Scheme of Valuation
Machine Learning
18IT505
B.Tech.,(Semester- V)

Date	:	February, 2021	Semester End Examination	:	Regular
Duration	:	3 hrs.	Maximum Marks	:	50

1. (1X10 = 10 Marks)

(a) Define Machine Learning CO1 L1

Answer : A machine learning algorithm is an algorithm that is able to learn from data. "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

(b) Write the difference between Classification and Regression CO1 L2

Answer : In case of Classification, the learning algorithm specifies which of k categories some input belongs to. The learning algorithm is usually asked to produce a function $f : R^n \rightarrow \{1, \ldots, k\}$.

In case of Regression, the predicted value is a numerical value given some input. The learning algorithm is asked to output a function $f : R^n \rightarrow R$.

(c) Write the difference between Supervised and Unsupervised learning CO1 L2

Answer : Supervised learning involves observing several examples of a random vector x and an associated value or vector y, then learning to predict y from x, usually by estimating $p(y|x)$

Unsupervised learning involves observing several examples of a random vector x and attempting to implicitly or explicitly learn the probability distribution $p(x)$, or some interesting properties of that distribution.

(d) What is Regularization? CO2 L2

Answer : Regularization allows complex models to be trained on data sets of limited size without severe over-fitting, essentially by limiting the effective model complexity.

(e) What is Type-1 error for a classifier? CO2 L2

Answer : A type 1 error is also known as a false positive and occurs when a classification model incorrectly predicts a true outcome for an originally false observation.

(f) What is a Confusion Matrix? CO2 L2

Answer : Confusion Matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

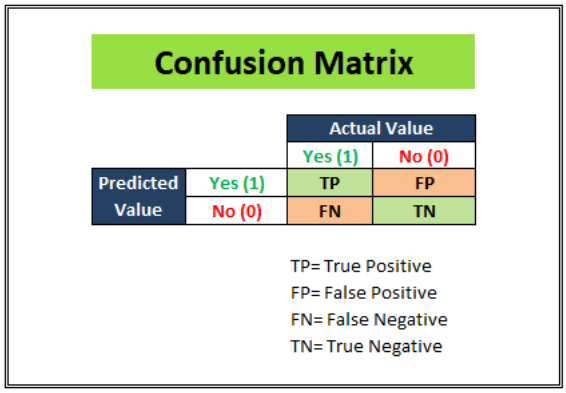


Figure 1: Confusion Matrix

(g) What is a Bootstrap sample? CO3 L2

Answer : Bootstrap sample is a sample that is drawn from the original dataset, with replacement. The size of the bootstrap sample is same as the size of the original dataset.

(h) What is a Dendrogram? CO3 L2

Answer : A dendrogram is a type of tree diagram showing hierarchical clustering — relationships between similar sets of data. They are frequently used in biology to show clustering between genes or samples, but they can represent any type of grouped data., based on similarity in a dataset.

(i) What is vanishing gradient problem, in the context of Artificial Neural Network? CO4 L2

Answer : As more layers using certain activation functions are added to neural networks, the gradients of the loss function approaches zero, making the network hard to train. Certain activation functions, like the sigmoid function, squishes a large input space into a small output space between 0 and 1. Therefore, a large change in the input of the sigmoid function will cause a small change in the output. Hence, the derivative becomes small.

(j) What is the role of learning rate in training an Artificial Neural Network? CO4 L2

Answer : Learning rate is a hyper-parameter, of a learning model that is used to tune how accurately a model converges on a result (classification/prediction, etc.). In the context of training an Artificial Neural Network learning rate controls how much the weights of the network are adjusted with respect to the loss gradient.

UNIT - I

2. (a) Describe elements of Machine Learning (CO1, L1, 5 Marks)

Answer :

