MACHINE LEARNING

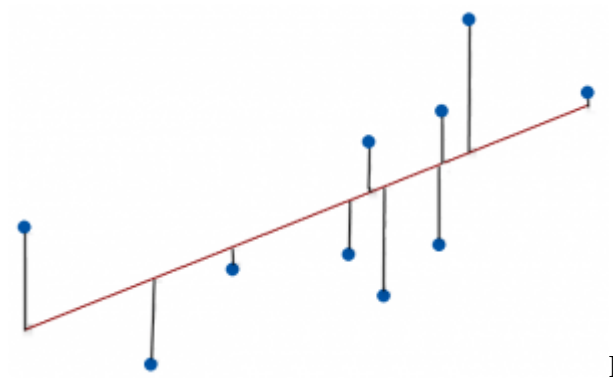
Q1 to Q15 are subjective answer type questions, Answer them briefly.

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: 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.

After fitting a linear regression model, you need to determine how well the model fits the data. Does it do a good job of explaining changes in the dependent variable? There are several key goodness-of-fit statistics for regression analysis. In this post, we'll examine R-squared (R^2), highlight some of its limitations, and discover some surprises. For instance, small R-squared values are not always a problem, and high R-squared values are not necessarily good!

Assessing Goodness-of-Fit in a Regression Model



Residuals are the distance between the observed value and the fitted value.

Linear regression identifies the equation that produces the smallest difference between all the observed values and their fitted values. To be precise, linear regression finds the smallest sum of squared residuals that is possible for the dataset.

Statisticians say that a regression model fits the data well if the differences between the observations and the predicted values are small and unbiased. Unbiased in this context means that the fitted values are not systematically too high or too low anywhere in the observation space.

However, before assessing numeric measures of goodness-of-fit, like R-squared, you should evaluate the residual plots. Residual plots can expose a biased model far more effectively than the numeric output by displaying problematic patterns in the residuals. If your model is biased, you cannot trust the results. If your residual plots look good, go ahead and assess your R-squared and other statistics.

R-squared and the Goodness-of-Fit

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 the percentage of the dependent variable variation that a linear model explains.

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

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 the variation in the response variable around its mean.

Usually, the larger the R^2, the better the regression model fits your observations. However, this guideline has important caveats that I'll discuss in both this post and the next post.

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 residual sum of squares is used to help you decide if a statistical model is a good fit for your data. It measures the overall difference between your data and the values predicted by your estimation model (a "residual" is a measure of the distance from a data point to a regression line).

In ANOVA, Total SS is related to the total sum and explained sum with the following formula:
Total SS = Explained SS + Residual Sum of Squares. Watch the video for a definition and calculation steps for Total (TSS), Between (BSS), and Within (WSS):

What is the Total Sum of Squares?

The Total SS (TSS or SST) tells you how much variation there is in the dependent variable.
Total SS = $\Sigma(Yi - \text{mean of } Y)^2$.
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:

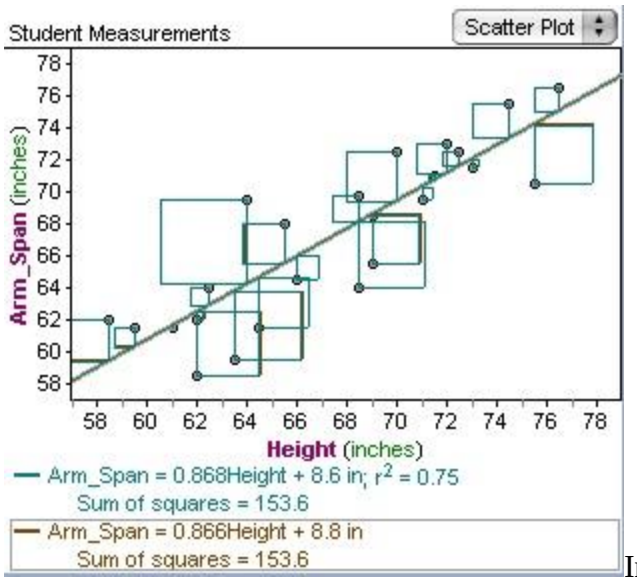
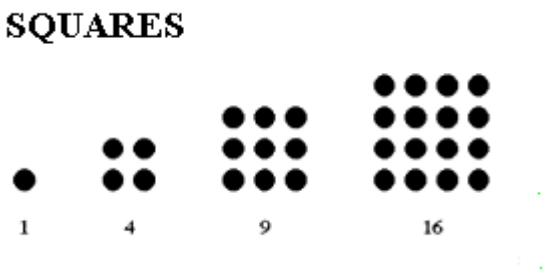


Image: University of Georgia.

Squares of numbers, as in 4^2 and 10^2 can be represented with actual geometric squares (image courtesy of UMBC.edu):



So the square shapes you see on regression lines are just representations of square numbers, like 5^2 or 9^2. When you're looking for a sum of squares, use the formula $y = Y - \overline{Y}$; to find the actual number that represents a sum of squares. A diagram (like the regression line above) is optional, and can supply a visual representation of what you're calculating.

Sample Question

Find the Sum of Sq. for the following numbers: 3,5,7.

Step 1: Find the mean by adding the numbers together and dividing by the number of items in the set:
(3 + 5 + 7) / 3 = 15 / 3 = 5

Step 2: Subtract the mean from each of your data items:
3 – 5 = -2
5 – 5 = 0
7 – 5 = 2

Step 3: Square your results from Step 3:
-2 x -2 = 4
0 x 0 = 0
2 x 2 = 4

Step 4: Sum (add up) all of your numbers:
4 + 4 + 0 = 8.

That's it!

Sum of Sq. in ANOVA and Regression

As you can probably guess, things get a little more complicated when you're calculating sum of squares in regression analysis or hypothesis testing. It is rarely calculated by hand; instead, software like Excel or SPSS is usually used to calculate the result for you.

For reference, sum of squares in regression uses the equation:

$$\Sigma(y - \overline{y})^2 = \Sigma(\hat{y} - \overline{y})^2 + \Sigma(y - \hat{y})^2$$

And in ANOVA it is calculated with:
The total SS = treatment sum of squares (SST) + SS of the residual error (SSE)

What is the 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.

What is the 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

If all those formulas look confusing, don't worry! It's very, very unusual for you to want to use them. Finding the sum by hand is tedious and time-consuming. 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 (R^2). The coefficient of determination is a ratio of the explained sum of squares to the total sum of squares.

Sum of Squares Within

Within-group variation is reported in ANOVA output as SS(W) or which means Sum of Squares Within groups or SSW: Sum of Squares Within. It is intrinsically linked to between group variation (Sum of Squares between), variance difference caused by how groups interact with each other.

