Machine Learning

Abhijit Bhattacharyya
vega@barc.gov.in

Bhabha Atomic Research Centre, Mumbai 400 085

Feb 8, 2024

Data

Engineering for outliers fixing / resampling:

Useless without any pattern

Data

Engineering for outliers fixing / resampling:

Useless without any pattern

Information (Average level of Surprise / Uncertainty)

Data shows pattern

Data

Engineering for outliers fixing / resampling:

Useless without any pattern

Information (Average level of Surprise / Uncertainty)

Data shows pattern

Knowledge

Learning is involved using information, provides interpretation, understanding of unknown phenomenon.

Data

Engineering for outliers fixing / resampling:

Useless without any pattern

Information (Average level of Surprise / Uncertainty)

Data shows pattern

Knowledge

Learning is involved using information, provides interpretation, understanding of unknown phenomenon.

Processor uses data to generate a connected mapping using some mathematical functions / models to understand unknown data.

Definition by Tom Mitchell (1998):

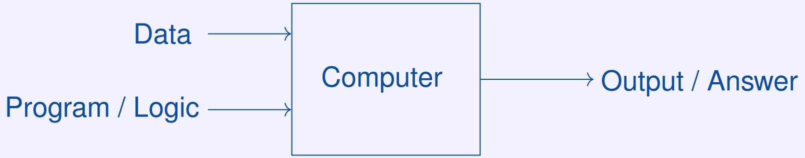
Machine Learning is the study of algorithms that

- improve their performance P
- for some task T
- with experience E.

A well defined learning task is given by $< P,\ T,\ E >$.

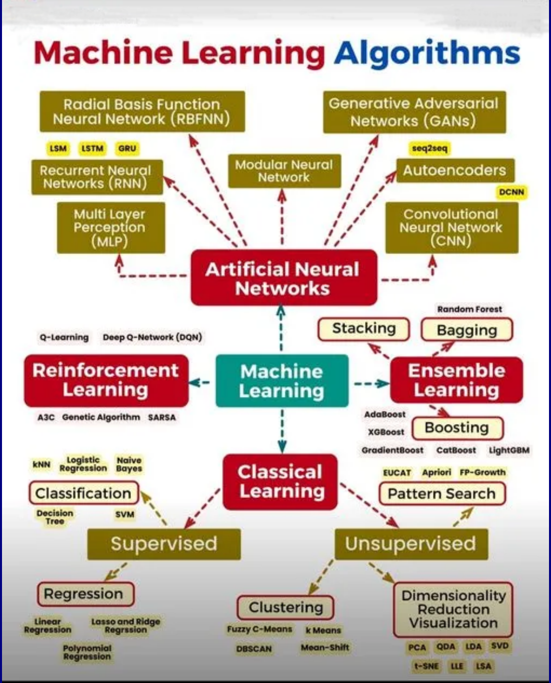
Improve on task T with respect to Performance metric P based on experience E

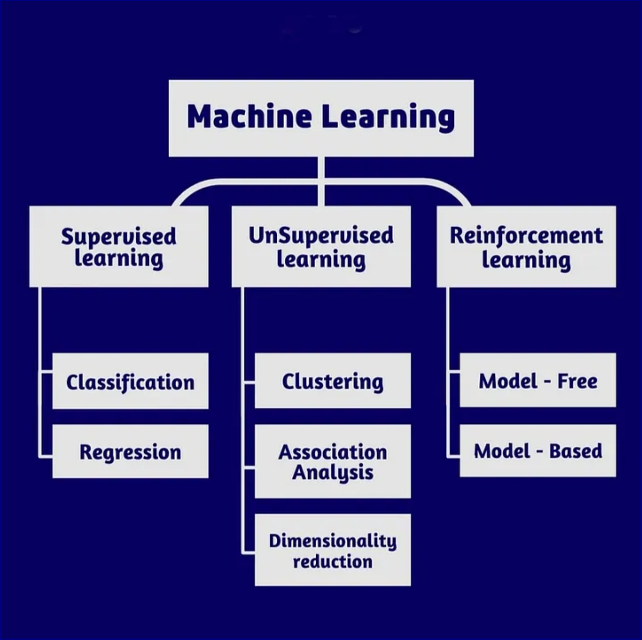
Traditional Programming



Machine Learning

Apply the generated logic to future unknown Data to get some output / result.


