Questions1. Typically, however, a smaller or lower value for the RSS is ideal in any model since it means there's less variation in the data set. In other words, **the lower the sum of squared residuals, the better the regression model is at explaining the data**

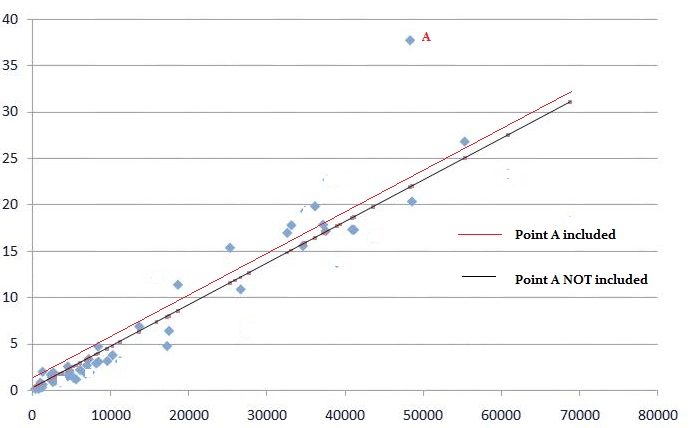
**Question2**

**RSS-**The residual sum of squares (RSS) is a statistical technique used to measure the amount of [variance](https://www.investopedia.com/terms/v/variance.asp) in a data set that is not explained by a regression model itself. Instead, it estimates the variance in the residuals, or [error term](https://www.investopedia.com/terms/e/errorterm.asp).

[Linear regression](https://www.investopedia.com/terms/r/regression.asp) is a measurement that helps determine the strength of the relationship between a dependent variable and one or more other factors, known as independent or explanatory variables.

**ESS-** **explained sum of squares** (**ESS**), alternatively known as the **model sum of squares** or **sum of squares due to regression** (**SSR** – not to be confused with the [residual sum of squares](https://en.wikipedia.org/wiki/Residual_sum_of_squares) (RSS) or sum of squares of errors), is a quantity used in describing how well a model, often a [regression model](https://en.wikipedia.org/wiki/Regression_analysis), represents the data being modelled. In particular, the explained sum of squares measures how much variation there is in the modelled values and this is compared to the [total sum of squares](https://en.wikipedia.org/wiki/Total_sum_of_squares) (TSS), which measures how much variation there is in the observed data, and to the [residual sum of squares](https://en.wikipedia.org/wiki/Residual_sum_of_squares), which measures the variation in the error between the observed data and modelled values.

TSS-he total sum of squares (TSS or SST) **tells you how far the data points in a dataset are from the center**. It's a descriptive statistic called a measure of spread or dispersion. Dividing the TSS by the number of observations in the dataset gives you the average variability within the data, which is called the variance.



Questionsn-3=Regularization consists of different techniques and methods used to address the issue of over-fitting by reducing the generalization error without affecting the training error much. Choosing overly complex models for the training data points can often lead to overfitting. On the other hand, a simpler model leads to underfitting the data. Hence choosing just the right amount of complexity in the model is critical. Since the complexity of the model can not be directly inferred from the available training data, it is often impossible to stumble upon the right model complexity for training

**Types of Regularization**

## Modify loss function

## L2 Regularization (strong):

## b.L1 Regularization (strong):

## c. Entropy Regularization (strong):

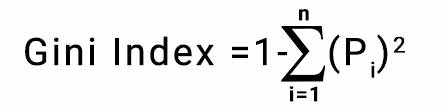
## Modify sampling method

## a . Data Augmntation (weak):

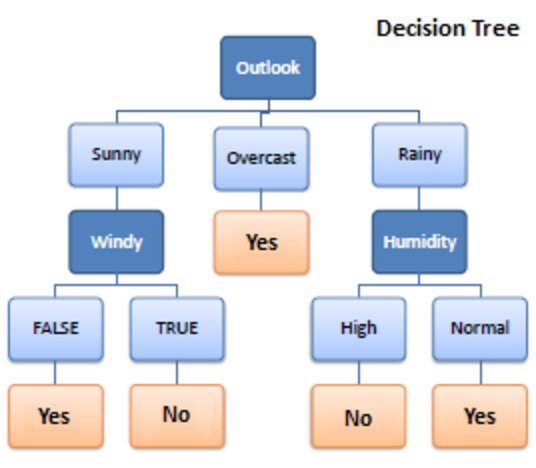
## b. K-fold Cross-Validation (medium):

## 3. Modify training algorithm

Questions-4Gini Index, also known as Gini impurity, **calculates the amount of probability of a specific feature that is classified incorrectly when selected randomly.**If all the elements are linked with a single class then it can be called pure.