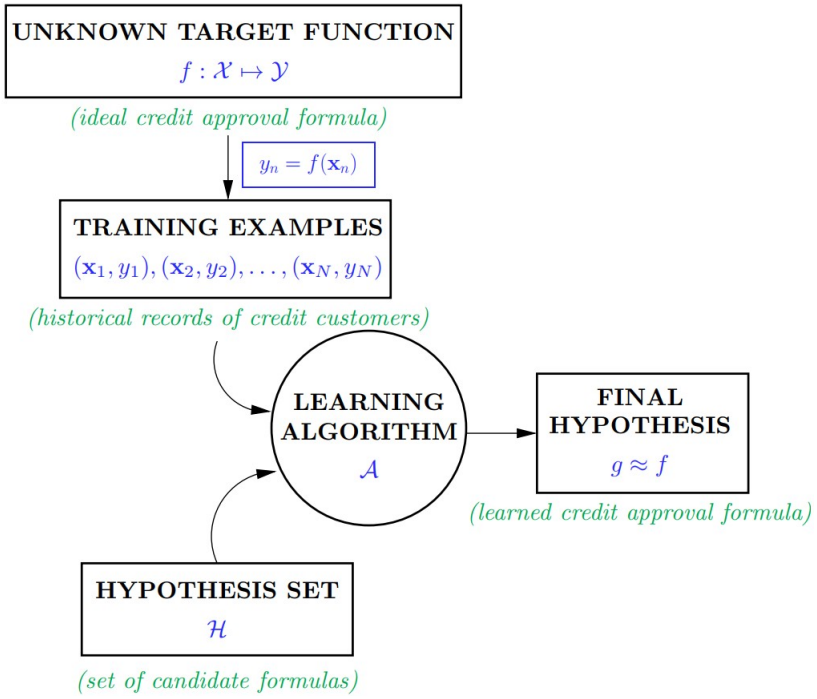


Figure 2: Elements of Machine Learning

(b) Write about commonly used Regression loss functions (CO1, L1, 5 Marks)

Answer :

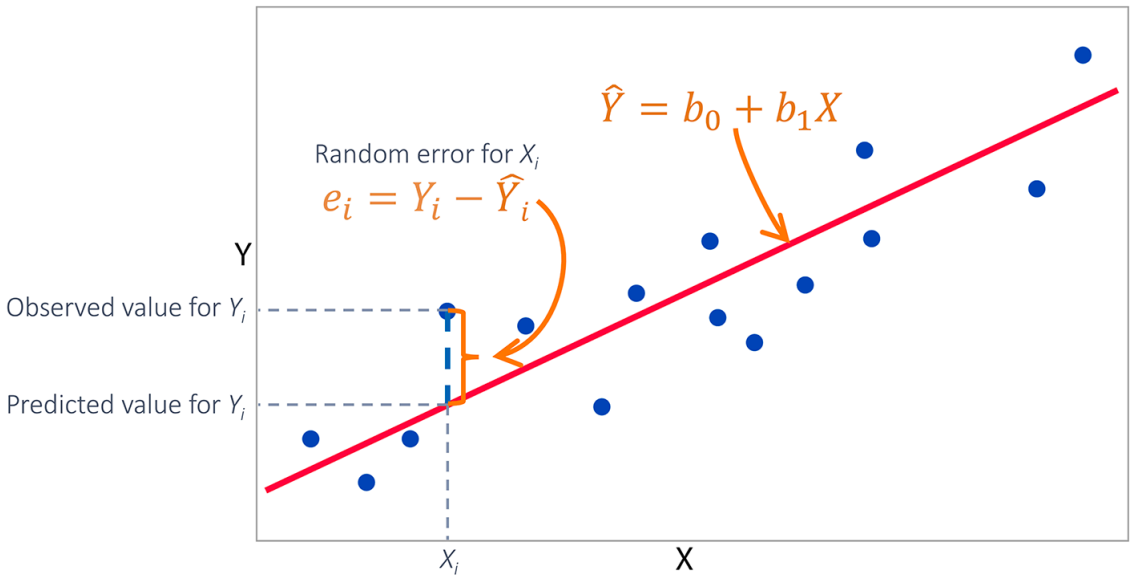


Figure 3: Regression error

NOTE
n - Number of training examples.
i - ith training example in a data set.
y_i - Observed value for ith training example.

$\hat{y}_i$ - Predicted value for ith training example.
Accuracy in % + Error in % = 100%.

- Mean Bias Error This is much less common in machine learning domain as compared to it's counterpart. This is same as MSE with the only difference that we don't take absolute values. Clearly there's a need for caution as positive and negative errors could cancel each other out. Although less accurate in practice, it could determine if the model has positive bias or negative bias.

$$MBE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i) \tag{1}$$

- Mean Absolute Error / $L1$ Loss Mean absolute error, on the other hand, is measured as the average of sum of absolute differences between predictions and actual observations. Like MSE, this as well measures the magnitude of error without considering their direction. Unlike MSE, MAE needs more complicated tools such as linear programming to compute the gradients. Plus MAE is more robust to outliers since it does not make use of square.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|(y_i - \hat{y}_i)| \tag{2}$$

- Mean Square Error / Quadratic Loss / $L2$ Loss: As the name suggests, Mean square error is measured as the average of squared difference between predictions and actual observations. It's only concerned with the average magnitude of error irrespective of their direction. However, due to squaring, predictions which are far away from actual values are penalized heavily in comparison to less deviated predictions. Plus MSE has nice mathematical properties which makes it easier to calculate gradients.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n} \tag{3}$$

- Root Mean Square Error This is calculated by applying square root function on MSE

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{4}$$

- Mean Percentage Error This is calculated as follows

$$MPE = \frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)}{y_i} \tag{5}$$

- Mean Absolute Percentage Error This is calculated as follows

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|(y_i - \hat{y}_i)|}{y_i} \tag{6}$$

OR

3. (a) Write about Batch Gradient descent optimization algorithm (CO1, L1, 5 Marks)

Answer :

The first stage in gradient descent is to pick a starting value (a starting point) for weights of the learning model. The starting point doesn't matter much; therefore, many algorithms simply set weights to 0 or pick a random value. The following figure 4 shows that we've picked a starting point slightly greater than 0. The gradient descent algorithm then calculates the gradient of the loss curve at the starting point. The gradient vector has both a direction and a magnitude.

