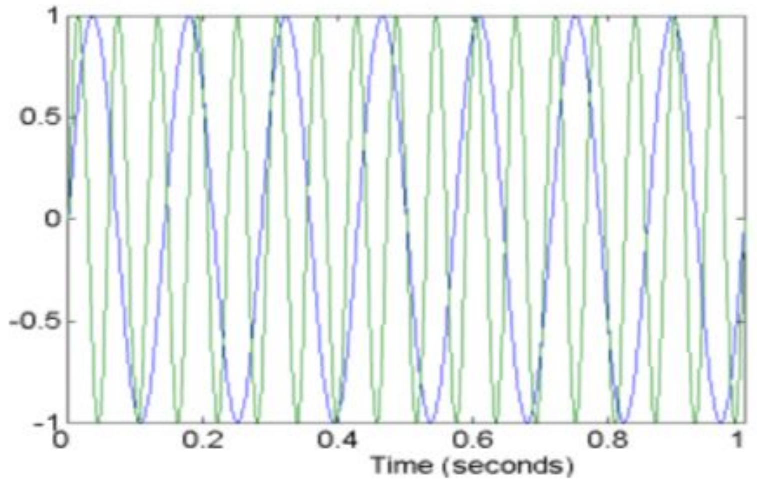
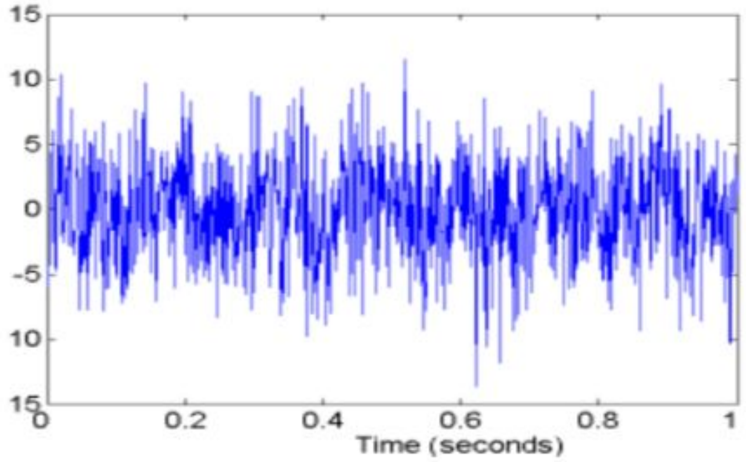
ML Revision Lecture 1

Noise

Noise refers to the modification/distortion of original values.



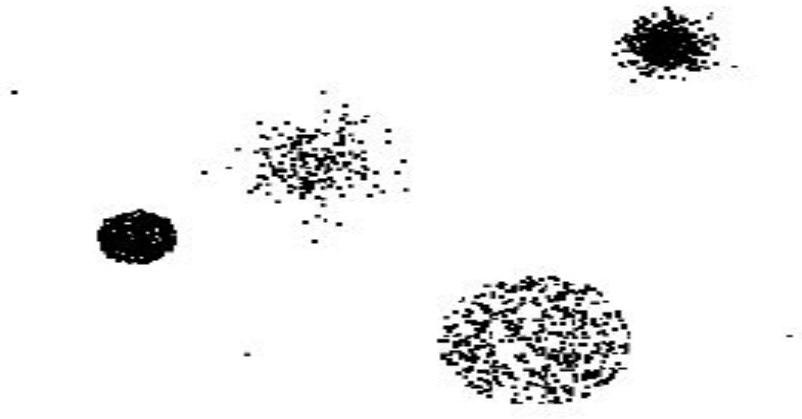
Two Sine Waves



Two Sine Waves + Noise

Outliers

Outliers are data objects with characteristics that are considerably different than most of the other data objects in the data set



Noise vs Outliers

(a) Can noise objects be outliers?

Noise vs Outliers

(a) Can noise objects be outliers?

Sol: Yes. Random distortion of the data is often responsible for outliers.

Noise vs Outliers

(b) Are noise objects always outliers?

Noise vs Outliers

(b) Are noise objects always outliers?

Sol: No. Random distortion can result in an object or value much like a normal one.

Noise vs Outliers

(c) Are outliers always noise objects?

Noise vs Outliers

(c) Are outliers always noise objects?

Sol: No. Often outliers merely represent a class of objects that are different from normal objects.

Linear Regression

Evaluation Metrics

1. R squared or Coefficient of Determination
2. Adjusted R squared
3. Mean Squared Error (MSE)
4. Root Mean Squared Error (RMSE)

R squared or Coefficient of Determination

The most commonly used metric for model evaluation in regression analysis is R squared.

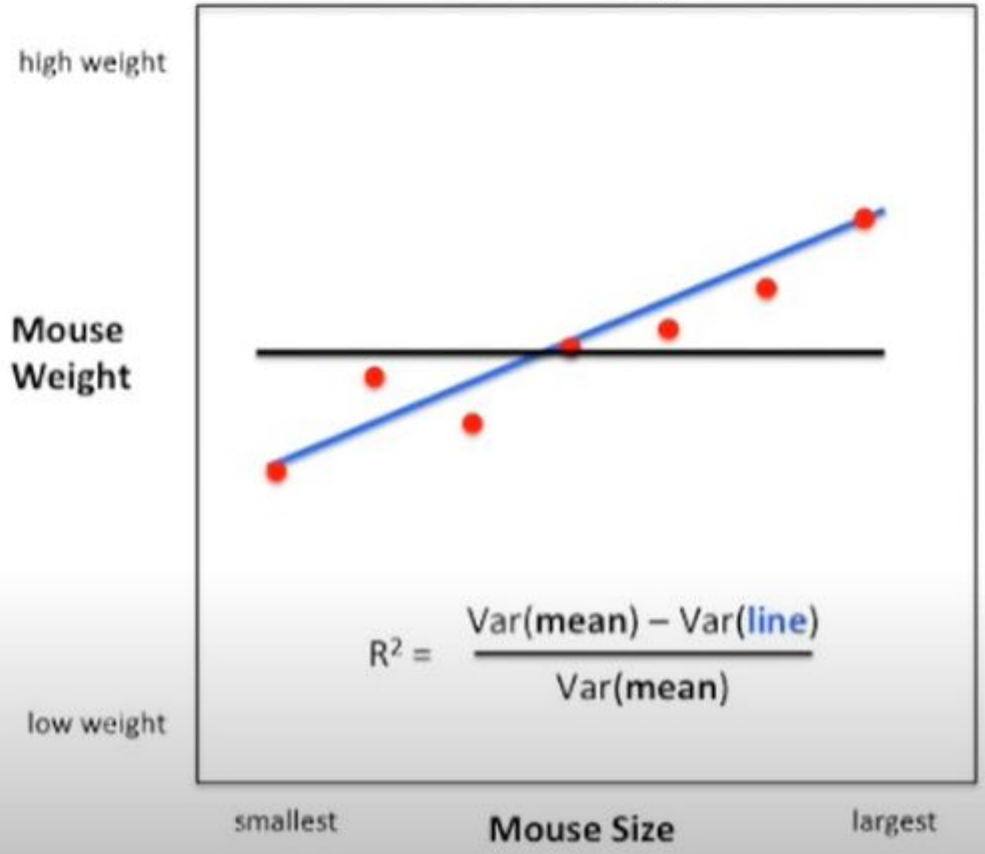
The value of R squared lies between 0 to 1, the value closer to 1 the better the model.

R^2 is goodness of fit measure which tells how well the model fits the data.

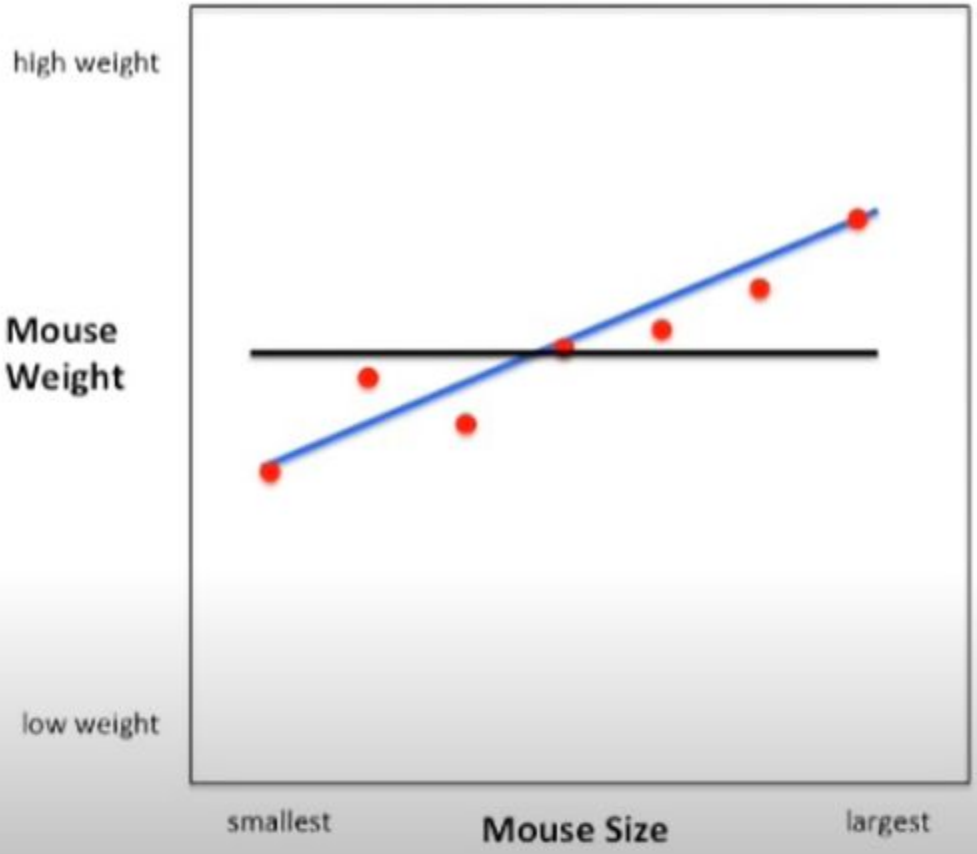
SS_{RES} is the Residual Sum of squares and SS_{TOT} is the Total Sum of squares

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

Quantifying the difference between the line and the mean. i.e. Calculating R^2



$$R^2 = \frac{\text{Var}(\textbf{mean}) - \text{Var}(\textbf{line})}{\text{Var}(\textbf{mean})}$$



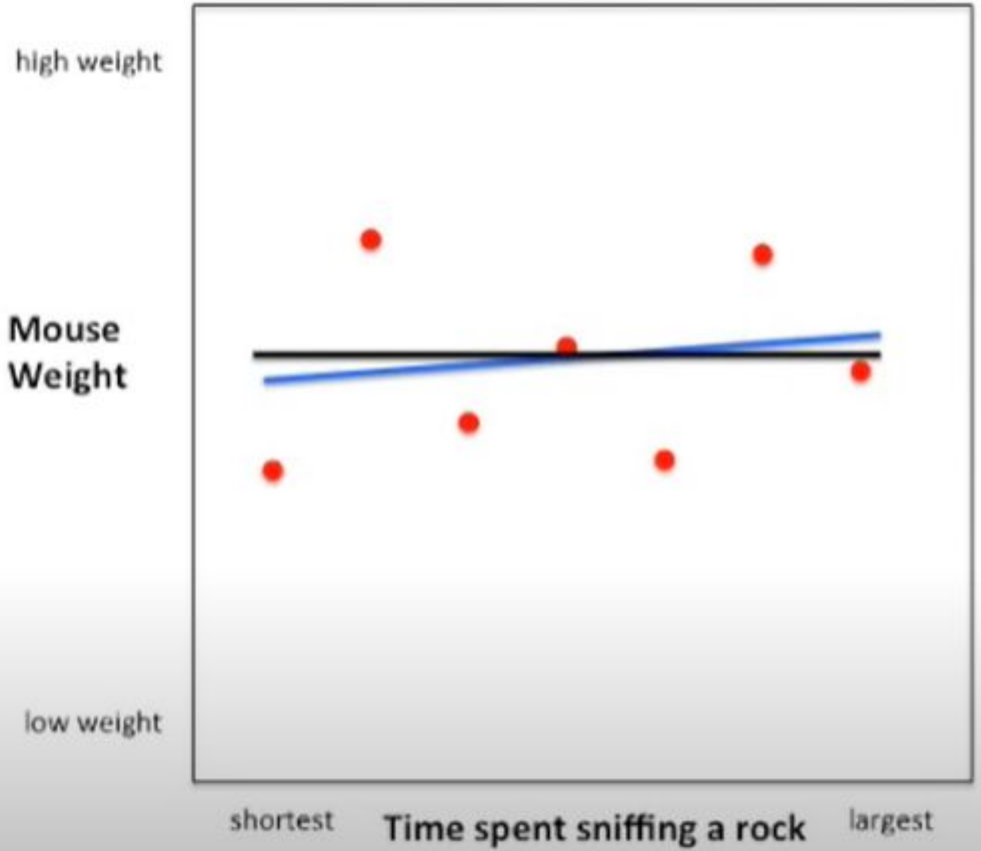
Var(**mean**) = 32

Var(**line**) = 6

$$R^2 = \frac{\text{Var}(\textbf{mean}) - \text{Var}(\textbf{line})}{\text{Var}(\textbf{mean})}$$

$$R^2 = \frac{32 - 6}{32}$$

$$R^2 = \frac{26}{32} = 0.81 = 81\%$$



Var(**mean**) = 32

Var(**line**) = 30

$$R^2 = \frac{\text{Var}(\textbf{mean}) - \text{Var}(\textbf{line})}{\text{Var}(\textbf{mean})}$$

$$R^2 = \frac{32 - 30}{32}$$

$$R^2 = \frac{2}{32} = 0.06 = 6\%$$

Adjusted R square

It is the improvement to R squared.