Question-5-Decision trees are a type of model used for both classification and regression. Trees answer sequential questions which send us down a certain route of the tree given the answer. The model behaves with “if this than that” conditions ultimately yielding a specific result. This is easy to see with the image below which maps out whether or not to play golf.

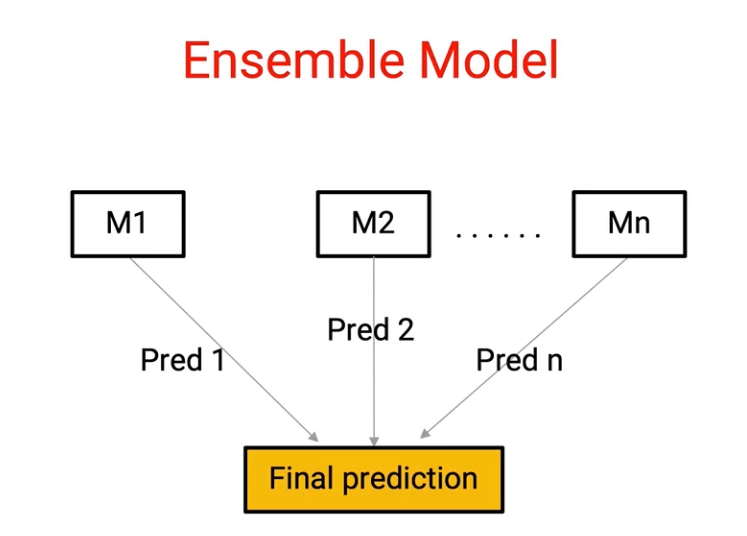


The flow of this tree works downward beginning at the top with the outlook. The outlook has one of three options: sunny, overcast, or rainy. If sunny, we travel down to the next level. Will it be windy? True or false? If true, we choose not to play golf that day. If false we choose to play. If the outlook was changed to overcast, we would end there and decide to play. If the outlook was rainy, we would then look at the humidity. If the humidity was high we would not play, if the humidity is normal we would play.

Tree depth is an important concept. This represents how many questions are asked before we reach our predicted classification. We can see that the deepest the tree gets in the example above is two. The sunny and rainy routes both have a depth of two. The overcast route only has a depth of one, although the overall tree depth is denoted by its longest route. Thus, this tree has a depth of two.

Question6-Ensembling is nothing but the technique to combine several individual predictive models to come up with the final predictive model. And in this article, we’re going to look at some of the ensembling techniques for both Classification and Regression problems such as Maximum voting, Averaging, Weighted Averaging, and Rank Averaging.

Note: If you are more interested in learning concepts in an Audio-Visual format, We have this entire article explained in the video below. If not, you may continue reading.



Here we have M1 to Mn individual models which are giving predictions from Pred 1 to Pred n respectively. And we combine them to come up with a single model. Now, the question is, how do we do this combination and there are several techniques to do this, and this is where the art of ensembling comes into the picture. So let’s start by understanding some of these basic techniques.

And in order to understand them, we’ll be using the Titanic survival example, where we have details about individual passengers who were traveling on Titanic and we have to predict which customer would survive the Titanic tragedy

Question7-Bagging is a method of merging the same type of predictions. Boosting is a method of merging different types of predictions.

Bagging decreases variance, not bias, and solves over-fitting issues in a model. Boosting decreases bias, not variance.

In Bagging, each model receives an equal weight. In Boosting, models are weighed based on their performance.

Models are built independently in Bagging. New models are affected by a previously built model’s performance in Boosting.

