MACHINE LEARNING
Worksheet Set - 5

Q11 to Q15 are subjective answer type questions, Answer them briefly.

1. R-squared or Residual Sum of Squares (RSS) which one of these two is a better measure of goodness of fit model in regression and why?

ANS: While working on a Linear Regression model and the R^2 is 0.89 which tells me my regression line is a good fit. When I calculated the residual sum squared, the number is around 4060, which is telling me the difference between the predicted and actual values are large.

Do the values R^2 = 0.89 and RSS = 4060 contradict each other

No, these numbers do not contradict each other. It sounds like you have an understanding of what R2 means: it represents the proportion of the variance in your data which is explained by your model; the closer to one, the better the fit.
The residual sum of squares (RSS) is the sum of the squared distances between your actual versus your predicted values:

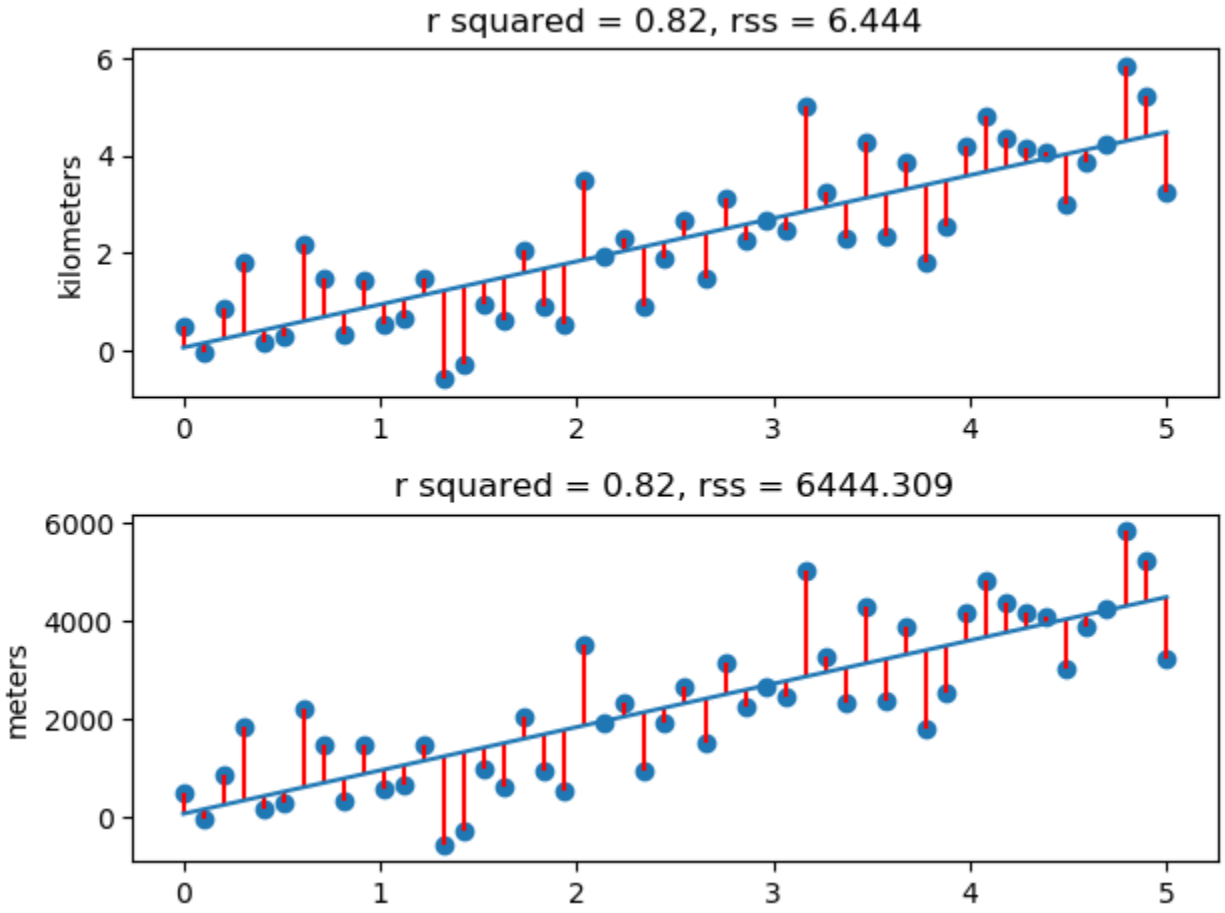
$$RSS=\sum_{i=1n}(y_i-y\hat{}_i)_2$$

Where yi is a given datapoint and y^i is your fitted value for yi.

