Data Science Interview Questions

1. What is upsampling and downsampling with examples?

The classification data set with skewed class proportions is called an imbalanced data set. Classes which make up a large proportion of the datasets are called majority classes. Those make up smaller proportions are minority classes.

Degree of imbalance Proportion of Minority Class

1. Mild 20-40% of the data set
2. Moderate 1-20% of the data set
3. Extreme <1% of the data set

If we have an imbalanced data set, first try training on the true distribution. If the model works well and generalises, you are done! If not, try the following up sampling and downsampling technique.

1. Up-sampling

Upsampling is the process of randomly duplicating observations from the minority class to reinforce its signal.

First, we will import the resampling module from Scikit-Learn: Module for resampling Python

1- From sklearn.utils import resample

Next, we will create a new Dataframe with an up-sampled minority class. Here are the steps:

1- First, we will separate observations from each class into different Data Frames.

2- Next, we will resample the minority class with replacement, setting the number of samples to match that of the majority class.

3- Finally, we'll combine the up-sampled minority class Data Frame with the original majority class Data Frame.

2. Down-sampling

Downsampling involves randomly removing observations from the majority class to prevent its signal from dominating the learning algorithm.

The process is similar to that of sampling. Here are the steps:

1-First, we will separate observations from each class into different Data Frames.

2-Next, we will resample the majority class without replacement, setting the number of samples to match that of the minority class.

3-Finally, we will combine the down-sampled majority class Data Frame with the original minority class Data Frame.

2.What is the Central limit theorem?

Central Limit Theorem Definition:

The theorem states that as the size of the sample increases, the distribution of the mean across multiple samples will approximate a Gaussian distribution (Normal). Generally, sample sizes equal to or greater than 30 are considered sufficient for the CLT to hold. It means that the distribution of the sample means is normally distributed. The average of the sample means will be equal to the population mean. This is the key aspect of the theorem.

Assumptions:

1. The data must follow the randomization condition. It must be sampled randomly

2. Samples should be independent of each other. One sample should not influence the other samples

3. Sample size should be no more than 10% of the population when sampling is done without replacement

4. The sample size should be sufficiently large.

The mean of the sample means is denoted as:

$$\mu_{\bar{X}} = \mu$$

Where,

$\mu_{\bar{X}}$ = Mean of the sample means

μ= Population mean and,

the standard deviation of the sample mean is denoted as:

$$\sigma_{\bar{X}} = \sigma/\text{sqrt}(n)$$

Where,

$\sigma_{\bar{X}}$ = Standard deviation of the sample mean

σ = Population standard deviation

n = sample size

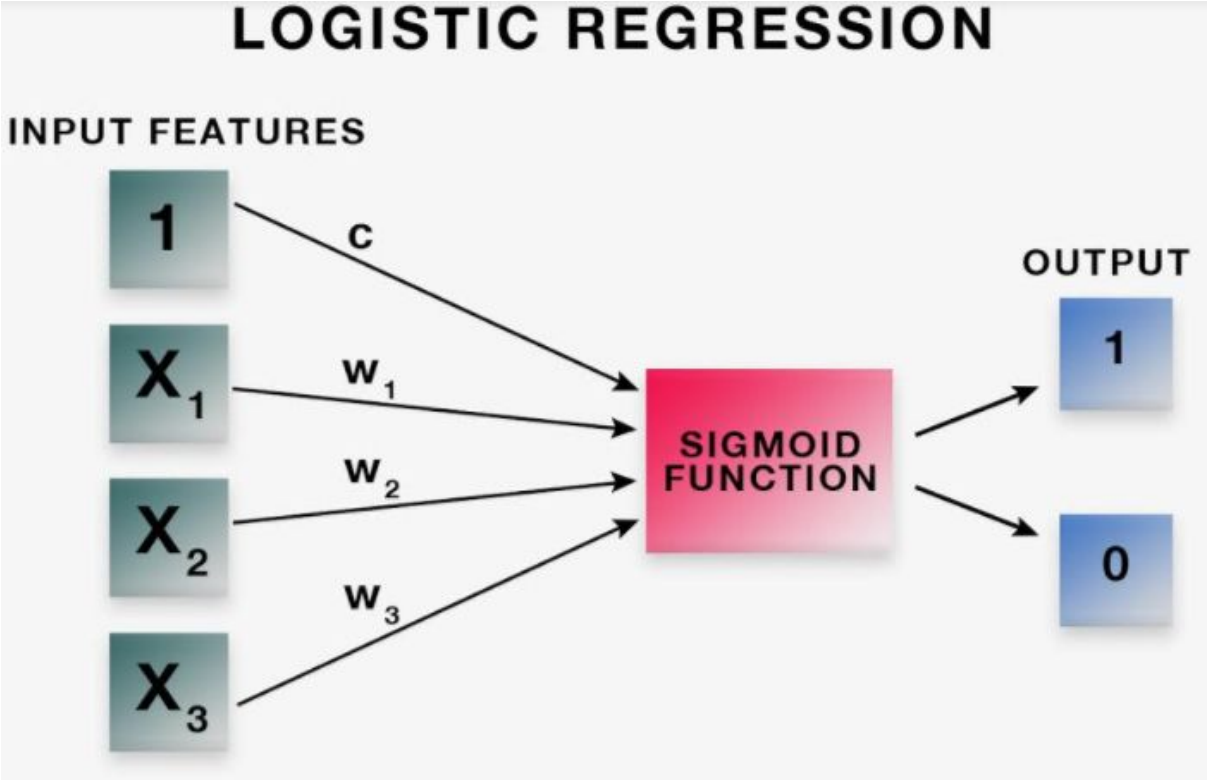
A sufficiently large sample size can predict the characteristics of a population accurately. For Example, we shall take uniformly distributed data.

