

[Upgrade](https://medium.com/membership?source=upgrade_membership---nav_full------------------------)



[[](https://towardsdatascience.com/?source=post_page-----24f9ff6ab222----------------------)](https://towardsdatascience.com/?source=post_page-----24f9ff6ab222----------------------)

* [DATA SCIENCE](https://towardsdatascience.com/data-science/home?source=post_page-----24f9ff6ab222----------------------)
* [MACHINE LEARNING](https://towardsdatascience.com/machine-learning/home?source=post_page-----24f9ff6ab222----------------------)
* [PROGRAMMING](https://towardsdatascience.com/programming/home?source=post_page-----24f9ff6ab222----------------------)
* [VISUALIZATION](https://towardsdatascience.com/data-visualization/home?source=post_page-----24f9ff6ab222----------------------)
* [AI](https://towardsdatascience.com/artificial-intelligence/home?source=post_page-----24f9ff6ab222----------------------)
* [VIDEO](https://towardsdatascience.com/video/home?source=post_page-----24f9ff6ab222----------------------)
* [ABOUT](https://towardsdatascience.com/about-us/home?source=post_page-----24f9ff6ab222----------------------)
* [CONTRIBUTE](https://towardsdatascience.com/contribute/home?source=post_page-----24f9ff6ab222----------------------)

Top highlight

**Comparative Study on Classic Machine learning Algorithms**

Quick summary on various ML algorithms

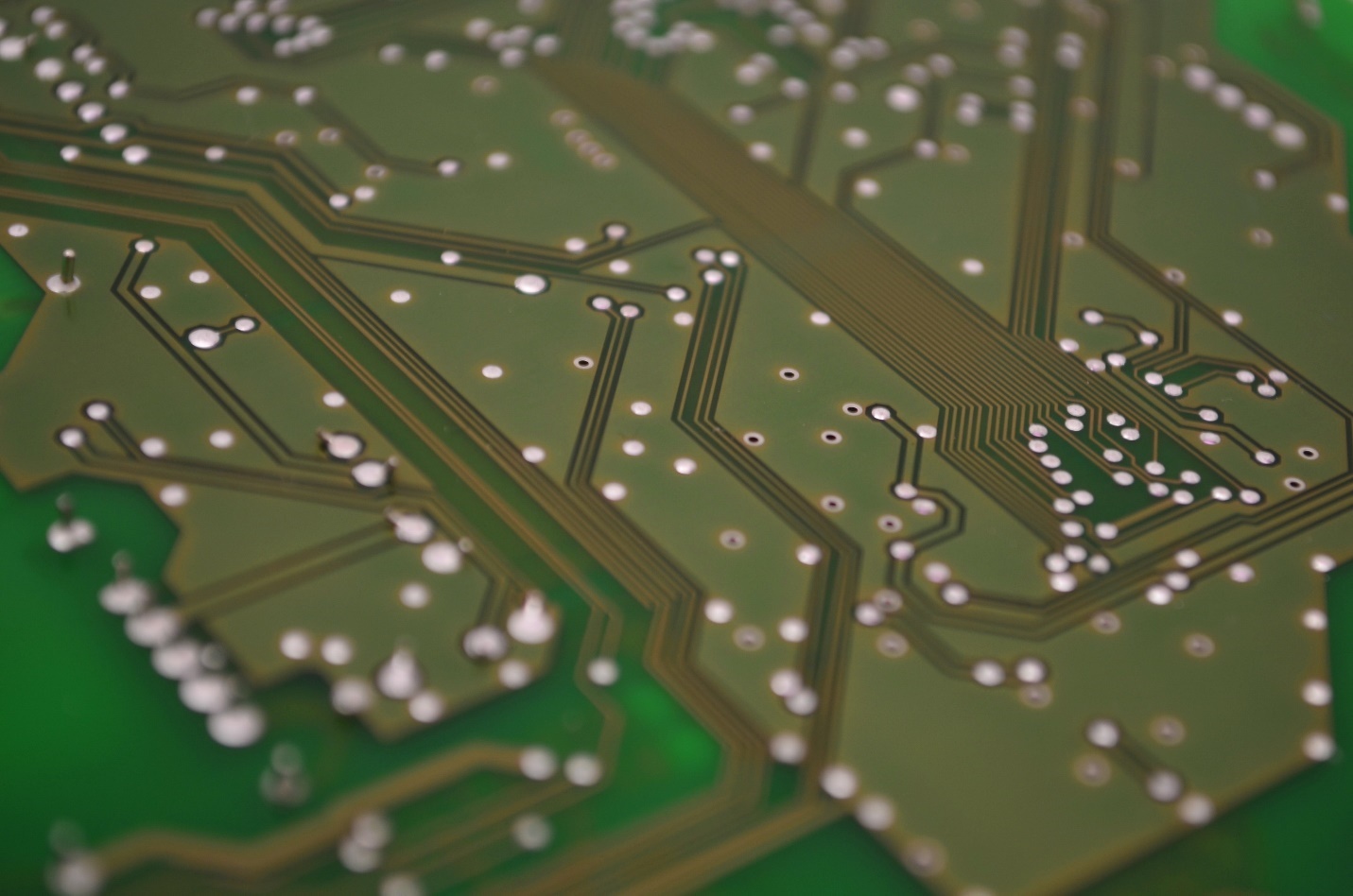
[](https://towardsdatascience.com/@dannymvarghese?source=post_page-----24f9ff6ab222----------------------)