In Bagging, training data subsets are drawn randomly with a replacement for the training dataset. In Boosting, every new subset comprises the elements that were misclassified by previous models.

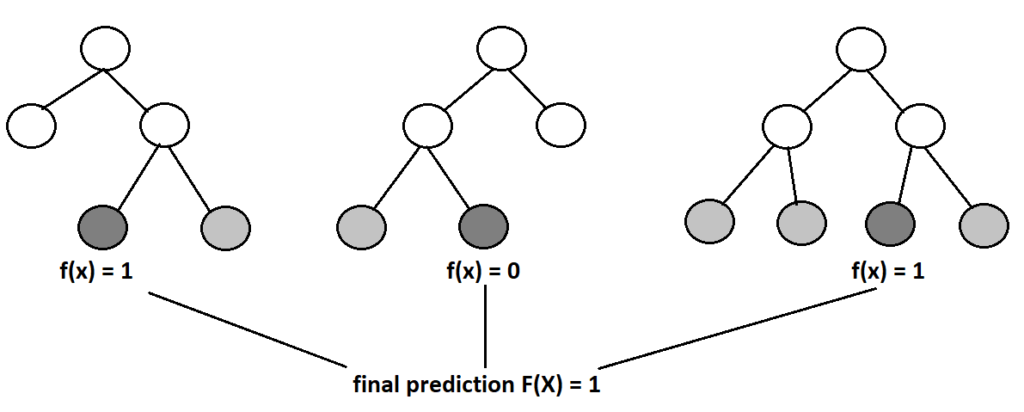
Bagging is usually applied where the classifier is unstable and has a high variance. Boosting is usually applied where the classifier is stable and simple and has high bias.

QUESTION-8**Out-of-bag** (**OOB**) **error**, also called **out-of-bag estimate**, is a method of measuring the [prediction error](https://en.wikipedia.org/wiki/Prediction_error) of [random forests](https://en.wikipedia.org/wiki/Random_forest), [boosted decision trees](https://en.wikipedia.org/wiki/Gradient_boosting), and other [machine learning](https://en.wikipedia.org/wiki/Machine_learning) models utilizing [bootstrap aggregating](https://en.wikipedia.org/wiki/Bootstrap_aggregating) (bagging). Bagging uses subsampling with replacement to create training samples for the model to learn from. OOB error is the mean prediction error on each training sample *xi*, using only the trees that did not have *xi* in their bootstrap sample.[[1]](https://en.wikipedia.org/wiki/Out-of-bag_error#cite_note-islr-1)

[Bootstrap aggregating](https://en.wikipedia.org/wiki/Bootstrap_aggregating) allows one to define an out-of-bag estimate of the prediction performance improvement by evaluating predictions on those observations that were not used in the building of the next base learner.

-Random forest is an ensemble learning method used for classification and regression. It relies on a set of decision trees to create a diverse prediction model. This algorithm is one of the most popular machine learning algorithms today because it offers robust accuracy, fast training time, and ease of interpretation.

**In the first place, random forest uses the ensemble technique called bagging.** Bagging is a technique for reducing variance when training a model on noisy datasets by creating multiple models so that each one compensates for the errors of the others. Basically, it means that a random forest is constructed of many decision trees, and the final prediction is calculated as the majority vote or average prediction of the trees:



## Out-of-bag Error in Random Forests

Generally, in machine learning and data science, it is crucial to create a trustful system that will work well with the new, unseen data. Overall, there are a lot of different approaches and methods to achieve this generalization. Out-of-bag error is one of these methods for validating the machine learning model.