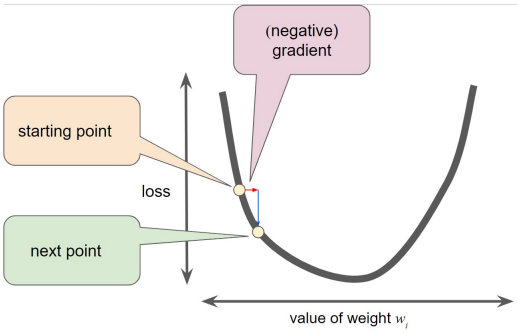


Figure 4: A gradient step moves us to the next point on the loss curve.

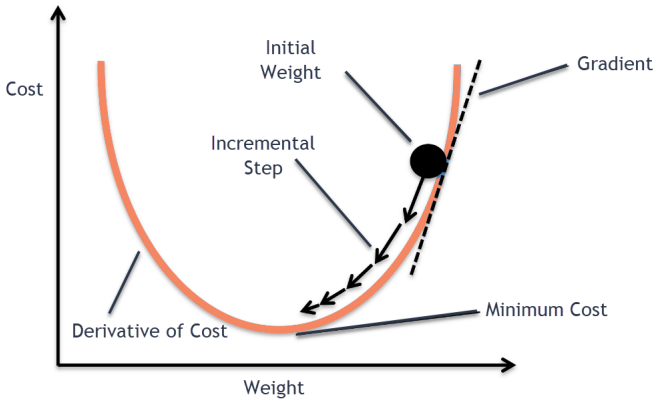


Figure 5: Cost function with one parameter

Here in figure 5, the gradient of the loss is equal to the derivative (slope) of the curve, and tells us which way is "increasing" or "decreasing". When there are multiple weights, the gradient is a vector of partial derivatives with respect to the weights. The gradient always points in the direction of steepest increase in the loss function.

The gradient descent algorithm takes a step in the direction of the negative gradient in order to reduce loss as quickly as possible. To determine the next point along the loss function curve, the gradient descent algorithm adds some fraction of the gradient's magnitude to the starting point as shown in the figure 5.

(b) Write about stratified K-Fold Cross Validation (CO1, L1, 5 Marks)

Answer : In stratified k-Fold cross-validation, first all the observations are considered separately as per their class label. K-sets of data with one having observation belonging to class 1, another having observations of class 2 so on

and kth set having observations of the class label k are formed. Each of these sets is then divided into say 5 folds and the first fold of all these k classes are joined together to form the fold 1. Similarly, the second fold of all these k sets of observations are joined together to form the second fold and so on. This way the class labels gets balanced across the training and testing set and we are able to counter the 'unlucky splits' from happening.

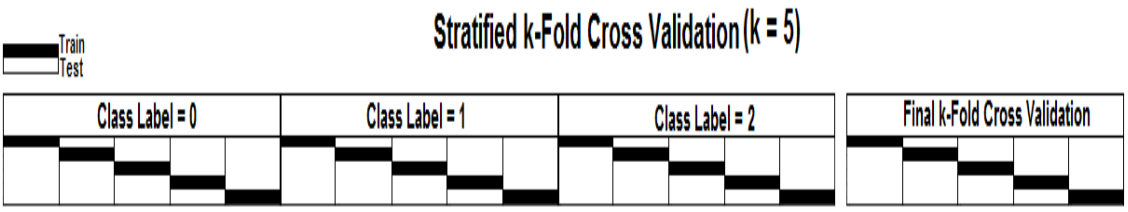


Figure 6: Stratified K-Fold Cross Validation

UNIT - II

4. (a) Write about Logistic Regression. (CO2, L1, 5 Marks)

Answer : Unlike actual regression, logistic regression does not try to predict the value of a numeric variable given a set of inputs. Instead, the output is a probability that the given input point belongs to a certain class. Let $P_+ \Rightarrow$ the probability that a certain data point belongs to the + class. Of course, $P_- = 1 - P_+$. Thus, the output of Logistic Regression always lies in $[0, 1]$.

Let $P(X)$ denote the probability of an event X occurring. In that case, the odds ratio ($OR(X)$) is defined by $P(X)/1 - P(X)$, which is essentially the ratio of the probability of the event happening, vs. it not happening.