Randomly distributed data: Even for a randomly (Exponential) distributed data the plot of the means is normally distributed.

The advantage of CLT is that we need not worry about the actual data since the means of it will always be normally distributed. With this, we can create component intervals, perform T-tests and ANOVA tests from the given samples.

3. What is Logistic Regression?

The logistic regression technique involves the dependent variable, which can be represented in the binary (0 or 1, true or false, yes or no) values, which means that the outcome could only be in either one form of two. For example, it can be utilized when we need to find the probability of a successful or failed event.



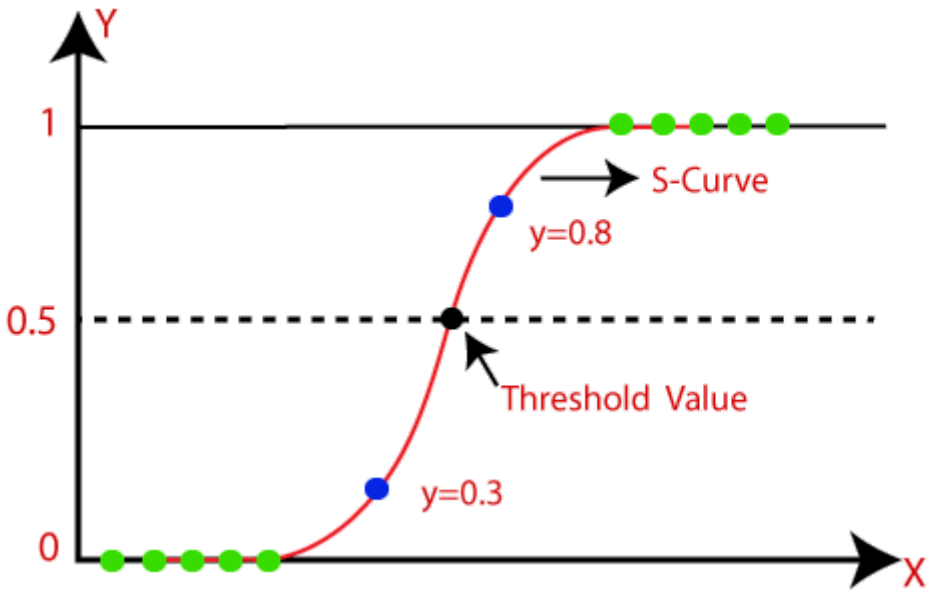
Model

Output = 0 or 1

$Z = WX + B$

$h\Theta(x) = \text{sigmoid}(Z)$

$h\Theta(x) = \log(P(X) / (1 - P(X)) = WX + B$



Cost Function

Logistic regression cost function

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

In [8]:
```
def cost_function(predicted_y):

    error=(-y*np.log(predicted_y)) - ((1-y)*np.log(1-predicted_y))
    cf=(1/X1.shape[0])*sum(error)

    return cf,error
```

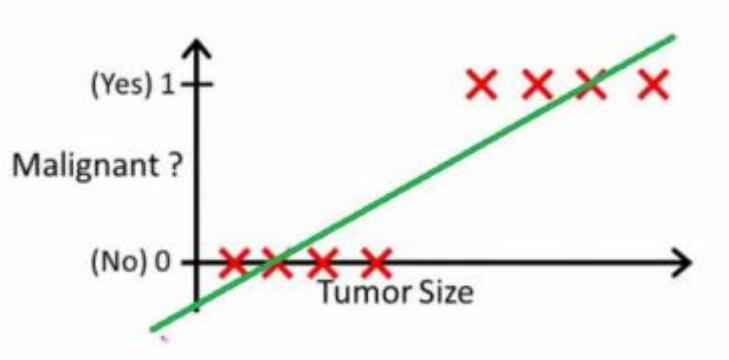
Cost ($h\Theta(x)$, Y(Actual)) = -log ($h\Theta(x)$) if y=1

-log (1 - $h\Theta(x)$) if y=0

This implementation is for binary logistic regression. For data with more than 2 classes, SoftMax regression has to be used.

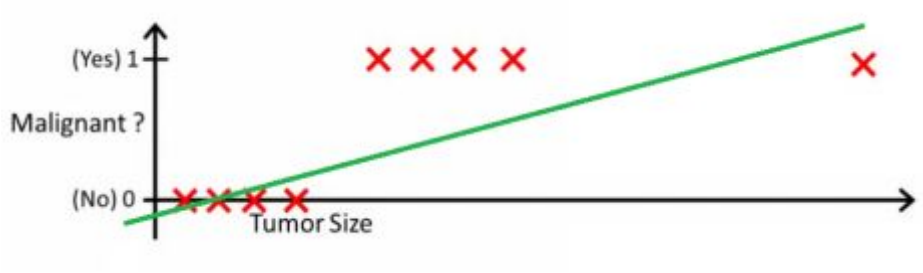
4. Why cannot classification problems be solved using Regression?

