**Ans1**. The kernel is a way of computing the dot product of two vectors x and y in some (very high dimensional) feature space, which is why kernel functions are sometimes called “generalized dot product.”

The three major kernel of svm are

1.Linear-**This kernel is used when data is linearly separable that is, it can be separated using a single Line.** It is mostly used when there are a Large number of Features in a particular Data Set. One of the examples where there are a lot of features, is **Text Classification**, as each alphabet is a new feature. So we mostly use Linear Kernel in Text Classification.

2.**Gaussian RBF(Radial Basis Function) is another popular Kernel method used in SVM models for more. RBF kernel is a function whose value depends on the distance from the origin or from some point**. Gaussian Kernel is of the following format;

Image for post

Image for post

||X1 — X2 || = Euclidean distance between X1 & X2

Using the distance in the original space we calculate the dot product (similarity) of X1 & X2.

Note: similarity is the angular distance between two points.

Parameters:

C: Inverse of the strength of regularization.

Behavior: As the value of ‘c’ increases the model gets overfits.

As the value of ‘c’ decreases the model underfits.

2. γ : Gamma (used only for RBF kernel)

Behavior: As the value of ‘ γ’ increases the model gets overfits.

As the value of ‘ γ’ decreases the model underfits.

3.**Polynomial Kernel: It represents the similarity of vectors in training set of data in a feature space over polynomials of the original variables used in kernel**.

In general, the polynomial kernel is defined as ;

Image for post

Image for post

b = degree of kernel & a = constant term.

in the polynomial kernel, we simply calculate the dot product by increasing the power of the kernel.

**Ans.2** **R-squared evaluates the scatter of the data points around the fitted regression line**. It is also called the coefficient of determination, or the coefficient of multiple determination for multiple regression. For the same data set, higher R-squared values represent smaller differences between the observed data and the fitted values. R-squared is always between 0 and 100%:

0% represents a model that does not explain any of the variation in the response variable around its mean. The mean of the dependent variable predicts the dependent variable as well as the regression model.

100% represents a model that explains all of the variation in the response variable around its mean.

{\displaystyle R^2 = \frac {\text{Variance explained by the model}}{\text{Total variance}}}

**So,r squared is a goodness-of-fit measure for linear regression models**. This statistic indicates the percentage of the variance in the dependent variable that the independent variables explain collectively. R-squared measures the strength of the relationship between your model and the dependent variable on a convenient 0 – 100% scale..

**Whereas , The residual sum of squares measures the amount of error remaining between the regression function and the data set**. A smaller residual sum of squares figure represents a regression function. Residual sum of squares–also known as the sum of squared residuals–essentially determines how well a regression model explains or represents the data in the model, Ideally, the sum of squared residuals should be a smaller or lower value in any regression model.

The residual sum of squares (RSS) is the sum of the squared distances between your actual versus your predicted values:

RSS=∑i=1n(yi−y^i)2

**Standalone residual sum of squares doesn’t provide valuable information**.

**Ans.3** Total SS is related to the total sum and explained sum with the following formula:  
**Total SS = Explained SS + Residual Sum of Squares.**

Total sum of squares

**The Total SS (TSS or SST) tells you how much variation there is in the dependent variable**.  
Total SS = Σ(Yi – mean of Y)2.. Sum of squares is a measure of how a data set varies around a central number (like the mean).

sum of squares in regression uses the equation:  
[ss](https://www.statisticshowto.com/wp-content/uploads/2015/04/ss.png)

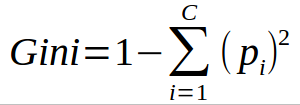
Explained sum of squares,

**The Explained SS tells you how much of the variation in the dependent variable your model explained**.  
Explained SS = Σ(Y-Hat – mean of Y)2.

Residual sum of squares

**The residual sum of squares tells you how much of the dependent variable’s variation your model did not explain.** It is the sum of the squared differences between the actual Y and the predicted Y:  
Residual Sum of Squares = Σ e2

**Ans.4** **Gini index or Gini impurity measures the degree or probability of a particular variable being wrongly classified when it is randomly chosen**. The degree of Gini index varies between 0 and 1, where 0 denotes that all elements belong to a certain class or if there exists only one class, and 1 denotes that the elements are randomly distributed across various classes. A Gini Index of 0.5 denotes equally distributed elements into some classes.



where pi  is the probability of an object being classified to a particular class.

While building the decision tree, we would prefer choosing the attribute/feature with the least Gini index as the root node.