The boundary function of logistic regression actually defines the log-odds of the + class. Given a point (a, b), this is what Logistic regression would do-

i. Compute the boundary function (alternatively, the log-odds function) value, $w_0 + w_1 x_1 + w_2 x_2$. Lets call this value t for short.
ii. Compute the Odds Ratio, by doing $OR_+ = e^t$. (Since t is the logarithm of OR_+).
iii. Knowing OR_+, it would compute P_+ using the simple mathematical relation

$$P_+ = \frac{OR_+}{1 + OR_+}$$

.

In fact, once you know t from step 1, you can combine steps 2 and 3 to give you

$$P_+ = \frac{e^t}{1 + e^t}$$

The RHS of the above equation is called the logistic function. Hence the name given to this model of learning

(b) Write about Lasso and Ridge Regression. (CO2, L1, 5 Marks)

Answer : Ridge Regression uses something called $L2$ Regularization. Regularization is used to reduce overfitting. $L2$ Regularization reduces model complexity and helps bring the weights in our model closer to zero, in essence, decreasing variance and shifting us more left on the bias-variance curve. Linear Regression is prone to overfitting and high variance, to counter that bias is introduced. In return for adding bias, the model has decreased variance.

A traditional linear regression model minimizes the sum of squared residuals, whereas in ridge regression, the sum of squared residuals plus a λ times the sum of the coefficients squared is minimized. This can be considered as a penalty term. If $\lambda = 0$, ridge cost function equals linear regression cost function.

$$Cost = \sum_{i=1}^{M} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{M} \left(y_i - \sum_{j=0}^{d} w_j \times x_{ij} \right)^2 + \lambda \sum_{j=1}^{d} (w_j)^2 \tag{7}$$

LASSO regression uses $L1$ regularization. LASSO regression permits certain weights to reach 0 whereas in ridge regression, weights can only approach 0. This type of regularization is extremely useful when we have a lot of features in

our data. $L1$ regularization uses feature selection to help reduce model complexity.

$$Cost = \sum_{i=1}^{M} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{M} \left(y_i - \sum_{j=0}^{d} w_j \times x_{ij} \right)^2 + \lambda \sum_{j=0}^{d} (|w_j|) \tag{8}$$

where $y_i, \hat{y}_i$ are the actual and predicted target values respectively of the ith example in training set. M is the number of examples in training set, d is the number of features of each training example x and w is the weight or coefficient of the learning model.

OR

5. (a) Write about Decision Tree classifier (CO2, L1, 5 Marks)

Answer : A decision tree has three main components :

- Root Node : The top most node is called Root Node. It implies the best predictor (independent variable).
- Decision / Internal Node : The nodes in which predictors (independent variables) are tested and each branch represents an outcome of the test
- Leaf / Terminal Node : It holds a class label (category) - Yes or No (Final Classification Outcome).

Types of Decision Tree

- Regression Tree is used when the dependent variable is continuous. The value obtained by leaf nodes in the training data is the mean response of observation falling in that region. Thus, if an unseen data observation falls in that region, its prediction is made with the mean value. This means that even if the dependent variable in training data was continuous, it will only take discrete values in the test set. A regression tree follows a top-down greedy approach. Classification Tree
- Classification tree is used when the dependent variable is categorical. The value obtained by leaf nodes in the training data is the mode response of observation falling in that region It follows a top-down greedy approach.

Together they are called as CART(Classification And Regression Tree)

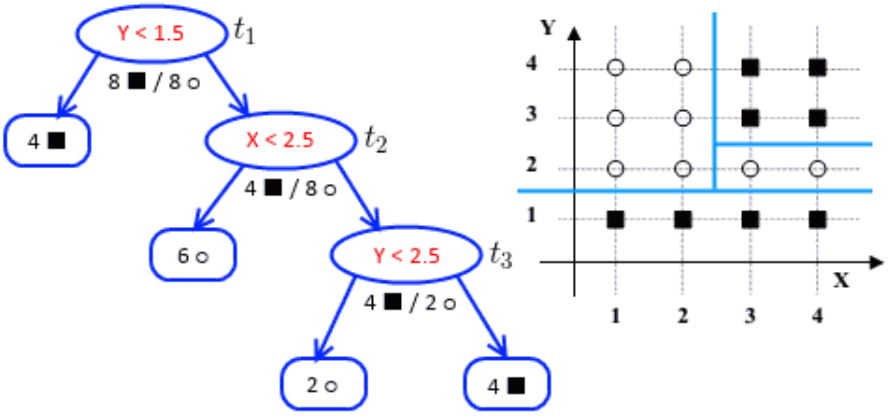


Figure 7: Decision tree

(b) Write about attribute selection methods used in Decision Tree classifier (CO2, L1, 5 Marks)