SSW is one component of total sum of squares (the other is between sum of squares). Within sum of squares represents the variation due to individual differences in the score. In other words, it's the variation of individual scores around the group mean; it is variation *not* due to the treatment (Newsom, 2013).

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

Ans : Regularization refers to techniques that are used to calibrate machine learning models in order **to minimize the adjusted loss function and prevent overfitting or underfitting**. Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it

Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting.

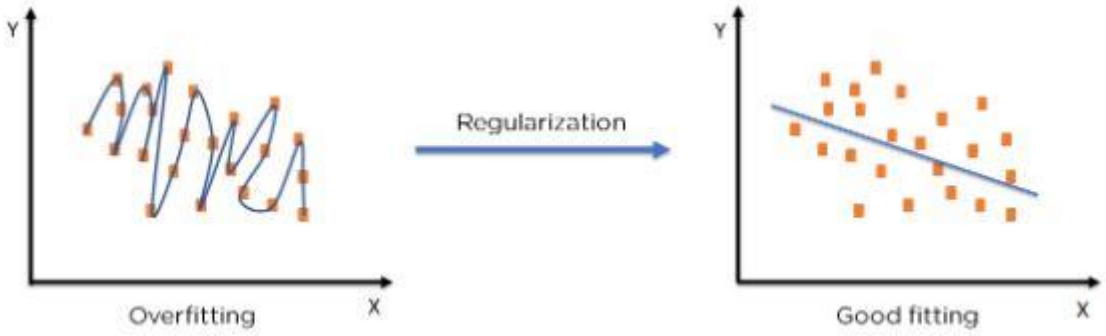


Figure 5: Regularization on an over-fitted model

Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it.

While training a machine learning model, the model can easily be overfitted or under fitted. To avoid this, we use regularization in machine learning to properly fit a model onto our test set. Regularization techniques help reduce the chance of overfitting and help us get an optimal model. In this article titled 'The Best Guide to Regularization in Machine Learning', you will learn all you need to know about regularization.

What Are Overfitting and Underfitting?

To train our machine learning model, we give it some data to learn from. The process of plotting a series of data points and drawing the best fit line to understand the relationship between the variables is called Data Fitting. Our model is the best fit when it can find all necessary patterns in our data and avoid the random data points and unnecessary patterns called Noise.

If we allow our machine learning model to look at the data too many times, it will find a lot of patterns in our data, including the ones which are unnecessary. It will learn really well on the test dataset and fit very well to it. It will learn important patterns, but it will also learn from the noise in our data and will not be able to predict on other datasets.

A scenario where the machine learning model tries to learn from the details along with the noise in the data and tries to fit each data point on the curve is called Overfitting.

In the figure depicted below, we can see that the model is fit for every point in our data. If given new data, the model curves may not correspond to the patterns in the new data, and the model cannot predict very well in it.

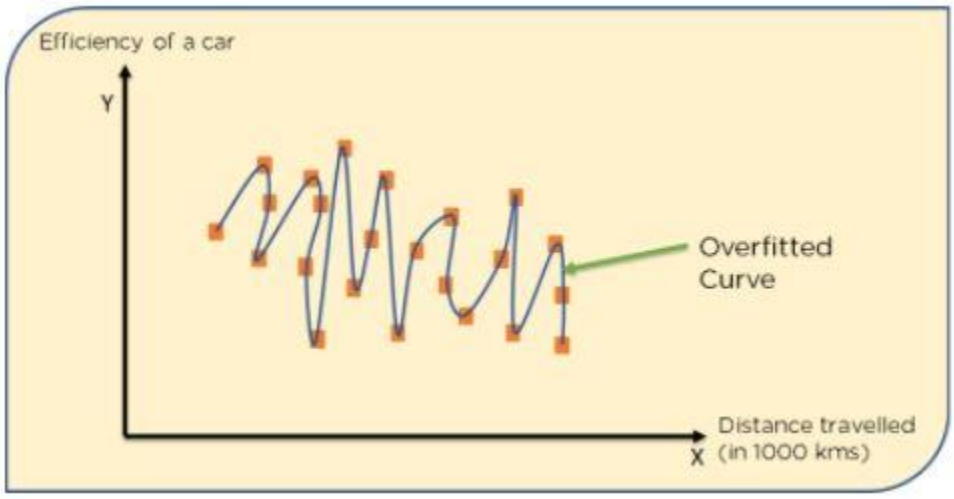


Figure 1: Overfitted Model

Conversely, in a scenario where the model has not been allowed to look at our data a sufficient number of times, the model won't be able to find patterns in our test dataset. It will not fit properly to our test dataset and fail to perform on new data too.

A scenario where a machine learning model can neither learn the relationship between variables in the testing data nor predict or classify a new data point is called Underfitting.

The below diagram shows an under-fitted model. We can see that it has not fit properly to the data given to it. It has not found patterns in the data and has ignored a large part of the dataset. It cannot perform on both known and unknown data.

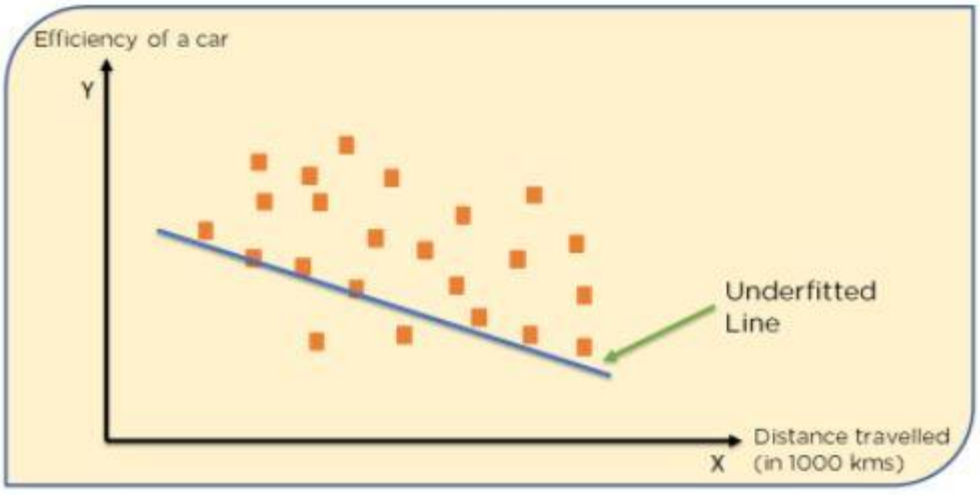


Figure 2: Underfitted Model

What are Bias and Variance?