The actual number you get depends largely on the scale of your response variable. Taken alone, the RSS isn't so informative.
Example:
Picture your residuals as a vertical line connecting your actual values to your predicted value (red traces in the plot below).

You can imagine that if your y-axis is on a different scale, the number you get will be very different.

For instance, consider that your y-axis were kilometers, and a given point is about 0.5km away from your line of best fit. Then, the residual on that given datapoint is 0.5. However, if your scale is meters, then that same datapoint has a residual of 500. Your RSS will be much larger, but the fit does not change at all; in fact the data don't change at all either. But the RSS changes drastically.



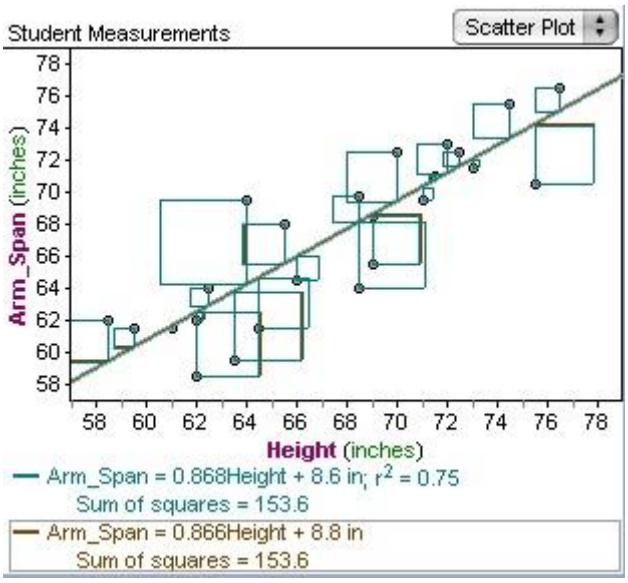
2. What are TSS (Total Sum of Squares), ESS (Explained Sum of Squares) and RSS (Residual Sum of Squares) in regression. Also mention the equation relating these three metrics with each other. ?

Ans: 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.
Note: Sigma (Σ) is a mathematical term for summation or "adding up." It's telling you to add up all the possible results from the rest of the equation.

Sum of squares is a measure of how a data set varies around a central number (like the mean). You might realize by the phrase that you're summing (*adding up*) squares—but squares of what? You'll sometimes see this formula:

$$y = Y - \overline{Y}$$

Other times you might see actual "squares", like in this regression line:



Explained 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

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

It involves a *lot* of subtracting, squaring and summing. Your calculations will be prone to errors, so you're much better off using software like Excel to do the calculations. You won't even need to know the actual formulas, as Excel works them behind the scenes.

Uses

The smaller the residual sum of squares, the better your model fits your data; The greater the residual sum of squares, the poorer your model fits your data. A value of zero means your model is a perfect fit. One major use is in finding the coefficient of determination (R2). The coefficient of determination is a ratio of the explained sum of squares to the total sum of squares.

3. What is the need of regularization in machine learning?

Ans: It is said that regularization can help us obtain simple models over complex ones to avoid over-fitting. But for a linear classification problem:

$$F(x) = Wx$$

The complexity of the model is somewhat specified: it's linear, not quadratic or something more complex. We need regularization because:
1) learn the data too well, even the noises which is called over fitting
2) do not learn from the data, cannot find the pattern from the data which is called under fitting.
Now, both over fitting and underfitting are problems one need to address while building models.

Regularization in Machine Learning is used to minimize the problem of overfitting, the result is that the model generalizes well on the unseen data once overfitting is minimized.
To avoid overfitting, regularization discourages learning a more sophisticated or flexible model. Regularization will try to minimize a loss function by inducing penalty.
For Example
The residual sum of squares is our optimization function or loss function in simple linear regression (RSS).

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

Here ,

y is the dependent variable,
x_1, x_2, x_3,..... x_n are independent variables.
b_0, b_1 ,b_2..... b_n, are the coefficients estimates for different variables of x, these can also be called weights or magnitudes
Regularization will shrink these coefficients towards Zero,
Minimizing the loss means less error and model will be a good fit.

The way regularization can be done is by
1) RIDGE also know as L-2 Regularization

2) LASSO (Least Absolute and Selection Operator) also known as L-1 Regularization

4. What is Gini–impurity index?

ANS: The Gini Index or Gini Impurity is calculated by subtracting the sum of the squared probabilities of each class from one. It favours mostly the larger partitions and are very simple to implement. In simple terms, it calculates the probability of a certain randomly selected feature that was classified incorrectly.
Gini Impurity is a measurement used to build Decision Trees to determine how the features of a dataset should split nodes to form the tree

5. Are unregularized decision-trees prone to overfitting? If yes, why?

ANS: 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.