Machine Learning Algorithms
Radial Basis Function Neural Network (RBFNN)
Generative Adversarial Networks (GANs)
LSM
LSTM
GRU
seq2seq
Recurrent Neural Networks (RNN)
Modular Neural Network
Autoencoders
DCNN
Multi Layer Perception (MLP)
Convolutional Neural Network (CNN)
Artificial Neural Networks
Random Forest
Stacking
Bagging
Q-Learning
Deep Q-Network (DQN)
Reinforcement Learning
Machine Learning
Ensemble Learning
A3C
Genetic Algorithm
SARSA
AdaBoost
XGBoost
Boosting
GradientBoost
CatBoost
LightGBM
kNN
Logistic Regression
Naive Bayes
Classification
Classical Learning
EUCAT
Apriori
FP-Growth
Pattern Search
Decision Tree
SVM
Supervised
Unsupervised
Regression
Clustering
Dimensionality Reduction Visualization
Linear Regression
Lasso and Ridge Regrssion
Polynomial Regression
Fuzzy C-Means
k Means
DBSCAN
Mean-Shift
PCA
QDA
LDA
SVD
t-SNE
LLE
LSA



Machine Learning
Supervised learning
UnSupervised learning
Reinforcement learning
Classification
Regression
Clustering
Association Analysis
Dimensionality reduction
Model - Free
Model - Based

Data Engineering

TOOLS:

- TensorFlow: Open source ML Lib developed by Google.
- PyTorch: Open source Deep learning framework know for dynamic computation graph, intuitive design and support for dynamic NN.
- Scikit-learn: Library with collection of tools for data preprocessing, feature selection, model evaluation etc.
- Keras: A Deep learning API on the top of TensorFlow or PyTorch.

Big data Technologies for processing and Storing ML Data

- Hadoop: Open source Big data framework includes Hadoop Distributed File System (HDFS) for distributed storage. Hadoop has the MapReduce programming model for processing large datasets.
- Apache Spark: Open source framework for distributed data processing provides libraries for various tasks including data preprocessing, ML, graph processing. Its in-memory processing capabilities accelerate computations using AI/ML. It overcomes limitations of Hadoop.

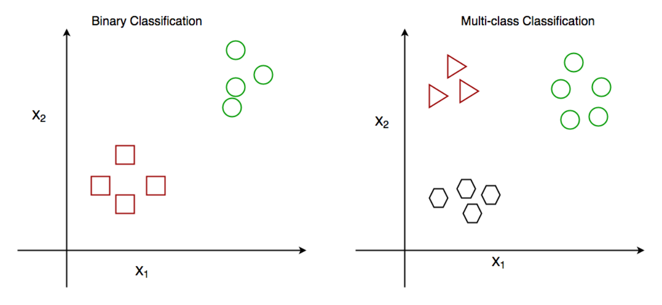
Data / Outliers / Anomalies

- Outliers: Distribution or dataset having unusual input for training.
- Overfitting: Outliers cause Overfitting.
- Sorting, grouping may help to detect Outliers.
- Anomaly may represent distribution or pattern but does not accurately reflect dataset. Outliers may be Anomalies while Anomalies are not Outliers.

Classification Algorithm:Supervised Learning

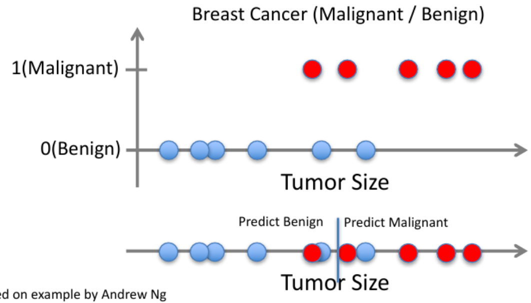
Classification problem:

Model or function to separate data into multiple categorical classes *i*.e. discrete values.



Supervised Learning: Classification

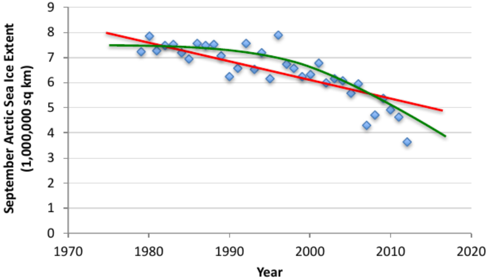
- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is categorical == classification



Based on example by Andrew Ng

Supervised Learning: Regression

- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is real-valued == regression