[Danny Varghese](https://towardsdatascience.com/@dannymvarghese?source=post_page-----24f9ff6ab222----------------------)

Follow

[Dec 6, 2018](https://towardsdatascience.com/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222?source=post_page-----24f9ff6ab222----------------------) · 12 min read





Machine learning is a scientific technique where the computers learn how to solve a problem, without explicitly program them. Deep learning is currently leading the ML race powered by better algorithms, computation power and large data. Still ML classical algorithms have their strong position in the field.

I will be doing a comparative study over different machine learning supervised techniques like***Linear Regression, Logistic Regression, K nearest neighbors and Decision Trees***in this story***.***In the [next story](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-part-2-5ab58b683ec0), I’ll be covering ***Support Vector machine, Random Forest and Naive Baye*s**. There are so many better blogs about the in-depth details of algorithms, so we will only focus on their comparative study. We will look into their ***basic logic, advantages, disadvantages, assumptions, effects of co-linearity & outliers, hyper-parameters****,****mutual comparisons****etc*.

[**please refer Part-2 of this series for remaining algorithms.**](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-part-2-5ab58b683ec0)

**1. Linear Regression**

If you want to start machine learning, Linear regression is the best place to start. Linear Regression is a regression model, meaning, it’ll take features and predict a continuous output, eg : stock price,salary etc. Linear regression as the name says, finds a linear curve solution to every problem.

**Basic Theory :**

LR allocates weight parameter, theta for each of the training features. The predicted output(h(θ)) will be a linear function of features and θ coefficients.



linear regression output. Eqn (1)

During the start of training, each theta is randomly initialized. But during the training, we correct the theta corresponding to each feature such that, the loss (metric of the deviation between expected and predicted output) is minimized. Gradient descend algorithm will be used to align the θ values in the right direction. In the below diagram, each red dots represent the training data and the blue line shows the derived solution.



*gif credits :*<https://medium.com/@kabab/linear-regression-with-python-d4e10887ca43>

**Loss function :**

In LR, we use mean squared error as the metric of loss. The deviation of expected and actual outputs will be squared and sum up. Derivative of this loss will be used by gradient descend algorithm.

**Advantages :**

* Easy and simple implementation.
* Space complex solution.
* Fast training.
* Value of θ coefficients gives an assumption of feature significance.

**Disadvantages :**

* Applicable only if the solution is linear. In many real life scenarios, it may not be the case.
* Algorithm assumes the input residuals (error) to be normal distributed, but may not be satisfied always.
* Algorithm assumes input features to be mutually independent(no co-linearity).

**Hyperparameters :**

* Regularization parameter (λ) : Regularization is used to avoid over-fitting on the data. Higher the λ, higher will be regularization and the solution will be highly biased. Lower the λ, solution will be of high variance. An intermediate value is preferable.
* learning rate (α) : it estimates, by how much the θ values should be corrected while applying gradient descend algorithm during training. α should also be a moderate value.

**Assumptions for LR :**

* Linear relationship between the independent and dependent variables.
* Training data to be homoskedastic, meaning the variance of the errors should be somewhat constant.
* Independent variables should not be co-linear.

**Colinearity & Outliers :**

Two features are said to be colinear when one feature can be linearly predicted from the other with somewhat accuracy.

* colinearity will simply inflate the standard error and causes some significant features to become insignificant during training. Ideally, we should calculate the colinearity prior to training and keep only one feature from highly correlated feature sets.

Outlier is another challenge faced during training. They are data-points that are extreme to normal observations and affects the accuracy of the model.

* outliers inflates the error functions and affects the curve function and accuracy of the linear regression. Regularization (especially L1 ) can correct the outliers, by not allowing the θ parameters to change violently.
* During Exploratory data analysis phase itself, we should take care of outliers and correct/eliminate them. Box-plot can be used for identifying them.

**Comparison with other models:**

As the linear regression is a regression algorithm, we will compare it with other regression algorithms. One basic difference of linear regression is, LR can only support linear solutions. There are no best models in machine learning that outperforms all others(no free Lunch), and efficiency is based on the type of training data distribution.

**LR vs Decision Tree** :

* Decision trees supports non linearity, where LR supports only linear solutions.
* When there are large number of features with less data-sets(with low noise), linear regressions may outperform Decision trees/random forests. In general cases, Decision trees will be having better average accuracy.
* For categorical independent variables, decision trees are better than linear regression.
* Decision trees handles colinearity better than LR.

**LR vs SVM :**

* SVM supports both linear and non-linear solutions using kernel trick.
* SVM handles outliers better than LR.
* Both perform well when the training data is less, and there are large number of features.