Answer :

i. Information gain computes the difference in entropy before partitioning and average entropy after partitioning of the dataset based on a given attribute value. ID3 (Iterative Dichotomiser) decision tree algorithm uses information gain.

$$\begin{aligned} &\text{Info(D)} = -\sum_{i=1}^{m} p_i \log_2 p_i \\ &\text{Info}_\text{A}\text{(D)} = \sum_{j=1}^{v} |D_j|/|D| \times \text{Info}\,(\text{D}_\text{j}) \\ &\text{Gain(A)} = \text{Info(D)} - \text{Info}_\text{A}\text{(D)} \end{aligned} \tag{9}$$

Where, p_i is the probability that an arbitrary tuple in D belongs to class C_i, m is the number of categories in the dataset, v is the number of partitions of D based on attribute A, $Info(D)$ is the average amount of information needed to identify the class label of a tuple in D, $|D_j|/|D|$ acts as the weight of the jth partition and $Info_A(D)$ is the expected information required to classify a tuple from D based on the partitioning by A.

The attribute A with the highest information gain, Gain(A), is chosen as the partitioning attribute at node N of a Decision Tree.

ii. Gain Ratio Information gain is biased for the attribute that has highest partitions on D. C4.5, an improvement of ID3, uses gain ratio. Gain ratio handles the issue of bias by normalizing the information gain using Split Info.

$$\begin{aligned} &\text{SplitInfo }_A(D) = -\sum_{j=1}^{v} |D_j|/|D| \times \log_2 (|D_j|/|D|) \\ &\text{GainRatio}(A) = \text{Gain}(A)/\text{SplitInfo}_A(D) \end{aligned} \tag{10}$$

Where, $|D_j|/|D|$ acts as the weight of the jth partition and v is the number of partitions of D based on attribute A.

iii. Gini index or Gini impurity measures the degree or probability of a particular example from D being wrongly classified when it is randomly chosen. If all the examples in D belong to a single class, then it can be called pure. The degree of Gini index varies between 0 and 1, where 0 denotes that all elements belong to a certain class or if there exists only one class, and 1 denotes that the elements are randomly distributed across various classes. A Gini Index of 0.5 denotes equally distributed elements into some classes. The attribute A with the least Gini index, $\text{GiniIndex}_\text{A}\text{(D)}$, is chosen as the partitioning attribute at node N of a Decision Tree.

$$\begin{aligned} &\text{GiniIndex(D)} = 1 - \sum_{i=1}^{m} {p_i}^2 \\ &\text{GiniIndex}_\text{A}\text{(D)} = \sum_{j=1}^{v} |D_j|/|D| \times \text{GiniIndex}\,(\text{D}_\text{j}) \end{aligned} \tag{11}$$

UNIT - III

6. (a) Write about K-Means algorithm (CO4, L1, 5 Marks)

Answer :

The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares (see below). This algorithm requires the number of clusters to be specified. It scales well to large number of samples and has been used across a large range of application areas in many different fields. The k-means algorithm divides a set of N samples X into k disjoint clusters C, each described by the mean μ_j of the samples in the cluster.

Algorithm 0.1 k-means. The k-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- k: the number of clusters,
- D: a data set containing n objects.

Output: A set of k clusters.

Method:

arbitrarily choose k objects from D as the initial cluster centers;
repeat
(re)assign each object to the cluster to which the object is the most similar,

based on the mean value of the objects in the cluster; update the cluster means, that is, calculate the mean value of the objects for each cluster;
until no change;

The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum-of-squares criterion:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} \text{dist}\,(\boldsymbol{p}, \boldsymbol{\mu_i})^2 \tag{12}$$

(b) Compare Partitional and Hierarchical clustering methods (CO4, L2, 5 Marks)

Answer :

Partitional method	Hierarchical method
Scalable for large datasets	Computationally more complex
Assumptions such as number of clusters and the initial cluster centers are made	Only similarity measure has to be assumed
Require the number of clusters to start running	Does not require any input parameters
Returns exactly k clusters	Returns a much more meaningful and subjective division of clusters
Is a non deterministic process	Is a deterministic process, meaning cluster assignments won't change when the algorithm is run twice on the same input data.

OR

7. (a) Write about Random forest algorithm (CO3, L1, 5 Marks)

Answer : In Random Forest, multiple trees are grown as opposed to a single tree in CART model. Each tree is grown as follows:

i. Let the training set has N number of examples. A sample of these N cases (2/3) is taken at random but with replacement. This sample will be the training set for growing the tree.

ii. If there are M input variables, a number $m < M$ is specified such that at each node, m variables ($\sqrt{M}$ for classification and $M/3$ for regression) are selected at random out of the M. The best split on these m is used to split the node. The value of m is held constant while we grow the forest.

iii. Each tree is grown to the largest extent possible and there is no pruning.

iv. Predict new data by aggregating the predictions of the n trees (i.e., majority votes for classification, average for regression).