There are several approaches to avoiding overfitting in building decision trees. Pre-pruning that stop growing the tree earlier, before it perfectly classifies the training set. Post-pruning that allows the tree to perfectly classify the training set, and then post prune the tree.

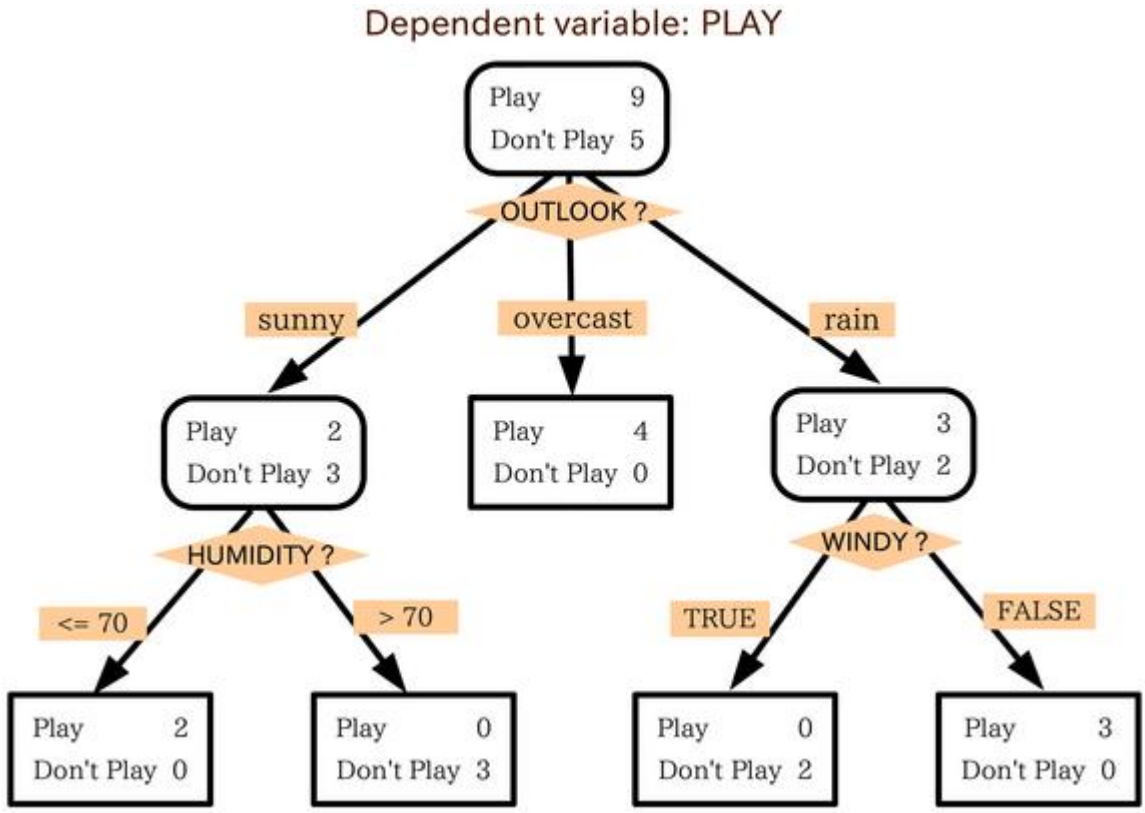
6. What is an ensemble technique in machine learning?

ANS: Ensemble methods are techniques that aim at improving the accuracy of results in models by combining multiple models

instead of using a single model. The combined models increase the accuracy of the results significantly. This has boosted the popularity of ensemble methods in machine learning.

- Ensemble methods aim at improving predictability in models by combining several models to make one very reliable model.
- The most popular ensemble methods are boosting, bagging, and stacking.
- Ensemble methods are ideal for regression and classification, where they reduce bias and variance to boost the accuracy of models.

largely utilize Decision Trees to outline the definition and practicality of Ensemble Methods (however it is important to note that Ensemble Methods do not only pertain to Decision Trees).