**LR vs KNN :**

* KNN is a non -parametric model, whereas LR is a parametric model.
* KNN is slow in real time as it have to keep track of all training data and find the neighbor nodes, whereas LR can easily extract output from the tuned θ coefficients.

**LR vs Neural Networks :**

* Neural networks need large training data compared to LR model, whereas LR can work well even with less training data.
* NN will be slow compared to LR.
* Average accuracy will be always better with neural networks.

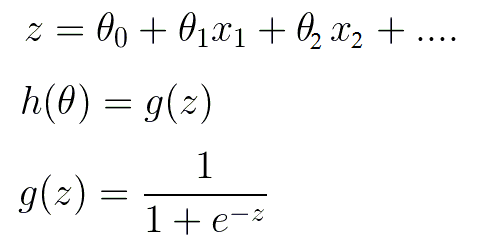
**2. Logistic Regression**

Just like linear regression, Logistic regression is the right algorithm to start with classification algorithms. Eventhough, the name ‘Regression’ comes up, it is not a regression model, but a classification model. It uses a logistic function to frame binary output model. The output of the logistic regression will be a probability (0≤x≤1), and can be used to predict the binary 0 or 1 as the output ( if x<0.5, output= 0, else output=1).

**Basic Theory :**

Logistic Regression acts somewhat very similar to linear regression. It also calculates the linear output, followed by a stashing function over the regression output. Sigmoid function is the frequently used logistic function. You can see below clearly, that the z value is same as that of the linear regression output in Eqn(1).





The h(θ) value here corresponds to P(y=1|x), ie, probability of output to be binary 1, given input x. P(y=0|x) will be equal to 1-h(θ).

when value of z is 0, g(z) will be 0.5. Whenever z is positive, h(θ) will be greater than 0.5 and output will be binary 1. Likewise, whenever z is negative, value of y will be 0. As we use a linear equation to find the classifier, the output model also will be a linear one, that means it splits the input dimension into two spaces with all points in one space corresponds to same label.

The figure below shows the distribution of a sigmoid function.





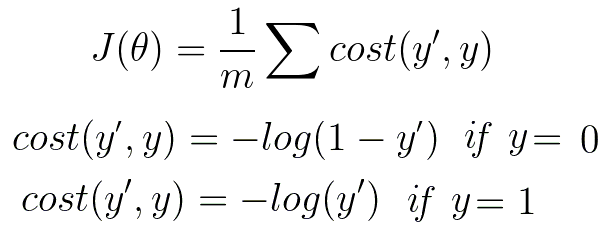
sigmoid function Eqn(3)

**Loss function :**

We can’t use mean squared error as loss function(like linear regression), because we use a non-linear sigmoid function at the end. MSE function may introduce local minimums and will affect the gradient descend algorithm.

So we use cross entropy as our loss function here. Two equations will be used, corresponding to y=1 and y=0. The basic logic here is that, whenever my prediction is badly wrong, (eg : y’ =1 & y = 0), cost will be -log(0) which is infinity.





cross-entropy loss Eqn(4)

In the equation given, m stands for training data size, y’ stands for predicted output and y stands for actual output.

**Advantages :**

* Easy, fast and simple classification method.
* θ parameters explains the direction and intensity of significance of independent variables over the dependent variable.
* Can be used for multiclass classifications also.
* Loss function is always convex.

**Disadvantages :**

* Cannot be applied on non-linear classification problems.
* Proper selection of features is required.
* Good signal to noise ratio is expected.
* Colinearity and outliers tampers the accuracy of LR model.

**Hyperparameters :**

Logistic regression hyperparameters are similar to that of linear regression. Learning rate(α) and Regularization parameter(λ) have to be tuned properly to achieve high accuracy.

**Assumptions of LR :**

Logistic regression assumptions are similar to that of linear regression model. please refer the above section.

**Comparison with other models :**

**Logistic regression vs SVM :**

* SVM can handle non-linear solutions whereas logistic regression can only handle linear solutions.
* Linear SVM handles outliers better, as it derives maximum margin solution.
* Hinge loss in SVM outperforms log loss in LR.

**Logistic Regression vs Decision Tree :**

* Decision tree handles colinearity better than LR.
* Decision trees cannot derive the significance of features, but LR can.
* Decision trees are better for categorical values than LR.

**Logistic Regression vs Neural network :**

* NN can support non-linear solutions where LR cannot.
* LR have convex loss function, so it wont hangs in a local minima, whereas NN may hang.
* LR outperforms NN when training data is less and features are large, whereas NN needs large training data.

**Logistic Regression vs Naive Bayes :**

* Naive bayes is a generative model whereas LR is a discriminative model.
* Naive bayes works well with small datasets, whereas LR+regularization can achieve similar performance.
* LR performs better than naive bayes upon colinearity, as naive bayes expects all features to be independent.

**Logistic Regression vs KNN :**

* KNN is a non-parametric model, where LR is a parametric model.
* KNN is comparatively slower than Logistic Regression.
* KNN supports non-linear solutions where LR supports only linear solutions.
* LR can derive confidence level (about its prediction), whereas KNN can only output the labels.