With linear regression you fit a polynomial through the data - say, like on the example below, we fit a straight line through {tumor size, tumor type} sample set:



Above, malignant tumors get 1, and non-malignant ones get 0, and the green line is our hypothesis h(x). To make predictions, we may say that for any given tumor size x, if h(x) gets bigger than 0.5, we predict malignant tumors. Otherwise, we predict benignly. It looks like this way, we could correctly predict every single training set sample, but now let's change the task a bit.

Intuitively it's clear that all tumors with larger certain thresholds are malignant. So, let's add another sample with huge tumor size, and run linear regression again:



Now our h(x)>0.5→malignant doesn't work anymore. To keep making correct predictions, we need to change it to h(x)>0.2 or something - but that is not how the algorithm should work.

We cannot change the hypothesis each time a new sample arrives. Instead, we should learn it off the training set data, and then (using the hypothesis we've learned) make correct predictions for the data we haven't seen before.

Linear regression is unbounded.

5. Explain Decision Tree?

A decision tree is a type of supervised learning algorithm that can be used in classification as well as regressor problems. The input to a decision tree can be both continuous as well as categorical. The decision tree works on an if-then statement. Decision tree tries to solve a problem by using tree representation (Node and Leaf)

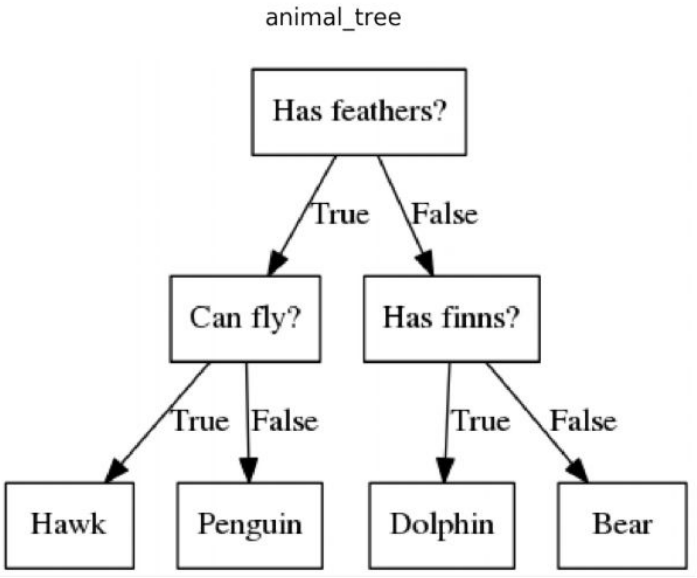
Assumptions while creating a decision tree:

1) Initially all the training set is considered as a root

2) Feature values are preferred to be categorical, if continuous then they are discretized

3) Records are distributed recursively on the basis of attribute values

4) Which attributes are considered to be in the root node or internal node is done by using a statistical approach.



6. Explain the terms Entropy, Information Gain, Gini Index, Reducing Impurity?

There are different attributes which define the split of nodes in a decision tree. There are few algorithms to find the optimal split.

1) ID3(Iterative Dichotomiser 3): This solution uses Entropy and Information gain as metrics to form a better decision tree. The attribute with the highest information gain is used as a root node, and a similar approach is followed after that. Entropy is the measure that characterizes the impurity of an arbitrary collection of examples.

Entropy

Entropy $H(S)$ is a measure of the amount of uncertainty in the (data) set S (i.e. entropy characterizes the (data) set S).

$$H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$$

Where,

- S – The current (data) set for which entropy is being calculated (changes every iteration of the ID3 algorithm)
- C – Set of classes in S C={ yes, no }
- $p(c)$ – The proportion of the number of elements in class c to the number of elements in set S

When $H(S) = 0$, the set S is perfectly classified (i.e. all elements in S are of the same class).

In ID3, entropy is calculated for each remaining attribute. The attribute with the **smallest** entropy is used to split the set S on this iteration. The higher the entropy, the higher the potential to improve the classification here.

Entropy varies from 0 to 1. 0 if all the data belong to a single class and 1 if the class distribution is equal. In this way, entropy will give a measure of impurity in the dataset.
Steps to decide which attribute to split:

1. Compute the entropy for the dataset

2. For every attribute:

- Calculate entropy for all categorical values.
- Take average information entropy for the attribute.
- Calculate gain for the current attribute.

