* **K- nearest neighbours classifier:**

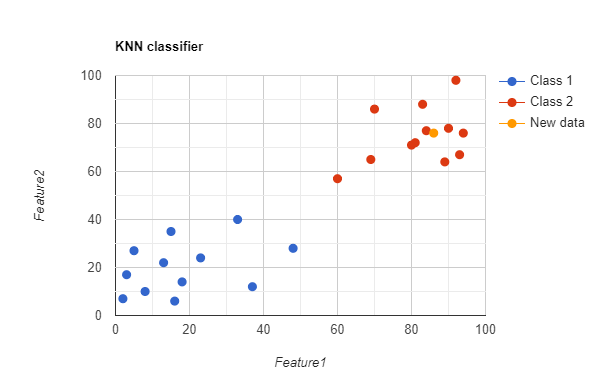
K-nearest neighbours classifier is a supervised learning technique which is non-parametric i.e. it makes no assumptions about the underlying distributions of the data. The working principle of the algorithm is the prediction of the classes based on some similarity measures (generally Euclidian distance). The new data point is classified by observing the k nearest data points around it and the most common class among the k neighbours is assigned the class to the new data point.

Euclidian distance is the measure of distance between two points in a Cartesian coordinate system. For instance, let us assume two data points M and N having coordinates (xm, ym) and (xn, yn) respectively. Then the Euclidian distance between the points can be calculated as:

D(M, N) =

For classification using the knn algorithm, the dataset is first split into the training and testing part. Then for each new data point, the k nearest neighbours are considered and assigned the most common class among the neighbours.

The number of neighbours i.e. K is the hyperparameter that needs to be adjusted carefully to avoid problems like overfitting or underfitting.



* **Support Vector classifier:**

Support vector classifier (SVC) is a classification technique used for supervised learning problems. SVC finds out the optimal separating hyperplane also called the decision boundary that separates the data points belonging to different classes. The goal of SVC is to maximise the distance between data points close to the margin and the separating hyperplane. The data points closest to the margin are called support vectors and hence the name.

SVC has a few important parameters which need to be tuned carefully for proper results:

1. c: Regularization parameter with default value 1.0, type float.
2. kernels: Out of several choices for the kernels the appropriate one must be chosen (more on kernels later).

Popular choices for kernels:

1. Linear: It calculates the dot product of input vectors.
2. Radial Basis Function (RBF): It uses a Gaussian function to map the input vectors to an infinite dimensional feature space.
3. Polynomial: It is a non-linear kernel that calculates the similarity between two vectors using the following relation:
4. Sigmoid: It uses a tan hyperbolic function to compute the similarity between the inputs. It is a non-linear kernel.

Where x,x’ are the input vectors, is the hyperparameter controlling the width of the Gaussian kernel, d is the hyperparameter degree and coef0 is the coefficient hyperparameter.

The Support Vector machine uses hinge loss as a loss function. The goal of SVM is to minimise hinge loss. The loss function is given as:

Where yi and xi are the ith instance in the training set and b is the bias.