The problem/drawback with R2 is that as the features increase, the value of R2 also increases which gives the illusion of a good model.

So the Adjusted R2 solves the drawback of R2.

Adjusted R2 is always lower than R2.

Adjusted R Square

$$Adjusted\ R^2 = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

Where

R^2	Sample R-Squared
N	Total Sample Size
p	Number of independent variable

Mean Square Error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE	= mean squared error
n	= number of data points
Y_i	= observed values
$\hat{Y}_i$	= predicted values

Root Mean Square Error

It is square root of MSE.

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

Question

Which one is the disadvantage of Linear Regression?

(A) The assumption of linearity between the dependent variable and the independent variables. In the real world, the data is not always linearly separable.

(B) Linear regression is very sensitive to outliers

(C) Before applying Linear regression, multicollinearity should be removed because it assumes that there is no relationship among independent variables.

(D) All of the above

Question

Which one is the disadvantage of Linear Regression?

(A) The assumption of linearity between the dependent variable and the independent variables. In the real world, the data is not always linearly separable.

(B) Linear regression is very sensitive to outliers

(C) Before applying Linear regression, multicollinearity should be removed because it assumes that there is no relationship among independent variables.

(D) All of the above

Question

The value of R-Squared __________ with addition of every new independent variable?

A. May increase or decrease
B. Always increases
C. Always decreases

Question

The value of R-Squared __________ with addition of every new independent variable?

A. May increase or decrease
B. Always increases
C. Always decreases

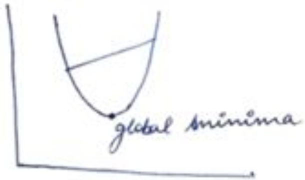
$\Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i))^2$ = error function

By diff w_0 & w_1, we get diff error functions
we want to minimize the error

$\Rightarrow \min \Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i))^2$

When the function is convex, it will have a global minima & it is solvable to find point of min error

Convex function : connect any two points & if straight line is above the curve then it convex function



$E = \Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i))^2$ is a convex function

so $\frac{\partial E}{\partial w_0} = 0$ and $\frac{\partial E}{\partial w_1} = 0$

$$\frac{\partial E}{\partial w_0} = \frac{\partial}{\partial w_0} \left(\Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i))\right)^2$$

$$\frac{\partial E}{\partial w_0} = -\frac{2}{n} \Sigma (y_i - (w_0 + w_1 x_i)) = 0$$

Convex function : connect any two points & if straight line is above the curve then it convex function

$E = \Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i))^2$ is a convex function

so $\frac{\partial E}{\partial w_0} = 0$ and $\frac{\partial E}{\partial w_1} = 0$

$$\frac{\partial E}{\partial w_0} = \frac{\partial}{\partial w_0} \left(\Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i))\right)^2$$

$$\frac{\partial E}{\partial w_0} = -\frac{2}{n} \Sigma (y_i - (w_0 + w_1 x_i)) = 0$$

$$\Sigma (y_i - (w_0 + w_1 x_i)) = 0$$

$$\Sigma y_i - n w_0 - w_1 \Sigma x_i = 0$$

$$\Sigma y_i = n w_0 + w_1 \Sigma x_i \qquad \textcircled{1}$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial}{\partial w_1} \left(\Sigma \frac{1}{n} (y_i - (w_0 + w_1 x_i)\right)^2$$

$$\frac{\partial E}{\partial w_1} = \frac{2}{n} \Sigma (y_i - (w_0 + w_1 x_i) \cdot (-x_i) = 0$$

$$\Sigma (y_i - (w_0 + w_1 x_i)) . x_i = 0$$

$$\Sigma y_i x_i - w_0 \Sigma x_i - w_1 \Sigma x_i^2 = 0$$

$$\Sigma y_i x_i = w_0 \Sigma x_i + w_1 \Sigma x_i^2 \qquad \textcircled{2}$$

$$\begin{bmatrix} n & \Sigma x_i \\ \Sigma x_i & \Sigma x_i^2 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} \Sigma y_i \\ \Sigma y_i x_i \end{bmatrix} \qquad \textcircled{3}$$

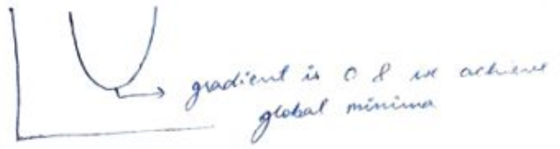
Solve $\textcircled{3}$ to get w_0 & w_1

$$AW = B$$

$$W = A^{-1} B$$

Here it is guaranteed to get a solution but the solution has higher time complexity

Gradient Descent

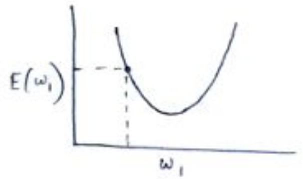


$$E(w_0, w_1) = \frac{1}{2n} \sum \left((w_0 + w_1 x_i) - y_i \right)^2$$

Assume intercept is 0 i.e. $w_0 = 0$

$$E(w_1) = \frac{1}{n} \sum (w_1 x_i - y_i)^2$$

Take a random w_1 and find $E(w_1)$



Now I have to decrease the gradient so I change w_1 to move in the direction to decrease gradient

$$w_1' = w_1 - \eta \left.\frac{\partial E}{\partial w}\right|_{w = w_1}$$

Now we need to prove $E(w_1) > E(w_1')$

$$E(w_1) = \frac{1}{n} \sum (w_1 x_i - y_i)^2$$

$$E(w_1') = E\left(w_1 - \eta \left.\frac{\partial E}{\partial w}\right|_{w = w_1}\right)$$

Taylor's theorem

$$f(x_0+h) = f(x_0) + h f'(x)\big|_{x=x_0} + \frac{h^2}{2!} f''(x)\big|_{x=x_0} + \cdots$$

$$E(w_1') = E\left(w_1 - \eta \frac{\partial E}{\partial w_1}\right)$$

$$= E(w_1) - \eta \frac{\partial E}{\partial w_1} \frac{\partial E}{\partial w_1}$$

$$= E(w_1) - \eta \left(\frac{\partial E}{\partial w_1}\right)^2$$

$\left(\frac{\partial E}{\partial w_1}\right)^2$ is a +ve quantity

so $E(w_1') < E(w_1)$

Similary with each iteration the error goes down.

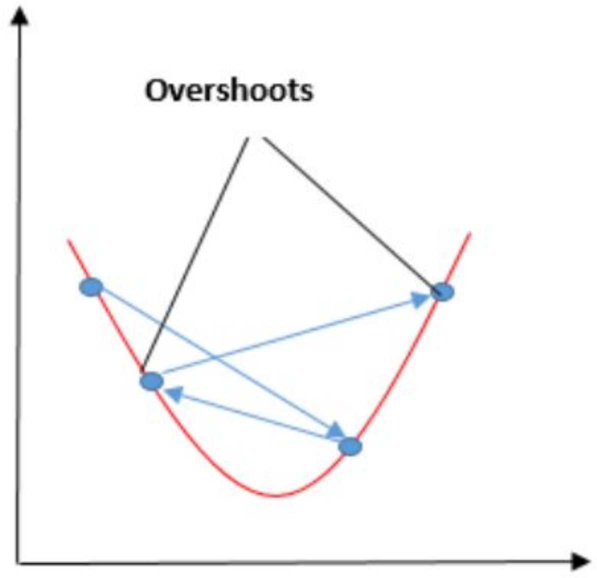
→ Difference between error values for 2 consecutive iterations keep on decreasing.

→ We don't get the exact answer always, we get the approximate answer

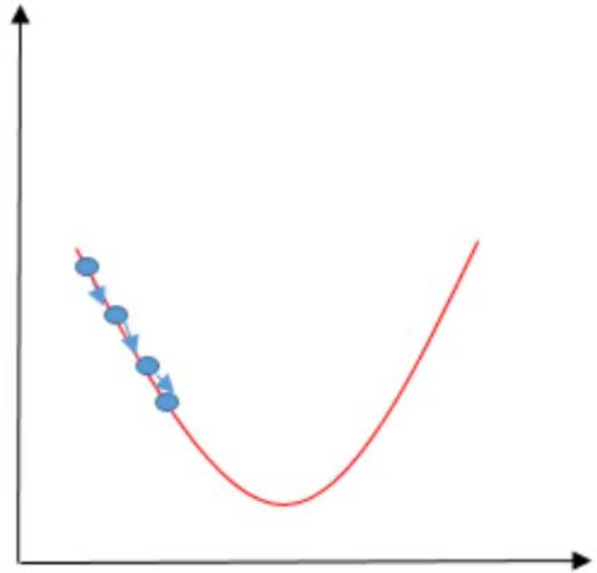
→ If η is very small, the more time consumed to reach global minima

→ If η is too large, then instead of convergence we might get divergence

Learning Rate



High learning rate



Low learning rate

Now if w_0 is also there

$$\nabla E = \begin{bmatrix} \frac{\partial E}{\partial w_0} \\ \frac{\partial E}{\partial w_1} \end{bmatrix}$$

$$\begin{bmatrix} w_0' \\ w_1' \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial E}{\partial w} \Big| w = w_0 \\ \frac{\partial E}{\partial w_1} \Big| w = w_1 \end{bmatrix}$$

$$\begin{bmatrix} w_0' \\ w_1' \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial}{\partial w_0} \left(\frac{1}{n} \Sigma((w_0 + w_1 x_i) - y_i)^2 \right. \\ \frac{\partial}{\partial w_1} \frac{1}{n} \Sigma((w_0 + w_1 x_i) - y_i)^2 \end{bmatrix}$$

$$\begin{bmatrix} w_0' \\ w_1' \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} - \eta \begin{bmatrix} \frac{2}{n} \Sigma((w_0 + w_1 x_i) - y_i) \\ \frac{2}{n} \underset{\downarrow}{\Sigma} (w_0 + w_1 x_i - y_i) x_i \end{bmatrix}$$

O(N) computation

For each iteration we need to calculate the sum in gradient descent

So to reduce complexity, instead of finding of gradient including all points, find the gradient for only one point

SGD randomly picks one data point from the whole dataset at each iteration to reduce computations enormously.

GD vs SGD

SGD often converges much faster compared to GD

but

the error function is not as well minimized as in the case of GD

Logistic Regression

Logistic Regression

Logistic Regression is one of the basic and popular algorithms to solve a classification problem.

Logistic Regression is a “Supervised machine learning” algorithm that can be used to model the probability of a certain class or event.

It is named 'Logistic Regression' because its underlying technique is quite the same as Linear Regression

Logistic Regression

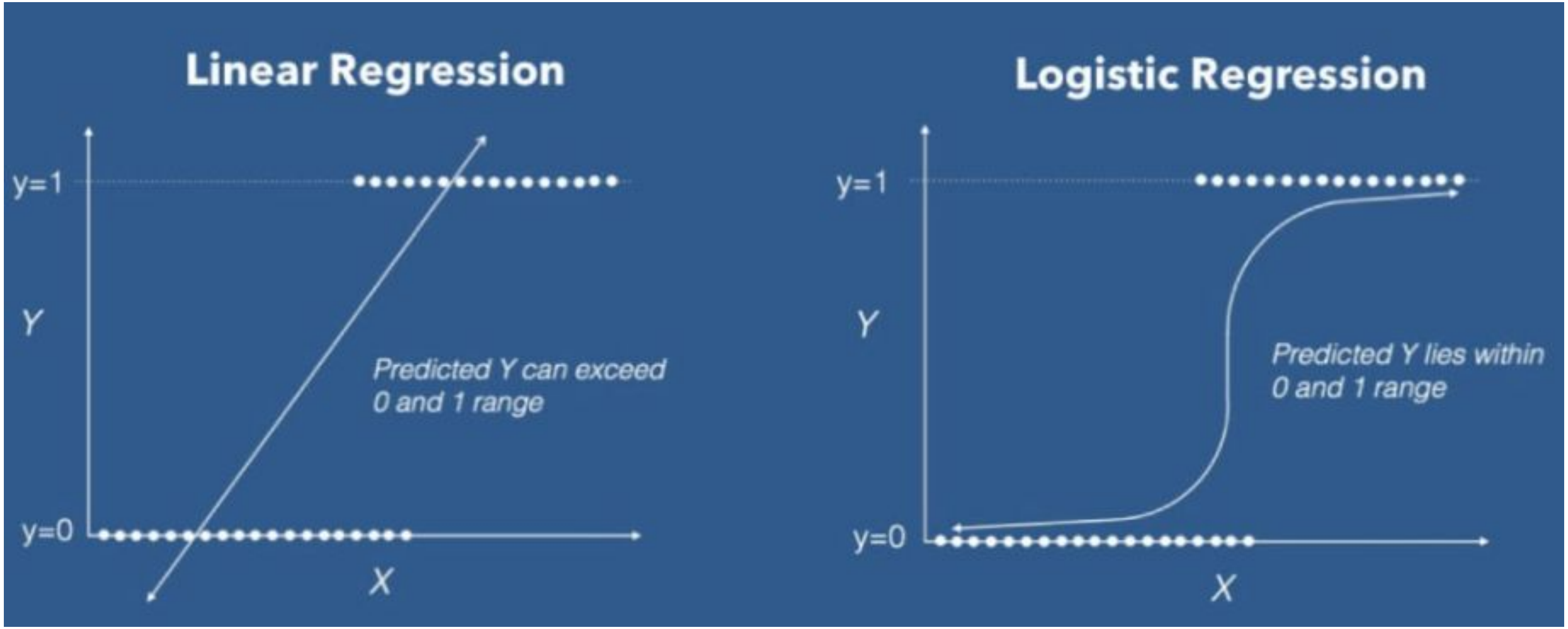
We started with a linear equation and ended up with a logistic regression model with the help of a sigmoid function.