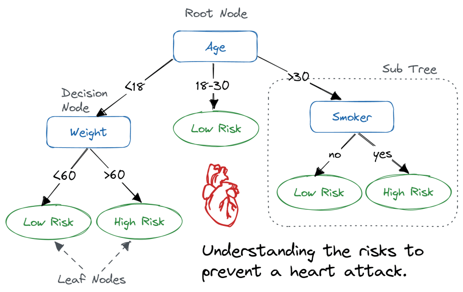
Data from G. Witt. Journal of Statistics Education, Volume 21, Number 1 (2013)

Decision Tree : Classification

- Consecutive set of questions (nodes).
- Only TWO possible answers per question.
- Each question depends on previous answers.
- Final verdict (leaf) is reached after a given maximum number of nodes.

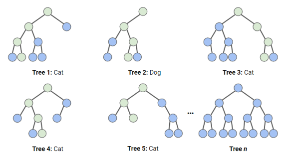
- Easy to understand/interpret
- Good with multivariate data
- Fast training
- Single tree is NOT very strong $\Rightarrow$ Random Forests

Decision Tree : Classification



- The topmost node is called Root Node.
- Root node learns to make partition on the basis of attribute values.
- Partitioning is done in recursive manner.
- Best attribute selection is heuristic and best attribute becomes a decision node.
- Easily capture non-linear patterns.
- It can be used for feature engineering to predict missing data etc.
- Sensitive to noisy data and may overfit.
- Small variation may produce different tree which can be fixed by bagging and boosting.

Random Forest: (Classification && Regression)



Green circle is hypothetical path the tree took to reach decision

- Random forest can be used for Regression (numeric target) and Classification (categorical target).
- Multiple decision trees are created using different random sets of data and features.
- Predictions are made by voting for classification and by averaging for regression.

Supervised Learning

Supervised Learning:

Training data : $\{(x_1, y_1), \ldots (x_N, y_N)\}$
x_i: feature vector, y_i: label (class) of i^{th} data, $g \in G$: Hypothesis space.
A learning algorithm seeks a function $g : X \to Y$, where X is input space and Y is output space.

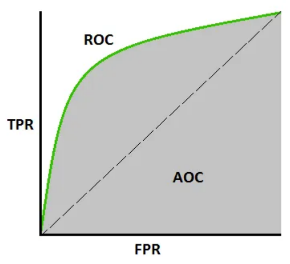
Logistic Regression Model:
1. Estimates the probablity of occurrence of an event based on given dataset of independent variables.
2. It is probability of a class.
3. Since outcome is probablity, dependent variable is bounded in $[0,\ 1]$

Supervised Learning:

Training data : $\{(x_1, y_1), \ldots (x_N, y_N)\}$
x_i: feature vector, y_i: label (class) of i^{th} data, $g \in G$: Hypothesis space.
A learning algorithm seeks a function $g : X \rightarrow Y$, where X is input space and Y is output space.

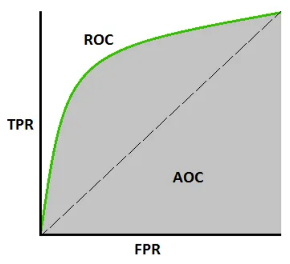
Support Vector Machine (SVM):
1. Goal is to creat decision boundary segregating n-dim space into classes.
2. Best decision boundary is called Hyperplane.
3. SVM : (a) Linear and (b) Non-linear.

Performance Measurement for Classification



AUC-ROC :Performance measurement.
AUC:- Area Under the Curve.
ROC:- Receiver Operating Characteristics.

Performance Measurement for Classification



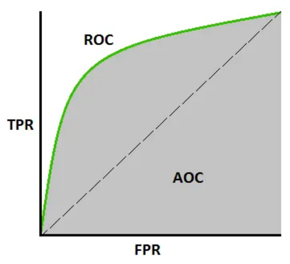
AUC-ROC :Performance measurement.

AUC:- Area Under the Curve.

ROC:- Receiver Operating Characteristics.

ROC is probability curve plotted for “True Positive Rate (TPR)” <Y-axis> against “False Positive Rate (FPR)” <X-axis>.

Performance Measurement for Classification



AUC-ROC :Performance measurement.

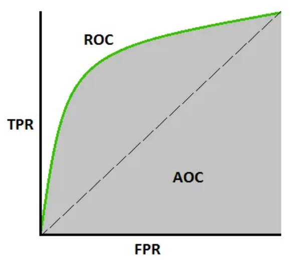
AUC:- Area Under the Curve.

