Machine Learning

—

By Roshan Sharma

Machine Learning is the Study of Algorithms and Scientific Models that can perform tasks without human intervention

Types of

Machine Learning

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Let’s get Started

Supervised Learning

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labeled training data consisting of a set of training examples

Unsupervised Learning

Unsupervised learning is a type of self-organized Hebbian learning that helps find previously unknown patterns in data set without pre-existing labels. It is also known as self-organization and allows modeling probability densities of given inputs.

Reinforcement Learning

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning

Types of Supervised Learning

- Classification Analysis
- Regression Analysis

Types of Unsupervised Learning

- Dimensionality Reduction
- Clustering Analysis

Classification Algorithms

A **classification algorithm**, in general, is a function that weighs the input features so that the output separates one class into positive values and the other into negative values.

The Main Purpose of Classification Algorithms is to Classify the target variables into two or more classes.

Types of Classification Algorithms

- Logistic Regression
- Support Vector Machine
- K Nearest Neighbors
- Adaptive Boost
- Random Forest
- Lg Boost
- XgBoost
- Gradient Boosting
- Naive Bayes
- Decision Trees
- Cat Boost

Types of Regression Algorithms

- Linear Regression
- Lasso
- Ridge
- Elastic Net
- Polynomial Regression
- Principal Components Regression
- Quantile Regression

Linear Regression

linear regression is a linear approach to modeling the relationship between a scalar response and one or more explanatory variables. The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression.

Linear Regression

Formula: Y(pred) = b0 + b1*x

The values b0 and b1 must be chosen so that they minimize the error. If sum of squared error is taken as a metric to evaluate the model, then goal to obtain a line that best reduces the error.

$$\text{Error} = \sum_{i=1}^{n} (actual\_output - predicted\_output) ** 2$$

If the model does not include x=0, then the prediction will become meaningless with only b0. For example, we have a dataset that relates height(x) and weight(y). Taking x=0(that is height as 0), will make equation have only b0 value which is completely meaningless as in real-time height and weight can never be zero. This resulted due to considering the model values beyond its scope.

$$b_0 = \bar{y} - b_1\bar{x}$$

- If b1 > 0, then x(predictor) and y(target) have a positive relationship. That is increase in x will increase y.
- If b1 < 0, then x(predictor) and y(target) have a negative relationship. That is increase in x will decrease y.

$$b_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Logistic Regression

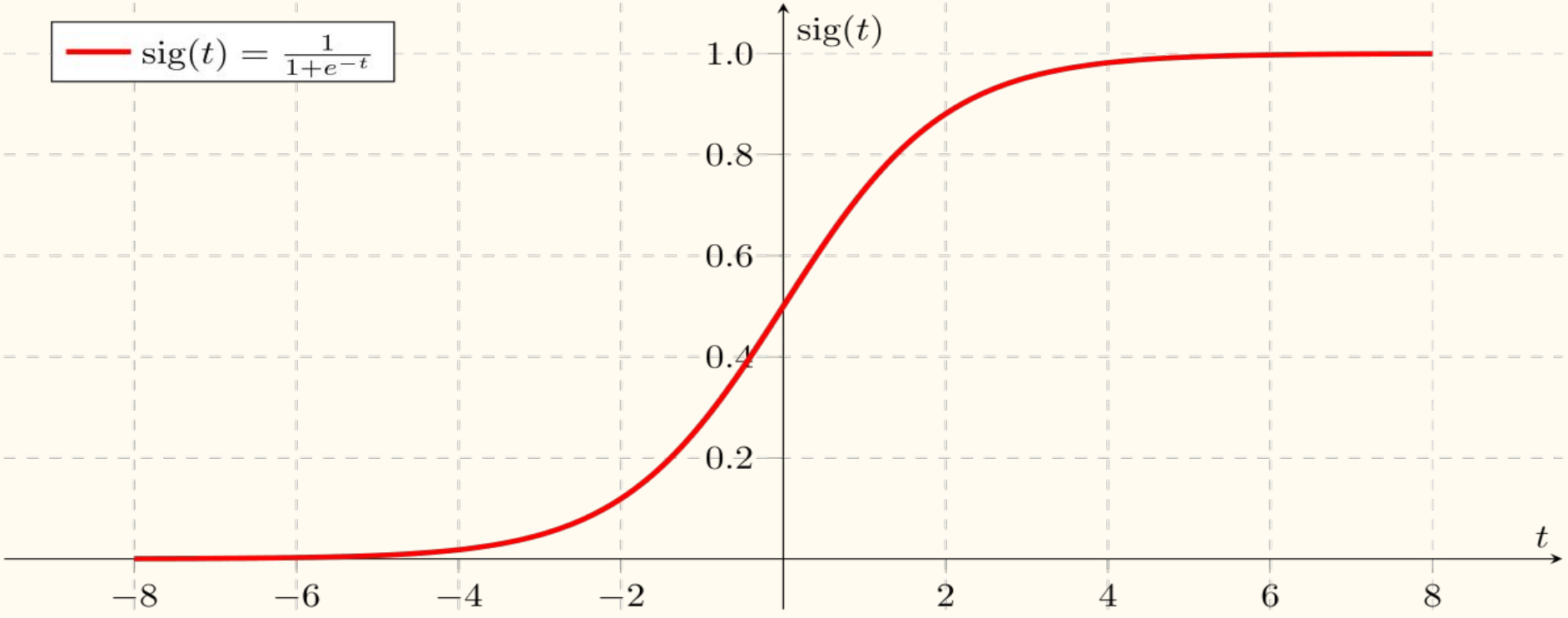
Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

