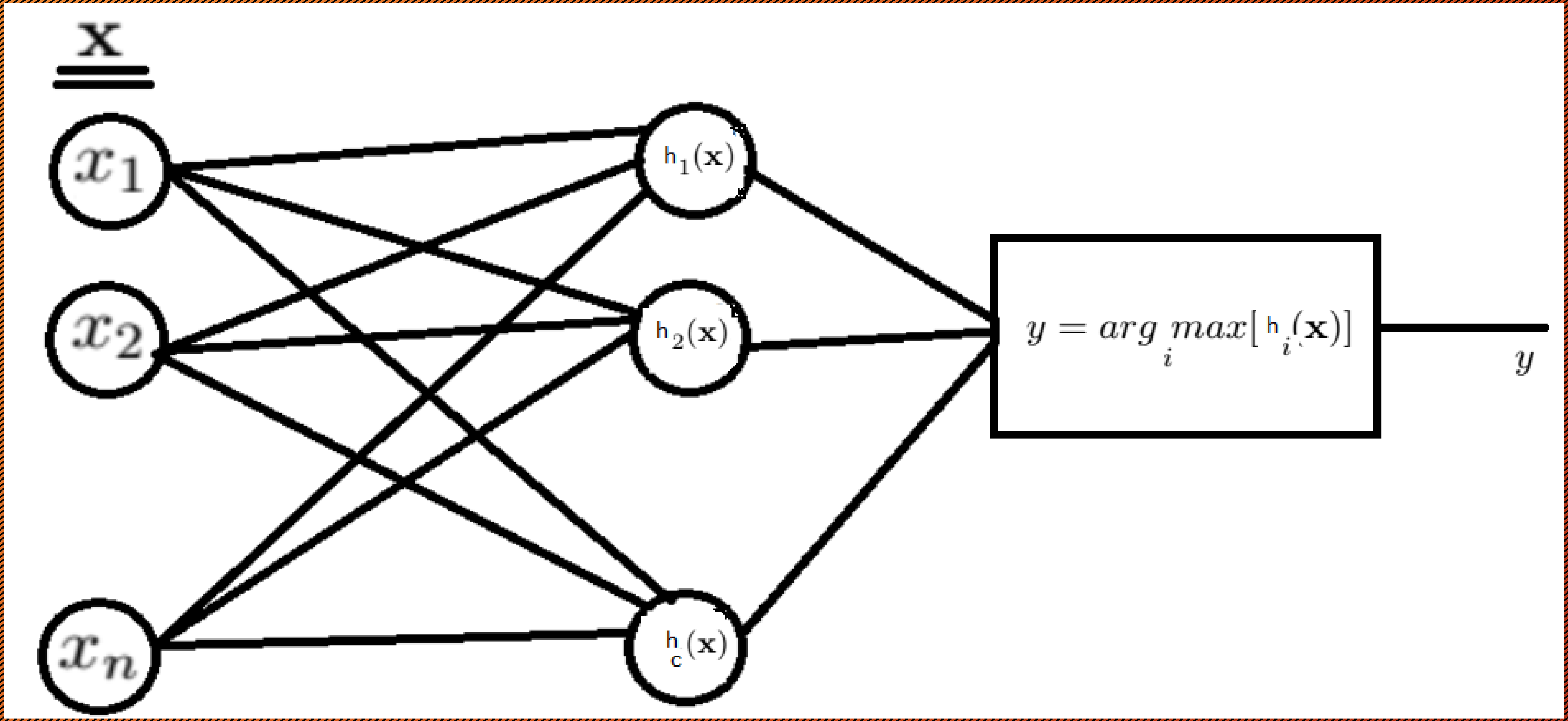
For equal priors, the decision rule for LRT is given by

$$\underset{ML}{\Delta(\mathbf{x})} = \frac{P(\mathbf{x}/y_1)}{P(\mathbf{x}/y_2)} \underset{y_2}{\overset{y_1}{\gtrless}} 1$$

The ML decision rule for multiclass classification is given by

$$y = \arg\max_{k} P(x/y_k)$$

ML decision rule for Multiclass Classification



$\underline{\underline{\mathbf{x}}}$
x_1
x_2
x_n
$h_1(\mathbf{x})$
$h_2(\mathbf{x})$
$h_c(\mathbf{x})$
$y = arg\,\max_i[h_i(\mathbf{x})]$
y

$$h_i(\mathbf{x}) = P(\mathbf{x}/y_i) = P([x_1\ x_2,\ \ldots,\ x_n]/y_i)$$

$$y = arg\,\max_i[h_i(\mathbf{x})]$$