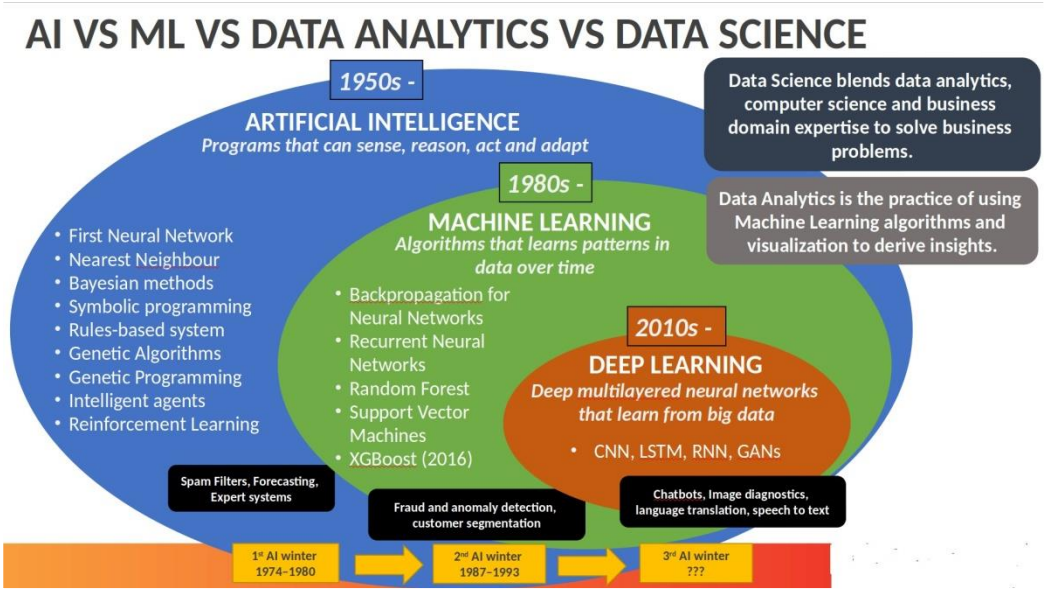
Data Science Interview Questions

(30 days of Interview Preparation)

Q1. What is the difference between AI, Data Science, ML, and DL?

Ans 1 :



Artificial Intelligence: AI is purely math and scientific exercise, but when it became computational, it started to solve human problems formalized into a subset of computer science. Artificial intelligence has changed the original computational statistics paradigm to the modern idea that machines could mimic actual human capabilities, such as decision making and performing more “human” tasks. Modern AI into two categories

1. General AI - Planning, decision making, identifying objects, recognizing sounds, social & business transactions
2. Applied AI - driverless/ Autonomous car or machine smartly trade stocks

Machine Learning: Instead of engineers “teaching” or programming computers to have what they need to carry out tasks, that perhaps computers could teach themselves – learn something without being explicitly programmed to do so. ML is a form of AI where based on more data, and they can change actions and response, which will make more efficient, adaptable and scalable. e.g., navigation apps and recommendation engines. Classified into:-

1. Supervised
2. Unsupervised
3. Reinforcement learning

Data Science: Data science has many tools, techniques, and algorithms called from these fields, plus others –to handle big data

The goal of data science, somewhat similar to machine learning, is to make accurate predictions and to automate and perform transactions in real-time, such as purchasing internet traffic or automatically generating content.

Data science relies less on math and coding and more on data and building new systems to process the data. Relying on the fields of data integration, distributed architecture, automated machine learning, data visualization, data engineering, and automated data-driven decisions, data science can cover an entire spectrum of data processing, not only the algorithms or statistics related to data.

Deep Learning: It is a technique for implementing ML.

ML provides the desired output from a given input, but DL reads the input and applies it to another data. In ML, we can easily classify the flower based upon the features. Suppose you want a machine to look at an image and determine what it represents to the human eye, whether a face, flower, landscape, truck, building, etc.

Machine learning is not sufficient for this task because machine learning can only produce an output from a data set – whether according to a known algorithm or based on the inherent structure of the data. You might be able to use machine learning to determine whether an image was of an "X" – a flower, say – and it would learn and get more accurate. But that output is binary (yes/no) and is dependent on the algorithm, not the data. In the image recognition case, the outcome is not binary and not dependent on the algorithm.