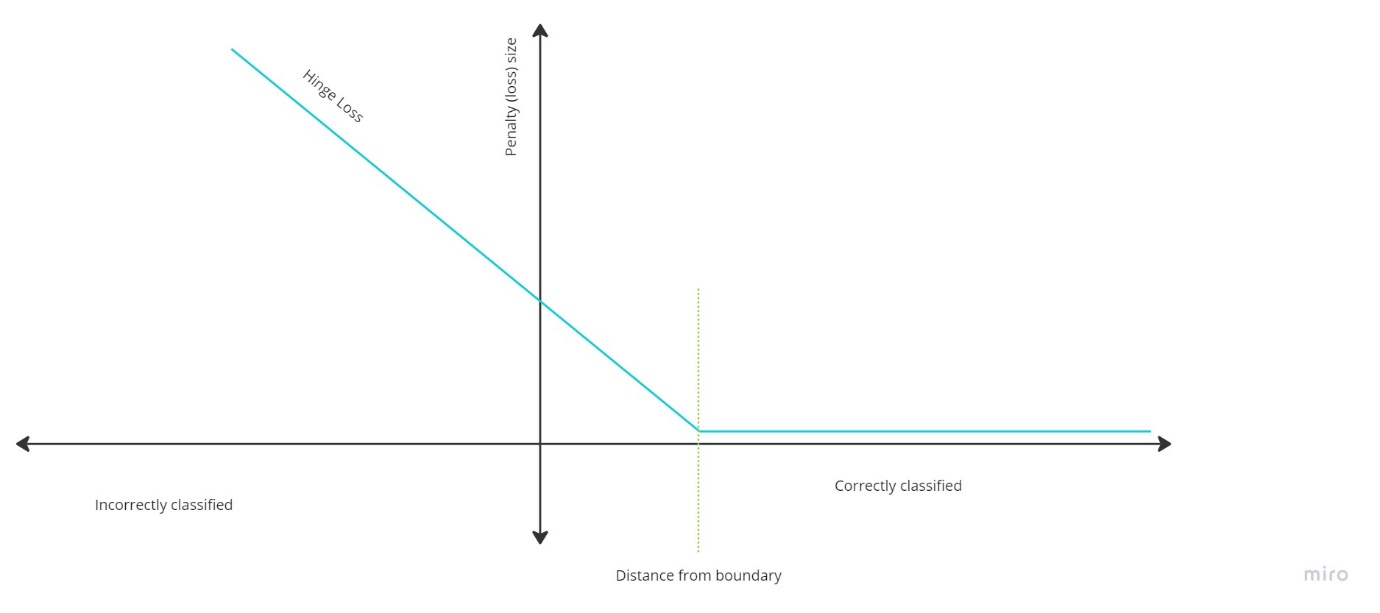
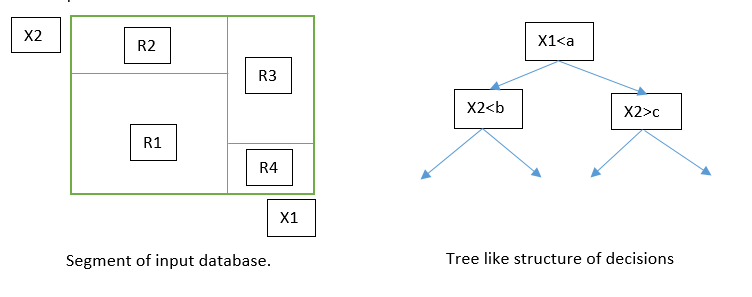


Figure showing the graph of hinge loss.

* **Decision Tree Classifier:**

The decision tree classifier is a supervised learning technique used for the classification of input. It works by continuously separating the input space into several segments and making decisions based on certain criteria. This algorithm can be easily visualised as a tree of decisions which has consequences at each decision node.



After building the tree, sometimes it tends to take a very complex form which has very high computational complexity. In order to reduce the computational load the tree is pruned which also avoids overfitting.

The decision tree algorithm uses various metrics to select the feature for the best split. Some of the commonly used metrics are:

1. Information Gain:

1. Gini Index:

Where D is the instance of the dataset, c is the number of possible classes and pi is the probability that Di belongs to class i.

* **Random forest classifiers:**

Random forest classifier is a bit different classification algorithm. It is an ensemble learning technique which uses multiple decision trees to generate its output (a group of trees is called a forest).

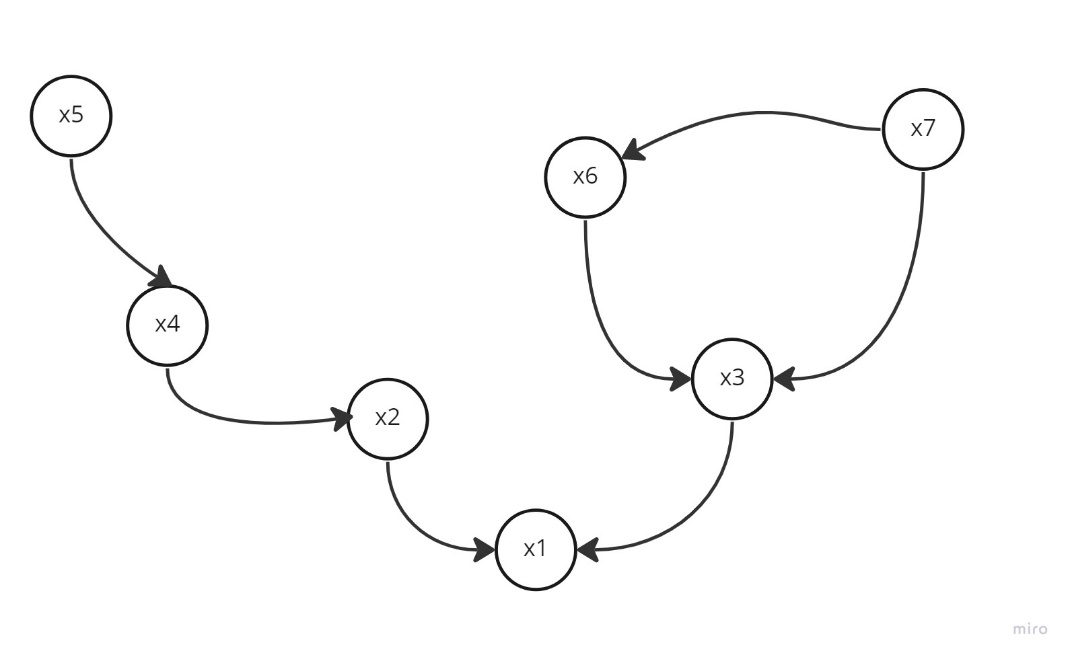
Bagging of these individual trees is done to reduce the correlation between trees. This algorithm creates multiple decision trees where each tree constructed uses a different subset of features, selected with replacement also called Bootstrapping.