ROC:- Receiver Operating Characteristics.

ROC is probability curve plotted for “True Positive Rate (TPR)” <Y-axis> against “False Positive Rate (FPR)” <X-axis>.

AUC is degree of separability. Efficiency of a model to separate classes.

Performance Measurement for Classification



AUC-ROC :Performance measurement.
AUC:- Area Under the Curve.
ROC:- Receiver Operating Characteristics.

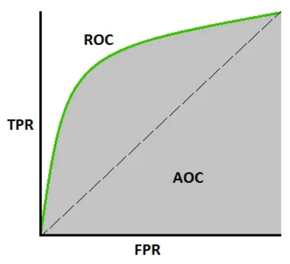
ROC is probability curve plotted for "True Positive Rate (TPR)" <Y-axis> against "False Positive Rate (FPR)" <X-axis>.

AUC is degree of separability. Efficiency of a model to separate classes.

$$\text{TPR/Recall/Sensitivity} = \frac{\text{True Positive}}{\text{True Positive} + \text{Full Negative}}$$

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{Full Positive}}$$

Performance Measurement for Classification



AUC-ROC :Performance measurement.
AUC:- Area Under the Curve.
ROC:- Receiver Operating Characteristics.

ROC is probability curve plotted for "True Positive Rate (TPR)" <Y-axis> against "False Positive Rate (FPR)" <X-axis>.

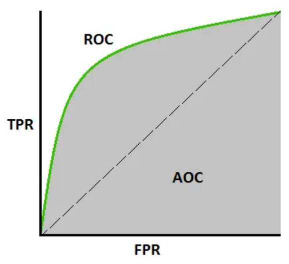
AUC is degree of separability. Efficiency of a model to separate classes.

$$\text{TPR/Recall/Sensitivity} = \frac{\text{True Positive}}{\text{True Positive} + \text{Full Negative}}$$

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{Full Positive}}$$

$$\text{FPR} = 1\text{-Specificity} = \frac{\text{Full Positive}}{\text{True Negative} + \text{Full Positive}}.$$

Performance Measurement for Classification



AUC-ROC :Performance measurement.

AUC:- Area Under the Curve.

ROC:- Receiver Operating Characteristics.

ROC is probability curve plotted for “True Positive Rate (TPR)” <Y-axis> against “False Positive Rate (FPR)” <X-axis>.

AUC is degree of separability. Efficiency of a model to separate classes.

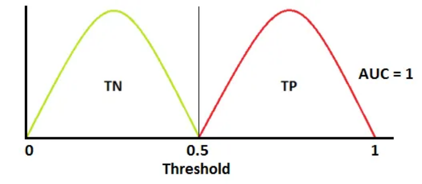
$$\text{TPR/Recall/Sensitivity} = \frac{\text{True Positive}}{\text{True Positive} + \text{Full Negative}}$$

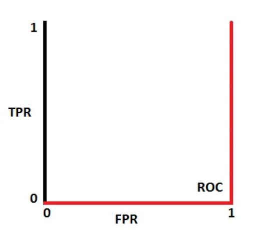
$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{Full Positive}}$$

$$\text{FPR} = 1\text{-Specificity} = \frac{\text{Full Positive}}{\text{True Negative} + \text{Full Positive}}.$$

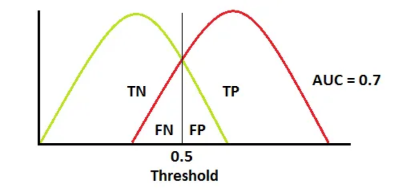
A Good model has AUC$\rightarrow$ 1 means good separability of classes.

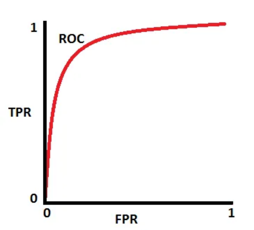
ROC: Probability Curve.





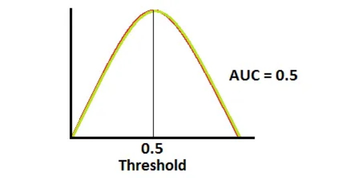
Ideal measure of probability. Positive class (e.g. patient with disease, Negative class (e.g. No disease))

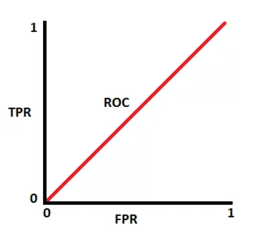




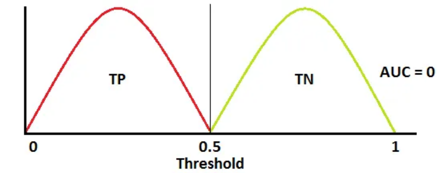
AUC=0.7 means the model may separate classes with 70% probability.