Linear model: $\hat{y} = b_0 + b_1 x$

Sigmoid function: $\sigma(z) = 1/(1+e^{-z})$

Logistic regression model: $\hat{y} = \sigma(b_0 + b_1 x) = 1/(1+e^{-(b_0+b_1 x)})$

Logistic Regression


Linear Regression
y=1
Y
y=0
X
Predicted Y can exceed 0 and 1 range
Logistic Regression
y=1
Y
y=0
X
Predicted Y lies within 0 and 1 range

Logistic Regression

To evaluate the performance of the model, we calculate the loss. The most commonly used loss function is the mean squared error.

But in logistic regression, as the output is a probability value between 0 or 1, mean squared error wouldn’t be the right choice. So, instead, we use the cross-entropy loss function.

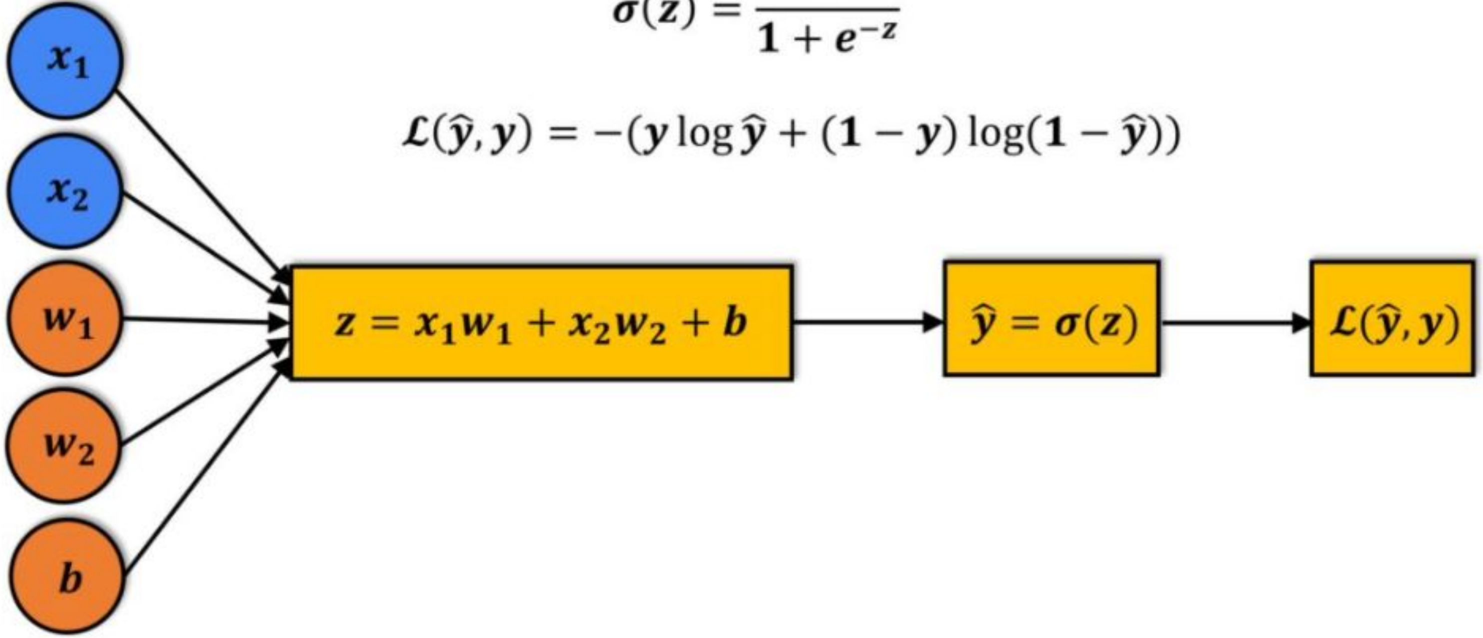
Cross-Entropy Loss Function is also called logarithmic loss, log loss or logistic loss.

Here also to reduce the loss function we can use gradient descent.

Logistic Regression

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

$$\mathcal{L}(\hat{y}, y) = -(y \log \hat{y} + (1-y)\log(1-\hat{y}))$$



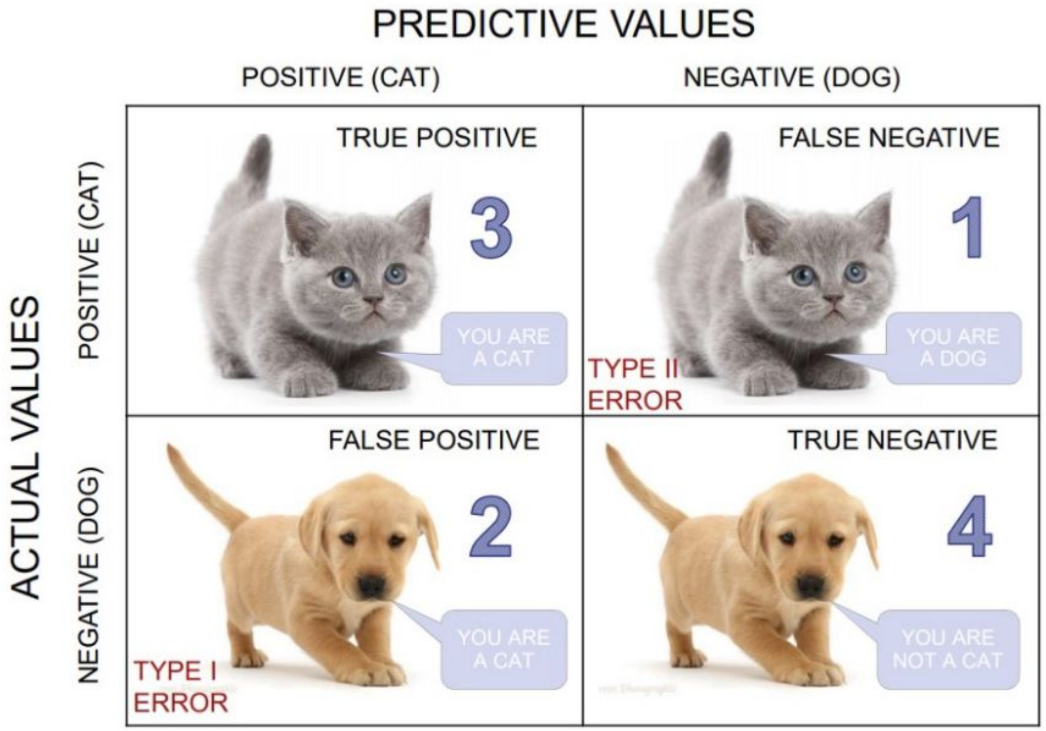
Logistic Regression Evaluation Metrics

- Confusion Matrix
- AUC-ROC curve

Logistic Regression Evaluation Metrics

- Confusion Matrix
- AUC-ROC curve

Confusion Matrix



Confusion Matrix

True Positive: You predicted positive and it’s true. You predicted that an animal is a cat and it actually is.

True Negative: You predicted negative and it’s true. You predicted that animal is not a cat and it actually is not (it’s a dog).

False Positive (Type 1 Error): You predicted positive and it’s false. You predicted that animal is a cat but it actually is not (it’s a dog).

False Negative (Type 2 Error): You predicted negative and it’s false. You predicted that animal is not a cat but it actually is.

Classification Accuracy

$$\boldsymbol{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Recall (aka Sensitivity)

Recall is defined as the ratio of the total number of correctly classified positive classes divide by the total number of positive classes. This is also called as True Positive Rate.

Recall should be high.

$$\textbf{\textit{Recall}} = \frac{TP}{TP + FN}$$

Precision

Precision is defined as the ratio of the total number of correctly classified positive classes divided by the total number of predicted positive classes.

Precision should be high.

$$\textbf{\textit{Precision}} = \frac{TP}{TP + FP}$$

F Score or F1 Score

It is difficult to compare two models with different Precision and Recall. So to make them comparable, we use F-Score.

It is the Harmonic Mean of Precision and Recall.

F-score should be high.

$$\textbf{\textit{F - score}} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

Specificity

Specificity determines the proportion of actual negatives that are correctly identified.

$$\textbf{\textit{Specificity}} = \frac{TN}{TN + FP}$$

Important Terms

TPR = TP/Total Positives = TP/(TP+FN) = Sensitivity = Recall

TPR tells us what proportion of the positive class got correctly classified.

FPR = FP/Total Negatives = FP/(TN+FP)

FPR tells us what proportion of the negative class got incorrectly classified by the classifier.

Important Terms

TNR = TN/Total Negatives = TN/(TN+FP) = Specificity

TNR tells us what proportion of the negative class got correctly classified.

FNR = FN/Total Positives = FN/(TP+FN)

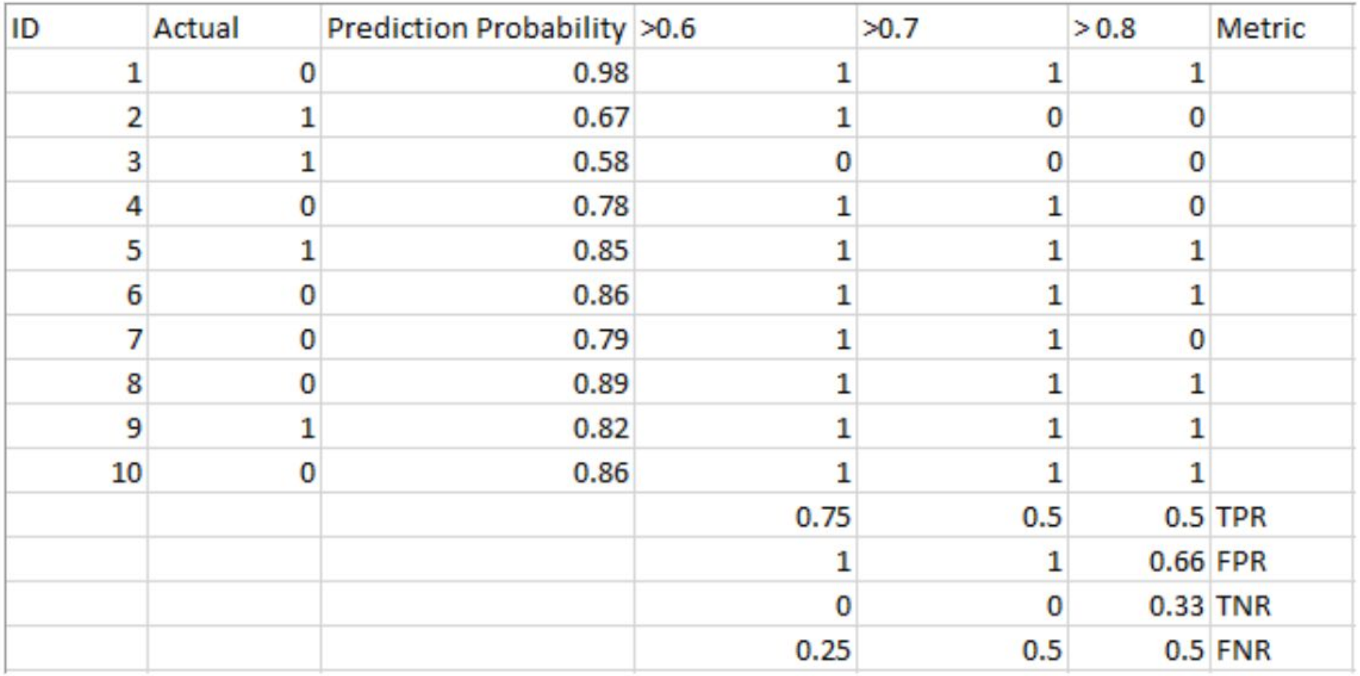
FNR tells us what proportion of the positives class got incorrectly classified by the classifier.

AUC-ROC

It is one of the most important evaluation metrics for checking any classification model's performance.

It is also written as AUROC (Area Under the Receiver Operating Characteristics)

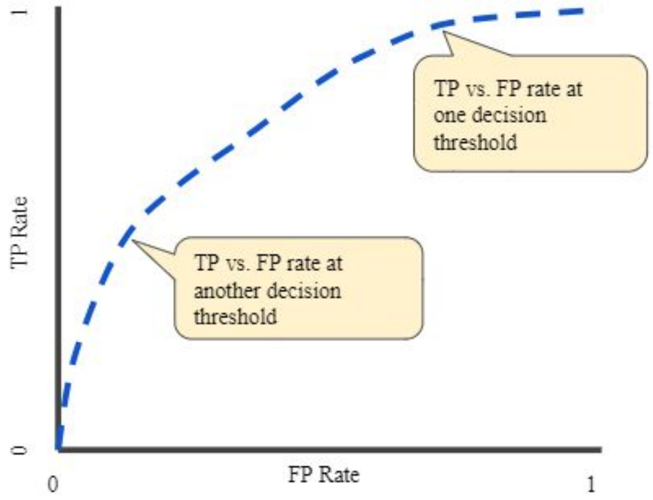
What is AUC-ROC curve?