**3. K-nearest neighbors**

K-nearest neighbors is a non-parametric method used for classification and regression. It is one of the most easy ML technique used. It is a lazy learning model, with local approximation.

**Basic Theory :**

The basic logic behind KNN is to explore your neighborhood, assume the test datapoint to be similar to them and derive the output. In KNN, we look for k neighbors and come up with the prediction.

In case of KNN classification, a majority voting is applied over the k nearest datapoints whereas, in KNN regression, mean of k nearest datapoints is calculated as the output. As a rule of thumb, we selects odd numbers as k. KNN is a lazy learning model where the computations happens only runtime.



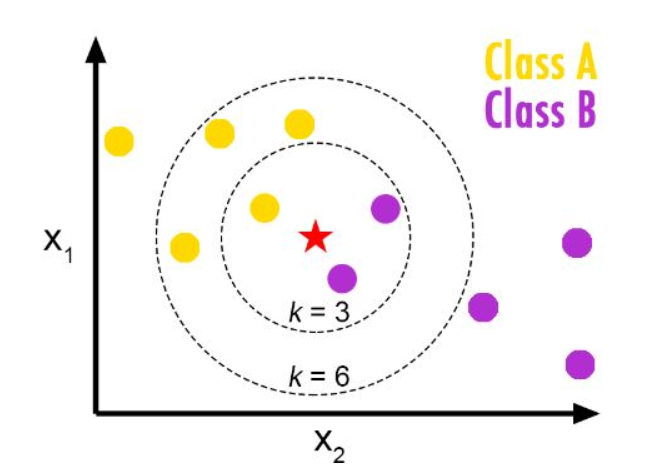


image credits : <https://www.fromthegenesis.com/pros-and-cons-of-k-nearest-neighbors/>

In the above diagram yellow and violet points corresponds to Class A and Class B in training data. The red star, points to the testdata which is to be classified. when k = 3, we predict Class B as the output and when K=6, we predict Class A as the output.

**Loss function :**

There is no training involved in KNN. During testing, k neighbors with minimum distance, will take part in classification /regression.

**Advantages :**

* Easy and simple machine learning model.
* Few hyperparameters to tune.

**Disadvantages :**

* k should be wisely selected.
* Large computation cost during runtime if sample size is large.
* Proper scaling should be provided for fair treatment among features.

**Hyperparameters :**

KNN mainly involves two hyperparameters, K value & distance function.

* K value : how many neighbors to participate in the KNN algorithm. k should be tuned based on the validation error.
* distance function : Euclidean distance is the most used similarity function. Manhattan distance, Hamming Distance, Minkowski distance are different alternatives.

**Assumptions :**

* There should be clear understanding about the input domain.
* feasibly moderate sample size (due to space and time constraints).
* colinearity and outliers should be treated prior to training.

**Comparison with other models :**

A general difference between KNN and other models is the large real time computation needed by KNN compared to others.

**KNN vs naive bayes :**

* Naive bayes is much faster than KNN due to KNN’s real-time execution.
* Naive bayes is parametric whereas KNN is non-parametric.

**KNN vs linear regression :**

* KNN is better than linear regression when the data have high SNR.

**KNN vs SVM :**

* SVM take cares of outliers better than KNN.
* If training data is much larger than no. of features(m>>n), KNN is better than SVM. SVM outperforms KNN when there are large features and lesser training data.

**KNN vs Neural networks :**

* Neural networks need large training data compared to KNN to achieve sufficient accuracy.
* NN needs lot of hyperparameter tuning compared to KNN.

**4. Decision Tree**

Decision tree is a tree based algorithm used to solve regression and classification problems. An inverted tree is framed which is branched off from a homogeneous probability distributed root node, to highly heterogeneous leaf nodes, for deriving the output. Regression trees are used for dependent variable with continuous values and classification trees are used for dependent variable with discrete values.

**Basic Theory :**

Decision tree is derived from the independent variables, with each node having a condition over a feature. The nodes decides which node to navigate next based on the condition. Once the leaf node is reached, an output is predicted. The right sequence of conditions makes the tree efficient. entropy/Information gain are used as the criteria to select the conditions in nodes. A recursive, greedy based algorithm is used to derive the tree structure.



credits : [https://brookewenig.github.io](https://brookewenig.github.io/)

In the above diagram, we can see a tree with set of internal nodes(conditions) and leaf nodes with labels( decline/accept offer).

**Algorithm to select conditions :**

* for CART(classification and regression trees), we use gini index as the classification metric. It is a metric to calculate how well the datapoints are mixed together.



the attribute with maximum gini index is selected as the next condition, at every phase of creating the decision tree. When set is unequally mixed, gini score will be maximum.

* For Iterative Dichotomiser 3 algorithm, we use entropy and information gain to select the next attribute. In the below equation, H(s) stands for entropy and IG(s) stands for Information gain. Information gain calculates the entropy difference of parent and child nodes. The attribute with maximum information gain is chosen as next internal node.