7. What is the difference between Bagging and Boosting techniques?

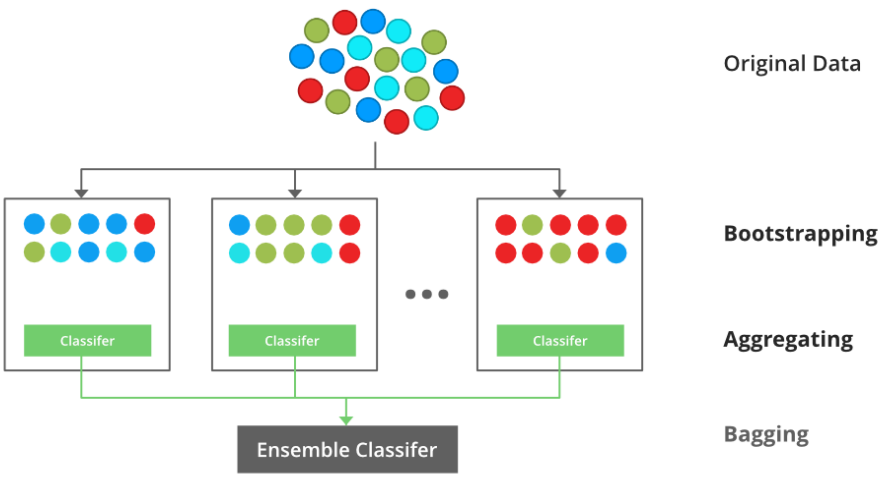
ANS:

Bagging:

1. is a technique for reducing prediction variance by producing additional data for training from a dataset by combining repetitions with combinations to create multi-sets of the original data.

2. It is a homogeneous weak learners' model that learns from each other independently in parallel and combines them for determining the model average.

3. Bootstrap Aggregating, also known as bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It decreases the variance and helps to avoid overfitting. It is usually applied to decision tree methods. Bagging is a special case of the model averaging approach.



An illustration for the concept of bootstrap aggregating (Bagging)

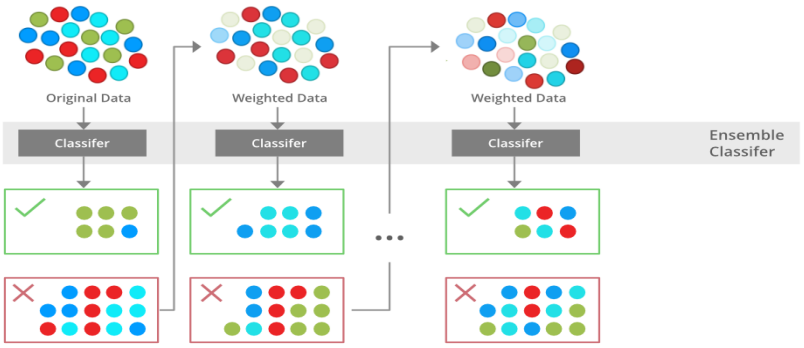
Example of Bagging
The Random Forest model uses Bagging, where decision tree models with higher variance are present. It makes random feature selection to grow trees. Several random trees make a Random Forest.

Boosting:

1. is an iterative strategy for adjusting an observation's weight based on the previous classification.

2. It is also a homogeneous weak learners' model but works differently from Bagging. In this model, learners learn sequentially and adaptively to improve model predictions of a learning algorithm.

3. Boosting is an ensemble modeling technique that attempts to build a strong classifier from the number of weak classifiers. It is done by building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models is added.



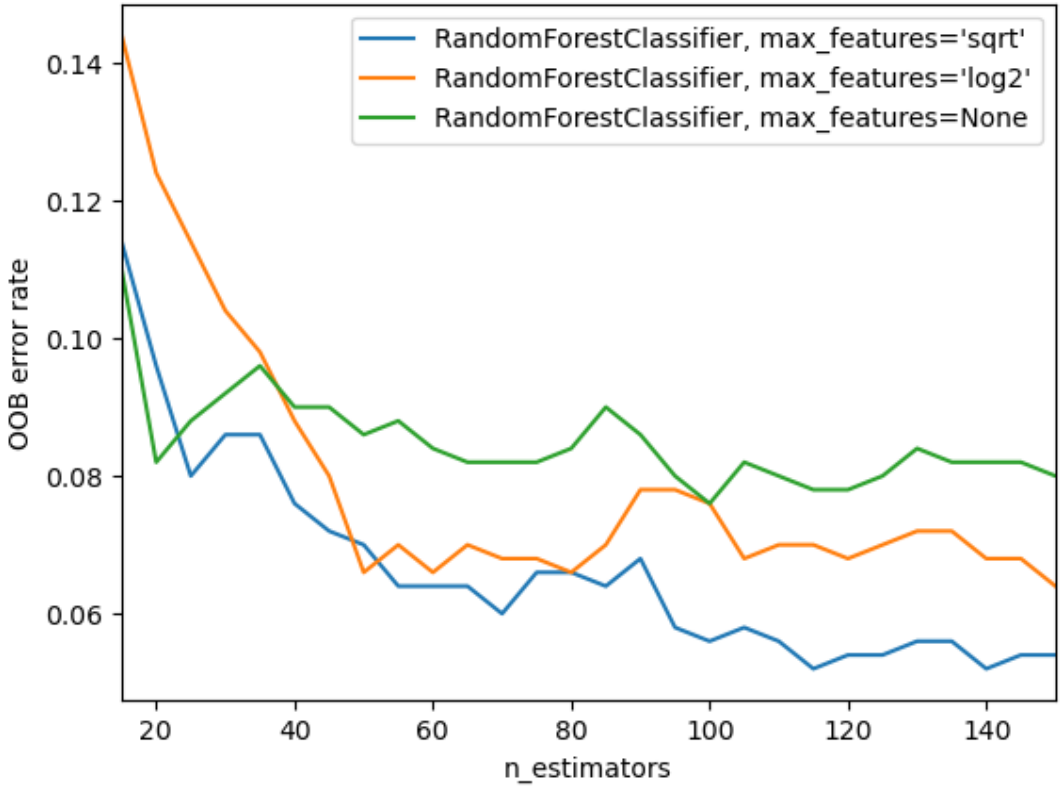
An illustration presenting the intuition behind the boosting algorithm