The neural network performs MICRO calculations with computational on many layers. Neural networks also support weighting data for 'confidence. These results in a probabilistic system, vs. deterministic, and can handle tasks that we think of as requiring more 'human-like' judgment.

Q2. What is the difference between Supervised learning, Unsupervised learning and Reinforcement learning?

Ans 2:

Machine Learning

Machine learning is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead.

Building a model by learning the patterns of historical data with some relationship between data to make a data-driven prediction.

Types of Machine Learning

• Supervised Learning

• Unsupervised Learning

• Reinforcement Learning

Supervised learning

In a supervised learning model, the algorithm learns on a labeled dataset, to generate reasonable predictions for the response to new data. (Forecasting outcome of new data)

• Regression

• Classification

Unsupervised learning

An unsupervised model, in contrast, provides unlabelled data that the algorithm tries to make sense of by extracting features, co-occurrence and underlying patterns on its own. We use unsupervised learning for

• Clustering

• Anomaly detection

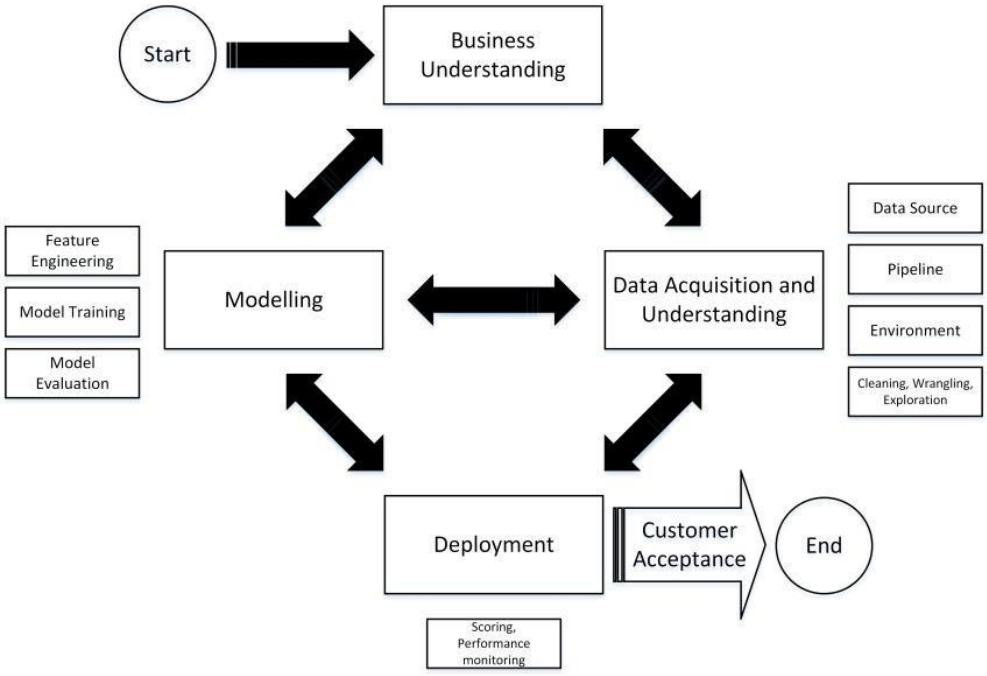
• Association

• Autoencoders

Reinforcement Learning

Reinforcement learning is less supervised and depends on the learning agent in determining the output solutions by arriving at different possible ways to achieve the best possible solution.

Q3. Describe the general architecture of Machine learning.



Business understanding: Understand the give use case, and also, it's good to know more about the domain for which the use cases are built.

Data Acquisition and Understanding: Data gathering from different sources and understanding the data. Cleaning the data, handling the missing data if any, data wrangling, and EDA(Exploratory data analysis).

Modeling: *Feature Engineering* - scaling the data, feature selection - not all features are important. We use the backward elimination method, correlation factors, PCA and domain knowledge to select the features.