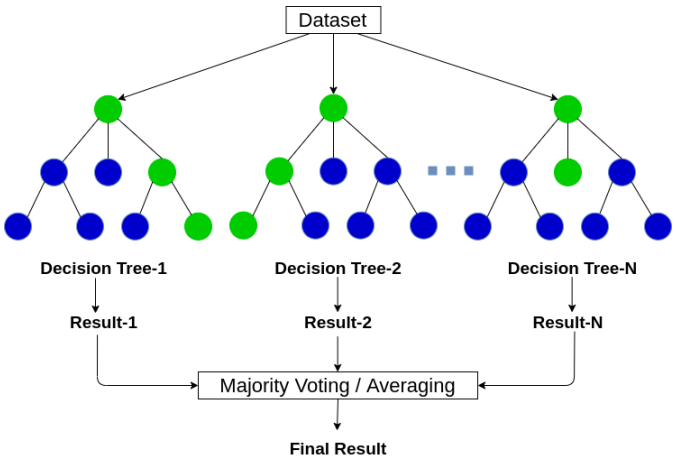


Figure 8: Random forest algorithm

(b) Compare Classification and Regression Tree (CART) and Random forest algorithm. (CO3, L2, 5 Marks)

Answer :

CART	Random Forest
Only one tree is used for the model	A large number of relatively uncorrelated models (trees) operating as a committee are used to outperform any of the individual constituent models.
Deep decision trees may suffer from overfitting	Random forests prevent overfitting by creating trees on random subsets.
Computationally faster	Relatively more complex
A decision tree is easily interpretable and can be converted to rules	Random forest model is difficult to interpret

UNIT - IV

8. (a) What are the guidelines to fix the design parameters of an Artificial Neural Network? (CO5, L1, 5 Marks)

Answer :

The first layer is the input layer and has as many neurons as the number of available features that are used as inputs. The last one, the output layer, will have as many neurons as the needed outputs.

Hidden layers are required if and only if the data must be separated non-linearly. In real-world problems, there is no way to determine the best number of hidden layers and neurons without trying.

Guidelines to know the number of hidden layers and neurons per each hidden layer in a classification problem:

i. Based on the data, draw an expected decision boundary to separate the classes.
ii. Express the decision boundary as a set of lines. Note that the combination of such lines must yield to the decision boundary.
iii. The number of selected lines represents the number of hidden neurons in the first hidden layer.
iv. To connect the lines created by the previous layer, a new hidden layer is added. Note that a new hidden layer is added each time you need to create connections among the lines in the previous hidden layer.
v. The number of hidden neurons in each new hidden layer equals the number of connections to be made.

(b) Write about commonly used activation functions in an Artificial Neural Network (CO5, L1, 5 Marks)

Answer :

- Sigmoid Function

$$A = \frac{1}{1 + e^{-x}} \tag{13}$$

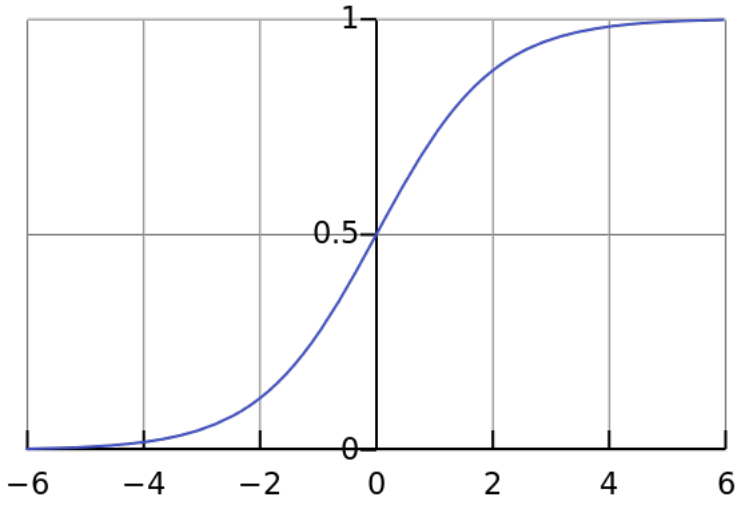


Figure 9: Sigmoid

Towards either end of the sigmoid function, the Y values tend to respond very less to changes in X. What does that mean? The gradient at that region is going to be small. It gives rise to a problem of "vanishing gradients".