3. Pick the attribute with the highest information gain.

4. Repeat until we get the desired tree.

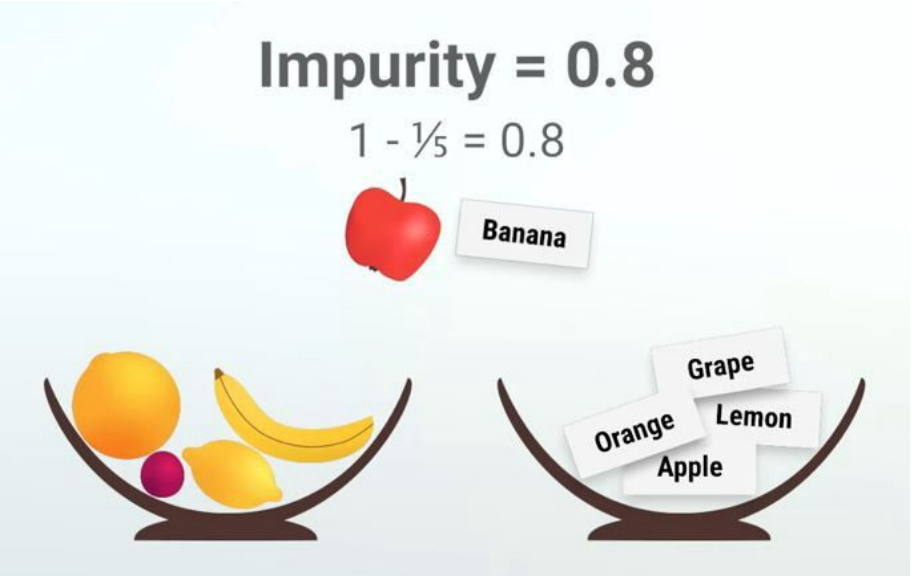
A leaf node is decided when entropy is zero

Information Gain = 1 - Σ (Sb/S) *Entropy

(Sb) Sb - Subset, S - entire data

2) CART Algorithm (Classification and Regression trees): In CART, we use the GINI index as a metric. Gini index is used as a cost function to evaluate split in a dataset Steps to calculate Gini for a split:

1. Calculate Gini for subnodes, using formula sum of the square of probability for success and failure (p2+q2).
2. Calculate Gini for split using the weighted Gini score of each node of that split. Choose the split based on higher Gini value



7. How to control leaf height and Pruning?

To control the leaf size, we can set the parameters: -

1. Maximum depth: Maximum tree depth is a limit to stop the further splitting of nodes when the specified tree depth has been reached during the building of the initial decision tree. NEVER use maximum depth to limit the further splitting of nodes. In other words: use the largest possible value.

2. Minimum split size: Minimum split size is a limit to stop the further splitting of nodes when the number of observations in the node is lower than the minimum split size. This is a good way to limit the growth of the tree. When a leaf contains too few observations, further splitting will result in overfitting (modeling of noise in the data).

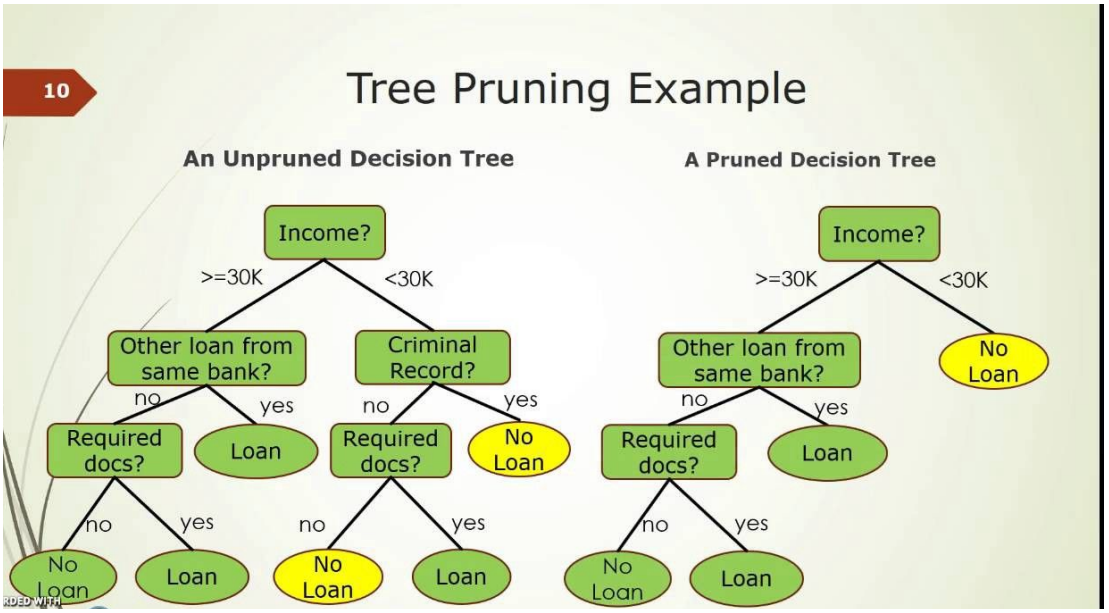
3. Minimum leaf size: Minimum leaf size is a limit to split a node when the number of observations in one of the child nodes is lower than the minimum leaf size. Pruning is mostly done to reduce the chances of overfitting the tree to the training data and reduce the overall complexity of the tree.

There are two types of pruning: **Pre-pruning** and **Post-pruning.**

1. **Pre-pruning** is also known as the early stopping criteria. As the name suggests, the criteria are set as parameter values while building the model. The tree stops growing when it meets any of these pre-pruning criteria, or it discovers the pure classes.

2. **In Post-pruning**, the idea is to allow the decision tree to grow fully and observe the CP value. Next, we prune/cut the tree with the optimal CP (Complexity Parameter) value as the parameter.

The CP (complexity parameter) is used to control tree growth. If the cost of adding a variable is higher, then the value of CP, tree growth stops.



8. How to handle a decision tree for numerical and categorical data?

Decision trees can handle both categorical and numerical variables at the same time as features. There is not any problem in doing that.

Every split in a decision tree is based on a feature.