Differences Between Bagging and Boosting

S.NO	Bagging	Boosting
1.	The simplest way of combining predictions that belong to the same type.	A way of combining predictions that belong to the different types.
2.	Aim to decrease variance, not bias.	Aim to decrease bias, not variance.
3.	Each model receives equal weight.	Models are weighted according to their performance.
4.	Each model is built independently.	New models are influenced by the performance of previously built models.
5.	Different training data subsets are selected using row sampling with replacement and random sampling methods from the entire training dataset.	Every new subset contains the elements that were misclassified by previous models.
6.	Bagging tries to solve the over-fitting problem.	Boosting tries to reduce bias.
7.	If the classifier is unstable (high variance), then apply bagging.	If the classifier is stable and simple (high bias) the apply boosting.
8.	In this base classifiers are trained parallelly.	In this base classifiers are trained sequentially.
9	Example: The Random forest model uses Bagging.	Example: The AdaBoost uses Boosting techniques

8. What is out-of-bag error in random forests?

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

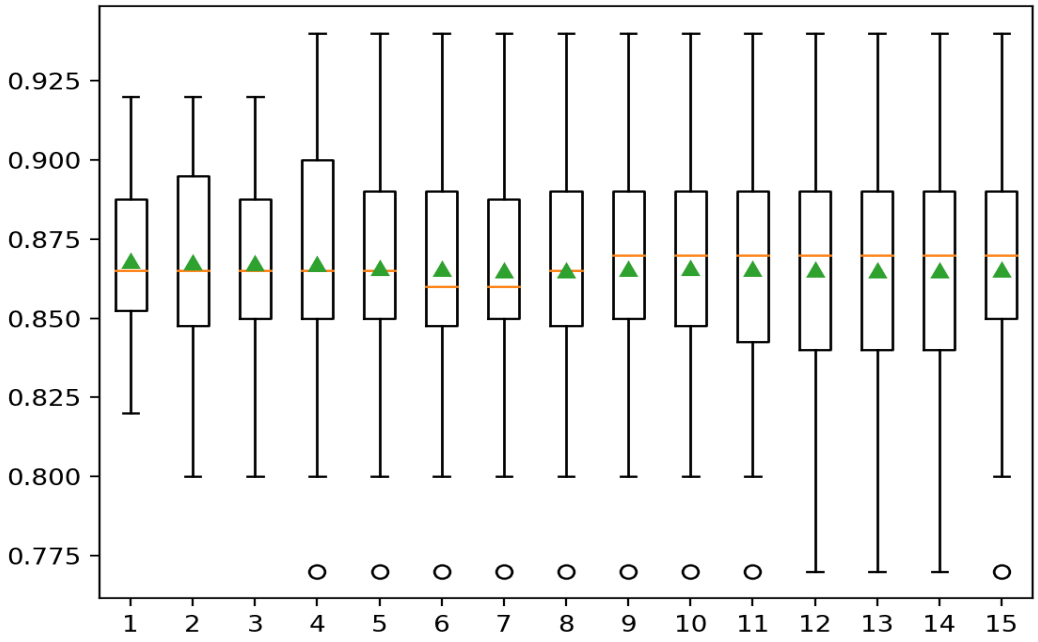


Out-of-bag (OOB) error, also called out-of-bag estimate, is a method of measuring the prediction error of random forests, boosted decision trees, and other machine learning models utilizing bootstrap aggregating (bagging). Bagging uses subsampling with replacement to create training samples for the model 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.