- Tanh Function

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{14}$$

$$\tanh(x) = 2\,\text{sigmoid}(2x) - 1 \tag{15}$$

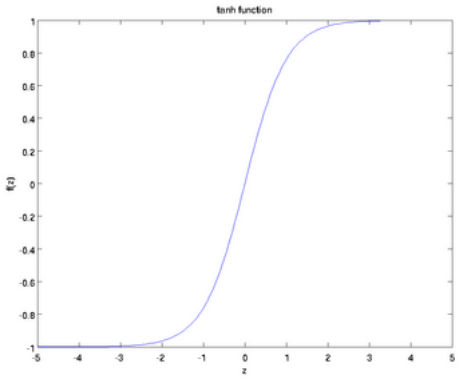


Figure 10: Tanh function

The gradient is stronger for tanh than sigmoid (derivatives are steeper).

- ReLu function The ReLu function gives an output x if x is positive and 0 otherwise.

$$f(x) = max(0, x) \tag{16}$$

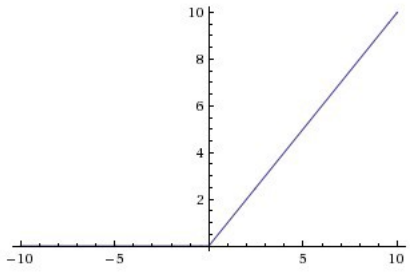


Figure 11: ReLu function

ReLu is nonlinear in nature. And combinations of ReLu are also non linear. Any function can be approximated with combinations of ReLu. The range of ReLu is $[0, inf)$. This means it can blow up the activation. Because of the horizontal line in ReLu(for negative X), the gradient can go towards 0. For activations in that region of ReLu, gradient will be 0 because of which the weights will not get adjusted during descent. That means, those neurons which go into that state will stop responding to variations in error/ input (simply because gradient is 0, nothing changes). This is called dying ReLu problem. ReLu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. That is a good point to consider when we are designing deep neural nets.

OR

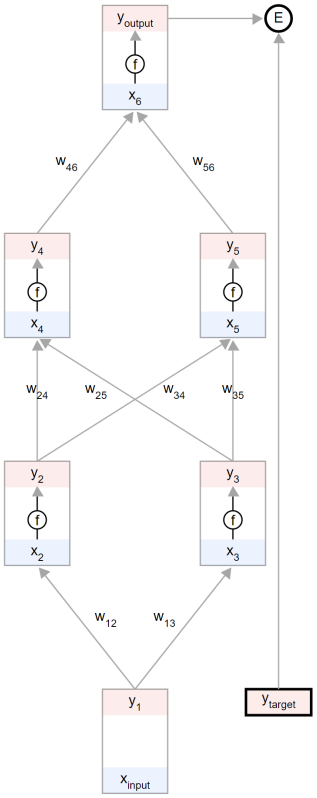
9. How is backpropagation algorithm used to train an Artificial Neural Network? CO5, L2, 10 Marks

Answer :

Error function

The goal of backpropagation algorithm is to learn the weights of the network automatically from data such that the predicted output y_{output} is close to the target y_{target} for all inputs x_{input}.

To measure how far we are from the goal, we use an error function E. A commonly used error function is $E(y_{output}, y_{target}) = 1/2(y_{output} - y_{target})^2$.



Forward propagation

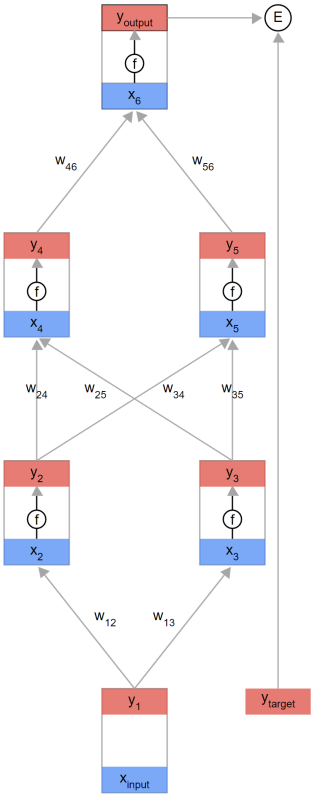
We begin by taking an input example (x_{input}, y_{target}) and updating the input layer of the network.

For consistency, we consider the input to be like any other node but without an activation function so its output is equal to its input, i.e. $y_1 = x_{input}$.

Using these 2 formulas we propagate for the rest of the network and get the final output of the network.

$$y = f(x)$$

$$x_j = \sum_{i \in in(j)} w_{ij} \cdot y_i + b_j$$



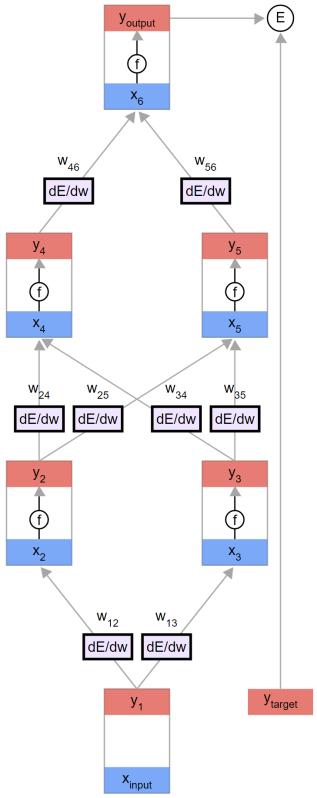
Error derivative

The backpropagation algorithm decides how much to update each weight of the network after comparing the predicted output with the desired output for a particular example. For this, we need to compute how the error changes with respect to each weight dE/dw_{ij}.