1. If the feature is categorical, the split is done with the elements belonging to a particular class.
2. If the feature is continuous, the split is done with the elements higher than a threshold.

At every split, the decision tree will take the best variable at that moment. This will be done according to an impurity measure with the split branches. And the fact that the variable used to do split is categorical or continuous is irrelevant (in fact, decision trees categorize continuous variables by creating binary regions with the threshold).

At last, the good approach is to always convert your categoricals to continuous using **LabelEncoder** or **OneHotEncoding**.

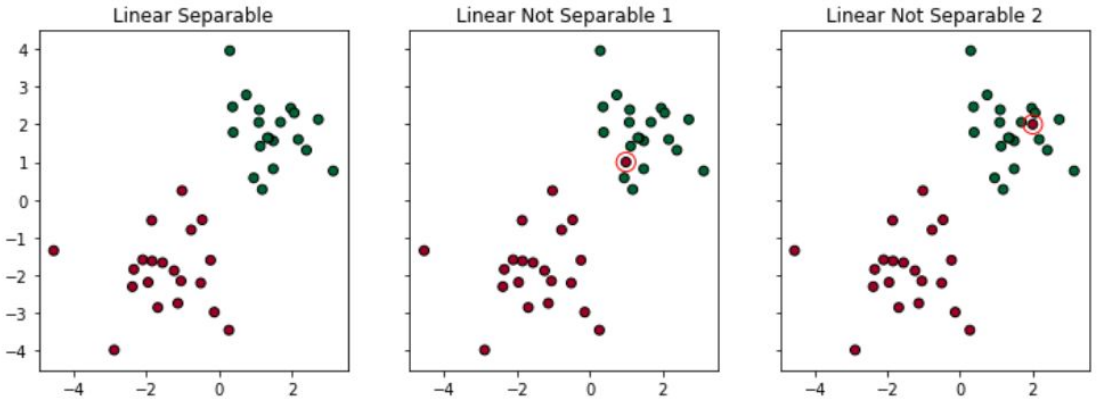
9. What is SVM Classification?

SVM or Large margin classifier is a supervised learning algorithm that uses a powerful technique called SVM for classification.

We have two types of SVM classifiers:

1) Linear SVM: In Linear SVM, the data points are expected to be separated by some apparent gap. Therefore, the SVM algorithm predicts a straight hyperplane dividing the two classes. The hyperplane is also called as maximum margin hyperplane

2) Non-Linear SVM: It is possible that our data points are not linearly separable in a p-dimensional space, but can be linearly separable in a higher dimension. Kernel tricks make it possible to draw nonlinear hyperplanes. Some standard kernels are a) Polynomial Kernel b) RBF kernel (mostly used).



Advantages of SVM classifier:

- SVMs are effective when the number of features is quite large.
- It works effectively even if the number of features is greater than the number of samples.
- Non-Linear data can also be classified using customized hyperplanes built by using kernel trick.
- It is a robust model to solve prediction problems since it maximizes margin.

Disadvantages of SVM classifier:

- The biggest limitation of the Support Vector Machine is the choice of the kernel. The wrong choice of the kernel can lead to an increase in error percentage.
- With a greater number of samples, it starts giving poor performances.
- SVMs have good generalization performance, but they can be extremely slow in the test phase.
- SVMs have high algorithmic complexity and extensive memory requirements due to the use of quadratic programming.

10. What is the Random Forest Algorithm?

Random Forest is an ensemble machine learning algorithm that follows the bagging technique. The base estimators in the random forest are decision trees. Random forest randomly selects a set of features that are used to decide the best split at each node of the decision tree.

Looking at it step-by-step, this is what a random forest model does:

- Random subsets are created from the original dataset (bootstrapping).
- At each node in the decision tree, only a random set of features are considered to decide the best split.
- A decision tree model is fitted on each of the subsets.
- The final prediction is calculated by averaging the predictions from all decision trees.

To sum up, the Random forest randomly selects data points and features and builds multiple trees (Forest).

Random Forest is used for feature importance selection. The attribute (.feature_importances_) is used to find feature importance.

Some Important Parameters: -

1. n_estimators:- It defines the number of decision trees to be created in a random forest.

2. criterion:- "Gini" or "Entropy."

3. min_samples_split:- Used to define the minimum number of samples required in a leaf node before a split is attempted

4. max_features:- It defines the maximum number of features allowed for the split in each decision tree.

5. n_jobs:- The number of jobs to run in parallel for both fit and predict. Always keep (-1) to use all the cores for parallel processing.

11. What is Variance and Bias tradeoff?

In predicting models, the prediction error is composed of two different errors

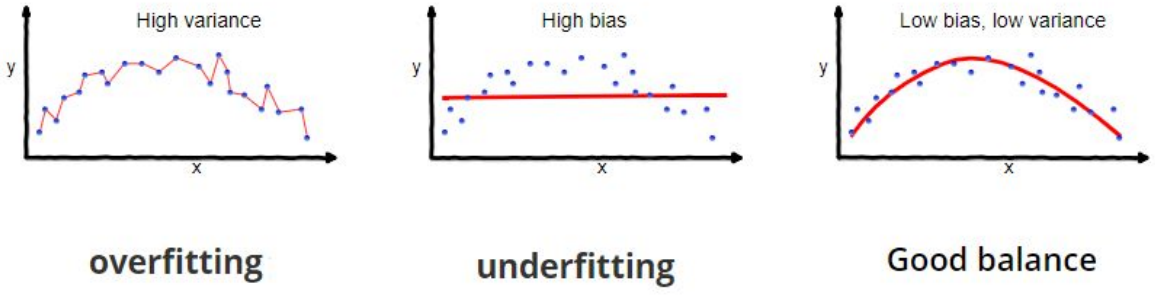
1. Bias

2. Variance

It is important to understand the variance and bias trade-off which helps to minimize the Bias and Variance in the prediction and avoids overfitting & under fitting of the model.