ID	Actual	Prediction Probability	>0.6	>0.7	> 0.8	Metric
1	0	0.98	1	1	1	
2	1	0.67	1	0	0	
3	1	0.58	0	0	0	
4	0	0.78	1	1	0	
5	1	0.85	1	1	1	
6	0	0.86	1	1	1	
7	0	0.79	1	1	0	
8	0	0.89	1	1	1	
9	1	0.82	1	1	1	
10	0	0.86	1	1	1	
			0.75	0.5	0.5	TPR
			1	1	0.66	FPR
			0	0	0.33	TNR
			0.25	0.5	0.5	FNR

ROC

An ROC curve plots TPR vs. FPR at different classification thresholds. The following figure shows a typical ROC curve.

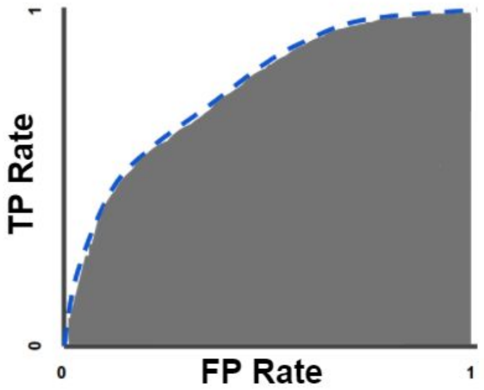


AUC

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1).

The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between the classes(https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/)

The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.



Bias

Bias are the simplifying assumptions made by a model to make the target function easier to learn.

Low Bias: Suggests less assumptions about the form of the target function.
High-Bias: Suggests more assumptions about the form of the target function to make the target function easier and simple.

Variance

Variance is the amount that the **estimate of the target function** will change if different training data was used.

Low Variance: Suggests small changes to the estimate of the target function with changes to the training dataset.

High Variance: Suggests large changes to the estimate of the target function with changes to the training dataset.

Bias Variance Tradeoff

The goal of any supervised machine learning algorithm is to achieve low bias and low variance.

Generally,

- Linear machine learning algorithms often have a high bias but a low variance.
- Nonlinear machine learning algorithms(Decision Tree) often have a low bias but a high variance.

Underfitting(Fitting the train data less than needed)

Underfit model is simpler than it should be.

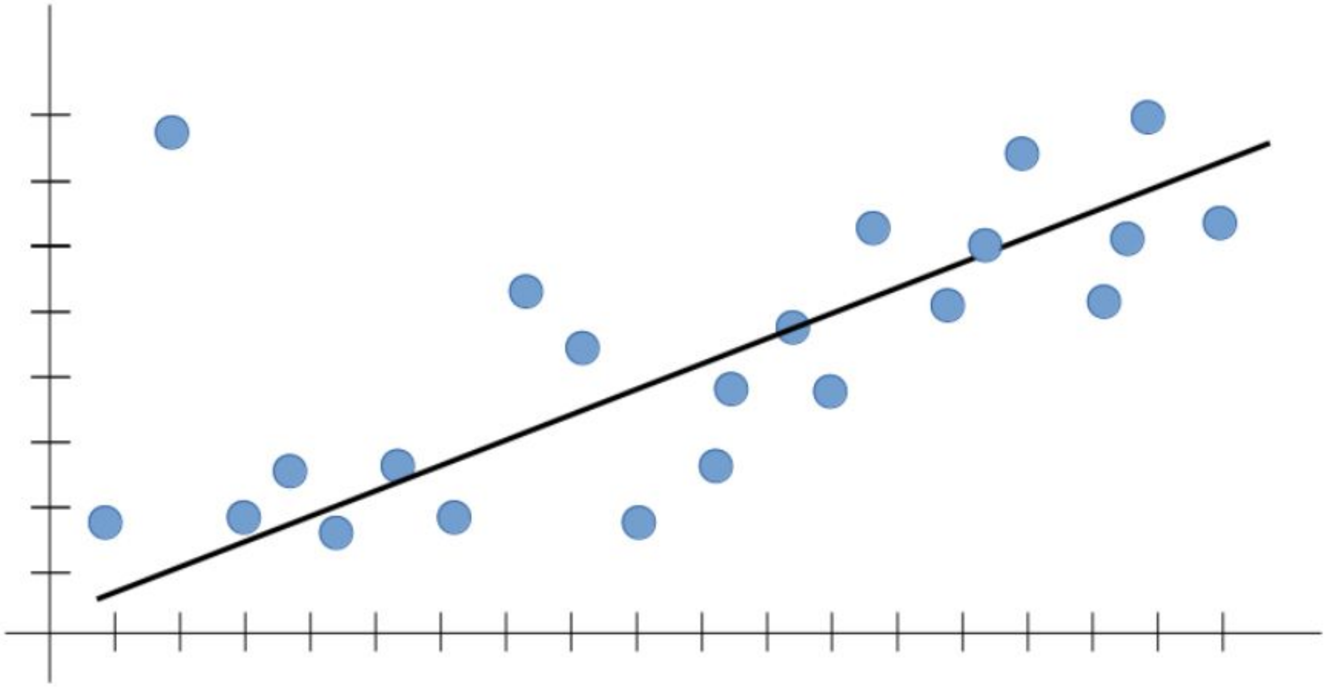
In other words, model fails to learn on the given data and acquire the intricate pattern of the dataset. So underfit models have high bias.

It means a biased model will not fit on the Training Dataset properly and hence will have low training accuracy (or high training loss).

Since the model is very simple, it does not change much with the new dataset so undefit models have low variance.

Underfit models: High bias and low variance

Underfit model



Overfitting(Fitting the train data more than needed)

If the model is overly complex it tries to fit a much more complex curve to a relatively simpler data. It fits the train data more than needed.

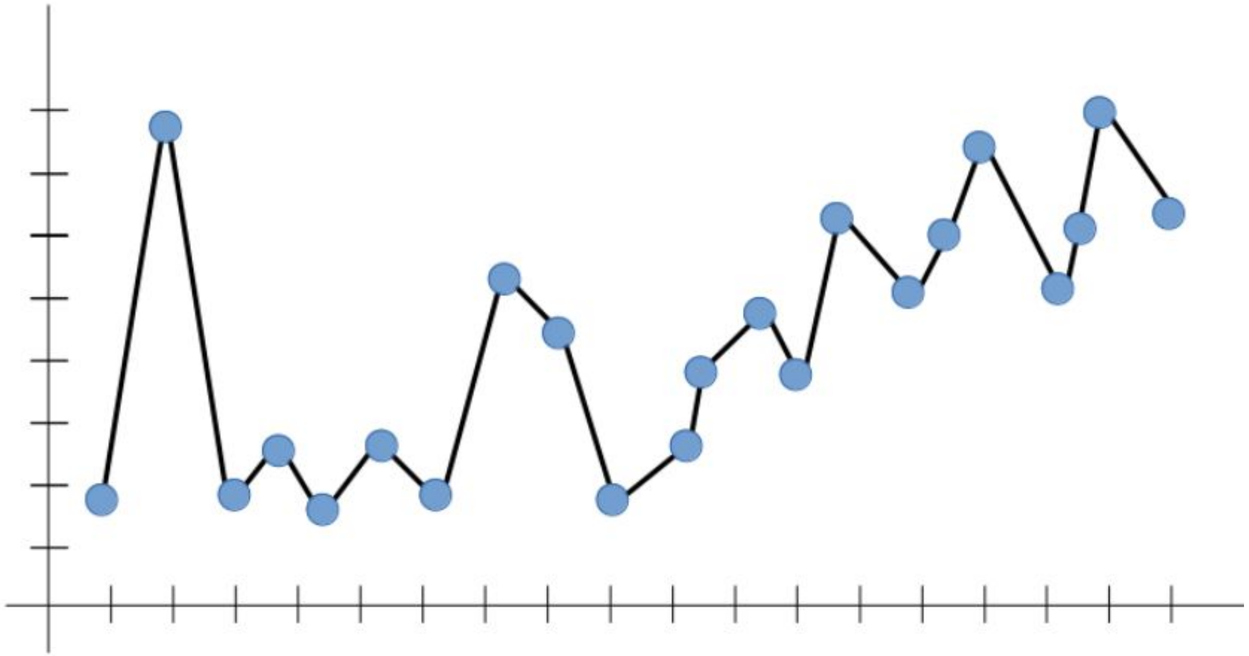
The overfit model will have less training error since the model fits the train data very well. But with the test data the model with give more error since overfit model is made exclusively for the training data. It cannot generalize on the test data.

So the overfit model has high variance and low bias.

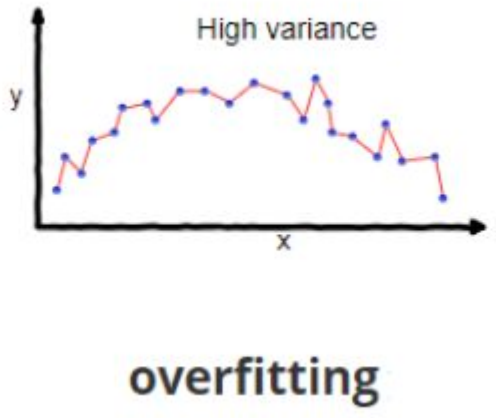
Low bias because model is not simplifying things and high variance because model does not fit the test data and will have to change a lot to fit test data.

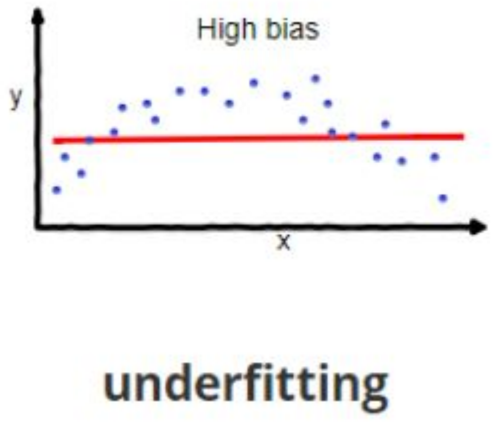
Overfitting: High variance and low bias

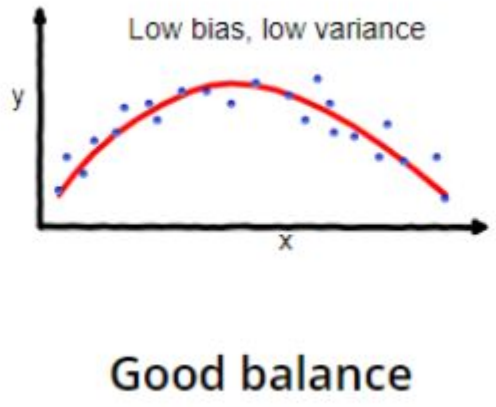
Overfit model



Desired: Low Bias & Low Variance

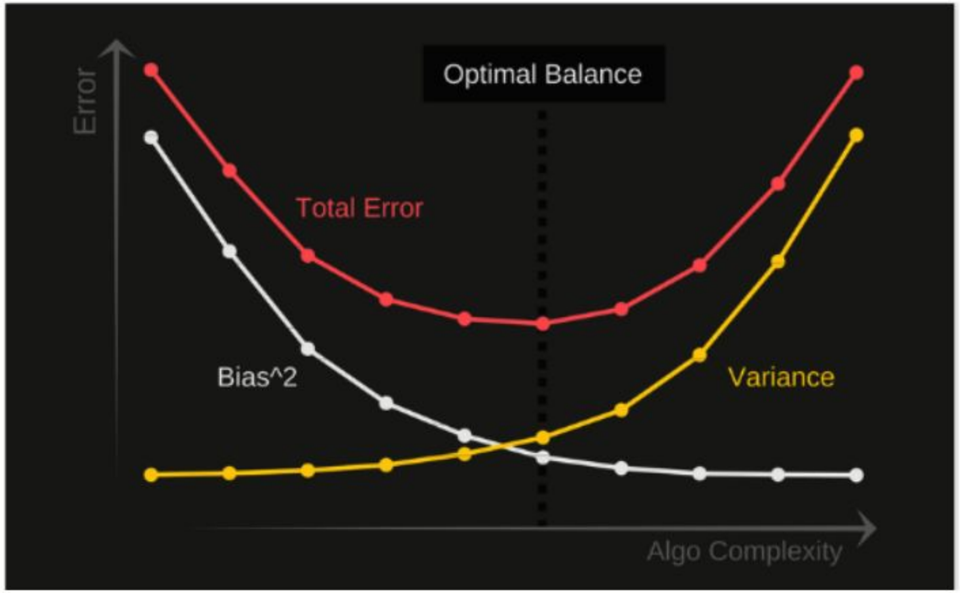






Total Error→ Bias Variance Tradeoff

Total Error = Bias^2 + Variance + Irreducible Error



Imbalance Data

Imbalanced data typically refers to a problem with classification problems where the classes are not represented equally.

For example, you may have a 2-class (binary) classification problem with 100 instances (rows). A total of 80 instances are labeled with Class-1 and the remaining 20 instances are labeled with Class-2.