Once we have the error derivatives, we can update the weights using a simple update rule:

$$w_{ij} = w_{ij} - \alpha \frac{dE}{dw_{ij}}$$

where α is a positive constant, referred to as the learning rate, which we need to fine-tune empirically. The update rule is very simple: if the error goes down when the weight increases ($dE/dw_{ij} < 0$), then increase the weight, otherwise if the error goes up when the weight increases ($dE/dw_{ij} > 0$), then decrease the weight.

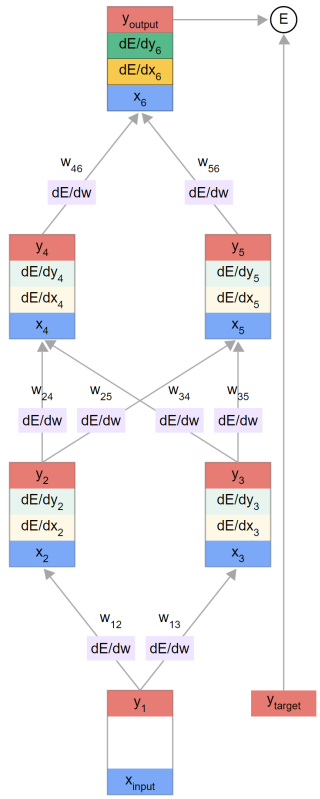


Back propagation

Now that we have dE/dy we can get dE/dx using the chain rule.

$\partial E/\partial x = dy/dx \partial E/\partial y = d/dx f(x) \partial E/\partial y$

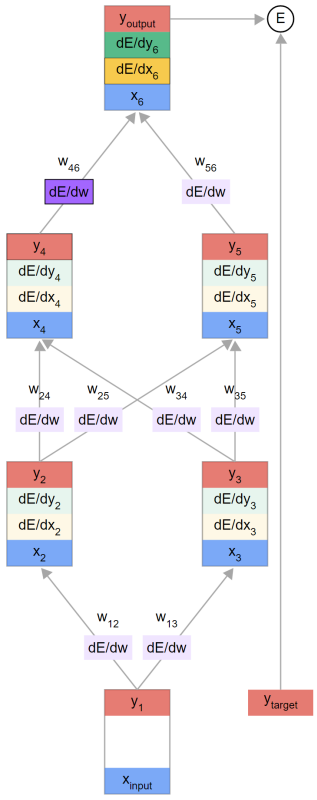
where $d/dx f(x) = f(x)(1 - f(x))$ when $f(x)$ is the Sigmoid activation function.



Back propagation

As soon as we have the error derivative with respect to the total input of a node, we can get the error derivative with respect to the weights coming into that node.

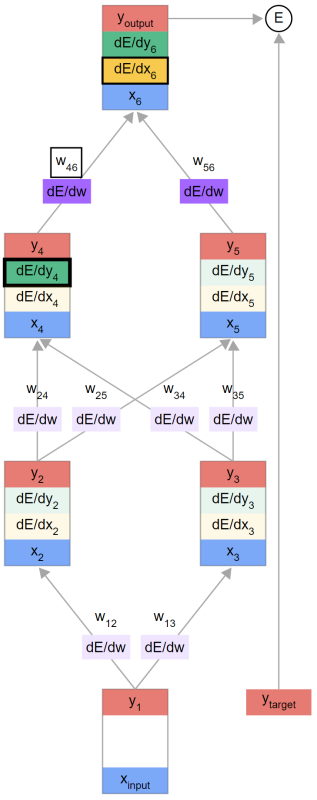
$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial x_j}{\partial w_{ij}} \frac{\partial E}{\partial x_j} = y_i \frac{\partial E}{\partial x_j}$$



Back propagation

And using the chain rule, we can also get dE/dy from the previous layer. We have made a full circle.

$$\frac{\partial E}{\partial y_i} = \sum_{j \in \text{out}(i)} \frac{\partial x_j}{\partial y_i} \frac{\partial E}{\partial x_j} = \sum_{j \in \text{out}(i)} w_{ij} \frac{\partial E}{\partial x_j}$$



Scheme prepared by

(N.Sivaram Prasad, IT Dept.)

Signature of HOD

Signatures of evaluators

SNo	Name of the evaluator	College Name	Signature