Bias: It is the difference between the expected or average prediction of the model and the correct value which we are trying to predict. Imagine if we are trying to build more than one model by collecting different data sets, and later on, evaluating the prediction, we may end up with different predictions for all the models. So, bias is something which measures how far these model predictions from the correct prediction. It always leads to a high error in training and test data.

Variance: Variability of a model prediction for a given data point. We can build the model multiple times, so the variance is how much the predictions for a given point vary between different realizations of the model.



For example: Voting Republican - 13 Voting Democratic - 16 Non-Respondent - 21 Total - 50 The probability of voting Republican is 13/(13+16), or 44.8%. We put out our press release that the Democrats are going to win by over 10 points; but, when the election comes around, it turns out they lose by 10 points. That certainly reflects poorly on us. Where did we go wrong in our model?

Bias scenarios: using a phonebook to select participants in our survey is one of our sources of bias. By only surveying certain classes of people, it skews the results in a way that will be consistent if we repeated the entire model building exercise. Similarly, not following up with respondents is another source of bias, as it consistently changes the mixture of responses we get. On our bulls-eye diagram, these move us away from the centre of the target, but they would not result in an increased scatter of estimates.

Variance scenarios: the small sample size is a source of variance. If we increased our sample size, the results would be more consistent each time we repeated the survey and prediction. The results still might be highly inaccurate due to our large sources of bias, but the variance of predictions will be reduced

Q12. Explain Ensemble Methods?

Bagging and Boosting

Decision trees have been around for a long time and also known to suffer from bias and variance. You will have a large bias with simple trees and a large variance with complex trees.

Ensemble methods - which combines several decision trees to produce better predictive performance than utilizing a single decision tree. The main principle behind the ensemble model is that a group of weak learners come together to form a strong learner.

Two techniques to perform ensemble decision trees:

1. Bagging

2. Boosting

Bagging (Bootstrap Aggregation) is used when our goal is to reduce the variance of a decision tree. Here the idea is to create several subsets of data from the training sample chosen randomly with replacement. Now, each collection of subset data is used to train their decision trees. As a result, we end up with an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree.

Boosting is another ensemble technique to create a collection of predictors. In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analysing data for errors. In other words, we fit consecutive trees (random sample), and at every step, the goal is to solve for net error from the prior tree.

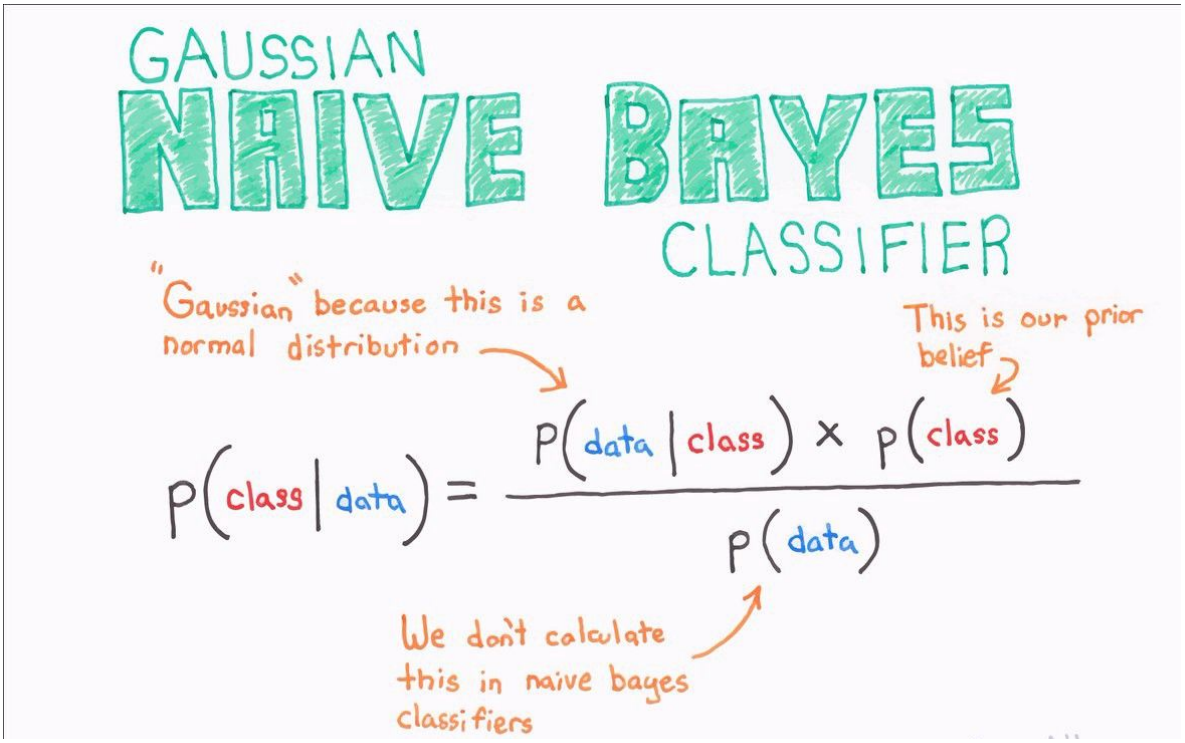
When a hypothesis misclassifies an input, its weight is increased, so that the next hypothesis is more likely to classify it correctly. By combining the whole set at the end converts weak learners into a better performing model.