A Bias occurs when an algorithm has limited flexibility to learn from data. Such models pay very little attention to the training data and oversimplify the model therefore the validation error or prediction error and training error follow similar trends. Such models always lead to a high error on training and test data. High Bias causes underfitting in our model.

Variance defines the algorithm's sensitivity to specific sets of data. A model with a high variance pays a lot of attention to training data and does not generalize therefore the validation error or prediction error are far apart from each other. Such models usually perform very well on training data but have high error rates on test data. High Variance causes overfitting in our model.

An optimal model is one in which the model is sensitive to the pattern in our model, but at the same time can generalize to new data. This happens when Bias and Variance are both optimal. We call this Bias-Variance Tradeoff and we can achieve it in over or under fitted models by using Regression.

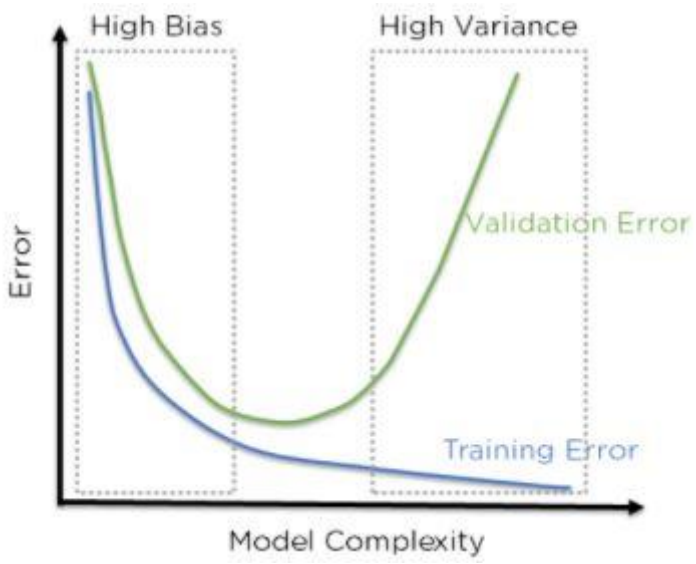


Figure 3: Error in testing and training datasets with high bias and variance

In the above figure, we can see that when bias is high, the error in both testing and training set is also high. When Variance is high, the model performs well on our training set and gives a low error, but the error in our testing set is very high. In the middle of this exists a region where the bias and variance are in perfect balance to each other, and here, but the training and testing errors are low.

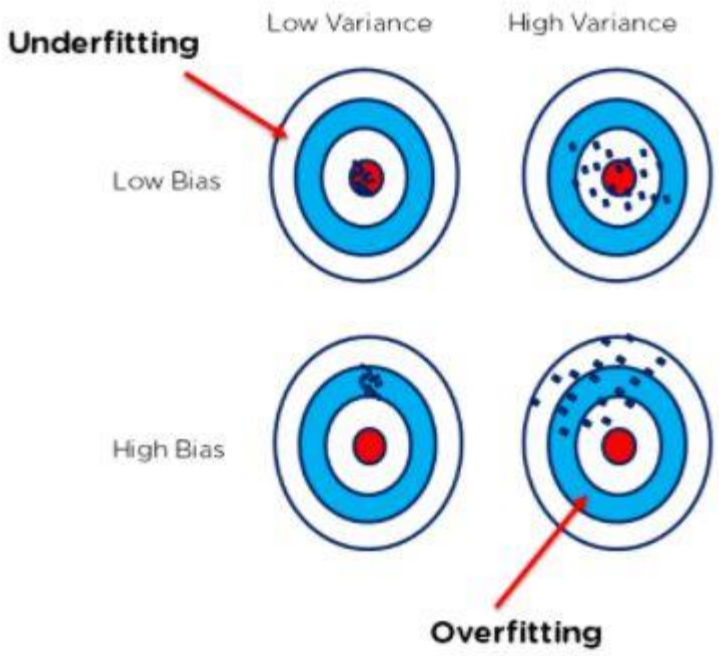


Figure 4: Bullseye diagram for different bias and variance levels

What is Regularization in Machine Learning?

Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting.

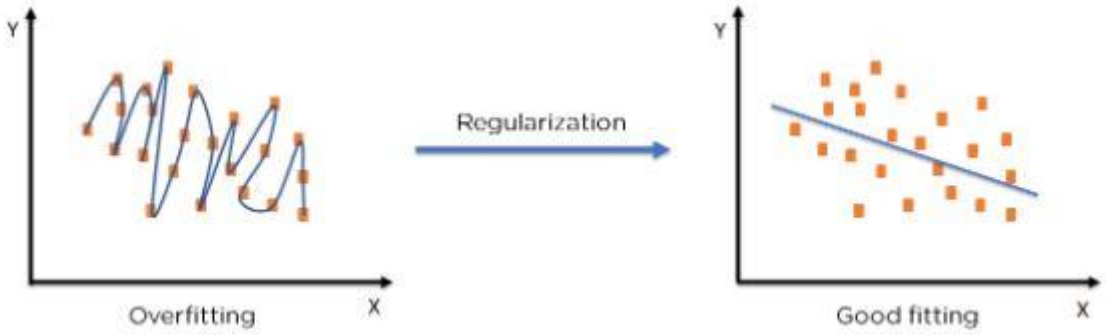


Figure 5: Regularization on an over-fitted model

Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it.

Regularization Techniques

There are two main types of regularization techniques: Ridge Regularization and Lasso Regularization.

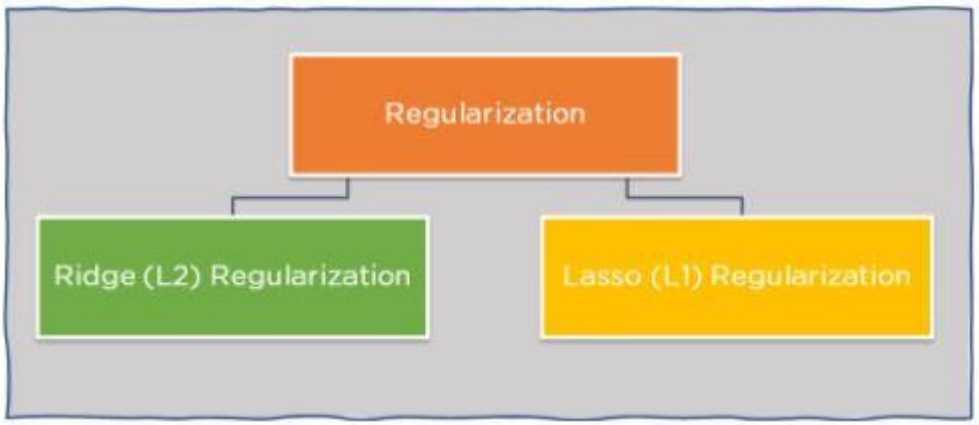


Figure 6: Regularization techniques

Ridge Regularization :

Also known as Ridge Regression, it modifies the over-fitted or under fitted models by adding the penalty equivalent to the sum of the squares of the magnitude of coefficients.