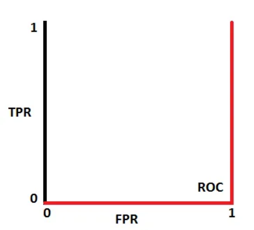
ROC: Probability Curve.





AUC=0.5 is the worst situation where model is unable to distinguish.



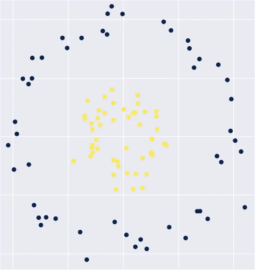


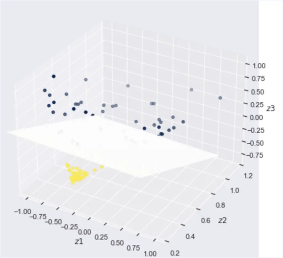
AUC=0 means the model is recognising a positive class as negative and vice versa.

The Kernels

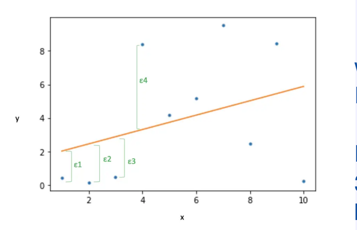
- Kernels (function) are a set of algorithms used for pattern matching.
- Usually non-linear problems are solved by linear classifier - “Kernel Tricks” « SVM ».
- The kernel function is applied on each data instance to map the original non-linear observations into a higher-dimensional space in which they become separable without computing the coordinates of the data in a higher dimensional space.
- Kernel Trick allows us to operate in the original feature space .

Kernels





- 2D dataset with 2 classes. Function to separate 2 classes is required. Data is NOT linealry separablein to 2 classes.
- One can fit a complex polynomial function to separate the data.
- Data may be transformed into 3D.
- A linear decision boundary may be found by fitting a linear classifier (a plane separating data) - *Hyperplane* .
- Map the linear decision boundary back into 2D space. The result will be a non-linear decision boundary in 2D



Let us consider a regression model:

$$y_i = w_0 + w_1 x_i + w_2 x_i + \epsilon_i$$
$$y_i = W^T x_i + \epsilon_i,$$

where, $W = \{w_0, w_1, w_2\}$ are weights.
Error: $\epsilon_i = (W^T x_i - y_i)$.

Let $x_a = [x_{a1}, x_{a2}]$ and $x_b = [x_{b1}, x_{b2}] \in \mathcal{R}^2$.
3D mapping, $x_i \longrightarrow \phi(x_i) : x_a^T x_b \longrightarrow \phi(x_a)^T \phi(x_b)$ and back to 2D,

$$K(x_a, x_b) = \phi(x_a)^T \phi(x_b)$$

Let, $K(x_a, x_b) = (x_a^T x_b)^2 = (x_{a1}x_{b1} + x_{a2}x_{b2})^2,$

$$= (x_{a1}^2 x_{b1}^2 + +2x_{a1}x_{a2}x_{b1}x_{b2} + x_{a2}^2 x_{b2}^2).$$

AND can be decomposed into $\phi(x_a) = \begin{pmatrix} x_{a1}^2 \\ \sqrt{2}x_{a1}x_{a2} \\ x_{a2}^2 \end{pmatrix}$ and $\phi(x_b) = \begin{pmatrix} x_{b1}^2 \\ \sqrt{2}x_{b1}x_{b2} \\ x_{b2}^2 \end{pmatrix}$.

In place of dot product we plug kernel K.

Kernels

Vectors: x_a and x_b.
Linear Kernel: $\mathrm{K}(x_a, x_b) = x_a\ x_b$.
Dot product measures similarity or distance in original feature space.

Polynomial Kernel: $\mathrm{K}(x_a, x_b) = (x_a\ x_b + c)d$,
d is the degree of the polynomial determines degree f nonlinearity.