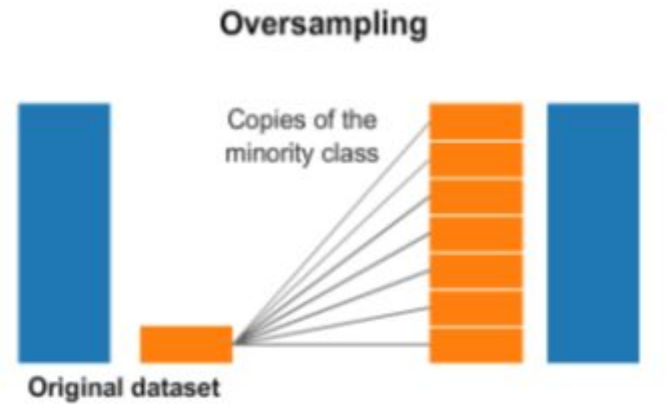
This is an imbalanced dataset and the ratio of Class-1 to Class-2 instances is 80:20 or more concisely 4:1.

How to handle imbalance data?

- Collect more data
- Resampling
- Use different evaluation metric

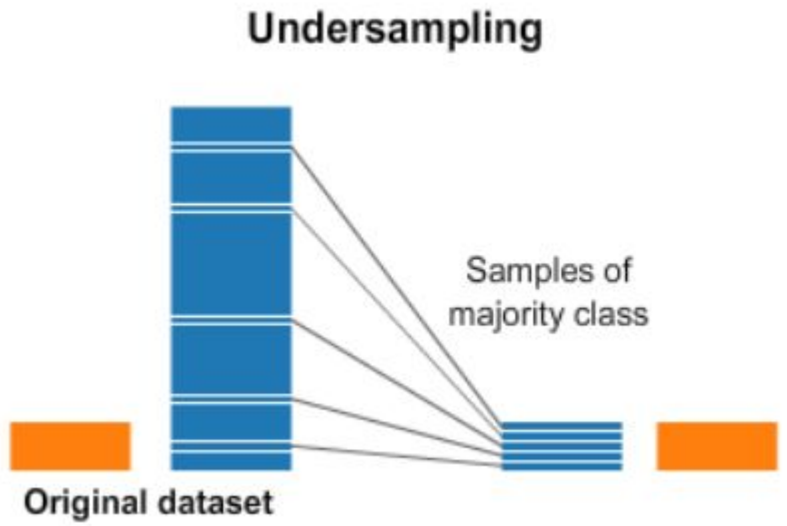
Oversampling

The simplest implementation of over-sampling is to duplicate random records from the minority class, which can lead to lower performance



Undersampling

In under-sampling, the simplest technique involves removing random records from the majority class, which can cause loss of information.



Feature Scaling

- Normalization
- Standardization

Normalization

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Here, Xmax and Xmin are the maximum and the minimum values of the feature respectively.

- When the value of X is the minimum value in the column, the numerator will be 0, and hence X' is 0
- On the other hand, when the value of X is the maximum value in the column, the numerator is equal to the denominator and thus the value of X' is 1
- If the value of X is between the minimum and the maximum value, then the value of X' is between 0 and 1

Standardization

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

$$X' = \frac{X - \mu}{\sigma}$$

Normalization vs Standardization

- Normalization is good to use when you know that the distribution of your data does not follow a Gaussian distribution.
- Standardization, on the other hand, can be helpful in cases where the data follows a Gaussian distribution. However, this does not have to be necessarily true.
- Standardization works better with dataset having outliers.
- However, at the end of the day, the choice of using normalization or standardization will depend on your problem and the machine learning algorithm you are using. There is no hard and fast rule to tell you when to normalize or standardize your data. You can always start by fitting your model to raw, normalized and standardized data and compare the performance for best results.

Is it important to standardize before applying PCA?

Is it important to standardize before applying PCA?

PCA finds new directions based on the covariance matrix of original variables. Since the covariance matrix is sensitive to the standardization of variables. Usually, we do standardization to assign equal weights to all the variables. If we use features of different scales, we get misleading directions. But, it is not necessary to standardize the variables, if all the variables are on the same scale.

Should one remove highly correlated variables before doing PCA?

Should one remove highly correlated variables before doing PCA?

No, PCA loads out all highly correlated variables on the same Principal Component(Eigenvector), not different ones.

How can you evaluate the performance of a dimensionality reduction algorithm on your dataset?

How can you evaluate the performance of a dimensionality reduction algorithm on your dataset?

Intuitively, a dimensionality reduction algorithm performs well if it eliminates a lot of dimensions from the dataset without losing too much information. Alternatively, if you are using dimensionality reduction as a preprocessing step before another Machine Learning algorithm (e.g., a Random Forest classifier), then you can simply measure the performance of that second algorithm; if dimensionality reduction did not lose too much information, then the algorithm should perform just as well as when using the original dataset.

What are the drawbacks of PCA?

What are the drawbacks of PCA?

- Some information is lost, possibly degrading the performance of subsequent training algorithms.
- It can be computationally intensive.
- Transformed features are often hard to interpret.

https://alekhyo.medium.com/interview-questions-on-pca-9cdc96ddaa9f

Is it important to standardize before applying PCA?

Is it important to standardize before applying PCA?

PCA finds new directions based on the covariance matrix of original variables. Since the covariance matrix is sensitive to the standardization of variables. Usually, we do standardization to assign equal weights to all the variables. If we use features of different scales, we get misleading directions. But, it is not necessary to standardize the variables, if all the variables are on the same scale.

Should one remove highly correlated variables before doing PCA?

Should one remove highly correlated variables before doing PCA?

No, PCA loads out all highly correlated variables on the same Principal Component(Eigenvector), not different ones.

How can you evaluate the performance of a dimensionality reduction algorithm on your dataset?

How can you evaluate the performance of a dimensionality reduction algorithm on your dataset?

Intuitively, a dimensionality reduction algorithm performs well if it eliminates a lot of dimensions from the dataset without losing too much information. Alternatively, if you are using dimensionality reduction as a preprocessing step before another Machine Learning algorithm (e.g., a Random Forest classifier), then you can simply measure the performance of that second algorithm; if dimensionality reduction did not lose too much information, then the algorithm should perform just as well as when using the original dataset.

What are the drawbacks of PCA?

What are the drawbacks of PCA?

- Some information is lost, possibly degrading the performance of subsequent training algorithms.
- It can be computationally intensive.
- Transformed features are often hard to interpret.

https://alekhyo.medium.com/interview-questions-on-pca-9cdc96ddaa9f

What is Entropy?

Entropy

Entropy is a degree of randomness or uncertainty or disorder.

Entropy = 0 means no impurity.

Higher the entropy means higher the impurity.

Lower entropy means more pure node and higher entropy means less pure nodes.

$$H(X) = H(p_1, ..., p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

How Does the Decision Tree Algorithm works?

How Does the Decision Tree Algorithm works?

The basic idea behind any decision tree algorithm is as follows:

1. Select the best Feature using Attribute Selection Measures(ASM) to split the records. (https://medium.com/analytics-steps/understanding-the-gini-index-and-information-gain-in-decision-trees-ab4720518ba8)

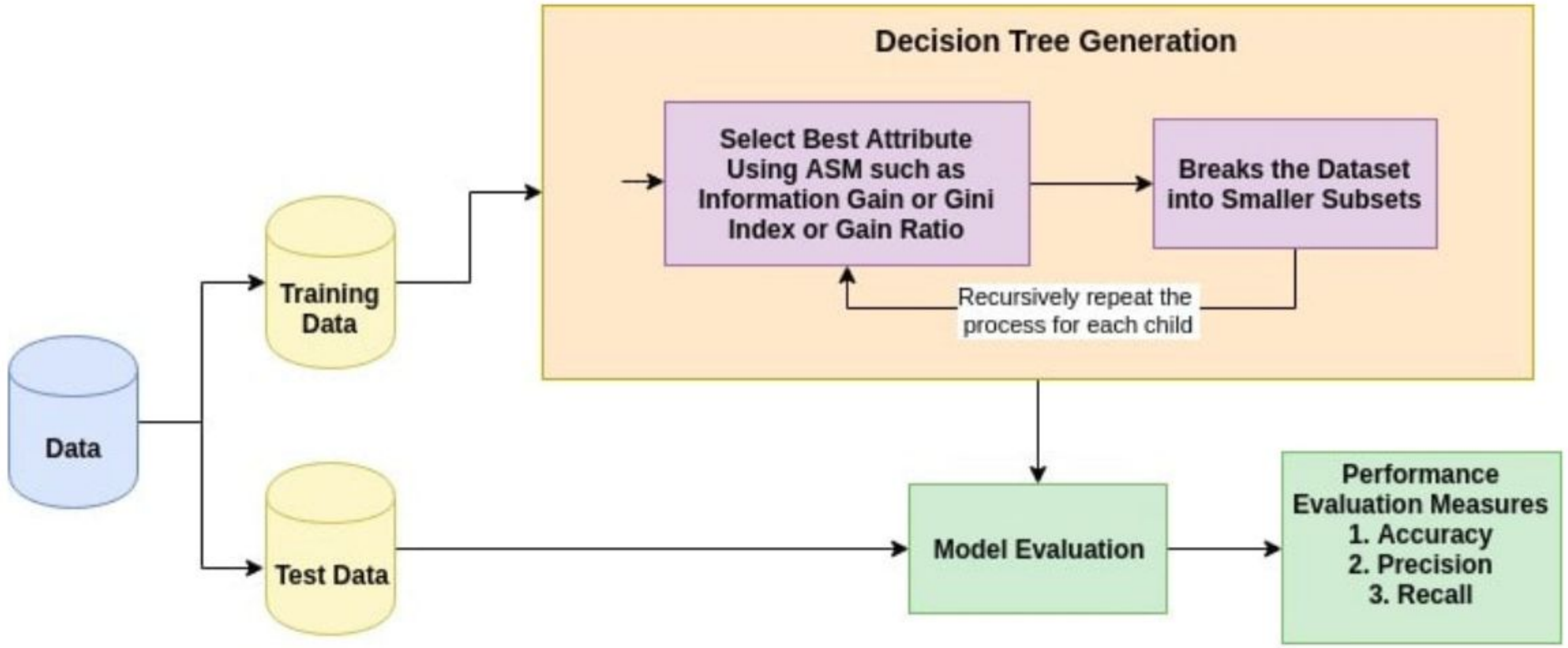
2. Make that attribute/feature a decision node and break the dataset into smaller subsets.

3. Start the tree-building process by repeating this process recursively for each child until one of the following condition is being achieved :

    a) All tuples belonging to the same attribute value.

    b) There are no more of the attributes remaining.

How Does the Decision Tree Algorithm works?


Decision Tree Generation
Select Best Attribute Using ASM such as Information Gain or Gini Index or Gain Ratio
Breaks the Dataset into Smaller Subsets
Recursively repeat the process for each child
Training Data
Data
Test Data
Model Evaluation
Performance Evaluation Measures
1. Accuracy
2. Precision
3. Recall

How does Random Forest ensure that different DTs are uncorrelated?

Ensuring that the Models Diversify Each Other

How does random forest ensure that the behavior of each individual tree is not too correlated with the behavior of any of the other trees in the model?

It uses the following two methods:

- Bagging
- Feature Randomness

Bagging

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.

Bagging

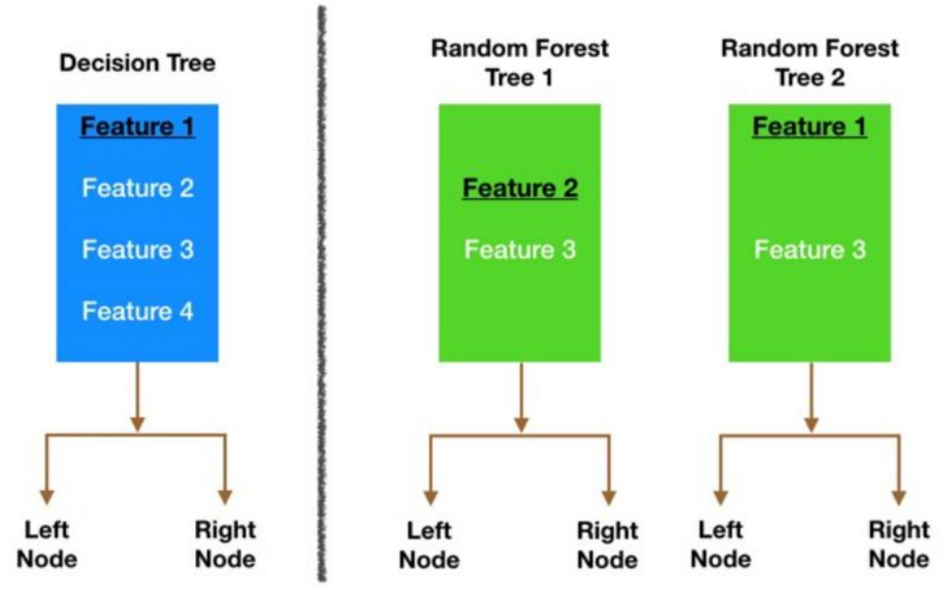
If we have a sample of size N, we are still feeding each tree a training set of size N.

But instead of the original training data, we take a random sample of size N with replacement.

For example, if our training data was [1, 2, 3, 4, 5, 6] then we might give one of our trees the following list [1, 2, 2, 3, 6, 6], other tree [1,1,1,2,5,6] and other tree [1,2,3,4,5,6].

Notice that all lists are of length and we sample with replacement.

Feature Randomness



Node splitting in a random forest model is based on a random subset of features for each tree.