This means that the mathematical function representing our machine learning model is minimized and coefficients are calculated. The magnitude of coefficients is squared and added. Ridge Regression performs regularization by shrinking the coefficients present. The function depicted below shows the cost function of ridge regression :

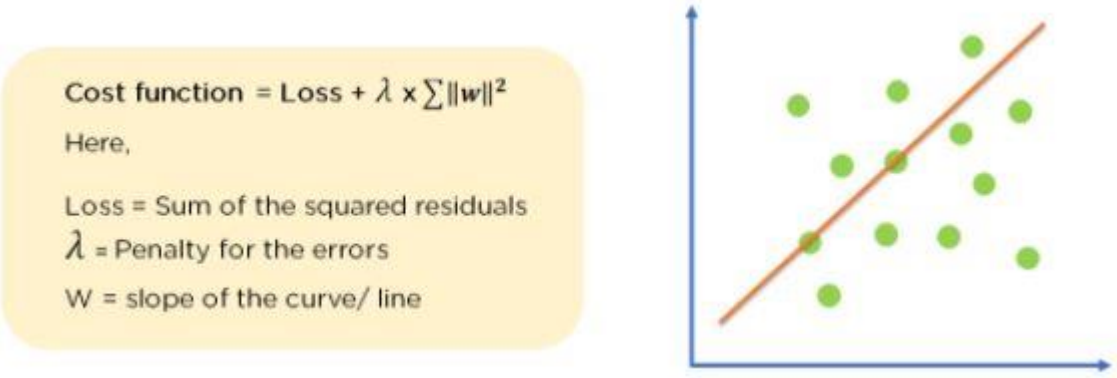


Figure 7: Cost Function of Ridge Regression

In the cost function, the penalty term is represented by Lambda λ. By changing the values of the penalty function, we are controlling the penalty term. The higher the penalty, it reduces the magnitude of coefficients. It shrinks the parameters. Therefore, it is used to prevent multicollinearity, and it reduces the model complexity by coefficient shrinkage.

Consider the graph illustrated below which represents Linear regression :

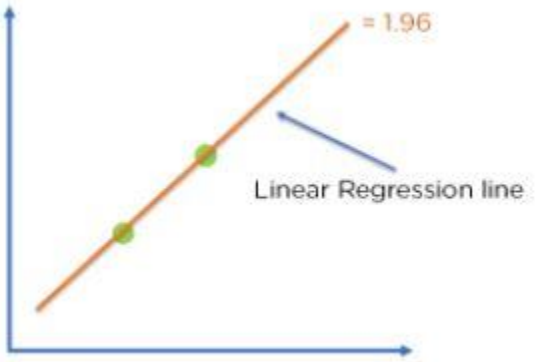


Figure 8: Linear regression model

Cost function = Loss + λ x∑‖w‖^2

Cost function = Loss + λ x∑‖w‖^2

For Linear Regression line, let’s consider two points that are on the line,

Loss = 0 (considering the two points on the line)

λ= 1

w = 1.4

Then, Cost function = 0 + 1 x 1.42

= 1.96

For Ridge Regression, let’s assume,

Loss = 0.32 + 0.22 = 0.13

λ = 1

w = 0.7

Then, Cost function = 0.13 + 1 x 0.72

= 0.62

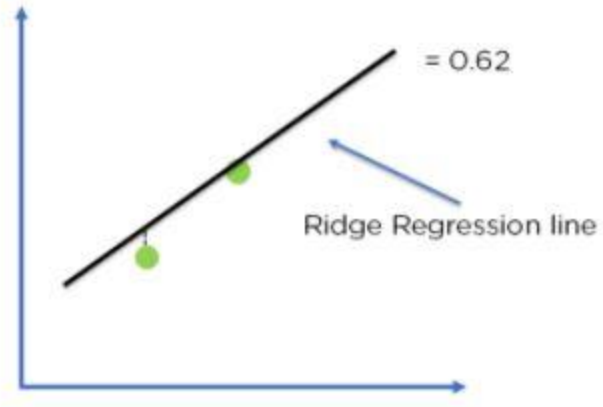


Figure 9: Ridge regression model

Comparing the two models, with all data points, we can see that the Ridge regression line fits the model more accurately than the linear regression line.

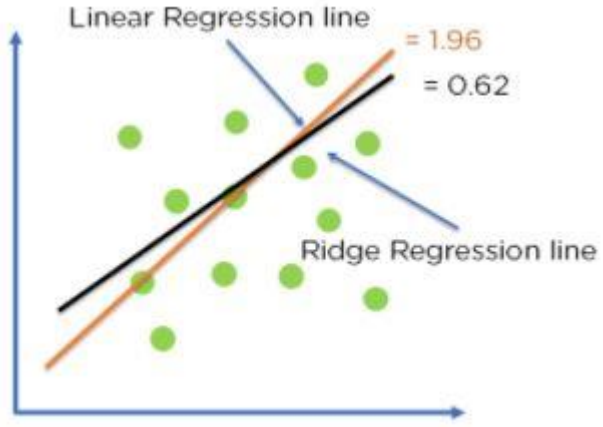


Figure 10: Optimization of model fit using Ridge Regression

What Are Overfitting and Underfitting?

To train our machine learning model, we give it some data to learn from. The process of plotting a series of data points and drawing the best fit line to understand the relationship between the variables is called Data Fitting. Our model is the best fit when it can find all necessary patterns in our data and avoid the random data points and unnecessary patterns called Noise.

If we allow our machine learning model to look at the data too many times, it will find a lot of patterns in our data, including the ones which are unnecessary. It will learn really well on the test dataset and fit very well to

it. It will learn important patterns, but it will also learn from the noise in our data and will not be able to predict on other datasets.

A scenario where the machine learning model tries to learn from the details along with the noise in the data and tries to fit each data point on the curve is called Overfitting.

In the figure depicted below, we can see that the model is fit for every point in our data. If given new data, the model curves may not correspond to the patterns in the new data, and the model cannot predict very well in it.