### Definition

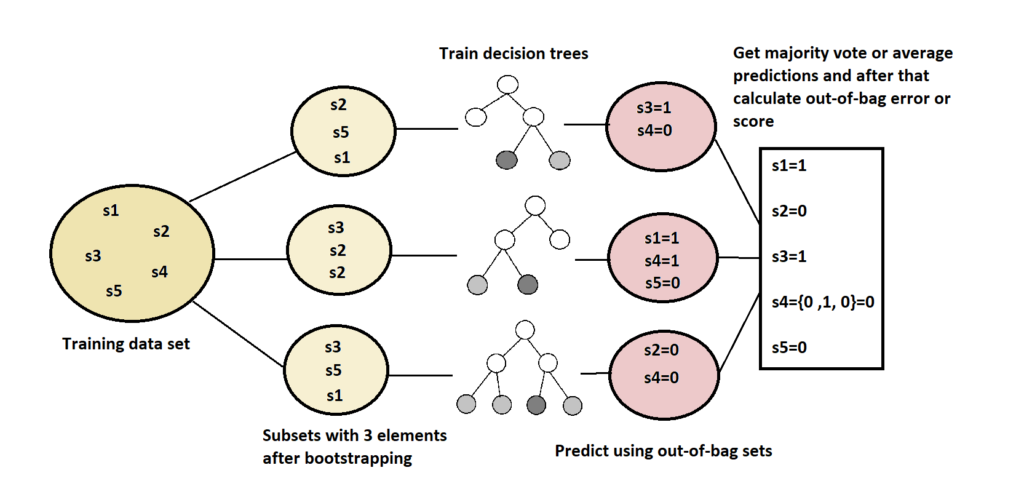
This approach utilizes the usage of bootstrapping in the random forest. Since the bootstrapping samples the data with the possibility of selecting one sample multiple times, it is very likely that we won’t select all the samples from the original data set. Therefore, one smart decision would be to exploit somehow these unselected samples, called out-of-bag samples.

Correspondingly, the error achieved on these samples is called out-of-bag error. What we can do is to use out-of-bag samples for each decision tree to measure its performance. This strategy provides reliable results in comparison to other validation techniques such as train-test split or cross-validation.

### Probability of Out-of-bag Sample

Theoretically, with the quite big data set and the number of sampling, it is expected that out-of-bag error will be calculated on 36% of the training set. To prove this, consider that our training set has  samples. Then, the probability of selecting one particular sample from the training set is .

Similarly, the probability of not selecting one particular sample is . Since we select the bootstrap samples with replacement, the probability of one particular sample not being selected  times is equal to . Now, if the number  is pretty big or if it tends to infinity, we’ll get a limit below



## 4. Conclusion

In this article, we’ve described one efficient way of measuring error on the random forest model. The out-of-bag method is empirically tested in many papers, and it provides approximately the same model performance measurements as the test set. Moreover, this method might be even more convenient than simple train-test validation because we don’t need to divide our data into training and test.

This way, for smaller data sets, the machine learning algorithm will have more data for training.

QUESTION-9

Let’s say that you have trained a machine learning model. Now, you need to find out how well this model performs. Is it accurate enough to be used? How does it compare to another model? There are several evaluation methods to determine this. One such method is called K-fold cross validation.

Cross validation is an evaluation method used in machine learning to find out how well your machine learning model can predict the outcome of unseen data. It is a method that is easy to comprehend, works well for a limited data sample and also offers an evaluation that is less biased, making it a popular choice.

The data sample is split into ‘k’ number of smaller samples, hence the name: K-fold Cross Validation. You may also hear terms like four fold cross validation, or ten fold cross validation, which essentially means that the sample data is being split into four or ten smaller samples respectively.

# How is k-fold cross validation performed?

The general stratergy is quite straight forward and the following steps can be used:

1. First, shuffle the dataset and split into k number of subsamples. (It is important to try to make the subsamples equal in size and ensure k is less than or equal to the number of elements in the dataset).
2. In the first iteration, the first subset is used as the test data while all the other subsets are considered as the training data.
3. Train the model with the training data and evaluate it using the test subset. Keep the evaluation score or error rate, and get rid of the model.
4. Now, in the next iteration, select a different subset as the test data set, and make everything else (including the test set we used in the previous iteration) part of the training data.

# How to determine the best value for ‘k’ in K-Fold Cross Validation?

Chosing a good value for k is important. A poor value for k can result in a poor evaluation of the model’s abilities. In other words, it can cause the measured ability of the model to be overestimated (high bias) or change widely depending on the training data used (high variance).