Feature Randomness

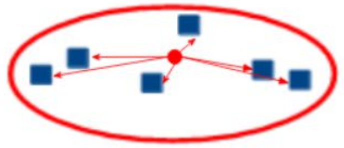
In a normal decision tree, when it is time to split a node, we consider every possible feature and pick the one that produces the most separation between the observations in the left node vs. those in the right node.

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.

So in our random forest, we end up with trees that are not only trained on different sets of data (thanks to bagging) but also use different features to make decisions.

What is Inertia?

Inertia


Intra cluster distance

Inertia actually calculates the sum of square of distances of all the points within a cluster from the centroid of that cluster.

This distance within the clusters is known as intracluster distance. So, inertia gives us the sum of square of intracluster distances.

We calculate this for all the clusters and the final inertial value is the sum of all these distances.

The distance between the points within a cluster should be as low as possible. Keeping this in mind, we can say that the lesser the inertia value, the better our clusters are.

What is the termination conditions in K means clustering?

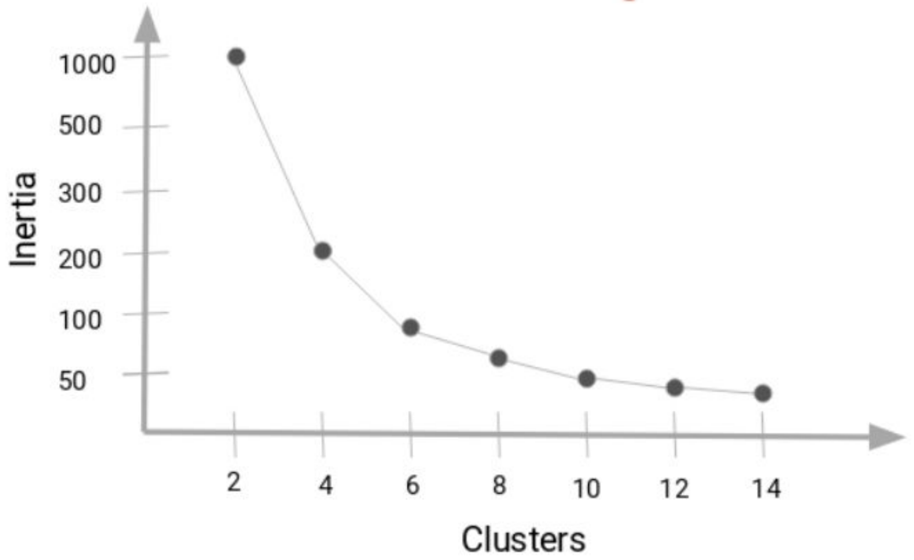
K Means Clustering : Stopping Criteria

There are essentially three stopping criteria that can be adopted to stop the K-means algorithm:

- Centroids of newly formed clusters do not change
- Points remain in the same cluster
- Maximum number of iterations are reached

What is Elbow method in Clustering?

How to Choose the Right Number of Clusters in K-Means Clustering? - Elbow Method



How to Choose the Right Number of Clusters in K-Means Clustering? - Elbow Method

When we changed the cluster value from 2 to 4, the inertia value reduced very sharply.

This decrease in the inertia value reduces and eventually becomes constant as we increase the number of clusters further.

So, the cluster value where this decrease in inertia value becomes constant can be chosen as the right cluster value for our data.

Why choose the elbow point : You must also look at the computation cost while deciding the number of clusters. If we increase the number of clusters, the computation cost will also increase.

Difference between parameter and hyperparameter

Parameter

A model parameter is a configuration variable that is internal to the model and whose value can be estimated from data.

They are estimated or learned from data.

They are often not set manually by the practitioner.

Example: The coefficients in a linear regression or logistic regression.

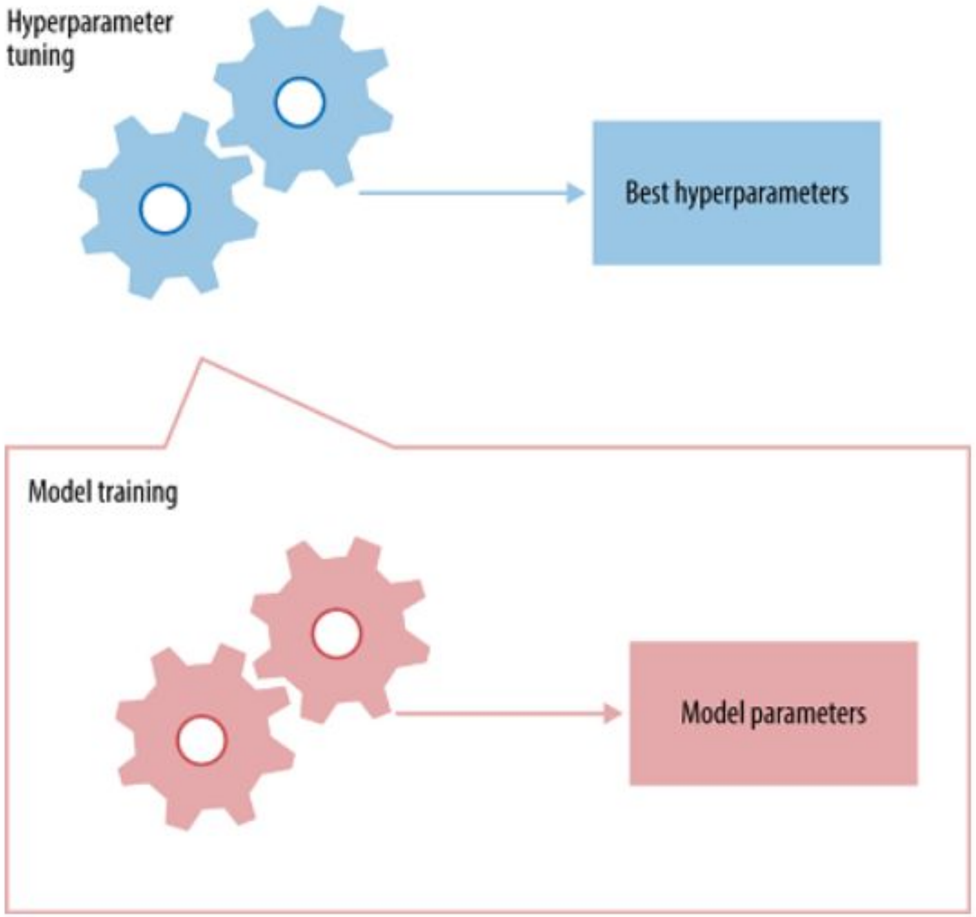
Hyperparameter

A model hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data.

They are often used in processes to help estimate model parameters.

Model parameters are learned from data and hyper-parameters are tuned to get the best fit.

Example: learning rate, number of estimators in decision tree, regularization parameter



Hyperparameters and Parameters

The best way to think about hyperparameters is like the settings of an algorithm that can be adjusted to optimize performance, just as we might turn the knobs of an AM radio to get a clear signal.

While model parameters are learned during training — such as the slope and intercept in a linear regression — hyperparameters must be set by the data scientist before training.

In the case of a random forest, hyperparameters include the number of decision trees in the forest and the number of features considered by each tree when splitting a node.

Hyperparameter Tuning

Scikit-Learn implements a set of sensible default hyperparameters for all models, but these are not guaranteed to be optimal for a problem.

The best hyperparameters are usually impossible to determine ahead of time, and tuning a model is where machine learning turns from a science into trial-and-error based engineering.

Hyperparameter Tuning

- Hyperparameter tuning relies more on experimental results than theory, and thus the best method to determine the optimal settings is to try many different combinations evaluate the performance of each model.
- However, evaluating each model only on the training set can lead to one of the most fundamental problems in machine learning: overfitting.
- If we optimize the model for the training data, then our model will score very well on the training set, but will not be able to generalize to new data, such as in a test set.

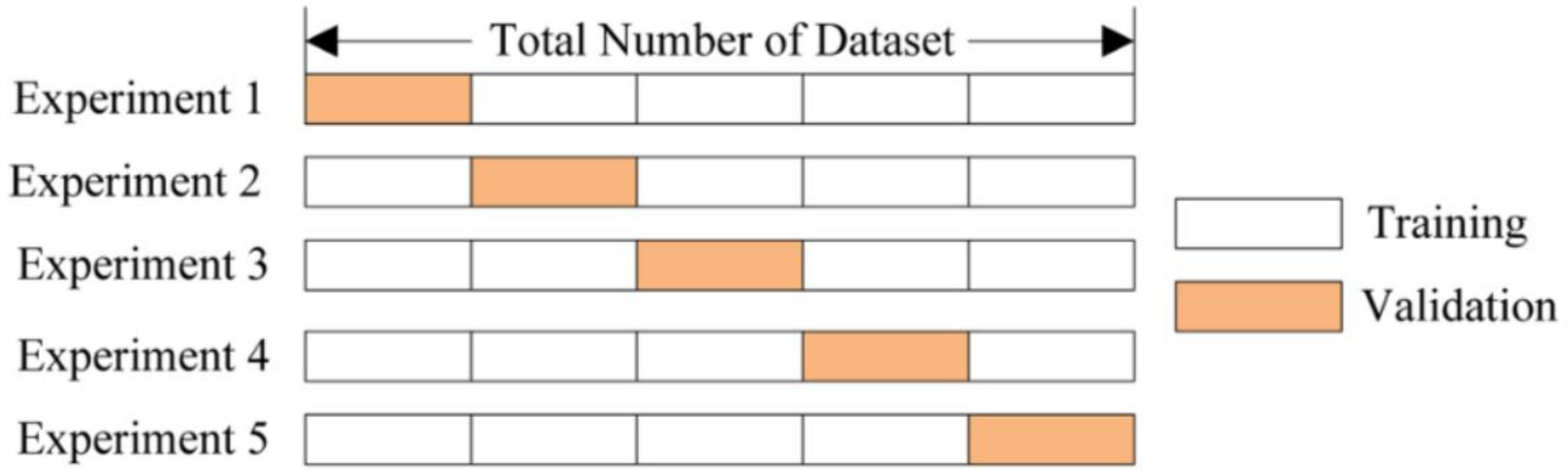
What is cross validation?

Cross Validation

- The technique of cross validation (CV) is best explained by example using the most common method, K-Fold CV.
- When we approach a machine learning problem, we make sure to split our data into a training and a testing set.
- In K-Fold CV, we further split our training set into K number of subsets, called folds. We then iteratively fit the model K times, each time training the data on K-1 of the folds and evaluating on the Kth fold (called the validation data).

Cross Validation

- As an example, consider fitting a model with K = 5.
- The first iteration we train on the first four folds and evaluate on the fifth.
- The second time we train on the first, second, third, and fifth fold and evaluate on the fourth.
- We repeat this procedure 3 more times, each time evaluating on a different fold.
- At the very end of training, we average the performance on each of the folds to come up with final validation metrics for the model.


Total Number of Dataset
Experiment 1
Experiment 2
Experiment 3
Experiment 4
Experiment 5
Training
Validation

Hyperparameter Tuning

- For hyperparameter tuning, we perform many iterations of the entire K-Fold CV process, each time using different model settings.
- We then compare all of the models, select the best one, train it on the full training set, and then evaluate on the testing set.
- But there is a problem.
- Each time we want to assess a different set of hyperparameters, we have to split our training data into K fold and train and evaluate K times.
- If we have 10 sets of hyperparameters and are using 5-Fold CV, that represents ?? training loops.

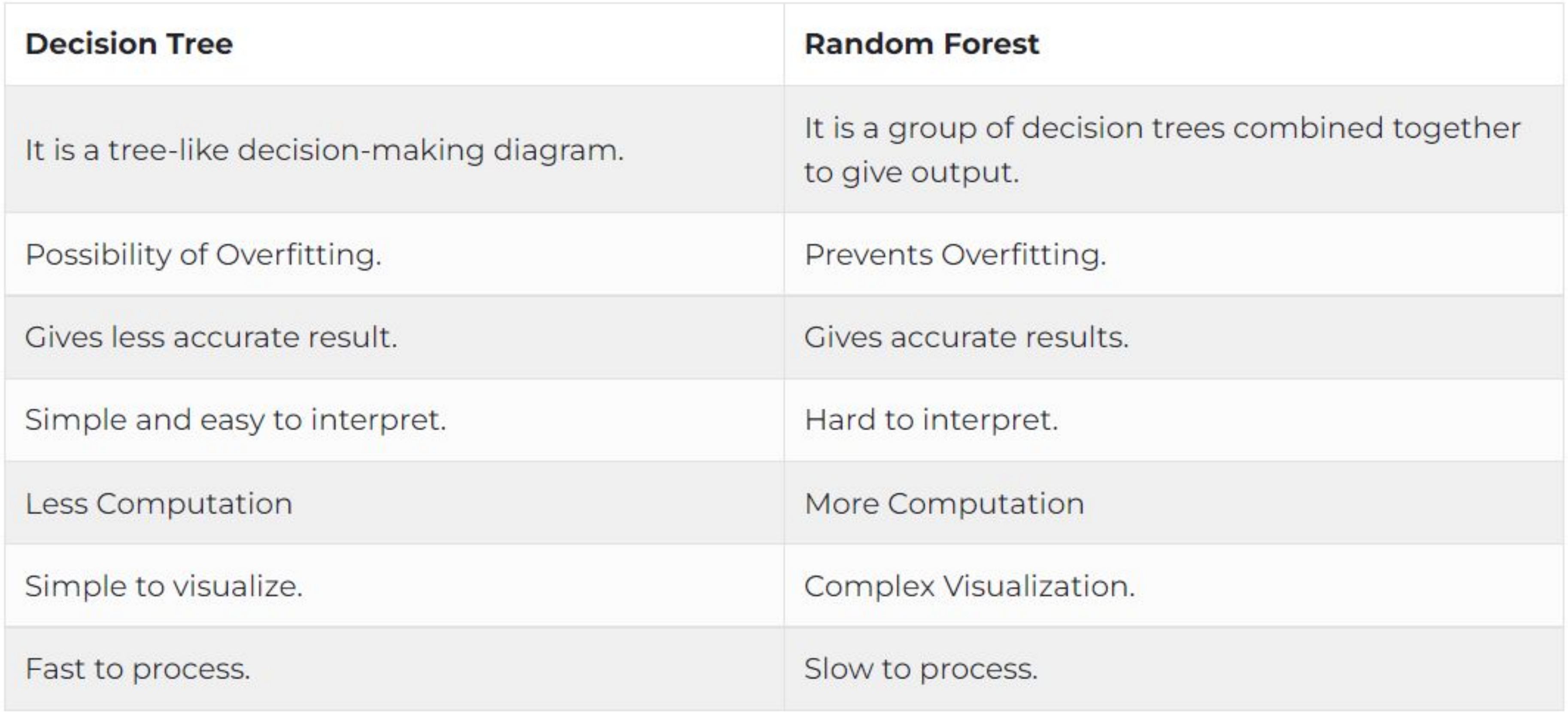
Grid Search CV

- Grid Search can be thought of as an exhaustive search for selecting a model.
- In Grid Search, the data scientist sets up a grid of hyperparameter values and for each combination, trains a model and scores.
- In this approach, every combination of hyperparameter values is tried which can be very inefficient. - Time & Space Complexity

Random Search CV

- Random Search sets up a grid of hyperparameter values and selects random combinations to train the model and score.
- This allows you to explicitly control the number of parameter combinations that are attempted.
- The number of search iterations is set based on time or resources.