**Advantages :**

* No preprocessing needed on data.
* No assumptions on distribution of data.
* Handles colinearity efficiently.
* Decision trees can provide understandable explanation over the prediction.

**Disadvantages :**

* Chances for overfitting the model if we keep on building the tree to achieve high purity. decision tree pruning can be used to solve this issue.
* Prone to outliers.
* Tree may grow to be very complex while training complicated datasets.
* Looses valuable information while handling continuous variables.

**Hyperparameters :**

Decision tree includes many hyperparameters and I will list a few among them.

* **criterion**: which cost function for selecting the next tree node. Mostly used ones are gini/entropy.
* **max depth :**it is the maximum allowed depth of the decision tree.
* **minimum samples split :**It is the minimum nodes required to split an internal node.
* **minimum samples leaf :**minimum samples that are required to be at the leaf node.

**Comparison with other Models :**

**Decision tree vs Random Forest :**

* Random Forest is a collection of decision trees and average/majority vote of the forest is selected as the predicted output.
* Random Forest model will be less prone to overfitting than Decision tree, and gives a more generalized solution.
* Random Forest is more robust and accurate than decision trees.

**Decision tree vs KNN :**

* Both are non-parametric methods.
* Decision tree supports automatic feature interaction, whereas KNN cant.
* Decision tree is faster due to KNN’s expensive real time execution.

**Decision tree vs naive Bayes :**

* Decision tree is a discriminative model, whereas Naive bayes is a generative model.
* Decision trees are more flexible and easy.
* Decision tree pruning may neglect some key values in training data, which can lead the accuracy for a toss.

**Decision tree vs neural network :**

* Both finds non-linear solutions, and have interaction between independent variables.
* Decision trees are better when there is large set of categorical values in training data.
* Decision trees are better than NN, when the scenario demands an explanation over the decision.
* NN outperforms decision tree when there is sufficient training data.

**Decision tree vs SVM :**

* SVM uses kernel trick to solve non-linear problems whereas decision trees derive hyper-rectangles in input space to solve the problem.
* Decision trees are better for categorical data and it deals colinearity better than SVM.

In the next story I will be covering the remaining algorithms like, naive bayes, Random Forest and Support Vector Machine.If you have any suggestions or corrections, please give a comment.

Happy Learning :)

**[Towards Data Science](https://towardsdatascience.com/?source=post_sidebar--------------------------post_sidebar-)**

A Medium publication sharing concepts, ideas, and codes.

Follow

708

* [Machine Learning](https://towardsdatascience.com/tagged/machine-learning)
* [Artificial Intelligence](https://towardsdatascience.com/tagged/artificial-intelligence)
* [Algorithms](https://towardsdatascience.com/tagged/algorithms)

Get started

[Open in app](https://rsci.app.link/?%24canonical_url=https%3A%2F%2Fmedium.com%2Fp%2F5ab58b683ec0&~feature=LoOpenInAppButton&~channel=ShowPostUnderUser&~stage=mobileNavBar&source=post_page-----5ab58b683ec0--------------------------------)

[Danny Varghese](https://medium.com/@dannymvarghese?source=post_page-----5ab58b683ec0--------------------------------)

#### 90 Followers

#### ·

#### [About](https://medium.com/@dannymvarghese/about?source=post_page-----5ab58b683ec0--------------------------------)

#### [Follow](https://medium.com/m/signin?actionUrl=%2F_%2Fsubscribe%2Fuser%2F208860849314&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40dannymvarghese%2Fcomparative-study-on-classic-machine-learning-algorithms-part-2-5ab58b683ec0&source=post_page-208860849314----5ab58b683ec0---------------------follow_header-----------)

Get started

# Comparative Study on Classic Machine learning Algorithms , Part-2

[Danny Varghese](https://medium.com/@dannymvarghese?source=post_page-----5ab58b683ec0--------------------------------)

#### [Danny Varghese](https://medium.com/@dannymvarghese?source=post_page-----5ab58b683ec0--------------------------------)

#### [Dec 11, 2018·9 min read](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-part-2-5ab58b683ec0?source=post_page-----5ab58b683ec0--------------------------------)

Quick summary on ML algorithms

In the previous [story](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222), I have already covered Linear Regression, Logistic Regression, KNN and Decision trees. In this Story, I will explain Support Vector Machines, Random Forest and Naive Bayes algorithm. The comparisons, that are already covered in the previous story wont be repeated here. Lets start the the phase 2.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1588/1\*eNvgdjHuBBUZ5p31KmX17w.jpeg" width="794" height="355" srcSet="https://miro.medium.com/max/552/1\*eNvgdjHuBBUZ5p31KmX17w.jpeg 276w, https://miro.medium.com/max/1104/1\*eNvgdjHuBBUZ5p31KmX17w.jpeg 552w, https://miro.medium.com/max/1280/1\*eNvgdjHuBBUZ5p31KmX17w.jpeg 640w, https://miro.medium.com/max/1400/1\*eNvgdjHuBBUZ5p31KmX17w.jpeg 700w" sizes="700px"/>

# 5. Support Vector Machine

Support Vector machine is a type of ML technique that can be used for both classification and regression. It have majorly two variants to support linear and non linear problems. Linear SVM has no kernel and finds a minimum margin linear solution to the problem. SVM with kernels are used when the solution is not linearly separable.