**Ans.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. Tree depth is an important concept. This represents how many questions are asked before we reach our predicted classification.  **Building decision trees require algorithms capable of determining an optimal choice at each node, this is a greedy model**, meaning it makes the most optimal decision at each step, but does not take into account the global optimum. **Decision trees are prone to overfitting, especially when a tree is particularly deep. This is due to the amount of specificity we look at leading to smaller sample of events that meet the previous assumptions**. This small sample could lead to unsound conclusions.**So we can conclude that yes unregularized decision-trees are prone to overfitting**.

**Ans.6** **Ensemble technique is a machine learning technique that combines several base models in order to produce one optimal predictive model.**For example, When making Decision Trees, there are several factors we must take into consideration: On what features do we make our decisions on? What is the threshold for classifying each question into a yes or no answer? This is where Ensemble Methods comes to rescue, Rather than just relying on one Decision Tree and hoping we made the right decision at each split, Ensemble Methods allow us to take a sample of Decision Trees into account, calculate which features to use or questions to ask at each split, and make a final predictor based on the aggregated results of the sampled Decision Trees.**So, rather than making one model and hoping this model is the best/most accurate predictor we can make, ensemble methods take a myriad of models into account, and average those models to produce one final model**.

**Ans.7** **Bagging and boosting are ensemble methods.the ensemble takes part in a bigger group of methods,**called multiclassifiers,where a set of hundreds or thousands of learners with a common objective are fused together to solve the problem. The main cause of error in learning are due to noise,bias and variance.Ensemble helps to minimize these factors.These methods are designed to improve the stability and accuracy of machine learning algorithms.**Bagging and boosting get N learners by generating additional data in the training stage**.N new training data sets are produced by random sampling with replacement from the original set.By sampling with replacement some observations may be repeated in each new training data set.In case of bagging,any element has the same probability to appear in new data set.However ,for boosting the observations are weighted and therefore some of them will take part in the new sets more often

**The main difference between the two methods is that while the training stage is parallel for bagging (i.e each model is build independently).Boosting builds the new learner in a sequential way.**Now coming to classification stage,**In boosting algorithms each classifier is trained on data,taking account the previous classifier’s success.**After each training step,the weights are redistributed.Misclassified data increases it’s weights to emphasise the most difficult cases.

**However,Boosting assigns a second set of weights,this time for the N classifiers,In order to take a weighted average of their estimates**.If the problem is that the single model gets a very low performance **,Bagging will rarely get a better bias .However boosting could generate a combined model with lower errors as it is optimises the advantages and reduces pitfalls of the single model**.By contrast ,if the difficulty of the single model is over fitting,in fact this technique is faced with this problem itself.For this reason ,Bagging is effective more than Boosting.

To sum up,

1.**the main difference between bagging and boosting is that in bagging each model is build independently,whereas boosting tries to add new models that do well where previous models** fail.(Bagging is a parallel technique ,boosting is a sequential technique)

2.**Only boosting determines weights for the data to tip the scales in favour of most difficult cases unlike bagging**  where there is equally weighted average

3.**Boosting tries to reduce bias,while bagging solves over fitting problem**.

**Ans.8** Random Forests algorithm is a classifier based on primarily two methods:

Bagging

Random subspace method.

If we take S number of trees in our random forest algorithm. Then we first create S datasets of "the same size as original" created from random resampling of data in T with-replacement (n times for each dataset). This will result in {T1, T2, ... TS} datasets. Each of them is called a bootstrap dataset.

Due to the "with-replacement" parameter, every dataset Ti can have duplicate data records and Ti can be missing several data records from original datasets. This is called Bootstrapping.

Bagging is the process of taking bootstraps & then aggregating the models learned on each bootstrap.

Random Forest creates an S number of trees and uses m (=sqrt(M) or =floor(lnM+1)) random subfeatures out of M possible features to build any tree. This is called a random subspace method.

So for each Ti bootstrap dataset, you create a tree, Ki. You can classify some input data D = {x1, x2, ..., xM} you can let it pass through each tree and produce S outputs which can be denoted by Y = {y1, y2, ..., ys}. The final prediction is a majority vote on this set.

Out-of-bag error:

**After building the classifiers (S trees), for each (Xi,yi) in the original training set i.e. T, select all Tk which does not include (Xi,yi). This subset, is a set of bootstrap datasets which do not contain a particular record from the original dataset. This set is called out-of-bag examples. There are n such subsets (one for each data record in original dataset T). OOB classifier is the aggregation of votes ONLY over Tk such that it does not contain (xi,yi).**

The out-of-bag estimate for the generalization error is the error rate of the out-of-bag classifier on the training set (compare it with known yi's).

The study of error estimates for bagged classifiers gives empirical evidence to show that the out-of-bag estimate is as accurate as using a test set of the same size as the training set. Therefore, using the out-of-bag error estimate removes the need for a set-aside test set.