Gaussian Kernel :: Radial Basis Function (RBF): $\mathrm{K}(x_a, x_b) = e^{-\gamma||x_a - x_b||^2}$.
The γ tunes the performance of the Gaussian kernel.

Laplace Kernel: $\mathrm{K}(x_a, x_b) = e^{-\gamma||x_a - x_b||}$.
$||\ x_a - x_b\ ||$ is Manhattan distance or L_1 norm between input vectors. It places less weight on large distance between input vectors than Gaussian kernel making it robust to Outliers.

Kernel Characteristics

- **Mercer's condition:** Ensures that the kernel function is positive semi definite, which means that it is always greater than or equal to zero.
- **Positive definiteness:** If kernel is always greater than zero except for when the inputs are equal to each other.
- **Non-negativity:** The kernel produces non-negative values for all inputs.
- **Symmetry:** A kernel function produces the same value regardless of the order in which the inputs are given.
- **Reproducing property:** A kernel function satisfies the reproducing property if it can be used to reconstruct the input data in the feature space.
- **Smoothness:** The kernel function produces a smooth transformation of the input data into the feature space.
- **Complexity:** More complex kernel functions may lead to over fitting and reduced generalization performance.

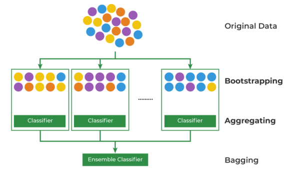
Ensemble Learning : Supervised

Ensemble Learning:

Combine the strengths of multiple models to make a single robust model less likely to overfit data.

Techniques:

- Averaging (for regression)
- Bagging (Bootstrap Aggregation),
- Boosting and
- Stacking (Stacked Generalization)



It can be used for both regression and classification.

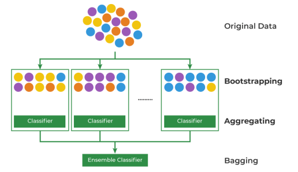
Bootstrap Sampling:
Randomly 'n' subsets of original data are sampled with replacement. Reduces of risks of overfitting increasing accuracy.

Original training dataset : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Resampled training set #1: [2, 3, 3, 5, 6, 1, 8, 10, 9, 1]
Resampled training set #2: [1, 1, 5, 6, 3, 8, 9, 10, 2, 7]
Resampled training set #3: [1, 5, 8, 9, 2, 10, 9, 7, 5, 4]

Some samples may be kept out of Sampling for verification of prediction.

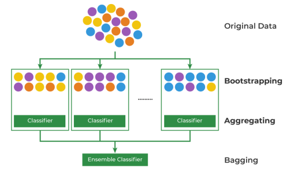
Bagging : Ensemble Learning



Base Model Training:

- Multiple base models are used.
- Each base model is independently trained using learning algorithm like **decision tree, SVM or Neural Networks**.
- Training is on different bootstrapped subset of data and can be parallelised.
- Each models are called "Weak Learners" as they may not be highly accurate of their own.

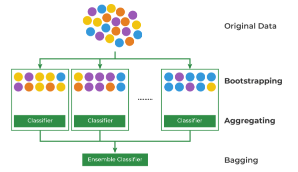
Bagging : Ensemble Learning



Aggregation:

- After training of all the base models, prediction is being made on unseen data.
- The Predicted class label is chosen on majority voting. <Classification>
- The final Prediction value is determined by averaging of the predictions from all base models. <Regression>

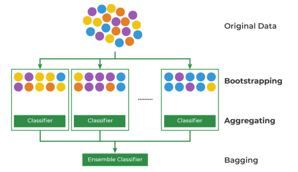
Bagging : Ensemble Learning



Out of Bag Evaluation:

- Some samples excluded in the bootstrapping are "Out-of-Bag" Samples.
- Out-of-Bag samples may be used to estimate the model performance

Bagging : Ensemble Learning



- Improved Predictive Performance: outperforms single classifier
- Robustness: Reduces impact of outliers and noises enhancing stability
- Reduced Variances: Since each base model is trained on different subsets, aggregated model's variance is reduced compared to indivudal model.
- Parallelization: Parallel processing of individual training reduces time.
- Flixibility: Wide range of algorithms can be used like DecisionTree, Random forests, support vector machine (SVM) etc.

Boosting : Ensemble Learning

- Boosting is a sequencial method

Boosting : Ensemble Learning

- Boosting is a sequencial method
- First a model is built from training data.