## Basic Theory:

Support Vector Machine is a supervised learning technique extensively used in text classification, image classification, bioinformatics etc.

In **Linear SVM**, the problem space must be linearly separable. A hyperplane is derived by the model, that maximizes the classification margin. The hyperplane will be an N-1 dimensional subspace if there are N features present. The boundary nodes in the feature space are called support vectors. Based on the their relative position, the maximum margin is derived and an optimal hyperplane is drawn in the midpoint.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1378/1\*VijGPT9cLN-4oNPrBm0dMA.png" width="689" height="490" srcSet="https://miro.medium.com/max/552/1\*VijGPT9cLN-4oNPrBm0dMA.png 276w, https://miro.medium.com/max/1104/1\*VijGPT9cLN-4oNPrBm0dMA.png 552w, https://miro.medium.com/max/1280/1\*VijGPT9cLN-4oNPrBm0dMA.png 640w, https://miro.medium.com/max/1378/1\*VijGPT9cLN-4oNPrBm0dMA.png 689w" sizes="689px"/>

credits : <https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/>

Value of margin(m) will be inversely proportional to ||w||, where w is the set of weight matrices. For maximizing margin, we’ll have to minimize||w||. The optimization problem will be,

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1460/1\*b4BqFvxEaGbcsIOzFghv7w.png" width="730" height="104" srcSet="https://miro.medium.com/max/552/1\*b4BqFvxEaGbcsIOzFghv7w.png 276w, https://miro.medium.com/max/1104/1\*b4BqFvxEaGbcsIOzFghv7w.png 552w, https://miro.medium.com/max/1280/1\*b4BqFvxEaGbcsIOzFghv7w.png 640w, https://miro.medium.com/max/1400/1\*b4BqFvxEaGbcsIOzFghv7w.png 700w" sizes="700px"/>

The above optimization works well for fully linearly separable solutions. For handling outliers, we need a slack term as below. The second term uses the hinge loss to get slack variable.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1120/1\*kxnqsdnzia6wkNPB5v3RTA.png" width="560" height="70" srcSet="https://miro.medium.com/max/552/1\*kxnqsdnzia6wkNPB5v3RTA.png 276w, https://miro.medium.com/max/1104/1\*kxnqsdnzia6wkNPB5v3RTA.png 552w, https://miro.medium.com/max/1120/1\*kxnqsdnzia6wkNPB5v3RTA.png 560w" sizes="560px"/>

C is the regularization parameter that balances the miss penalty and margin width. As the mathematical explanations are well above the scope of this story, I will not be explaining them in depth.

The basic logic is that, to minimize the cost function, w is forced to tune with maximum margin between the classes. C value will decide the level of regularization applied over the datasets. It decides the level (soft/hard) margin to be applied over the datasets. In short, C is the level of ignorance over outliers.

N**on-linear SVM** when the dataset is not linearly separable. A kernel function is used to derive a new hyperplane for all the training data. The distribution of labels in new hyperplane will be such that training data will be linearly separable. Later, a linear curve will classify the labels in the hyperplane. When the classification results are projected back to the feature space, we get a non linear solution.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1968/1\*vvCyZwnM9-Ex9L-RAr4kYQ.png" width="984" height="389" srcSet="https://miro.medium.com/max/552/1\*vvCyZwnM9-Ex9L-RAr4kYQ.png 276w, https://miro.medium.com/max/1104/1\*vvCyZwnM9-Ex9L-RAr4kYQ.png 552w, https://miro.medium.com/max/1280/1\*vvCyZwnM9-Ex9L-RAr4kYQ.png 640w, https://miro.medium.com/max/1400/1\*vvCyZwnM9-Ex9L-RAr4kYQ.png 700w" sizes="700px"/>

credits : <https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/>

The only change in the equation here is, the introduction of a new kernel function. The new equation will look like,

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/2208/1\*acjAZp5RJcTFMGIrpyQdIA.png" width="1104" height="94" srcSet="https://miro.medium.com/max/552/1\*acjAZp5RJcTFMGIrpyQdIA.png 276w, https://miro.medium.com/max/1104/1\*acjAZp5RJcTFMGIrpyQdIA.png 552w, https://miro.medium.com/max/1280/1\*acjAZp5RJcTFMGIrpyQdIA.png 640w, https://miro.medium.com/max/1400/1\*acjAZp5RJcTFMGIrpyQdIA.png 700w" sizes="700px"/>

The xi will be replaced by ϕ(xi) which will transform the dataset into the new hyperplane.