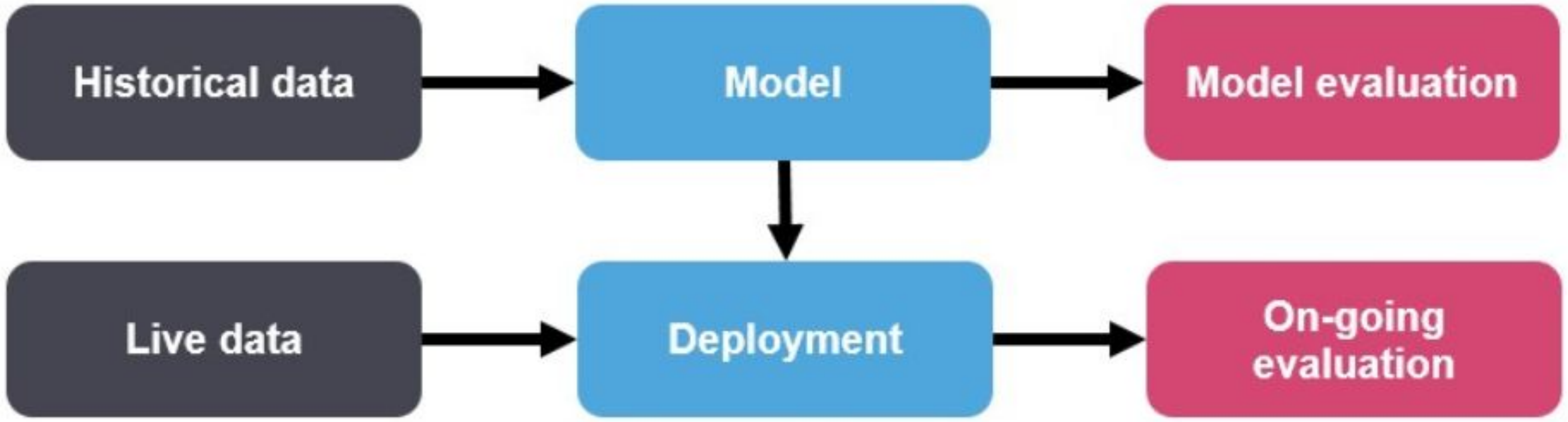
What is the difference between Random Forest & Decision Tree



Decision Tree	Random Forest
It is a tree-like decision-making diagram.	It is a group of decision trees combined together to give output.
Possibility of Overfitting.	Prevents Overfitting.
Gives less accurate result.	Gives accurate results.
Simple and easy to interpret.	Hard to interpret.
Less Computation	More Computation
Simple to visualize.	Complex Visualization.
Fast to process.	Slow to process.

Explain the process of ML deployment

Model Deployment



What is better - random forest or multiple decision trees?

What is better - random forest or multiple decision trees?

Random forest is better than multiple decision trees as random forests are much more robust, accurate, and lesser prone to overfitting as it is an ensemble method that ensures multiple weak decision trees learn strongly.

Question

A data set is given to you about utilities fraud detection(imbalanced data). You have built a classifier model and achieved a accuracy score of 98.5%. Is this a good model? If yes, justify. If not, what can you do about it?

Answer

Data set about utilities fraud detection is not balanced enough i.e. imbalanced. In such a data set, accuracy score cannot be the measure of performance as it may only be predict the majority class label correctly but in this case our point of interest is to predict the minority label. But often minorities are treated as noise and ignored. So, there is a high probability of misclassification of the minority label as compared to the majority label.

What is data leakage?

Data Leakage

In Machine learning, Data Leakage refers to a mistake that is made by the creator of a machine learning model in which they accidentally share the information between the test and training data sets.

Typically, when splitting a data set into testing and training sets, the goal is to ensure that no data is shared between these two sets. Ideally, there is no intersection between these two sets.

Data Leakage

When you split your data into training and testing subsets, some of your data present in the test set is also copied in the train set and vice-versa.

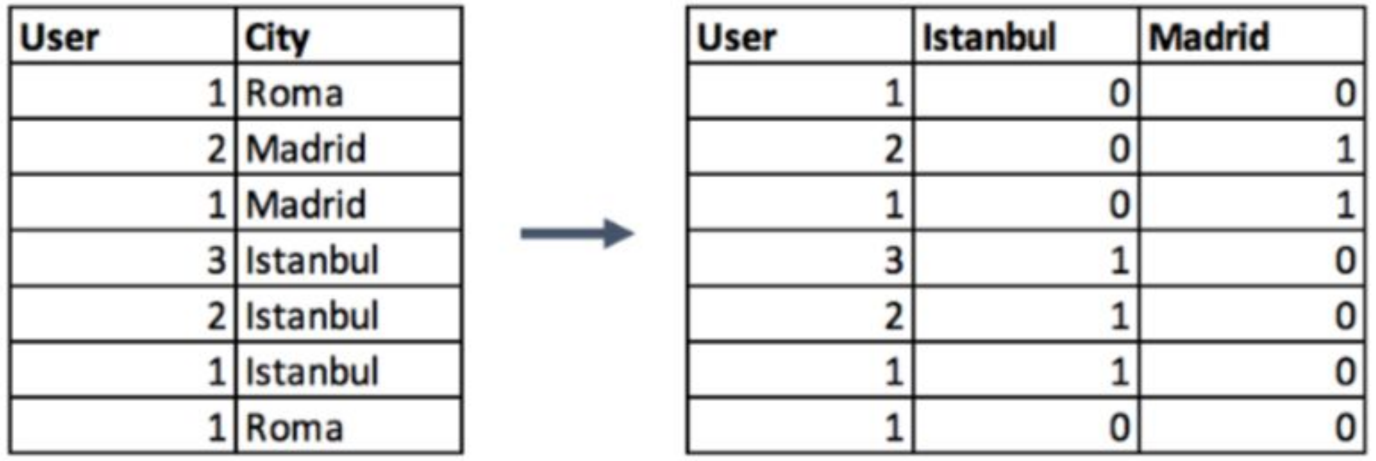
As a result of which when you train your model with this type of split it will give really good results on the train and test set i.e, both training and testing accuracy would be high.

https://www.analyticsvidhya.com/blog/2021/07/data-leakage-and-its-effect-on-the-performance-of-an-ml-model/

What is one hot encoding?

One hot encoding

One-hot encoding is one of the most common encoding methods in machine learning. This method spreads the values in a column to multiple flag columns and assigns 0 or 1 to them.



User	City
1	Roma
2	Madrid
1	Madrid
3	Istanbul
2	Istanbul
1	Istanbul
1	Roma

User	Istanbul	Madrid
1	0	0
2	0	1
1	0	1
3	1	0
2	1	0
1	1	0
1	0	0

One hot encoding

If you have N distinct values in the column, it is enough to map them to N-1 binary columns, because the missing value can be deducted from other columns. If all the columns in our hand are equal to 0, the missing value must be equal to 1.

What is imputation?

Imputation

Imputation is a technique used for replacing the missing data with some substitute value to retain most of the data/information of the dataset.

Many people say best imputation way is to use the medians of the columns. As the averages of the columns are sensitive to the outlier values, while medians are more solid in this respect.

What is Feature Engineering? List some techniques used in feature engineering.

Feature Engineering

Feature engineering is the process of transforming raw data into features that better represent the underlying problem.

Some techniques are:

- Scaling
- Imputation
- Handling outliers
- One hot encoding

https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114

What is difference between feature selection and feature extraction?

The key difference between feature selection and extraction is that feature selection keeps a subset of the original features while feature extraction creates brand new ones.

Support Vector Machine

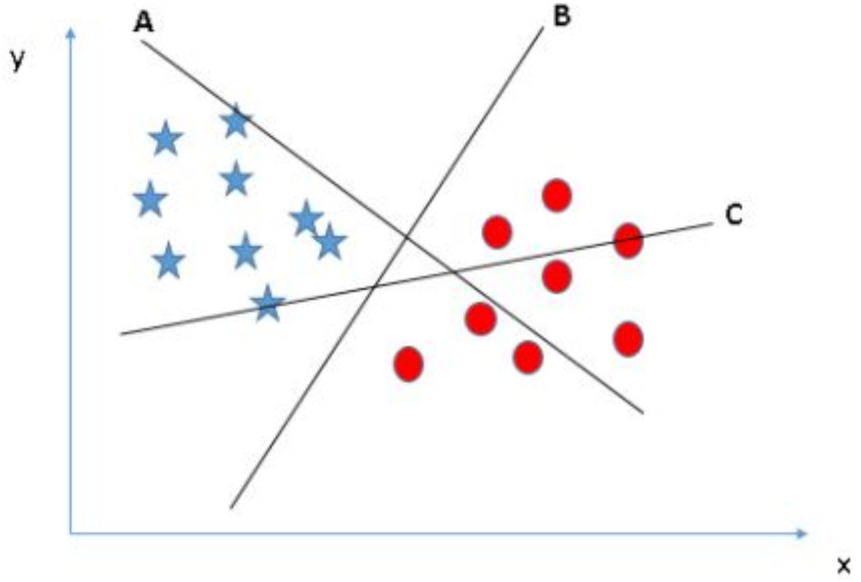
Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a **hyperplane**.

Types of SVM

- Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

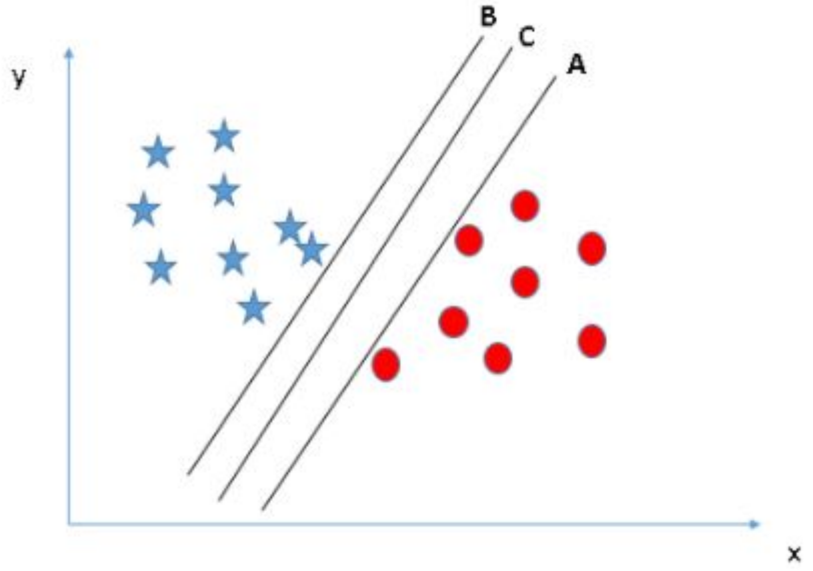
Identify the hyperplane



Identify the hyperplane

Thumb rule to identify the right hyper-plane: “Select the decision boundary which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.