Model Training based on trial and error method or by experience, we select the algorithm and train with the selected features.

Model evaluation Accuracy of the model , confusion matrix and cross-validation.

If accuracy is not high, to achieve higher accuracy, we tune the model...either by changing the algorithm used or by feature selection or by gathering more data, etc.

Deployment - Once the model has good accuracy, we deploy the model either in the cloud or Rasberry py or any other place. Once we deploy, we monitor the performance of the model.if its good...we go live with the model or reiterate the all process until our model performance is good.

It's not done yet!!!

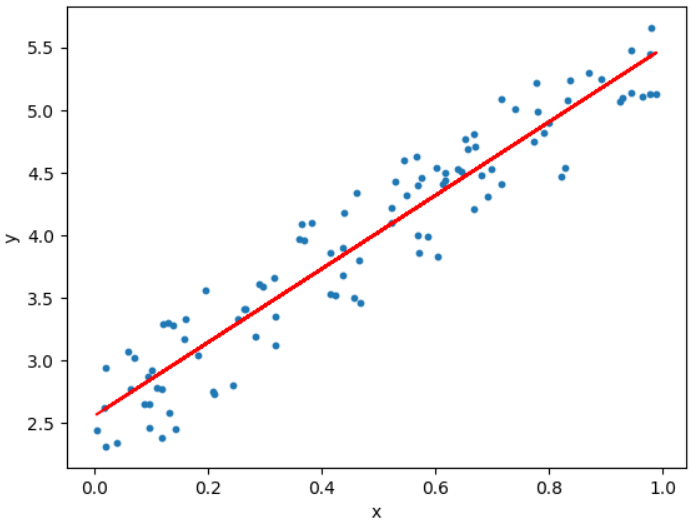
What if, after a few days, our model performs badly because of new data. In that case, we do all the process again by collecting new data and redeploy the model.

Q4. What is Linear Regression?

Ans 4:

Linear Regression tends to establish a relationship between a dependent variable(Y) and one or more independent variable(X) by finding the best fit of the straight line.

The equation for the Linear model is Y = mX+c, where m is the slope and c is the intercept



In the above diagram, the blue dots we see are the distribution of 'y' w.r.t 'x.' There is no straight line that runs through all the data points. So, the objective here is to fit the best fit of a straight line that will try to minimize the error between the expected and actual value.

Q5. OLS Stats Model (Ordinary Least Square)

Ans 5:

OLS is a stats model, which will help us in identifying the more significant features that can has an influence on the output. OLS model in python is executed as:

lm = smf.ols(formula = 'Sales ~ am+constant', data = data).fit() lm.conf_int() lm.summary()

And we get the output as below,

OLS Regression Results

Dep. Variable:	mpg	R-squared:	0.360
Model:	OLS	Adj. R-squared:	0.338
Method:	Least Squares	F-statistic:	16.86
Date:	Wed, 17 Jan 2018	Prob (F-statistic):	0.000285
Time:	14:07:51	Log-Likelihood:	-95.242
No. Observations:	32	AIC:	194.5
Df Residuals:	30	BIC:	197.4
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
constant	17.1474	1.125	15.247	0.000	14.851	19.444
am	7.2449	1.764	4.106	0.000	3.642	10.848

Omnibus:	0.480	Durbin-Watson:	1.065
Prob(Omnibus):	0.787	Jarque-Bera (JB):	0.589
Skew:	0.051	Prob(JB):	0.745
Kurtosis:	2.343	Cond. No.	2.46

The higher the t-value for the feature, the more significant the feature is to the output variable. And also, the p-value plays a rule in rejecting the Null hypothesis(Null hypothesis stating the features has zero significance on the target variable.). **If the p-value is less than 0.05(95% confidence interval) for a feature, then we can consider the feature to be significant.**

Q6. What is L1 Regularization (L1 = lasso) ?

Ans 6:

The main objective of creating a model(training data) is making sure it fits the data properly and reduce the loss. Sometimes the model that is trained which will fit the data but it may fail and give a poor performance during analyzing of data (test data). This leads to overfitting. Regularization came to overcome overfitting.