Generally, there are three ways to select k:

* Let k = 5, or k =10. Through experimentation, it has been found that selecting k to be 5 or 10 results in sufficiently good results.
* Let k = n, where n is the size of the dataset. This ensures each sample is used in the test data set.
* Another way is to chose k so that every split data sample is sufficiently large, ensuring they are statistically represented in the larger dataset.

QUESTION10-A Machine Learning model is defined as a mathematical model with a number of parameters that need to be learned from the data. By training a model with existing data, we are able to fit the model parameters.   
However, there is another kind of parameter, known as ***Hyperparameters***, that cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn.

Some examples of model hyperparameters include:

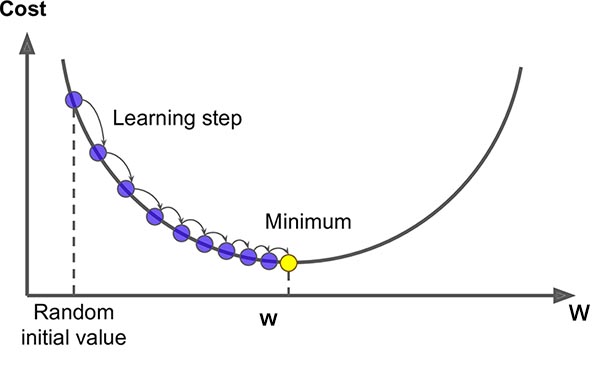
1. The penalty in Logistic Regression Classifier i.e. L1 or L2 regularization
2. The learning rate for training a neural network.
3. The C and sigma hyperparameters for support vector machines.
4. The k in k-nearest neighbors.

The aim of this article is to explore various strategies to tune hyperparameters for Machine learning models.

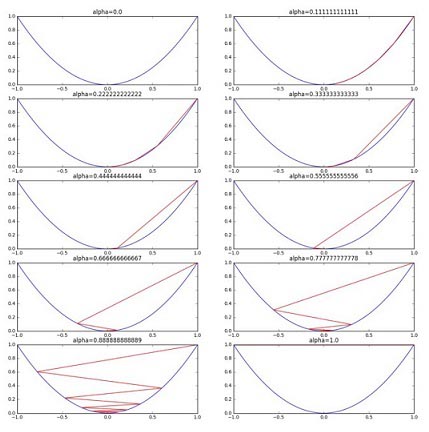
Models can have many hyperparameters and finding the best combination of parameters can be treated as a search problem. The two best strategies for Hyperparameter tuning are:

Hyperparameter tuning takes advantage of the processing infrastructure of Google Cloud **to test different hyperparameter configurations when training your model**. It can give you optimized values for hyperparameters, which maximizes your model's predictive accuracy.

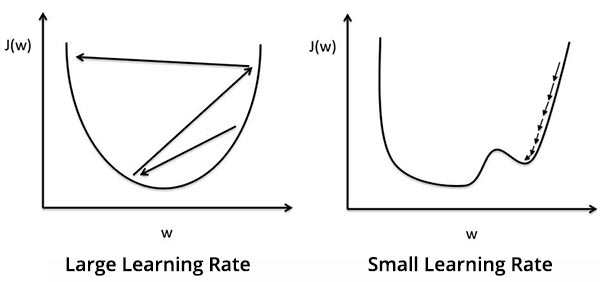
QUESTION11-The process of repeatedly nudging an input of a function by some multiple of the negative gradient is called gradient descent. It’s a way to converge towards some local minimum of a cost function basically valley in a graph.  According to Wikipedia, **Gradient descent** is a first-order iterative optimization algorithm for finding the minimum of a function. To find a local minimum or minimum cost of a function using gradient descent, one takes steps proportional to the negative of the **gradient** of the function at the current point.