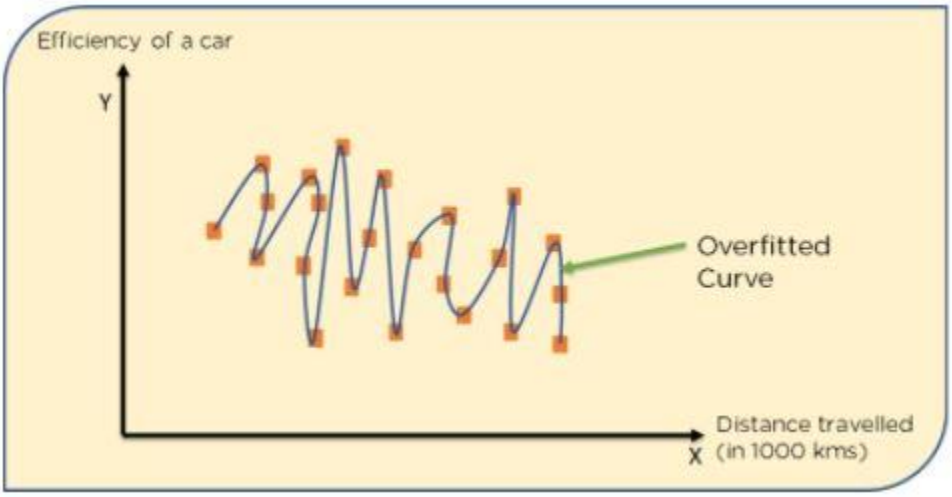


Figure 1: Overfitted Model

Conversely, in a scenario where the model has not been allowed to look at our data a sufficient number of times, the model won't be able to find patterns in our test dataset. It will not fit properly to our test dataset and fail to perform on new data too.

A scenario where a machine learning model can neither learn the relationship between variables in the testing data nor predict or classify a new data point is called Underfitting.

The below diagram shows an under-fitted model. We can see that it has not fit properly to the data given to it. It has not found patterns in the data and has ignored a large part of the dataset. It cannot perform on both known and unknown data.

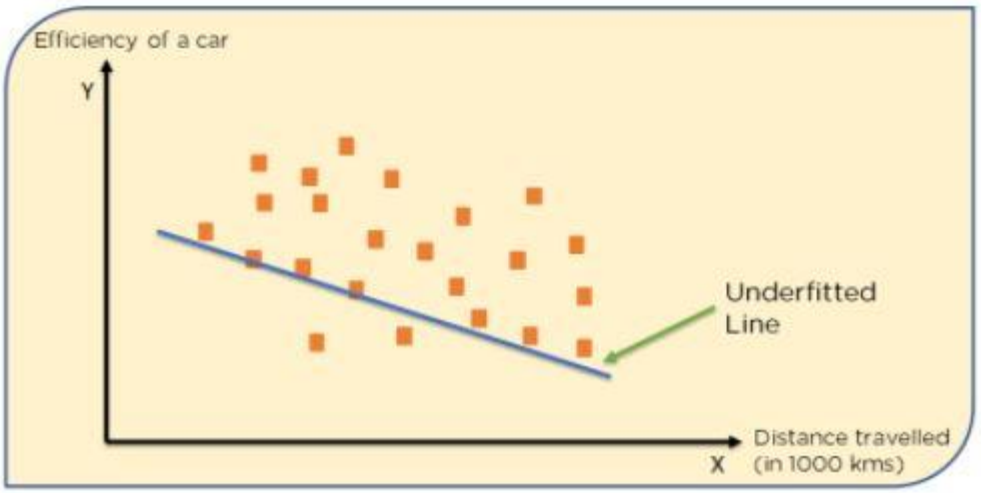


Figure 2: Underfitted Model

What are Bias and Variance?

A Bias occurs when an algorithm has limited flexibility to learn from data. Such models pay very little attention to the training data and oversimplify the model therefore the validation error or prediction error and training error follow similar trends. Such models always lead to a high error on training and test data. High Bias causes underfitting in our model.

Variance defines the algorithm's sensitivity to specific sets of data. A model with a high variance pays a lot of attention to training data and does not generalize therefore the validation error or prediction error are far apart from each other. Such models usually perform very well on training data but have high error rates on test data. High Variance causes overfitting in our model.

An optimal model is one in which the model is sensitive to the pattern in our model, but at the same time can generalize to new data. This happens when Bias and Variance are both optimal. We call this Bias-Variance Tradeoff and we can achieve it in over or under fitted models by using Regression.

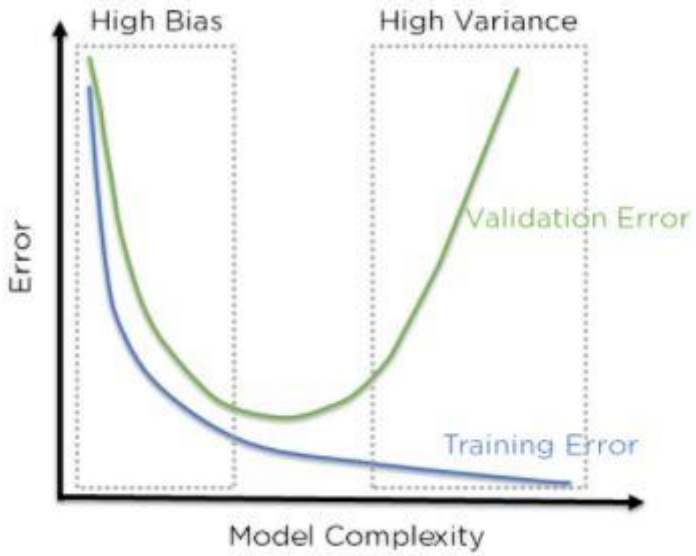


Figure 3: Error in testing and training datasets with high bias and variance

In the above figure, we can see that when bias is high, the error in both testing and training set is also high. When Variance is high, the model performs well on our training set and gives a low error, but the error in our testing set is very high. In the middle of this exists a region where the bias and variance are in perfect balance to each other, and here, but the training and testing errors are low.

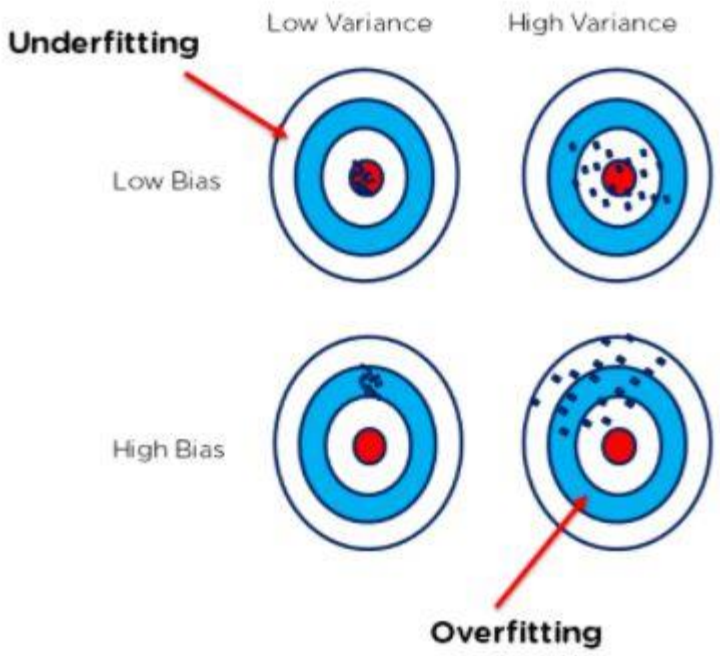


Figure 4: Bullseye diagram for different bias and variance levels

What is Regularization in Machine Learning?

Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting.

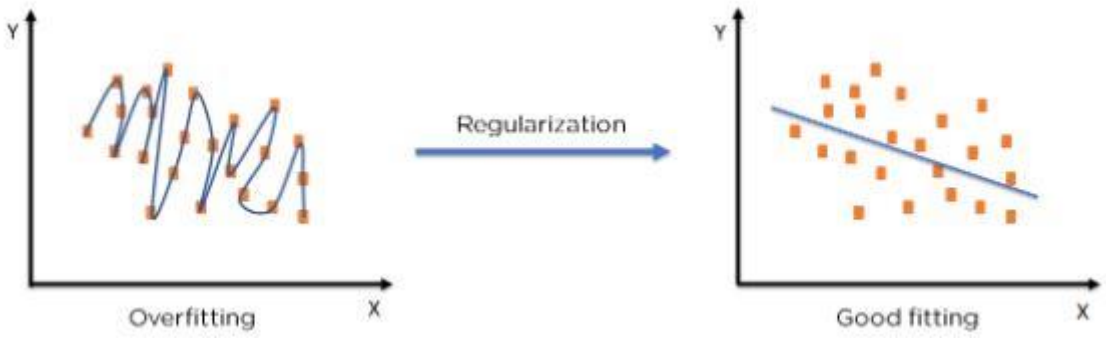


Figure 5: Regularization on an over-fitted model

Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it.

Regularization Techniques

There are two main types of regularization techniques: Ridge Regularization and Lasso Regularization.

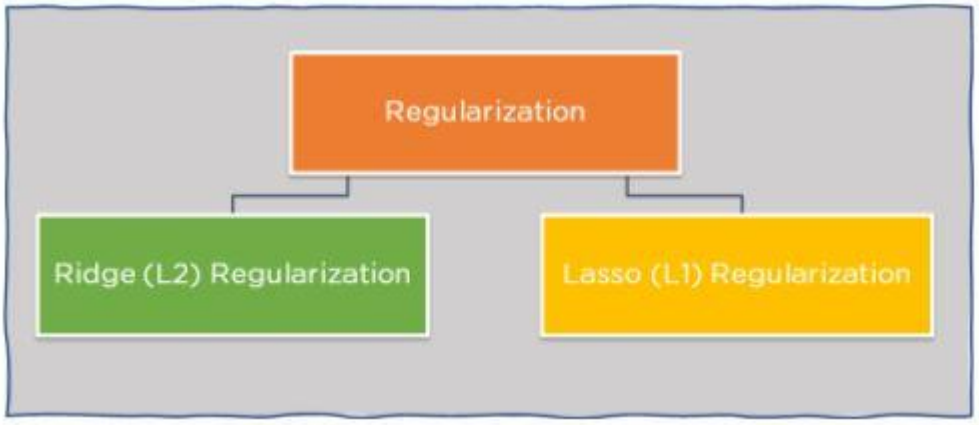


Figure 6: Regularization techniques

Ridge Regularization :

Also known as Ridge Regression, it modifies the over-fitted or under fitted models by adding the penalty equivalent to the sum of the squares of the magnitude of coefficients.

This means that the mathematical function representing our machine learning model is minimized and coefficients are calculated. The magnitude of coefficients is squared and added. Ridge Regression performs regularization by shrinking the coefficients present. The function depicted below shows the cost function of ridge regression :

Cost function = Loss + $\lambda \times \sum \|w\|^2$

Here,

Loss = Sum of the squared residuals

λ = Penalty for the errors

W = slope of the curve/ line

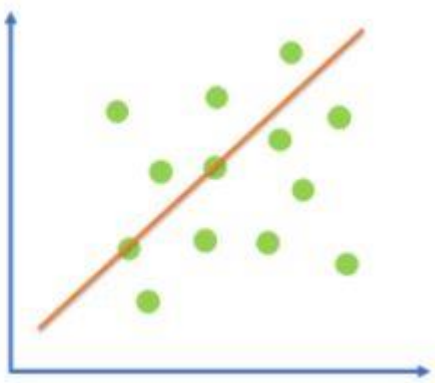


Figure 7: Cost Function of Ridge Regression

In the cost function, the penalty term is represented by Lambda λ. By changing the values of the penalty function, we are controlling the penalty term. The higher the penalty, it reduces the magnitude of coefficients. It shrinks the parameters. Therefore, it is used to prevent multicollinearity, and it reduces the model complexity by coefficient shrinkage.

Consider the graph illustrated below which represents Linear regression :

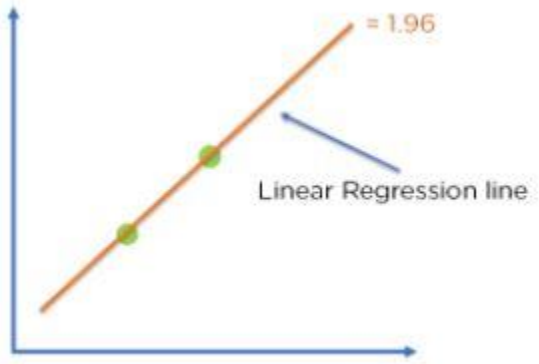


Figure 8: Linear regression model

Cost function = Loss + λ x∑‖w‖^2

For Linear Regression line, let’s consider two points that are on the line,

Loss = 0 (considering the two points on the line)

λ= 1

w = 1.4

Then, Cost function = 0 + 1 x 1.42

= 1.96

For Ridge Regression, let’s assume,

Loss = 0.32 + 0.22 = 0.13

λ = 1

w = 0.7

Then, Cost function = 0.13 + 1 x 0.72

= 0.62

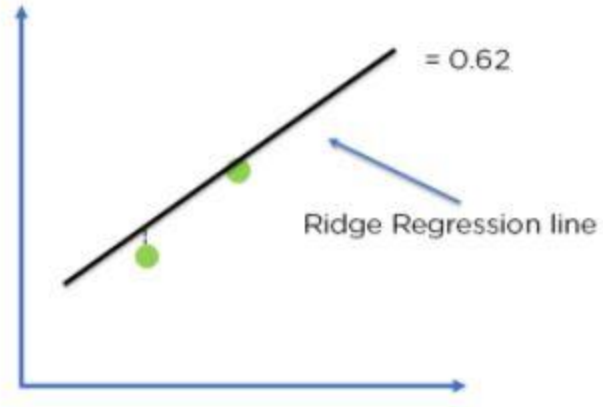


Figure 9: Ridge regression model

Comparing the two models, with all data points, we can see that the Ridge regression line fits the model more accurately than the linear regression line.