9. What is K-fold cross-validation?

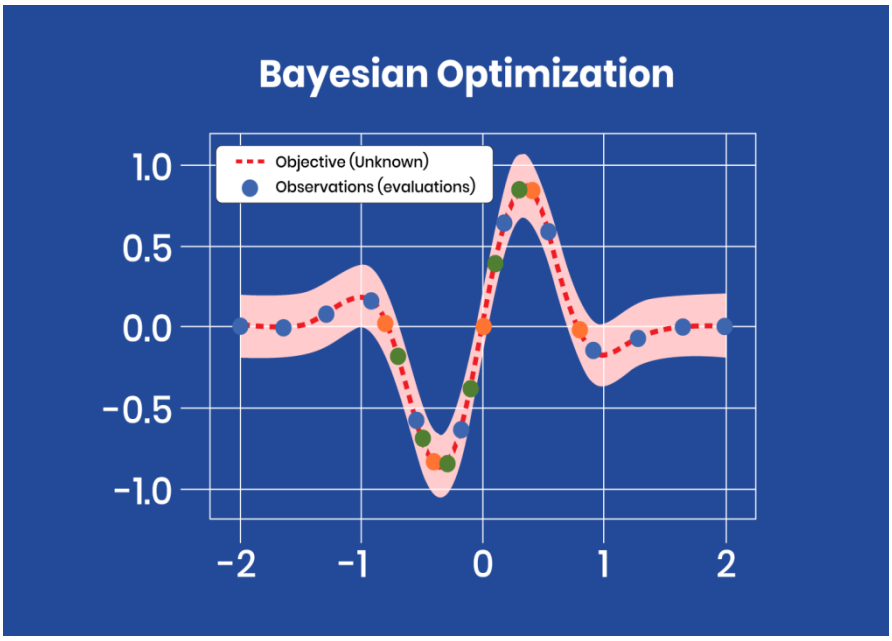
ANS: Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation.



K-fold Cross-Validation is when the dataset is split into a K number of folds and is used to evaluate the model's ability when given new data. K refers to the number of groups the data sample is split into. For example, if you see that the k-value is 5, we can call this a 5-fold cross-validation.

10. What is hyper parameter tuning in machine learning and why it is done?

ANS: Hyperparameter tuning consists of finding a set of optimal hyperparameter values for a learning algorithm while applying this optimized algorithm to any data set. That combination of hyperparameters maximizes the model's performance, minimizing a predefined loss function to produce better results with fewer errors.



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:

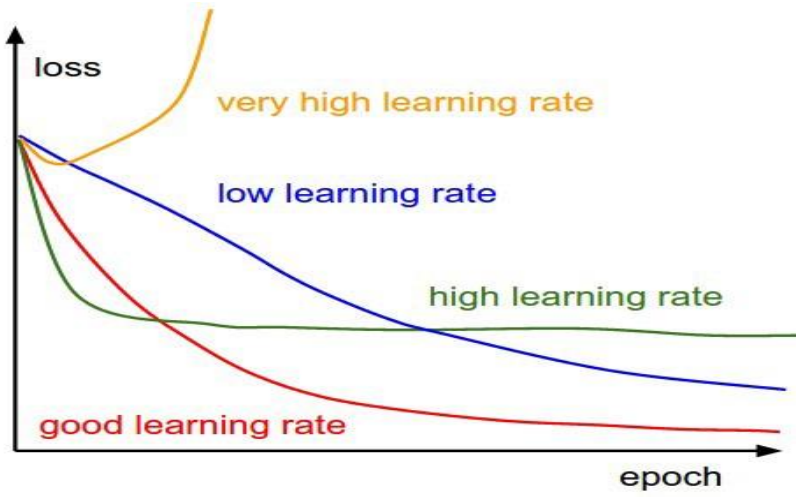
1. The penalty in Logistic Regression Classifier i.e. L1 or L2 regularization
2. The learning rate for training a neural network.
3. The C and sigma hyperparameters for support vector machines.
4. The k in k-nearest neighbors.

Models can have many hyperparameters and finding the best combination of parameters can be treated as a search problem.

11. What issues can occur if we have a large learning rate in Gradient Descent?
ANS: In order for Gradient Descent to work, we must set the learning rate to an appropriate value. This parameter determines how fast or slow we will move towards the optimal weights. If the learning rate is very large we will skip the optimal solution.

A learning rate that is too large can cause the model to converge too quickly to a suboptimal solution, whereas a learning rate that is too small can cause the process to get stuck.