The different types of boosting algorithms are:

1. AdaBoost
2. Gradient Boosting
3. XGBoost

13. What is Naive Bayes Classification and Gaussian Naive Bayes?

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:



Now, with regards to our dataset, we can apply Bayes' theorem in following way: P(y|X) = {P(X|y) P(y)}/{P(X)} where, y is class variable and X is a dependent feature vector (of size n) where: X = (x_1, x_2, x_3,..,x_n)

	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY GOLF
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

To clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of the dataset)

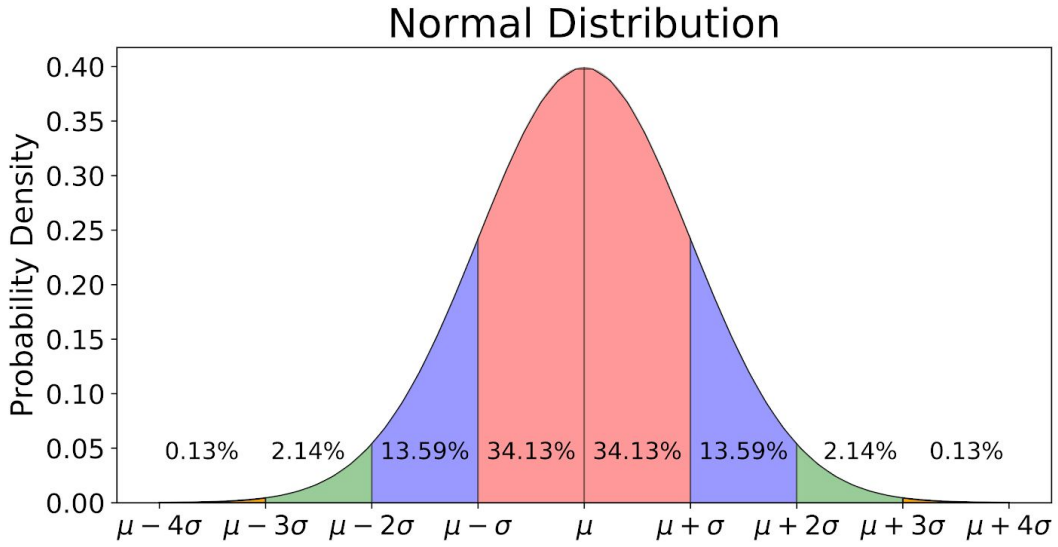
X = (Rainy, Hot, High, False) y = No So basically, P(X|y) here means, the probability of “Not playing golf” given that the weather conditions are “Rainy outlook”, “Temperature is hot”, “high humidity” and “no wind”.

Naive Bayes Classification:

- We assume that no pair of features are dependent. For example, the temperature being ‘Hot’ has nothing to do with the humidity, or the outlook being ‘Rainy’ does not affect the winds. Hence, the features are assumed to be independent.
- Secondly, each feature is given the same weight (or importance). For example, knowing the only temperature and humidity alone can’t predict the outcome accurately. None of the attributes is irrelevant and assumed to be contributing equally to the outcome

Gaussian Naive Bayes:

Continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution. When plotted, it gives a bell-shaped curve which is symmetric about the mean of the feature values as shown below:



This is as simple as calculating the mean and standard deviation values of each input variable (x) for each class value.

Mean (x) = 1/n * sum(x)

Where n is the number of instances, and x is the values for an input variable in your training data.

We can calculate the standard deviation using the following equation:

Standard deviation(x) = sqrt (1/n * sum(xi-mean(x)^2)) When to use what? Standard Naive Bayes only supports categorical features, while Gaussian Naive Bayes only supports continuously valued features.

14. Difference between logistic and linear regression?

Linear and Logistic regression are the most basic form of regression which are commonly used. The essential difference between these two is that Logistic regression is used when the dependent variable is binary. In contrast, Linear regression is used when the dependent variable is continuous, and the nature of the regression line is linear.

Key Differences between Linear and Logistic Regression

Linear regression models data using continuous numeric value. As against, logistic regression models the data in the binary values. Linear regression requires to establish the linear relationship among dependent and independent variables, whereas it is not necessary for logistic regression. In linear regression, the independent variable can be correlated with each other. On the contrary, in the logistic regression, the variable must not be correlated with each other.

15. What is the Confusion Matrix?

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class.

This is the key to the confusion matrix.

It gives us insight not only into the errors being made by a classifier but, more importantly, the types of errors that are being made.