Gradient Descent is a simple optimization technique that could be used in many machine learning problems. It involves reducing the cost function. The cost function could be anything like least square methods, cross entropy. The **cost function** is the relation between calculated output and actual output. Here’s the image from [an article](https://medium.com/@balamuralim.1993/importance-of-learning-rate-in-machine-learning-920a323fcbfb) of balamulari in the medium. In this experiment, balamulari is trying to find the minimum cost of function x^2.  Here, he started from (1,1) to well known minimum cost  (0,0) in the figure. He has tried various learning rate i.e. alpha = 0, 0.11111, 0.22222, 0.33333, 0.44444, 0.55556, 0.66667, 0.77778, 0.88889, 1.  At alpha = 0 there is no weight update and at alpha = 1 gets struck (it never converges as it hit the other end (-1,1)) . With alpha with 0.11111 converges very slowly and alpha with 0.66667 the fastest. And, alpha with 0.88889 oscillates several times before reaching the optimal point (0,0). If you want to learn more please visit the above mention link.

[](https://saugatbhattarai.com.np/wp-content/uploads/2018/06/learning-rate-in-gradient-descent.jpg)Fig: learning rate in cost function = x^2

Learning rate has greater importance in gradient descent. Choosing the very small learning rate leads to many iterations until convergence and the possibility of **trapping in local minima**. And choosing too large learning rate could result to **overshoot** the optimum value. Below image gives you some insight into what I mean to say.

[](https://saugatbhattarai.com.np/wp-content/uploads/2018/06/learning-rate-gradient-descent.jpg)

*In short, we use the gradient descent to find out the least cost for the function.*

#### ****ALGORITHM FOR GRADIENT DESCENT****

An algorithm for gradient descent: (repeat until convergence)

[gradient descent algorithm saugat](https://saugatbhattarai.com.np/wp-content/uploads/2018/06/gradient-descent-algo.png)

Question 12=It can only be used to predict discrete functions. Hence, the dependent variable of Logistic Regression is bound to the discrete number set. It is very fast at classifying unknown records. **Non-linear problems can't be solved with logistic regression because it has a linear decision surface**.

Logistic Regression has traditionally been used as a linear classifier, i.e. when the classes can be separated in the feature space by linear boundaries. That can be remedied however if we happen to have a better idea as to the shape of the decision boundary…

Logistic regression is known and used as a linear classifier. It is used to come up with a hyperplane in feature space to separate observations that belong to a class from all the other observations that do not belong to that class. The decision boundary is thus linear. Robust and efficient implementations are readily available (e.g. scikit-learn) to use logistic regression as a linear classifier

Questions-13

### ****The Comparison****

#### ****Loss Function:****

The technique of Boosting uses various loss functions. In case of Adaptive Boosting or AdaBoost, it minimises the exponential loss function that can make the algorithm sensitive to the outliers. With Gradient Boosting, any differentiable loss function can be utilised. Gradient Boosting algorithm is more robust to outliers than AdaBoost.

#### ****Flexibility****

AdaBoost is the first designed boosting algorithm with a particular loss function. On the other hand, Gradient Boosting is a generic algorithm that assists in searching the approximate solutions to the additive modelling problem. This makes Gradient Boosting more flexible than AdaBoost.

#### ****Benefits****

AdaBoost minimises loss function related to any classification error and is best used with weak learners. The method was mainly designed for binary classification problems and can be utilised to boost the performance of decision trees. Gradient Boosting is used to solve the differentiable loss function problem. The technique can be used for both classification and regression problems.

#### ****Shortcomings****

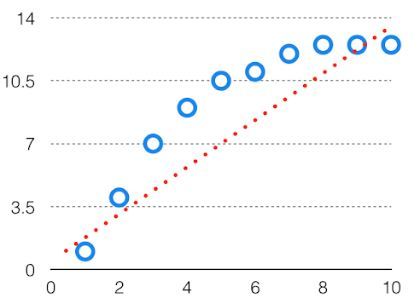
In the case of Gradient Boosting, the shortcomings of the existing weak learners can be identified by gradients and with AdaBoost, it can be identified by high-weight data points.

### ****Wrapping Up****

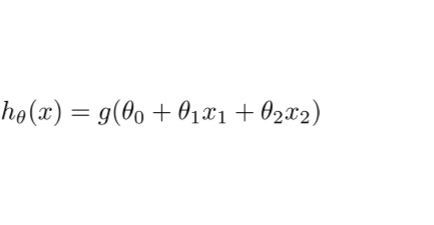
Though there are several differences between the two boosting methods, both the algorithms follow the [same path](https://arxiv.org/pdf/1403.1452.pdf) and share similar historic roots. Both the algorithms work for boosting the performance of a simple base-learner by iteratively shifting the focus towards problematic observations that are challenging to predict.