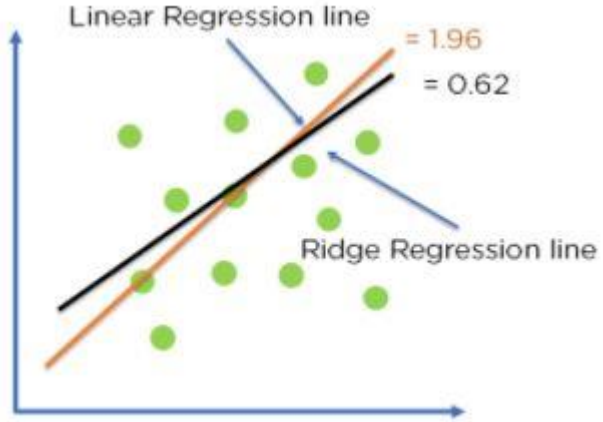


Figure 10: Optimization of model fit using Ridge Regression

Lasso Regression

It modifies the over-fitted or under-fitted models by adding the penalty equivalent to the sum of the absolute values of coefficients.

Lasso regression also performs coefficient minimization, but instead of squaring the magnitudes of the coefficients, it takes the true values of coefficients. This means that the coefficient sum can also be 0, because of the presence of negative coefficients. Consider the cost function for Lasso regression :

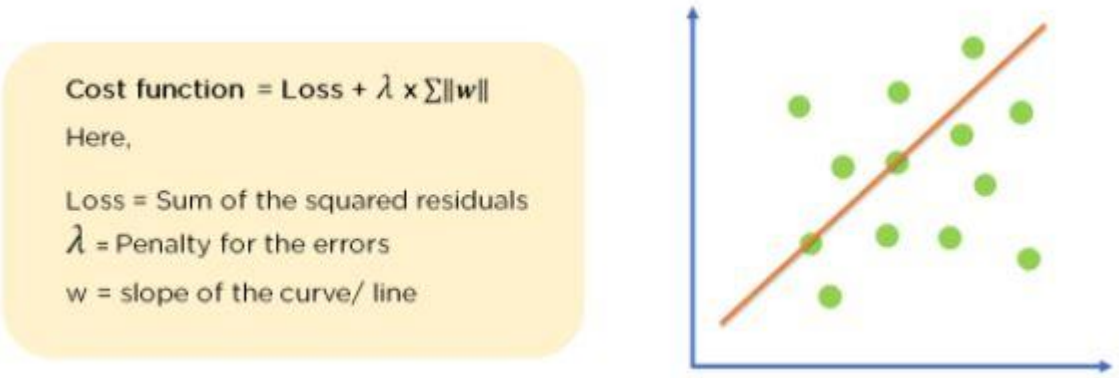


Figure 11: Cost function for Lasso Regression

We can control the coefficient values by controlling the penalty terms, just like we did in Ridge Regression. Again consider a Linear Regression model :

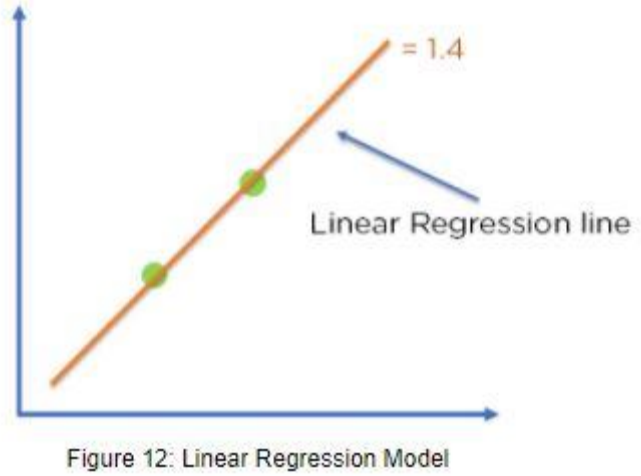


Figure 12: Linear Regression Model

Figure 12: Linear Regression Model

Cost function = Loss + λ x ∑||w||

For Linear Regression line, let's assume,

Loss = 0 (considering the two points on the line)

λ = 1

w = 1.4

Then, Cost function = 0 + 1 x 1.4

= 1.4

For Ridge Regression, let's assume,

Loss = 0.32 + 0.12 = 0.1

λ = 1

w = 0.7

Then, Cost function = 0.1 + 1 x 0.7

= 0.8

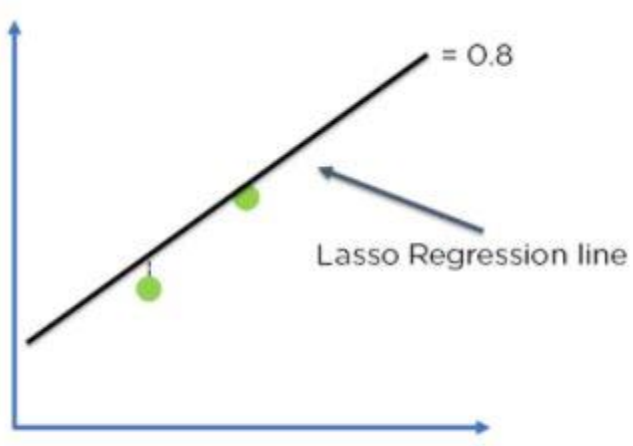


Figure 13: Lasso Regression

Comparing the two models, with all data points, we can see that the Lasso regression line fits the model more accurately than the linear regression line.

4. **What is Gini–impurity index?**

ANS :

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**.

In a nutshell, the Gini impurity index measures the diversity in a set. Let's say, for example, that we have a bag full of balls of several colors. A bag where all the balls have the same color, has a very low Gini impurity index (in fact, it is zero). A bag where all the balls have different colors has a very high Gini impurity index.