	Class 1 Predicted	Class 2 Predicted
Class 1 Actual	TP	FN
Class 2 Actual	FP	TN

Here,

- Class 1: Positive
- Class 2: Negative

Definition of the Terms:

1. Positive (P): Observation is positive (for example: is an apple).

2. Negative (N): Observation is not positive (for example: is not an apple).

3. True Positive (TP): Observation is positive, and is predicted to be positive.

4. False Negative (FN): Observation is positive, but is predicted negative.

5. True Negative (TN): Observation is negative, and is predicted to be negative.

6. False Positive (FP): Observation is negative, but is predicted positive.

16. What is Accuracy and Misclassification Rate?

Accuracy

Accuracy is defined as the ratio of the sum of True Positive and

True Negative by Total (TP+TN+FP+FN)

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

However, there are problems with accuracy. It assumes equal costs for both kinds of errors. A 99% accuracy can be excellent, good, mediocre, poor, or terrible depending upon the problem.

Misclassification Rate-

Misclassification Rate is defined as the ratio of the sum of False Positive and False Negative by Total (TP+TN+FP+FN) Misclassification Rate is also called Error Rate.

17. What is True Positive Rate & True Negative Rate ?

True Positive Rate:

Sensitivity (SN) is calculated as the number of correct positive predictions divided by the total number of positives. It is also called Recall (REC) or true positive rate (TPR). The best sensitivity is 1.0, whereas the worst is 0.0.

True Negative Rate:

Specificity (SP) is calculated as the number of correct negative predictions divided by the total number of negatives. It is also called a true negative rate (TNR). The best specificity is 1.0, whereas the worst is 0.0.

18. What is False Positive Rate & False Negative Rate?

False Positive Rate False positive rate (FPR) is calculated as the number of incorrect positive predictions divided by the total number of negatives. The best false positive rate is 0.0, whereas the worst is 1.0. It can also be calculated as 1 – specificity.

False Negative Rate False Negative rate (FPR) is calculated as the number of incorrect positive predictions divided by the total number of positives. The best false negative rate is 0.0, whereas the worst is 1.0.

	TRUE	FALSE
POSITIVE	True-Positive (Rule matched and attack present)	False-Positive (Rule matched and no attack present)
NEGATIVE	True-Negative (No rule matched and no attack present)	False-Negative (No rule matched and attack present)

Q19. What are F1 Score, precision and recall?

Recall:

Recall can be defined as the ratio of the total number of correctly classified positive examples divided to the total number of positive examples.

1. High Recall indicates the class is correctly recognized (small number of FN).
2. Low Recall indicates the class is incorrectly recognized (large number of FN).

Recall is given by the relation:

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

Precision:

To get the value of precision, we divide the total number of correctly classified positive examples by the total number of predicted positive examples.

1. High Precision indicates an example labeled as positive is indeed positive (a small number of FP).
2. Low Precision indicates an example labeled as positive is indeed positive (large number of FP).

The relation gives precision:

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

Remember:-

High recall, low precision: This means that most of the positive examples are correctly recognized (low FN), but there are a lot of false positives.
Low recall, high precision: This shows that we miss a lot of positive examples (high FN), but those we predict as positive are indeed positive (low FP).

F-measure/F1-Score:

Since we have two measures (Precision and Recall), it helps to have a measurement that represents both of them. We calculate an F-measure, which uses Harmonic Mean in place of Arithmetic Mean as it punishes the extreme values more.

The F-Measure will always be nearer to the smaller value of Precision or Recall.

$$F\text{-}measure = \frac{2*Recall*Precision}{Recall + Precision}$$

20. What is BaysianSearchCV?

Bayesian search, in contrast to the grid and random search, keeps track of past evaluation results, which they use to form a probabilistic model mapping hyperparameters to a probability of a score on the objective function.

21. What is RandomizedSearchCV?

Randomized search CV is used to perform a random search on hyperparameters. Randomized search CV uses a fit and score method, predict_proba, decision_func, transform, etc.., The parameters of the estimator used to apply these methods are optimized by cross-validated search over parameter settings.

In contrast to GridSearchCV, not all parameter values are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions. The number of parameter settings that are tried is given by n_iter. Code Example:

```
class sklearn.model_selection.RandomizedSearchCV(estimator, param_distributions,
n_iter=10, scoring=None, fit_params=None, n_jobs=None, iid='warn', refit=True,
cv='warn', verbose=0, pre_dispatch='2n_jobs', random_state=None, error_score='raise-
deprecating', return_train_score='warn')
```

22. What is GridSearchCV?

Grid search is the process of performing hyperparameter tuning to determine the optimal values for a given model. CODE Example:-

```
from sklearn.model_selection import GridSearchCV

from sklearn.svm import SVR gsc = GridSearchCV( estimator=SVR(kernel='rbf'),

                                    param_grid={ 'C': [0.1, 1, 100, 1000], 'epsilon': [0.0001,
0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10], 'gamma': [0.0001, 0.001, 0.005, 0.1, 1, 3,
5] }, cv=5, scoring='neg_mean_squared_error', verbose=0, n_jobs=-1)
```

Grid search runs the model on all the possible range of hyperparameter values and outputs the best model

23. What is ZCA Whitening?

Zero Component Analysis:

Making the covariance matrix as the Identity matrix is called whitening. This will remove the first and second-order statistical structure.

ZCA transforms the data to zero means and makes the features linearly independent of each other In some image analysis applications, especially when working with images of the colour and tiny type, it is frequently interesting to apply some whitening to the data before, e.g. training a classifier.