In the case of AdaBoost, the shifting is done by up-weighting observations that were misclassified before, while Gradient Boosting identifies the difficult observations by large residuals computed in the previous iterations

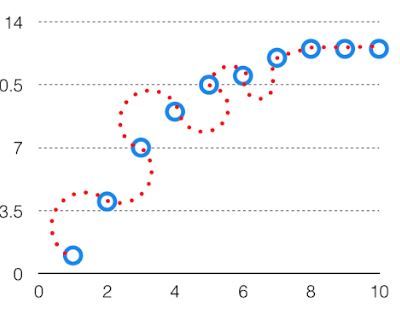
Questions14-**Bias**  
The bias is known as the difference between the prediction of the values by the ML model and the correct value. Being high in biasing gives a large error in training as well as testing data. Its recommended that an algorithm should always be low biased to avoid the problem of underfitting.  
By high bias, the data predicted is in a straight line format, thus not fitting accurately in the data in the data set. Such fitting is known as **Underfitting of Data**. This happens when the hypothesis is too simple or linear in nature. Refer to the graph given below for an example of such a situation.



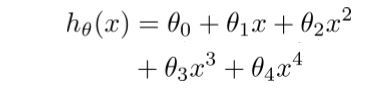
*HighBias*

In such a problem, a hypothesis looks like follows.  
  
**Variance**  
The variability of model prediction for a given data point which tells us spread of our data is called the variance of the model. The model with high variance has a very complex fit to the training data and thus is not able to fit accurately on the data which it hasn’t seen before. As a result, such models perform very well on training data but has high error rates on test data.  
When a model is high on variance, it is then said to as **Overfitting of Data**. Overfitting is fitting the training set accurately via complex curve and high order hypothesis but is not the solution as the error with unseen data is high.  
While training a data model variance should be kept low.

The high variance data looks like follows.

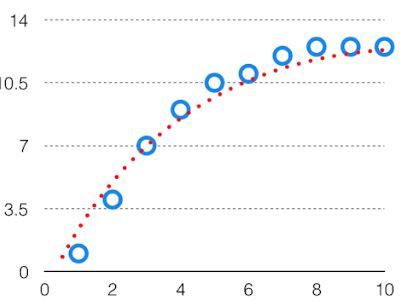


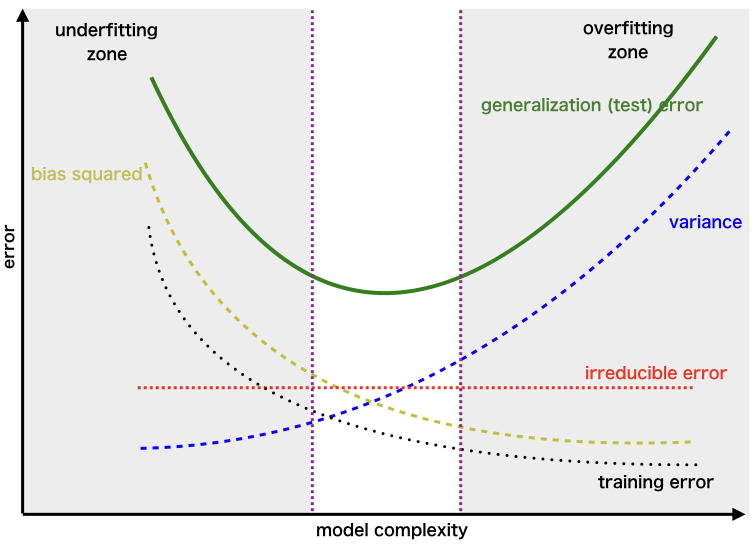
*High Variance*

In such a problem, a hypothesis looks like follows.  
  
**Bias Variance Tradeoff**

If the algorithm is too simple (hypothesis with linear eq.) then it may be on high bias and low variance condition and thus is error-prone. If algorithms fit too complex ( hypothesis with high degree eq.) then it may be on high variance and low bias. In the latter condition, the new entries will not perform well. Well, there is something between both of these conditions, known as Trade-off or Bias Variance Trade-off.