Lasso Regression (**Least Absolute Shrinkage and Selection Operator**) adds “Absolute value of magnitude” of coefficient, as penalty term to the loss function.

Lasso shrinks the less important feature's coefficient to zero; thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

Methods like Cross-validation, Stepwise Regression are there to handle overfitting and perform feature selection work well with a small set of features. These techniques are good when we are dealing with a large set of features.

Along with shrinking coefficients, the **lasso performs feature selection**, as well. (Remember the 'selection' in the lasso full-form?) Because some of the coefficients become exactly zero, which is equivalent to the particular feature being excluded from the model.

Q7. **L2 Regularization(L2 = Ridge Regression)**

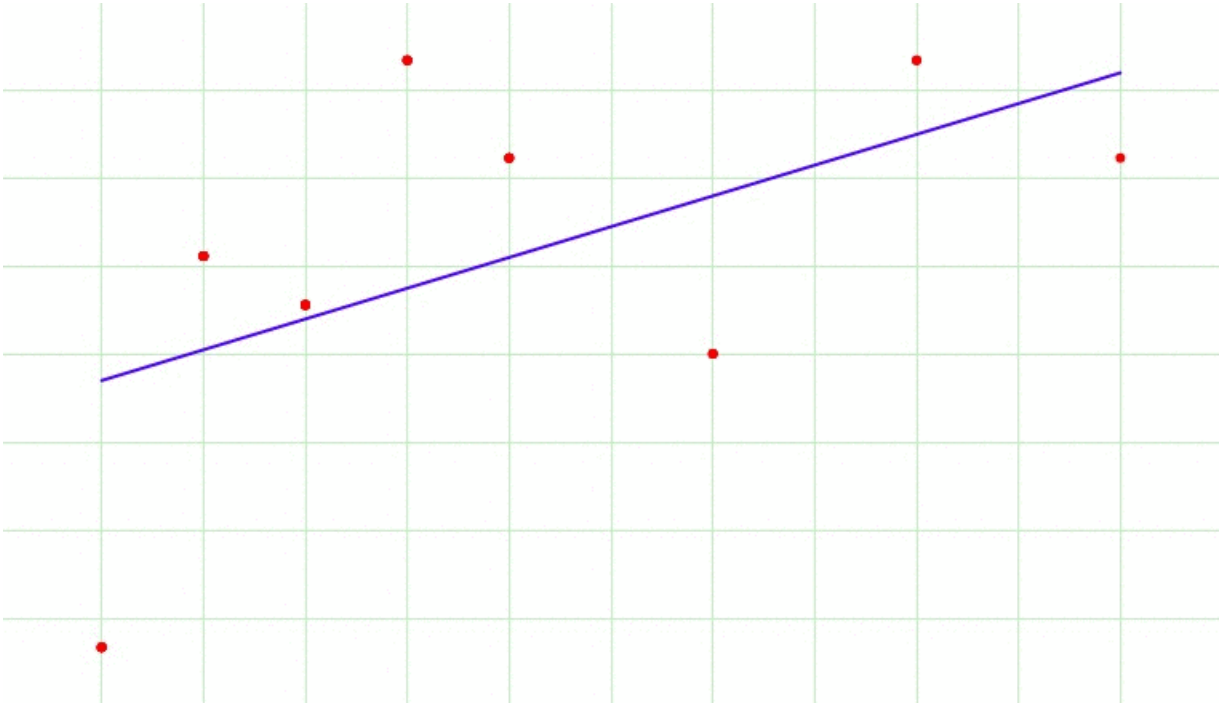
Ans 7:

$$Cost\,function = Loss + \frac{\lambda}{2m} * \sum \|w\|^2$$

Overfitting happens when the model learns signal as well as noise in the training data and wouldn't perform well on new/unseen data on which model wasn't trained on.

To avoid overfitting your model on training data like **cross-validation sampling**, **reducing the number of features, pruning**, **regularization**, etc.

So to avoid overfitting, we perform Regularization.



The Regression model that uses L2 regularization is called Ridge Regression.

The formula for Ridge Regression:-

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

$$\min_\theta J(\theta)$$

Regularization adds the penalty as model complexity increases. The regularization parameter (lambda) penalizes all the parameters except intercept so that the model generalizes the data and won't overfit.