The learning rate is the most important hyper-parameter for tuning neural networks. A good learning rate could be the difference between a model that doesn't learn anything and a model that presents state-of-the-art results.The below diagram demonstrates the different scenarios one can fall into when configuring the learning rate.



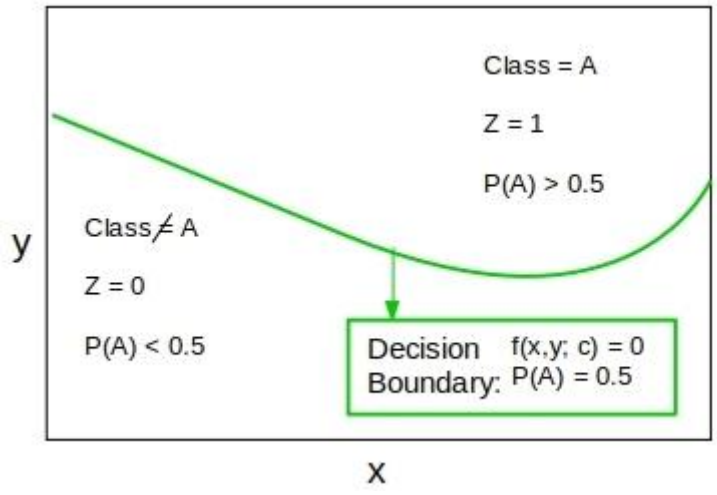
12. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?

ANS: Logistic regression is indeed non linear in terms of Odds and Probability, however it is linear in terms of Log Odds.

logistic regression is used when a variable is dichotomous. Since the variable can assume only value 1 or 0, fitting a line assumes a linear relationship which cannot hold for dichotomous outcomes. It can be proved that the linear probability model will not be efficient and, furthermore, nothing ensures that the estimated dependent variable will be bounded between 0 and 1. The logit can solve these problem.

The linear decision boundary is used for reasons of simplicity following the Zen mantra – when in doubt simplify. In those cases where we suspect the decision boundary to be nonlinear, it may make sense to formulate logistic regression with a nonlinear model and evaluate how much better we can do. That is what this post is about.

Briefly review the formulation of the likelihood function and its maximization. To keep the algebra at bay we stick to 2 classes, in a 2-d feature space. A point [x, y] in the feature space can belong to only one of the classes, and the function f (x,y; c) = 0



13. Differentiate between Adaboost and Gradient Boosting.

ANS: Comparison between AdaBoost and Gradient Boost

S.No	Adaboost	Gradient Boost
1	An additive model where shortcomings of previous models are identified by high-weight data points.	An additive model where shortcomings of previous models are identified by the gradient.
2	The trees are usually grown as decision stumps.	The trees are grown to a greater depth usually ranging from 8 to 32 terminal nodes.
3	Each classifier has different weights assigned to the final prediction based on its performance.	All classifiers are weighed equally and their predictive capacity is restricted with learning rate to increase accuracy.
4	It gives weights to both classifiers and observations thus capturing maximum variance within data.	It builds trees on previous classifier's residuals thus capturing variance in data.

AdaBoost

This is a type of ensemble technique, where a number of weak learners are combined together to form a strong learner. Here, usually, each weak learner is developed as **decision stumps** (A stump is a tree with just a single split and two terminal nodes) that are used to classify the observations.

Overview

At the end of this article, you will be able to-

1. Understand the working and math behind two ML techniques namely AdaBoost and Gradient Boost.
2. Similarities and differences between AdaBoost and Gradient Boost algorithms.

A brief introduction on Ensemble Techniques

Combining two or more classifiers or models and making a prediction (either through voting for classification or average for numerical) to increase the performance accuracy is called Ensemble models. They are more useful and proficient when the model is unstable(flexible) resulting in high variance between different sample sets taken out of the population. The basic intuition behind this is to expose the model to maximum variance within the given dataset itself. Here for simplicity, decision tree models are used as base classifiers.

AdaBoost

This is a type of ensemble technique, where a number of weak learners are combined together to form a strong learner. Here, usually, each weak learner is developed as **decision stumps** (A stump is a tree with just a single split and two terminal nodes) that are used to classify the observations.

In contrary to the random forest, here each classifier has different weights assigned to it based on the classifier's performance (more weight is assigned to the classifier when accuracy is more and vice-verse), weights are also assigned to the observations at the end of every round, in such a way that wrongly predicted observations have increased weight resulting in their probability of being picked more often in the next classifier's sample. So, the consecutive training set depends on their previous training set, and hence correlation exists between the built trees.