This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can’t be more complex and less complex at the same time. For the graph, the perfect tradeoff will be like.

  
The best fit will be given by hypothesis on the tradeoff point.

The error to complexity graph to show trade-off is given as –  
  
This is referred to as the best point chosen for the training of the algorithm which gives low error in training as well as testing data.

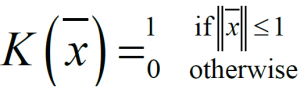
## Question 15-SVM Kernel Functions

SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types. For example**linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.**  
Introduce Kernel functions for sequence data, graphs, text, images, as well as vectors. The most used type of kernel function is **RBF.** Because it has localized and finite response along the entire x-axis.  
The kernel functions return the inner product between two points in a suitable feature space. Thus by defining a notion of similarity, with little computational cost even in very high-dimensional spaces.

‘

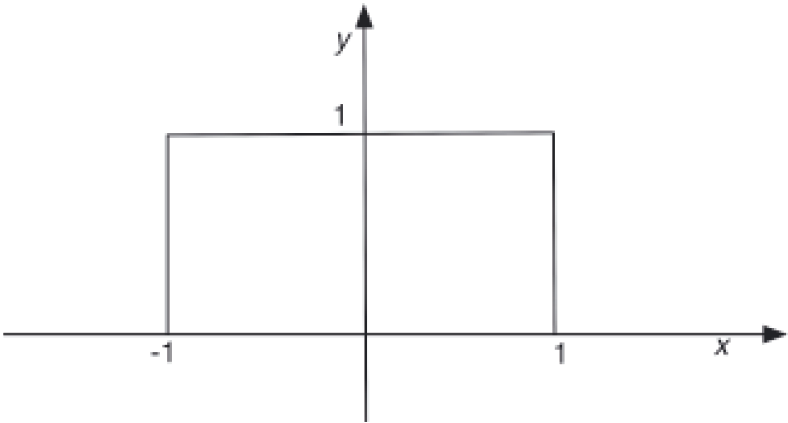
## Kernel Rules

Define kernel or a window function as follows:

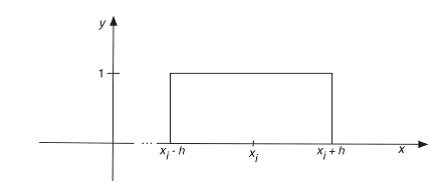
[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2017/08/kernal-rule.png)

*Kernel or a window function*

This value of this function is 1 inside the closed ball of radius 1 centered at the origin, and 0 otherwise . As shown in the figure below:



For a fixed xi, the function is K(z-xi)/h) = 1 inside the closed ball of radius h centered at xi, and 0 otherwise as shown in the figure below:



So, by choosing the argument of K(·), you have moved the window to be centered at the point xi and to be of radius h.

### Polynomial kernel

It is popular in image processing.

Polynomial kernel equation

### Gaussian kernel

It is a general-purpose kernel; used when there is no prior knowledge about the data. Equation is:

[Gaussian kernel equation](https://data-flair.training/blogs/wp-content/uploads/sites/2/2017/08/gaussian-kernel.png)

*Gaussian kernel equation*

### Gaussian radial basis function (RBF)

It is a general-purpose kernel; used when there is no prior knowledge about the data.  
Equation is:

[Gaussian radial basis function (RBF)](https://data-flair.training/blogs/wp-content/uploads/sites/2/2017/08/gaussian-radial-basis-function-RBF.png)

*Gaussian radial basis function (RBF)*

, for:

[Gaussian radial basis function (RBF)](https://data-flair.training/blogs/wp-content/uploads/sites/2/2017/08/gaussian-radial-basis-function-RBF-1.png)

*Gaussian radial basis function (RBF)*

Sometimes parametrized using:

[Gaussian radial basis function (RBF)](https://data-flair.training/blogs/wp-content/uploads/sites/2/2017/08/gaussian-radial-basis-function-RBF-2.png)

*Gaussian radial basis function (RBF)*