Boosting : Ensemble Learning

- Boosting is a sequencial method
- First a model is built from training data.
- Second model is built with an effort to correct errors in earlier model.

Boosting : Ensemble Learning

- Boosting is a sequencial method
- First a model is built from training data.
- Second model is built with an effort to correct errors in earlier model.
- Procedure continues

AND

models are added until **Either** the complete training data is predicted correctly **OR** maximum number of models have been added.

Boosting : Ensemble Learning

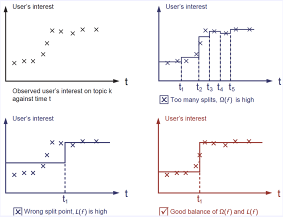
- Boosting is a sequencial method
- First a model is built from training data.
- Second model is built with an effort to correct errors in earlier model.
- Procedure continues
AND
models are added until **Either** the complete training data is predicted correctly **OR** maximum number of models have been added.
- Types (Important): Gradient Boosting, XG-Boost, AdaBoost, CatBoost

XGBoost (Extreme Gradient Boosting): Supervised Learning

Training data with multiple features x_i is used to predict target variable $\hat{y}_i$ by fitting.

- Model (*e.g.* Linear Model): Prediction $\hat{y}_i = \Sigma_j \theta_j x_{ij}$
- Training finds Best parameter θ_i using **Objective function** measuring degree of fitness.
- Objective function: $obj(\theta) = L(\theta) + \Omega(\theta)$:
  $L(\theta)$= training loss function, $\Omega(\theta)$ = regularization function.
- L: degree of prediction w.r.t. training data.
- Ω: controls complexity of the model helping to avoid Overfitting.

XGBoost (Extreme Gradient Boosting): Supervised Learning



$obj(\theta)$: (Loss $L(\theta)$ + Regularization $\Omega(\theta)$).

$\mathbf{L}(\theta)$:

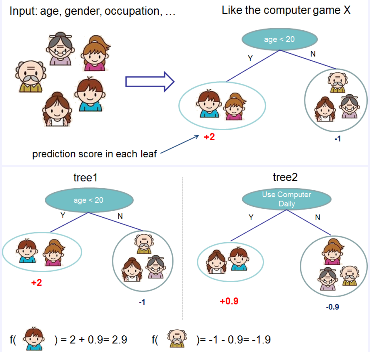
(a) Mean Squared Error (MSE): $L(\theta) = \Sigma_i (y_i - \hat{y}_i)^2$

(b) Logistic: $L(\theta) = \sum_i \left[y_i \ln\left(1 + e^{-\hat{y}_i}\right) + (1 - y_i)\ln\left(1 + e^{\hat{y}_i}\right)\right]$

$\mathbf{\Omega}(\theta)$: Fit a step function visually given input data points. Which of the 3 solutions is best fit?

Tradeoff between L and Ω is "**Bias-Variance trade-off**".

XGBoost (Extreme Gradient Boosting): Supervised Learning



Classify: who likes computer game.
Final score=Σ individual tree scores.

XGBoost: Decision Tree Ensemble: Set of Classification and Regression trees (CART).

Leaf also contains score.

2 trees complement each other:
$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i),\ f_k \in \mathcal{F}.$
$\mathcal{F}$: set of all possible CARTS, f_k : function in functional space $\mathcal{F}$, K: number of trees.

$obj(\theta) = \sum_i^n l(y_i,\ \hat{y}_i) + \sum_{k=1}^{K} w(f_k)$
$w(f_k)$: complexity depends on the scores.

Boosted Decision Tree Random Forrest!

ONE predictive service code : Different Training.

XGBoost (Extreme Gradient Boosting): Supervised Learning

Complexity

Define tree $f(x) = w_{q(x)}, \; w \in \mathcal{R}^T, \; q : R^d \rightarrow \{1, 2, 3, \ldots T\}$
w=scores on leaves, q= fn assigning each data point to corresponding leaf,
T= number of leaves.

$$w(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2.$$

XGBoost (Extreme Gradient Boosting): Supervised Learning

Loss Function

Prediction values:

$$\hat{y}_i^{(0)} = 0$$
$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)$$
$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i)$$
$$\ldots$$
$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

At each step, tree is selected by optimized objective function.

$$obj^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t)}\right) + \sum_{i=1}^{t} w\left(f_i\right) = \sum_{i=1}^{n} \left(y_i - \left(\hat{y}_i^{(t-1)} + f_t(x_i)\right)\right)^2 + \sum_{i=1}^{t} w(f_i), \text{ (MSE)}$$