Model

- Output = 0 or 1
- Hypothesis => Z = WX + B
- hΘ(x) = sigmoid (Z)
- Sigmoid Function: 1/1+e-t

Sigmoid Function



Decision Boundary

To predict which class a data belongs, a threshold can be set. Based upon this threshold, the obtained estimated probability is classified into classes.

Say, if predicted_value ≥ 0.5, then classify email as spam else as not spam.

Decision boundary can be linear or non linear. Polynomial order can be increased to get complex decision boundary.

Cost Function

$$\text{Cost}(h_\theta(x), Y(\text{actual})) = -\log(h_\theta(x)) \text{ if } y=1$$

$$-\log(1-h_\theta(x)) \text{ if } y=0$$

Decision Trees

A Decision Tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.

It is one way to display an algorithm that only contains conditional control statements.

Each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

Tree based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based methods empower predictive models with high accuracy, stability and ease of interpretation.

Unlike linear models, they map nonlinear relationships quite well. They are adaptable at solving any kind of problem at hand (classification or regression). Decision Tree algorithms are referred to as CART (Classification and Regression Trees).

Common Terms

- **Root Node:** It represents entire population or sample and this further gets divided into two or more homogeneous sets.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.

- **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.
- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
- **Branch / Sub-Tree:** A sub Section of entire tree is called branch or sub-tree.
- **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes whereas sub-nodes are the child of parent node.

Applications

- When the Object is to Optimize the Cost.
- When there are several courses of action.
- When there is a calculable measure of alternates.
- When Environmental Factors are involved, uncertain
- Uncertainty regarding Outcomes

Advantages

- **Easy to Understand and Interpret.**
- **Useful in Data Exploration.**
- **Implicitly perform feature selection.**
- **Lesser Data Cleaning and Preparation required.**
- **Can handle both continuous and categorical data.**

Disadvantages

- **Overfitting due to over complex trees.**
- **Not fit for continuous data, as it loses information while converting numerical data into categorical.**
- **The model can change completely due to a small change in the trees.**
- **It can create biased trees, if some classes dominate.**

- **Information gain in a decision tree with categorical variable can give a biased response for greater number of categories.**
- **Calculations become complex if there are many labels.**
- **Greedy algorithms do not guarantee to returrn the optimal decision tree as the features and samples are randomly sampled with placement.**

How to Decide where to Split the Decision Tree ?

There are Four Algorithms :

- **Gini Index**
- **Chi-Square**
- **Information Gain**
- **Reduction in Variance**

Gini Index

Gini index says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.

1. It works with categorical target variable “Success” or “Failure”.
2. It performs only Binary splits
3. Higher the value of Gini higher the homogeneity.
4. CART (Classification and Regression Tree) uses Gini method to create binary splits.

Steps to calculate Gini Index

1. Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure (p^2+q^2).
2. Calculate Gini for split using weighted Gini score of each node of that split

Chi Square

Chi Square an algorithm to find out the statistical significance between the differences between sub-nodes and parent node.