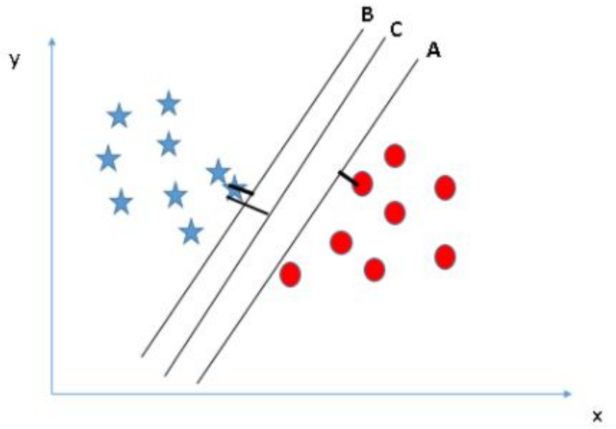
Identify the right hyper-plane



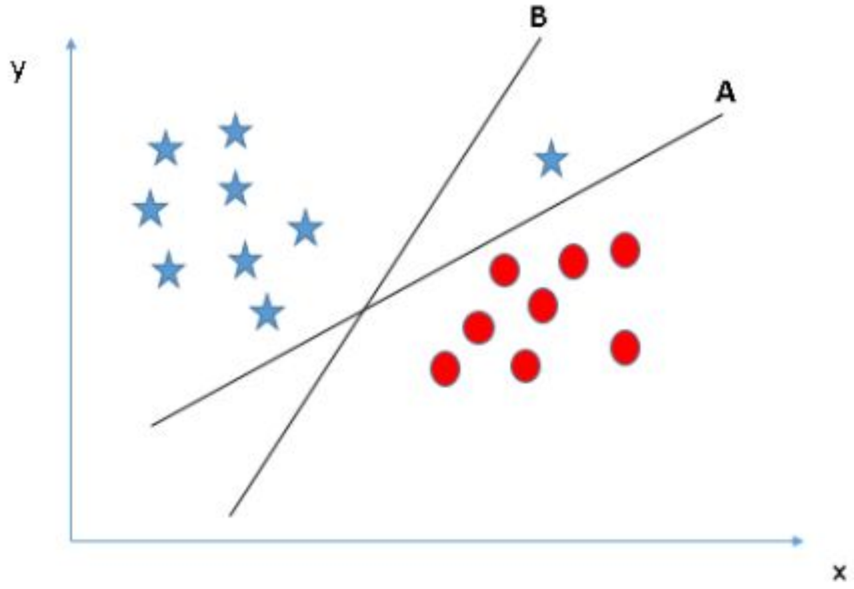
Identify the right hyper-plane

Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin

We can see that the margin for decision boundary C is high as compared to both A and B. Hence, we name the right hyper-plane as C.



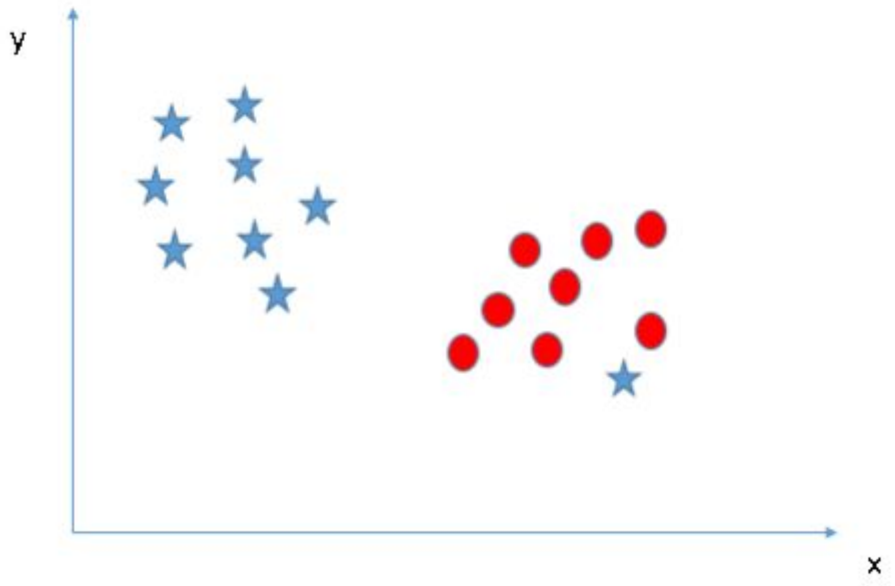
Identify the right hyper-plane



Identify the right hyper-plane

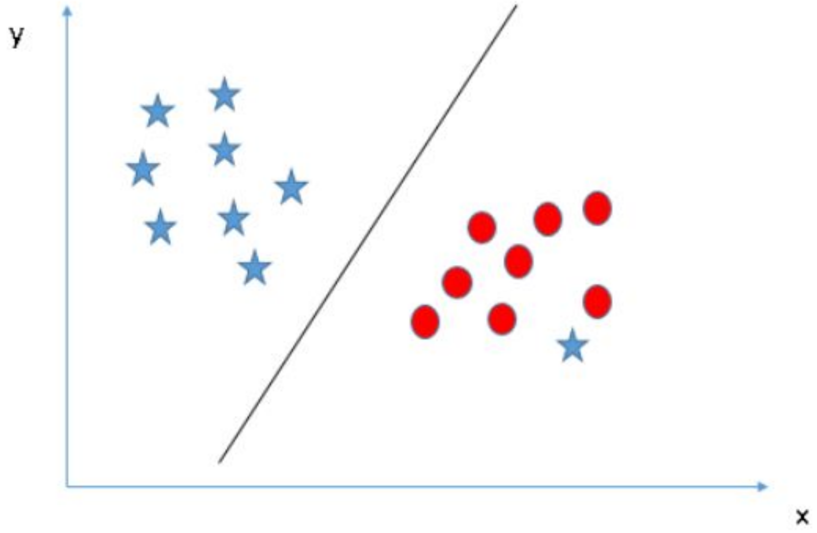
SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Therefore, the right hyper-plane is A.

Identify the hyperplane

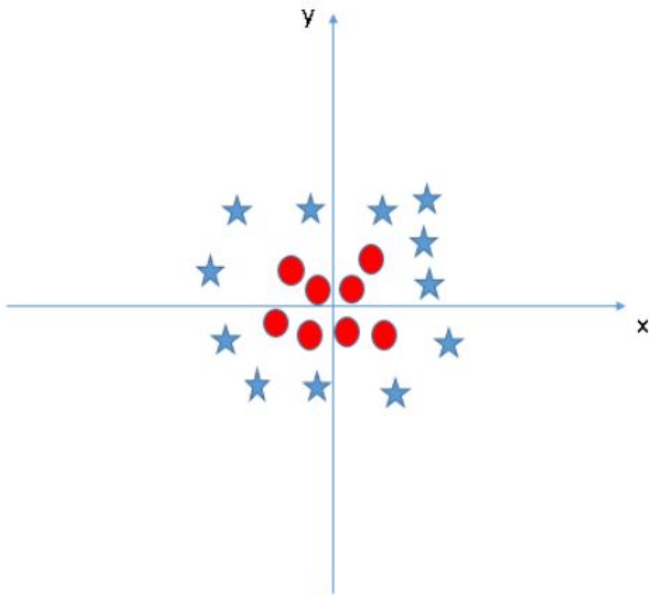


Identify the hyperplane

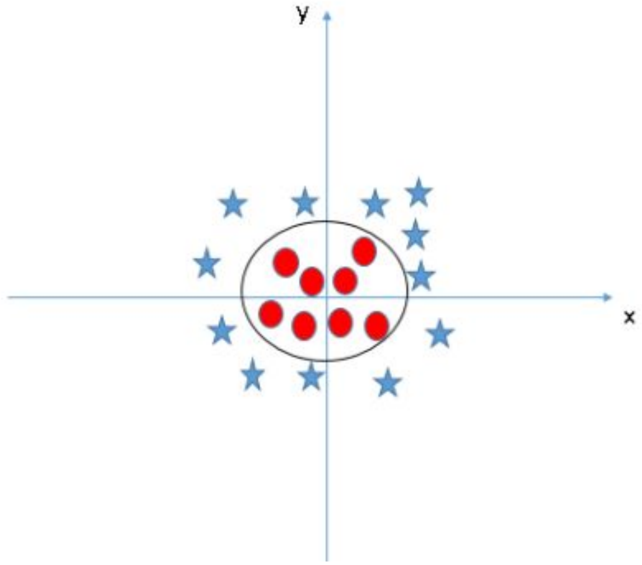
The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.



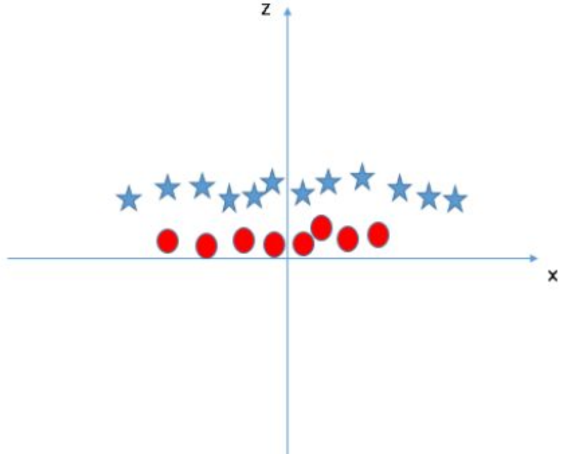
Identify the hyperplane



Identify the hyperplane



Kernel

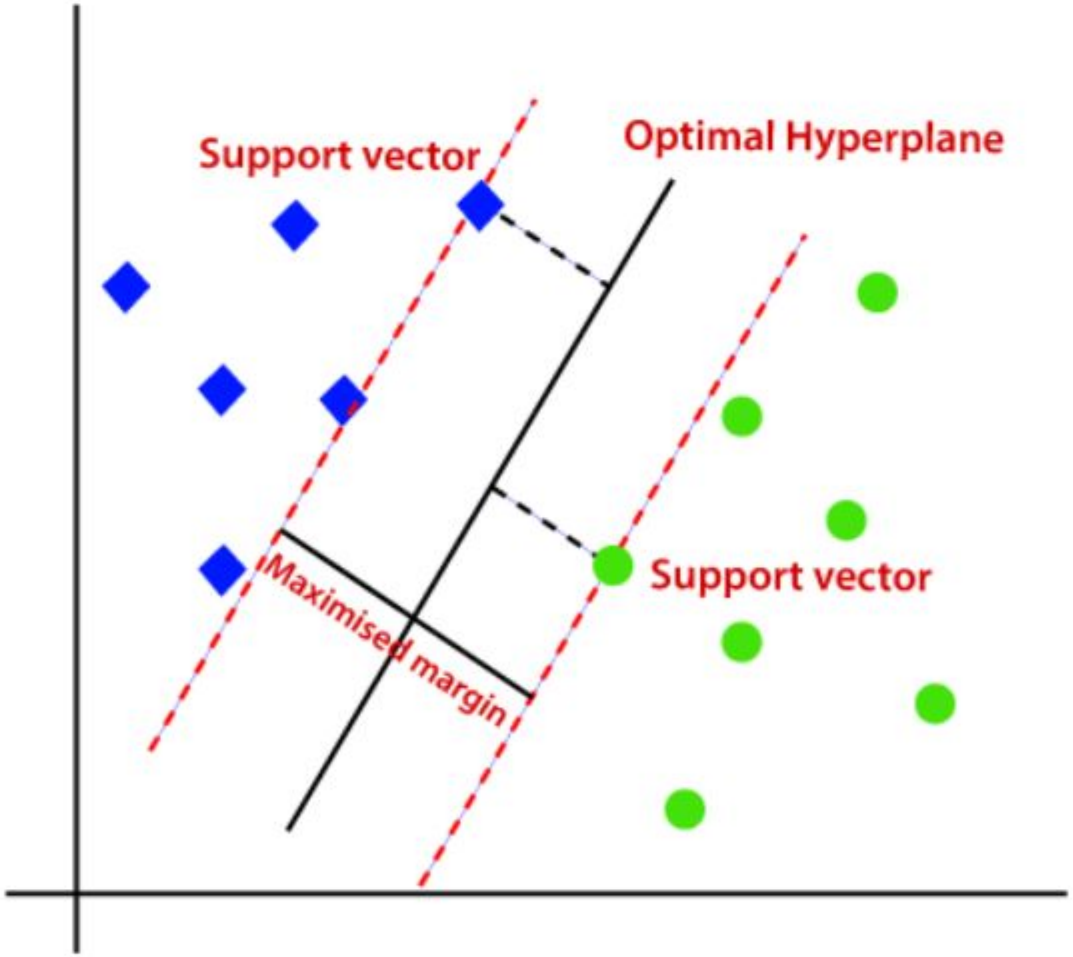


Here, we will add a new feature $z=x^2+y^2$

The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined.

Linear SVM

- SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane.
- SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors.
- The distance between the vectors and the hyperplane is called as margin. And the goal of SVM is to maximize this margin.
- The hyperplane with maximum margin is called the optimal hyperplane.


Support vector
Optimal Hyperplane
Support vector
Maximised margin

Support Vectors

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

Hyperplane

- There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points.
- This best boundary is known as the hyperplane of SVM.
- The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features, then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

SVM References

https://www.saedsayad.com/support_vector_machine.htm

https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

https://www.svm-tutorial.com/2015/06/svm-understanding-math-part-3/

https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/

https://stats.stackexchange.com/questions/37795/roc-curve-for-discrete-classifiers-like-svm-why-do-we-still-call-it-a-curve

https://towardsdatascience.com/support-vector-regression-svr-one-of-the-most-flexible-yet-robust-prediction-algorithms-4d25fbdaca60

https://www.analyticsvidhya.com/blog/2020/10/the-mathematics-behind-svm/

References

OOB Score:
https://towardsdatascience.com/what-is-out-of-bag-oob-score-in-random-forest-a7fa23d710

Multiclass Logistic Regression:

https://towardsdatascience.com/multiclass-logistic-regression-from-scratch-9cc0007da372