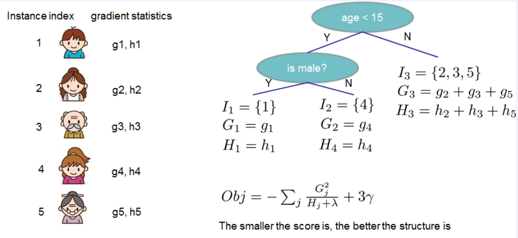
Objective at step $t \equiv$ goal for new tree

$$\sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + w(f_t)$$

.
$$g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)}) \qquad h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

XGBoost (Extreme Gradient Boosting): Supervised Learning

$$
\begin{aligned}
obj^{(t)} &\approx \sum_{i=1}^{n}\left[g_i w_{q(x_i)} + \frac{1}{2}h_i w_{q(x_i)}^2\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2, \\
&= \sum_{j=1}^{T}\left[w_j \sum_{i\in I_j} g_i + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right) w_j^2\right] + \gamma T, (\ \forall \text{ data in same leaf gets same score.}) \\
&= \sum_{j=1}^{T}\left[G_j w_j + \frac{1}{2}\left(H_j + \lambda\right) w_j^2\right] + \gamma T \quad \Rightarrow I_j = \{i \mid q(x_i) = j\}.
\end{aligned}
$$



w_j^s are independent.

$w_j^\star = -\frac{G_j}{H_j+\lambda}$

$obj^\star = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda} + \gamma T$

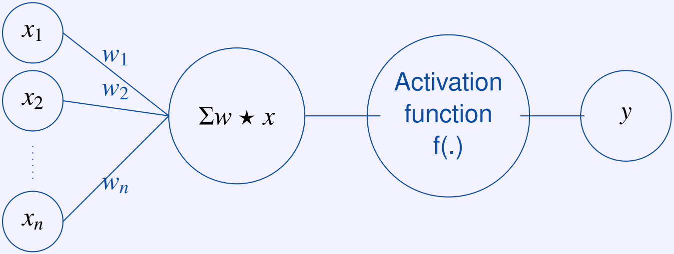
Smaller the score is,
the better the structure is.

AdaBoost (Adaptive Boosting) : Ensemble Learning

- Enhances weights of misclassified events after each training
- Reduces weights of correctly classified events so that future trees learn better
- Iteration continues until weight of misclassified . 50%
- Final weight is the sum of all classifiers weighted by their errors

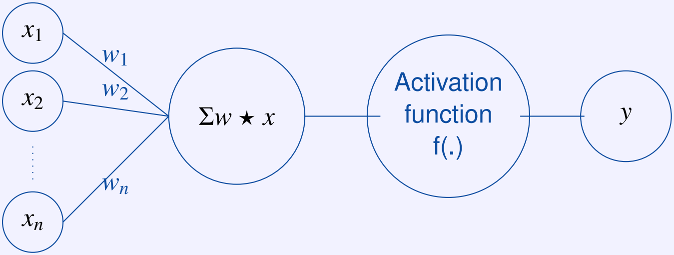
Neural Networks

- Artificial system being inspired from biological neural networks.



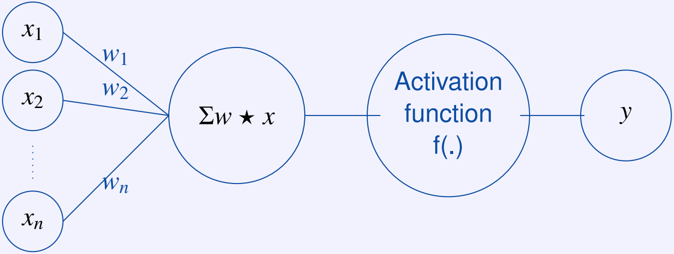
Neural Networks

- Artificial system being inspired from biological neural networks.
- The computational model is based on Threshold Logic.



Neural Networks

- Artificial system being inspired from biological neural networks.
- The computational model is based on Threshold Logic.
- It is a type of ML process that uses interconnected nodes/neurones in a layered structure called as Deep learning.



Neural Networks

- Artificial system being inspired from biological neural networks.
- The computational model is based on Threshold Logic.
- It is a type of ML process that uses interconnected nodes/neurones in a layered structure called as Deep learning.
- Algorithm updates itself through "backpropagation" as per optimization strategy.