We measure it by sum of squares of standardized differences between observed and expected frequencies of target variable.

$$\textbf{Chi-square} = ((\textbf{Actual} - \textbf{Expected})^2 / \textbf{Expected})^{1/2}$$

Steps to Calculate Chi Square

1. Calculate Chi-square for individual node by calculating the deviation for Success and Failure both
2. Calculated Chi-square of Split using Sum of all Chi-square of success and Failure of each node of the split

Information Gain

Less impure node requires less information to describe it. And, more impure node requires more information. Information theory is a measure to define this degree of disorganization in a system known as Entropy. If the sample is completely homogeneous, then the entropy is zero and if the sample is an equally divided (50% — 50%), it has entropy of one.

Entropy = -p log2 p — q log2q.

Here p and q is probability of success and failure respectively in that node. Entropy is also used with categorical target variable. It chooses the split which has lowest entropy compared to parent node and other splits. The lesser the entropy, the better it is.

Steps to Calculate Information Gain

1. Calculate entropy of parent node
2. Calculate entropy of each individual node of split and calculate weighted average of all sub-nodes available in split.

We can derive information gain from entropy as **1- Entropy.**

Reduction in Variance

Reduction in variance is an algorithm used for continuous target variables (regression problems). This algorithm uses the standard formula of variance to choose the best split. The split with lower variance is selected as the criteria to split the population:

$$\text{Variance} = \frac{\Sigma(X - \overline{X})^2}{n}$$

Steps to Calculate Variance

1. Calculate variance for each node.
2. Calculate variance for each split as weighted average of each node variance.

How to avoid Overfitting ?

- Setting Constraints on tree Size.
- Tree Pruning.

Setting Constraints

Minimum samples for a node split

- Defines the minimum number of samples (or observations) which are required in a node to be considered for splitting.
- Used to control over-fitting. Higher values prevent a model from learning relations which might be highly specific to the particular sample selected for a tree.
- Too high values can lead to under-fitting hence, it should be tuned using CV.

Minimum samples for a terminal node (leaf)

- Defines the minimum samples (or observations) required in a terminal node or leaf.
- Used to control over-fitting similar to min_samples_split.
- Generally lower values should be chosen for imbalanced class problems because the regions in which the minority class will be in majority will be very small.

Maximum depth of tree (vertical depth)

- The maximum depth of a tree.
- Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.
- Should be tuned using CV.

Maximum number of terminal nodes

- The maximum number of terminal nodes or leaves in a tree.
- Can be defined in place of max_depth. Since binary trees are created, a depth of ’n’ would produce a maximum of 2^n leaves.