After all, the trees are constructed and output generated, the final output is obtained by taking the majority voting or the mean of all predictions. A random forest works similarly to the decision tree and hence is mathematically identical. However, prediction is generated as follows:

* **Naïve Bayes Classifier:**

Naïve Bayes classifier is a probabilistic machine learning algorithm used for classification. The backbone for this algorithm is the Bayesian network, also known as the Bayes network or Belief network.

A Bayesian network is a directed probability graph which depicts the conditional dependency of the random variables on one another. The Bayesian network has to be an acyclic graph.



Let us consider the above Bayesian network. This network shows that x5 is an independent variable and x4 depends on x5 which is shown by an arrow from a random variable to the dependent variable. The joint probability distribution of the network can be factorized by using the chain rule. The factorization of the above network can be observed to be:

From this graph we can also say, knowing x2 (say) separates x1 and x4 i.e. x1 and x4 are conditionally independent if x2 is known.

The Naïve Bayes classification algorithm uses the principle of Bayes theorem assuming that all the features are independent of each other and hence the name. The model calculates the probability of an instance belonging to a class based on the features associated with it. This is done by calculating the conditional probability of each feature given the class and then calculating the total probability using the Bayes theorem.

The instance having the highest probability of belonging to a class is classified to be a point in that class.

Conditional probability is calculated using Bayes theorem as follows:

Naive Bayes has different variations and can be used for both binary and multi-class classification.

* **Logistic Regression:**

Logistic regression is a classification algorithm for binary class problems using a statistical approach. This algorithm works very well where the prediction of the probability of events is involved. The logistic regression algorithm uses a logit transformation of the probability as follows:

Solving which gives us:

Which is nothing but a sigmoid.

Where β0 and β are the weights and p(x) is the probability of instance x of the dataset.

Logistic regression uses the maximum likelihood estimation method to estimate the parameters of the linear combination and maximises the likelihood of the observed data given the model. An iterative algorithm such as gradient descent is used to estimate the parameters.

* **Extreme Gradient Boosting Classifier:**

Extreme Gradient boosting classifier, popularly known as XGBoost classifier, works using an ensemble of decision trees. This algorithm also uses multiple decision trees but in a sequential manner which is not the case in Random Forest. Here every subsequent tree tries to correct the errors of the previous tree.

The gradient-boosting algorithm is used to optimize an objective function:

Where loss is the loss function of the algorithm and Ω is the regularization function used for penalization. Actual and predicted are the vectors of actual labels and predicted labels respectively.

The loss function used can be any of the popular choices such as Mean Squared Error (MSE), log loss, hinge loss etc. The mean squared error can be calculated as:

Where yi is the actual label and f(xi) is the predicted label of the ith instance of the dataset having N data points.

The regularization parameter can use various methods like L1 or L2 norms. Here Ω using L2 norm is represented as:

Where Ὑ is the strength of regularization and w is the weight of the leaves.

XGBoost also uses pruning to decrease computational complexity and prevent overfitting.

* **Gradient Boosting classifier:**

The Gradient Boosting algorithm uses the idea of combining various weak classifiers to make a strong classifier altogether. It iteratively keeps on adding new trees to the ensemble and updates the weights of the training examples based on the errors of the previous trees.

The algorithm first initialises the ensemble with the mean of the target variables (say). Then for each iteration i, fits a regression tree to the negative gradient of the loss function. The negative gradient can be calculated as:

Where Loss() is the loss function used, yn is the actual label and Fn is the predicted label for the nth index of the training samples for the mth iteration.

Then the optimal leaf weight is calculated for the new tree minimises the loss function.

Where is the prediction of the current tree for the ith instance given that it falls in the jth leaf.

The weights are then updated with the new tree and the optimal weight.

These steps are repeated until the desired result is achieved.