## loss function:

The loss function in Eqn.1 can be divided into two parts as below :

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1072/1\*lohupo5gKMEo3ASvvXkErg.png" width="536" height="70" srcSet="https://miro.medium.com/max/552/1\*lohupo5gKMEo3ASvvXkErg.png 276w, https://miro.medium.com/max/1072/1\*lohupo5gKMEo3ASvvXkErg.png 536w" sizes="536px"/>

First term tries to minimize w parameters, and achieve high margin. Second term corresponds to **hinge loss.** It calculates the slack variable for each dataset. If any dataset comes in between the margin or in the wrong side, it will be given a penalty by the hinge loss.

The minimization of first term causes decrease in w and widening of the margin. The minimization of second term forces shortening of margin to reduce the hinge loss. Based on the value of C, we finally settles to a stable margin. Value of C decides a soft/hard margin in the curve.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1588/1\*gg821oUy3KsFsA9xPI6udA.png" width="794" height="444" srcSet="https://miro.medium.com/max/552/1\*gg821oUy3KsFsA9xPI6udA.png 276w, https://miro.medium.com/max/1104/1\*gg821oUy3KsFsA9xPI6udA.png 552w, https://miro.medium.com/max/1280/1\*gg821oUy3KsFsA9xPI6udA.png 640w, https://miro.medium.com/max/1400/1\*gg821oUy3KsFsA9xPI6udA.png 700w" sizes="700px"/>

credits : <https://datascience.stackexchange.com/questions/25977>

In the above diagram, it is evident about the effects of C in deriving the margin.

In non-linear kernels, we can use Gaussian kernel, polynomial kernel, Sigmoid kernel, Laplace RBF kernel etc.

## Advantages:

* SVM uses kernel trick to solve complex solutions.
* SVM uses a convex optimization function, due to which global minima is always achievable.
* Hinge loss provides higher accuracy.
* Outliers can be well handled using soft margin constant C.

## Disadvantages:

* Hinge loss leads to sparsity.
* Hyper parameters and kernels are to be carefully tuned for sufficient accuracy.
* Longer training time for larger datasets.

## Hyperparameters:

* Soft Margin Constant ( C ) : It is a hyperparameter that decides the level of penalty over the outliers. It is directly inverse to regularization parameter. When C is large, Outliers will be given high penalty and the a hard margin is formed. When C is small, the outliers are ignored and the margin will be wide.
* Degree of polynomial in Polynomial Kernel (d) : when d = 1, it is equivalent to a linear kernel. When d is higher, kernel is flexible enough to distinguish complex patterns by projecting them to a new hyperplane.
* Width Parameter in Gaussian Kernel (γ) : Gamma decides the width of the Gaussian curve. With increase in gamma, width also increases.

## Comparison with other Models :

**SVM vs Random Forest :**

* Random Forest supports multiclass classification,whereas SVM needs multiple models for the same.
* Random Forest can give a probability over the prediction, whereas SVM cannot give.
* Random Forest deals categorical data better than SVM.

**SVM vs Naive Bayes :**

* Both performs better with low amount of training data and large features.
* If features are mutually dependent, SVM outperforms Naive Bayes.
* SVM is a discriminative model whereas NB is generative model.

**SVM vs Neural Networks :**

* SVM have a convex optimization function , whereas NN may hung in local minima.
* SVM can perform better than NN when there are limited training data and many features. NN needs large training data for sufficient accuracy.
* Multi class classification requires multiple models for SVM, whereas NN can do it with a single model.

# 6. Random Forest

Random Forest is an ensemble model where, multiple decision trees are combined to get a stronger model. The derived model will be more robust, accurate and handles overfitting better than constituent models.

## Basic Theory :

Random Forest have a set of decision trees ensembled with “bagging method” to obtain classification and regression outputs. In classification, it calculates the output using majority voting , whereas in regression, mean is calculated.

please refer my previous [story](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222) to find more about decision trees.

Random Forest comes up with a robust, accurate model that can handle large varieties of input data with binary, categorical, continuous features.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1316/1\*D4ltClwXiyuuUJeHxwYpnQ.png" width="658" height="449" srcSet="https://miro.medium.com/max/552/1\*D4ltClwXiyuuUJeHxwYpnQ.png 276w, https://miro.medium.com/max/1104/1\*D4ltClwXiyuuUJeHxwYpnQ.png 552w, https://miro.medium.com/max/1280/1\*D4ltClwXiyuuUJeHxwYpnQ.png 640w, https://miro.medium.com/max/1316/1\*D4ltClwXiyuuUJeHxwYpnQ.png 658w" sizes="658px"/>

credits : <https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d>

## Loss function :

We use entropy/Gini score to calculate the loss value of the datasets. In the previous [story](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222), i have explained it in the decision tree section.

## Advantages :

* Accurate and powerful model.
* handles overfitting efficiently.
* Supports implicit feature selection and derives feature importance.

## Disadvantages :

* computationally complex and slower when forest becomes large.
* Not a well descriptive model over the prediction.

## Hyperparameters :

**n\_estimators :** It is the number of trees in the forest. With large number of trees comes high accuracy, but high computational complexity.

**maximum features :** maximum number of features allowed in an individual tree.

**minimum sample leaf** : It is the minimum number of samples required to split an internal node.

## Comparison with other models :

Random Forest comparison is pretty much similar to that of Decision tree comparisons. Please refer the [part-1](https://medium.com/@dannymvarghese/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222) of this series for more information.