Maximum features to consider for split

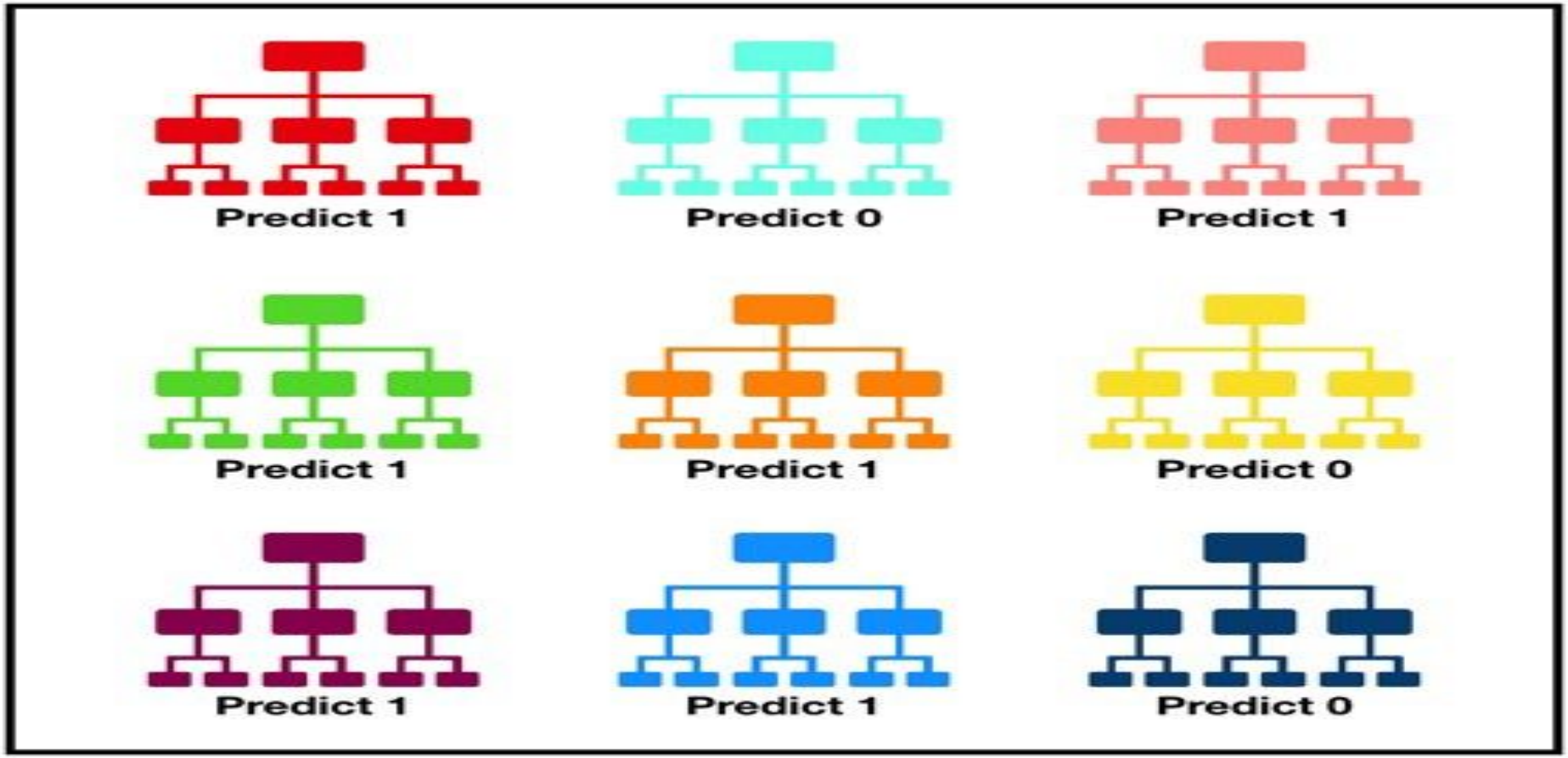
- The number of features to consider while searching for a best split. These will be randomly selected.
- As a thumb-rule, square root of the total number of features works great but we should check upto 30–40% of the total number of features.
- Higher values can lead to overfitting but depends on case to case.

Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees.

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.



Tally: Six 1s and Three 0s

Prediction: 1

Ensure Model Variance

Bagging (Bootstrap Aggregation)

Decisions trees are very sensitive to the data they are trained on small changes to the training set can result in significantly different tree structures.

Random forest takes advantage of this by allowing each individual tree to randomly sample from the dataset with replacement, resulting in different trees. This process is known as bagging.

Feature Randomness

In a normal decision tree, when it is time to split a node, we consider every possible feature and pick the one that produces the most separation between the observations in the left node vs. those in the right node. In contrast, each tree in a random forest can pick only from a random subset of features. This forces even more variation amongst the trees in the model and ultimately results in lower correlation across trees and more diversification.

Advantages of Random Forests

- The predictive performance can compete with the best supervised learning algorithms
- They provide a reliable feature importance estimate
- They offer efficient estimates of the test error without incurring the cost of repeated model training associated with cross-validation

Disadvantages of Random Forests

- An Ensemble model is inherently less interpretable than an individual decision tree
- Training a large number of deep trees can have high computational costs (but can be parallelized) and use a lot of memory
- Predictions are slower, which may create challenges for applications.

When to use Random Forest ?

- When you don't bother much about interpreting the model but want better accuracy.
- Random forest will reduce variance part of error rather than bias part, so on a given training data set decision tree may be more accurate than a random forest. But on an unexpected validation data set, Random forest always wins in terms of accuracy.

When to use Decision Tree ?

- When you want your model to be simple and explainable
- When you want non parametric model
- When you don't want to worry about feature selection or regularization or worry about multicollinearity.
- You can overfit the tree and build a model if you are sure of validation or test data set is going to be subset of training data set or almost overlapping instead of unexpected.

K Nearest Neighbors

The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph.