Low Gini impurity index

High Gini impurity index

Like everything in math, we need to attach a number to the Gini impurity index, and for this, we turn to our good old friend, probability. In a bag full of balls of different colors, we play the following game. We pick a ball out of this set, randomly, and we look at its color. Then we put the ball back. We proceed to randomly pick another ball from the bag (it could happen that it is the same ball, or a different one, we don't know). We look at its color. We record the two colors we obtained, and we check if they are equal, or different. Here is the main observation of Gini index

Notice that Set 4 is just the mirror image of Set 1, and Set 5 is the mirror image of Set 2. Therefore, their Gini impurity index should be the same as the original one. In other words, the Gini impurity index of Set 4 is 0.375, and that of Set 5 is 0.

Going back to App 1 and App 2 and summarizing, this is what we have calculated for the Gini impurity indices of our sets.

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.

Overfit condition arises **when the model memorizes the noise of the training data and fails to capture important patterns**. A perfectly fit decision tree performs well for training data but performs poorly for unseen test data

6. What is an ensemble technique in machine learning?

ANS: An ensemble method is **a technique which uses multiple independent similar or different models/weak learners to derive an output or make some predictions**. For e.g. A random forest is an ensemble of multiple decision trees

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

Bagging 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. Boosting is an iterative strategy for adjusting an observation's weight based on the previous classification. It attempts to increase the weight of an observation if it was erroneously categorized. Boosting creates good predictive models in general.

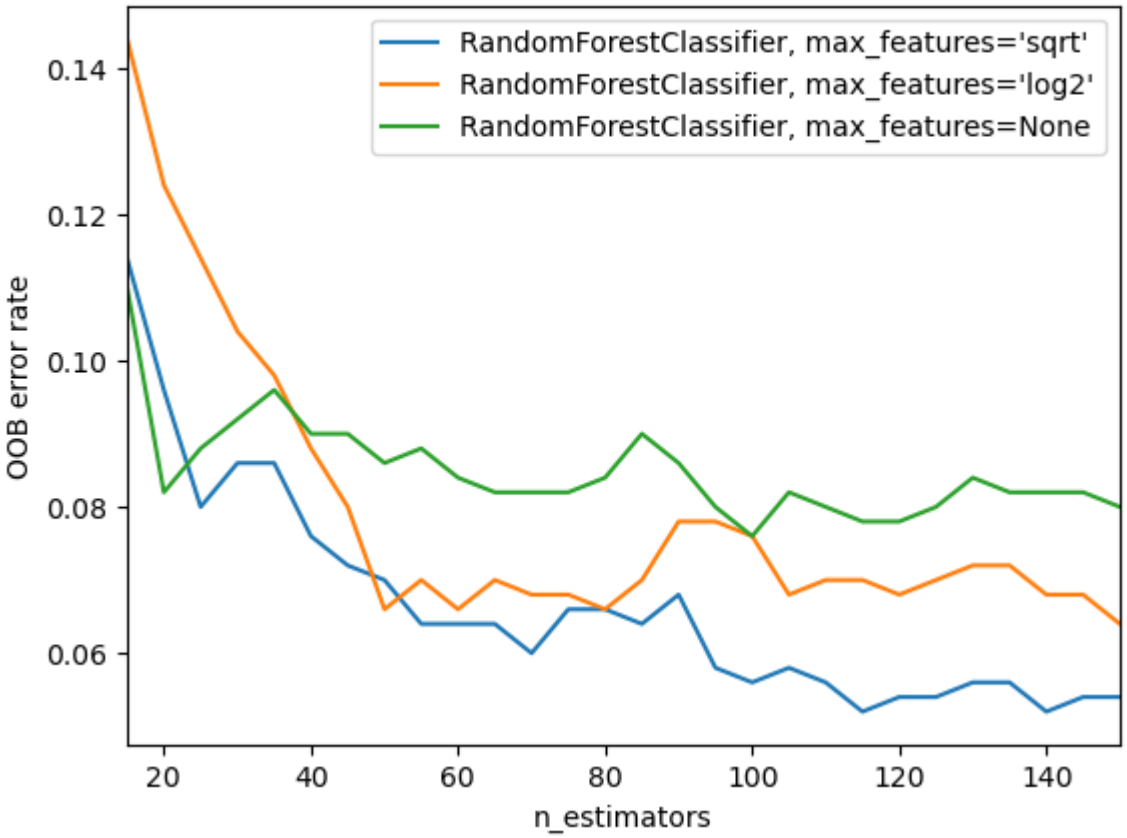
8. What is out-of-bag error in random forests?

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

The `RandomForestClassifier` is trained using *bootstrap aggregation*, where each new tree is fit from a bootstrap sample of the training observations zi=(xi,yi). 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 [1].

The example below demonstrates how the OOB error can be measured at the addition of each new tree during training. The resulting plot allows a practitioner to approximate a suitable value of `n_estimators` at which the error stabilizes.

[1]
T. Hastie, R. Tibshirani and J. Friedman, “Elements of Statistical Learning Ed. 2”, p592-593, Springer, 2009.



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.

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

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.

11. What issues can occur if we have a large learning rate in Gradient Descent?

Deep learning neural networks are trained using the stochastic gradient descent algorithm.

Stochastic gradient descent is an optimization algorithm that estimates the error gradient for the current state of the model using examples from the training dataset, then updates the weights of the model using the back-propagation of errors algorithm, referred to as simply backpropagation.
The amount that the weights are updated during training is referred to as the step size or the "*learning rate*."

Specifically, the learning rate is a configurable hyperparameter used in the training of neural networks that has a small positive value, often in the range between 0.0 and 1.0.

The learning rate controls how quickly the model is adapted to the problem. Smaller learning rates require more training epochs given the smaller changes made to the weights each update, whereas larger learning rates result in rapid changes and require fewer training epochs.
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 challenge of training deep learning neural networks involves carefully selecting the learning rate. It may be the most important hyperparameter for the model.

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

Logistic regression is considered a **generalized linear model** because the outcome always depends on the sum of the inputs and parameters. Or in other words, the output cannot depend on the product (or quotient, etc.)

No, logistic regression only forms linear decision surface, but the examples in the figure are not linearly separable.

13. Differentiate between Adaboost and Gradient Boosting.

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

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

Ans: n 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. The **bias–variance dilemma** or **bias–variance problem** is the conflict in trying to simultaneously minimize these two sources of error that prevent supervised learning algorithms from generalizing beyond their training set:[1][2]

- The *bias* error is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).
- The *variance* is an error from sensitivity to small fluctuations in the training set. High variance may result from an algorithm modeling the random noise in the training data (overfitting).

The **bias–variance decomposition** is a way of analyzing a learning algorithm's expected generalization error with respect to a particular problem as a sum of three terms, the bias, variance, and a quantity called the *irreducible error*, resulting from noise in the problem itself.

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

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.

t 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

Laplace 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

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.

FLIP ROBO