Thus, Adaboost increases the predictive accuracy by assigning weights to both observations at end of every tree and weights(scores) to every classifier. Hence, in Adaboost, every classifier has a different weightage on final prediction contrary to the random forest where all trees are assigned equal weights.

of correctly classified observation and $exp^{(alpha(i))}$ increases the weight of misclassified observation, thus, increasing their chance of getting picked. z_j is the normalization factor that is used to reassign weights of observation such that all weights sum up to 1.

The final prediction of each observation is made by aggregating the weighted average of the prediction made by each classifier. AdaBoost might result in overfitting. Hence, no. of trees should be checked and restricted.

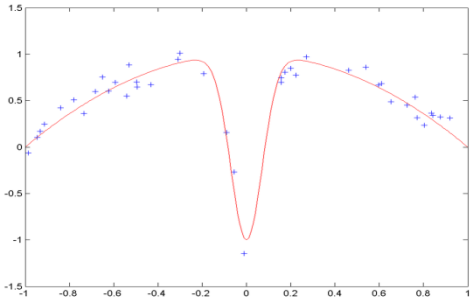
Gradient Boosting Machine (GBM)

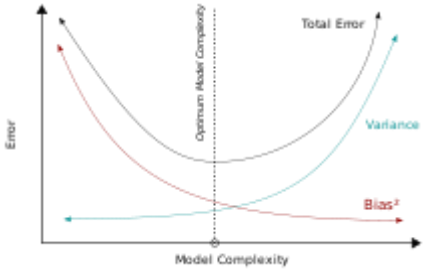
Just like AdaBoost, Gradient Boost also combines a no. of weak learners to form a strong learner. Here, the residual of the current classifier becomes the input for the next consecutive classifier on which the trees are built, and hence it is an additive model. The residuals are captured in a step-by-step manner by the classifiers, in order to capture the maximum variance within the data, this is done by introducing the learning rate to the classifiers.

14. What is bias-variance trade off in machine learning?

ANS: In statistics and machine learning, the bias–variance tradeoff is the property of a model that the variance of the parameter estimated across samples

can be reduced by increasing the bias in the estimated parameters.



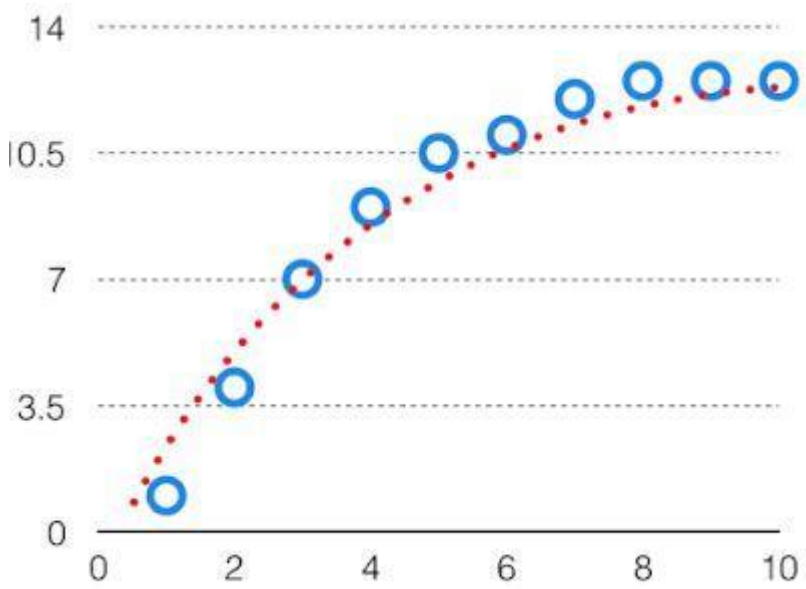


Bias Variance Tradeoff

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.



15. Give short description each of Linear, RBF, Polynomial kernels used in SVM.

ANS: 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.

Polynomial kernel

It is popular in image processing.
Equation is:

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$$

Polynomial kernel equation

where d is the degree of the polynomial.

Linear splines kernel in one-dimension

It is useful when dealing with large sparse data vectors. It is often used in text categorization. The splines kernel also performs well in regression problems. Equation is:

$$k(x,y) = 1 + xy + xy\ min(x,y) - \frac{x+y}{2}\ min(x,y)^2 + \frac{1}{3}\min(x,y)^3$$

Linear splines kernel equation in one-dimension

If you have any query about SVM Kernel Functions, So feel free to share with us. We will be glad to solve your queries.

RBF kernel

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

$$k(x,y) = \exp\left(-\frac{\|x-y\|}{\sigma}\right)$$

Laplace RBF kernel equation