Algorithm

1. Load the data
2. Initialize K to your chosen number of neighbors
3. For each example in the data
   a. Calculate the distance between the query example and the current example from the data.
   b. Add the distance and the index of the example to an ordered collection

4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances

5. Pick the first K entries from the sorted collection

6. Get the labels of the selected K entries

7. If regression, return the mean of the K labels

8. If classification, return the mode of the K labels

Advantages of KNN

1. The algorithm is simple and easy to implement.
2. There’s no need to build a model, tune several parameters, or make additional assumptions.
3. The algorithm is versatile. It can be used for classification, regression, and search

Disadvantages of KNN

The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

Applications of KNN

- Recommender Systems
- Classification and Regression
- Outlier Detection
- Search based applications

Support Vector Machines

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

Hyperplanes

Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3

Support Vectors

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

Margin Intuition

In **logistic** regression, we take the output of the linear function and squash the value within the range of [0,1] using the sigmoid function. If the squashed value is greater than a threshold value(0.5) we assign it a label 1, else we assign it a label 0. In **SVM**, we take the output of the linear function and if that output is greater than 1, we identify it with one class and if the output is -1, we identify is with another class. Since the threshold values are changed to 1 and -1 in SVM, we obtain this reinforcement range of values([-1,1]) which acts as margin.

Cost Function

$$(\theta) = C\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{i=1}^{n}\theta_i^2$$

The Cost Function is used to train the SVM. By minimizing the value of J(theta), we can ensure that the SVM is as accurate as possible. In the equation, the functions cost1 and cost0 refer to the cost for an example where y=1 and the cost for an example where y=0. Cost, for SVMs, is determined by kernel (similarity) functions.

Advantages of SVM

- SVM works relatively well when there is clear margin of separation between classes.
- SVM is more effective in high dimensional spaces.
- SVM is effective in cases where number of dimensions is greater than the number of samples.
- SVM is relatively memory efficient

Disadvantages of SVM

- SVM algorithm is not suitable for large data sets.
- SVM does not perform very well, when the data set has more noise i.e. target classes are overlapping.
- In cases where number of features for each data point exceeds the number of training data sample , the SVM will underperform.
- As the support vector classifier works by putting data points, above and below the classifying hyper plane there is no probabilistic explanation for the classification.

Types of Kernel Functions

- Polynomial Kernel
  - It is Popular in Image Processing
  - $k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$
- Gaussian Kernel
  - Used when there is no prior knowledge about data
  - Formula:

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- Laplace RBF:
  - Formula:

$$k(x, y) = \exp\left(-\frac{\|x - y\|}{\sigma}\right)$$

- Radial Basis Function
  - Formula:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2)$$

- Hyperbolic Tangent
  - Formula:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \tanh(\kappa \mathbf{x_i} \cdot \mathbf{x_j} + c)$$

Naive Bayes

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

Chain Rule :

$$P(y|x_1,...,x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

you can obtain the values for each by looking at the dataset and substitute them into the equation. For all entries in the dataset, the denominator does not change, it remain static. Therefore, the denominator can be removed and a proportionality can be introduced.

$$P(y|x_1,...,x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

Types of Naive Bayes Classifier

- **Multinomial Naive Bayes**
  - This is mostly used for document classification problem, i.e whether a document belongs to the category of sports, politics, technology etc. The features/predictors used by the classifier are the frequency of the words present in the document.
- **Bernoulli Naive Bayes**
  - This is similar to the multinomial naive bayes but the predictors are boolean variables. The parameters that we use to predict the class variable take up only values yes or no, for example if a word occurs in the text or not.

- **Gaussian Naive Bayes**
  - When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a gaussian distribution.

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Advantages of Naive Bayes

- Can be used in Sentiment Analysis
- Works well in Spam Filtering
- Can be effectively used for Recommendation Engines
- Fast to Implement
- Easy to Implement

Disadvantages of Naive Bayes

The biggest disadvantage of Naive Bayes Classifier in a real world scenario is that the requirement of predictors to be independent. In most of the real life cases, the predictors are dependent, this hinders the performance of the classifier.

Conclusion

In this Section, We have Studied the Basic Classification Models, In the Next Section We are going to Deal with the Regression Models.

We have Learnt Linear Regression, Logistic Regression, SVM, KNN, Decision Trees, Random Forests and Naive Bayes Classifier