**Thus, the out-of-bag (OOB) error is the average error for each zi calculated using predictions from the trees that do not contain zi in their respective bootstrap sample.** This allows the RandomForestClassifier to be fit and validated whilst being trained.

**Ans.9** **Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.**

**The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation**. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

**Ans.10** 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 parameters, 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.

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

GridSearchCV

RandomizedSearchCV

GridSearchCV  
In GridSearchCV approach, machine learning model is evaluated for a range of hyperparameter values. This approach is called GridSearchCV, because it searches for best set of hyperparameters from a grid of hyperparameters values.

RandomizedSearchCV  
RandomizedSearchCV solves the drawbacks of GridSearchCV, as it goes through only a fixed number of hyperparameter settings. It moves within the grid in random fashion to find the best set hyperparameters. This approach reduces unnecessary computation.

**So we can conclude Machine learning algorithms have hyperparameters that allow you to tailor the behavior of the algorithm to your specific dataset.(this is known as hyper parameter tuning)**

**Hyperparameters are different from parameters, which are the internal coefficients or weights for a model found by the learning algorithm. Unlike parameters, hyperparameters are specified by the practitioner when configuring the model.**

**Ans.11** The learning rate hyperparameter controls the rate or speed at which the model learns. Specifically, it controls the amount of apportioned error that the weights of the model are updated with each time they are updated, such as at the end of each batch of training examples. **Generally, a large learning rate allows the model to learn faster, at the cost of arriving on a sub-optimal final set of weights**. A smaller learning rate may allow the model to learn a more optimal or even globally optimal set of weights but may take significantly longer to train. **At extremes, a learning rate that is too large will result in weight updates that will be too large and the performance of the model (such as its loss on the training dataset) will oscillate over training epochs. Oscillating performance is said to be caused by weights that diverge (are divergent).**

**Ans.12** Bias is the simplifying assumptions made by the model to make the target function easier to approximate.

Variance is the amount that the estimate of the target function will change given different training data.

**Trade-off is tension between the error introduced by the bias and the variance**.

The parameterization of machine learning algorithms is often a battle to balance out bias and variance.

Below are two examples of configuring the bias-variance trade-off for specific algorithms:

The k-nearest neighbors algorithm has low bias and high variance, but the trade-off can be changed by increasing the value of k which increases the number of neighbors that contribute t the prediction and in turn increases the bias of the model.

The support vector machine algorithm has low bias and high variance, but the trade-off can be changed by increasing the C parameter that influences the number of violations of the margin allowed in the training data which increases the bias but decreases the variance.

**Ans.13** **Regularisation is a technique used to reduce the errors by fitting the function appropriately on the given training set and avoid overfitting**.  
The commonly used regularisation techniques are :

L1 regularisation

L2 regularisation

Dropout regularization

**Once the regression model is built and one of the following symptoms happen, we could apply one of the regularization techniques**.(use of regularization)

**Model lack of generalization**: Model found with higher accuracy fails to generalize on unseen or new data.

**Model instability**: Different regression models can be created with different accuracies. It becomes difficult to select one of them.

**Ans.14** **The basic idea of boosting (an ensemble learning technique) is to combine several weak learners into a stronger one. The general idea of boosting algorithms is to try predictors sequentially, where each subsequent model attempts to fix the errors of its predecessor**.

**In Adaboost, ‘shortcomings’ are identified by high-weight data points.  
  
In Gradient Boosting, ‘shortcomings’ (of existing weak learners) are identified by gradients**.  
  
**Adaboost is more about ‘voting weights’ and Gradient boosting is more about ‘adding gradient optimization’.**  
Adaboost increases the accuracy by giving more weightage to the target which is misclassified by the model. At each iteration, Adaptive boosting algorithm changes the sample distribution by modifying the weights attached to each of the instances. It increases the weights of the wrongly predicted instances and decreases the ones of the correctly predicted instances.  
  
Gradient boosting calculates the gradient (derivative) of the Loss Function with respect to the prediction (instead of the features). Gradient boosting increases the accuracy by minimizing the Loss Function (error which is difference of actual and predicted value) and having this loss as target for the next iteration.  
  
Gradient boosting algorithm builds first weak learner and calculates the Loss Function. It then builds a second learner to predict the loss after the first step. The step continues for third learner and then for fourth learner and so on until a certain threshold is reached.

**Ans.15** **No,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 to come up with a hyperplane that separates the feature space into classes.**

In very rare cases when we suspect that the decision boundary is nonlinear we may get better results by attempting some nonlinear functional forms for the logit function. Solving for the model parameters can be more challenging but the optimization modules in scipy can help.