**Random Forest vs Naive Bayes :**

* Random Forest is a complex and large model whereas Naive Bayes is a relatively smaller model.
* Naive Bayes performs better with small training data, whereas RF needs larger set of training data.

**Random Forest vs Neural Networks :**

* Both are very powerful and high accuracy algorithms.
* Both have feature interactions internally and are less explainable.
* Random Forest needs no feature scaling whereas NN needs features to be scaled.
* An ensemble version of both models will be powerful.

# 7. Naive Bayes

Naive bayes is a generative probability model used for classification problems. It is the prime model used for text classifications, where featureset is very large. It is extensively used for sentiment analysis, spam filtering etc.

## Basic Theory:

Naive bayes model is based on Thomas Baye’s bayes rule. The bayes rule can be stated as ,

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/602/1\*d73o9LgU0lL6RDnSI4HxsQ.png" width="301" height="91" srcSet="https://miro.medium.com/max/552/1\*d73o9LgU0lL6RDnSI4HxsQ.png 276w, https://miro.medium.com/max/602/1\*d73o9LgU0lL6RDnSI4HxsQ.png 301w" sizes="301px"/>

In the above equation,

P(A|B) : (posterior probability) probability of event A to happen when event B is true.

P(A) ,P(B) : probability of event A and event B to happen.

P(B|A) : (likelihood) probability of event B to happen when event A is true.

The basic logic is to derive the probability of output label(Y) given the input X , from individual probabilities of features(Xi) given output label as Y, from the training data.

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1470/1\*Ftfa9XFDqaUBRwHthuctZw.png" width="735" height="81" srcSet="https://miro.medium.com/max/552/1\*Ftfa9XFDqaUBRwHthuctZw.png 276w, https://miro.medium.com/max/1104/1\*Ftfa9XFDqaUBRwHthuctZw.png 552w, https://miro.medium.com/max/1280/1\*Ftfa9XFDqaUBRwHthuctZw.png 640w, https://miro.medium.com/max/1400/1\*Ftfa9XFDqaUBRwHthuctZw.png 700w" sizes="700px"/>

Image for post

<img alt="Image for post" class="du hs dv ht t" src="https://miro.medium.com/max/1652/1\*moAH4YFlUdVGV97xID0vCA.png" width="826" height="104" srcSet="https://miro.medium.com/max/552/1\*moAH4YFlUdVGV97xID0vCA.png 276w, https://miro.medium.com/max/1104/1\*moAH4YFlUdVGV97xID0vCA.png 552w, https://miro.medium.com/max/1280/1\*moAH4YFlUdVGV97xID0vCA.png 640w, https://miro.medium.com/max/1400/1\*moAH4YFlUdVGV97xID0vCA.png 700w" sizes="700px"/>

Please notice in the above equation that naive bayes assumes all the features to be independent. The word “naive” itself is used to remind this. In case of multiple class labels, P(Ci|X) is calculated for each labels and label with maximum probability is chosen as the output.

There are few alternatives of naive bayes theorem as listed below :

* Gaussian : Gaussian naive bayes assumes features to be following Gaussian distribution.
* Multinomial : Multinomial naive bayes distribution is used when the data is assumed to be in multinomial distribution.

## Assumption :

The major assumption of naive bayes is that all features tend to be mutually independent. But in real scenarios, this may not be true.

ie , P(X1|X2, X3, X4, …,Xn, C) = P(X1|C)

In the above equation, X1 feature is assumed to be independent of other features, so other features can be excluded. Xi is the ith feature and C is the class label.

## Advantages :

* works well with less training data.
* If NB conditional independence is satisfied, it converges faster than other discriminative models.
* Handles irrelevant features.
* Supports binary and multi-class classification problems.

## Disadvantages :

* expects the features to be strictly independent to each other, which is not applicable in real life scenarios.
* While training sample of a large population, and if we have a feature with P(X=feature|Y) as zero, the posterior probability will become zero. This happens when the sample is not representing the population properly.
* continuous variables are binned to extract discrete values from features. This task should be carefully done to avoid data loss.

## When to Use :

* Naive bayes is preferred when the features are mutually independent and have limited training data.
* Naive bayes is highly used in text classification, spam filtering, recommender systems etc.

## ****Conclusion :****

In the two stories, we learned about various classical Machine learning Algorithms and their general properties. None of the algorithms works best in all cases(No Free Lunch), that means, distribution of training data is the key criteria for selecting a suitable algorithm. Still, Some general assumptions can be made on selection of algorithms, based on training size, type of features, number of features, computational and space complexity etc. Once we try various ML models with different hyperparameters on the data, we will get to know a clear view of the Algorithms. I hope this story helped you to have an idea over various ML algorithms and their comparisons.

Please comment your corrections, concerns and opinions below.

Thank You all :)

Written by

## [Danny Varghese](https://medium.com/@dannymvarghese?source=post_sidebar--------------------------post_sidebar-----------)