Ridge regression adds "squared magnitude of the coefficient" as penalty term to the loss function. Here the box part in the above image represents the L2 regularization element/term.

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p}x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

Lambda is a hyperparameter.

If lambda is zero, then it is equivalent to OLS. But **if lambda is very large, then it will add too much weight, and it will lead to under-fitting**.

Ridge regularization **forces the weights to be small but does not make them zero** and **does not give the sparse solution**.

Ridge is **not robust to outliers** as square terms blow up the error differences of the outliers, and the regularization term tries to fix it by penalizing the weights

Ridge regression performs better when all the input features influence the output, and all with **weights are of roughly equal size**.

L2 regularization can **learn complex data patterns**.

Q8. What is R square(where to use and where not)?

Ans 8.

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

The definition of R-squared is the percentage of the response variable variation that is explained by a linear model.

R-squared = Explained variation / Total variation

R-squared is always between 0 and 100%.

0% indicates that the model explains none of the variability of the response data around its mean.

100% indicates that the model explains all the variability of the response data around its mean.

In general, the higher the R-squared, the better the model fits your data.

Sum Squared Regression Error

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}}$$

Sum Squared Total Error

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

There is a problem with the R-Square. The problem arises when we ask this question to ourselves.** Is it good to help as many independent variables as possible?**

The answer is No because we understood that each independent variable should have a meaningful impact. But, even** if we add independent variables which are not meaningful**, will it improve R-Square value?

Yes, this is the basic problem with R-Square. How many junk independent variables or important independent variable or impactful independent variable you add to your model, the R-Squared value will always increase. It will never decrease with the addition of a newly independent variable, whether it could be an impactful, non-impactful, or bad variable, so we need another way to measure equivalent R-Square, which penalizes our model with any junk independent variable.

So, we calculate the **Adjusted R-Square** with a better adjustment in the formula of generic R-square.

$$R^2_{adjusted} = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

where

R^2 = sample R-square

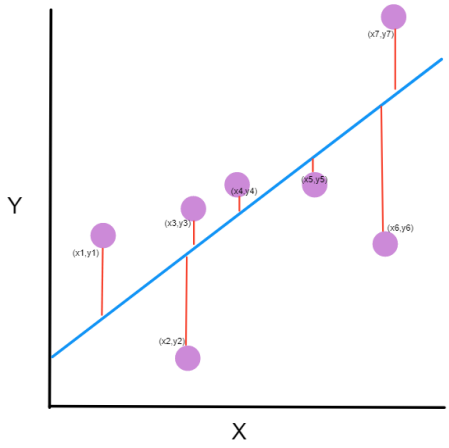
p = Number of predictors

N = Total sample size.

Q9. What is Mean Square Error?

The mean squared error tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the "errors") and squaring them.

Giving an intuition



The line equation is **y=Mx+B**. We want to find **M (slope)** and **B (y-intercept)** that minimizes the squared error.

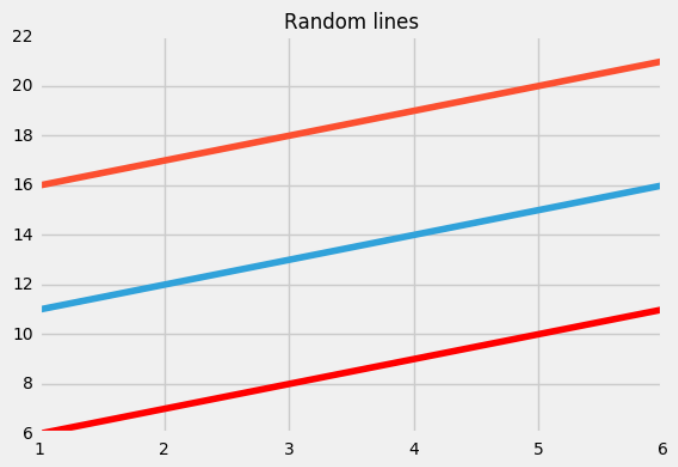
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Q10. Why Support Vector Regression? Difference between SVR and a simple regression model?

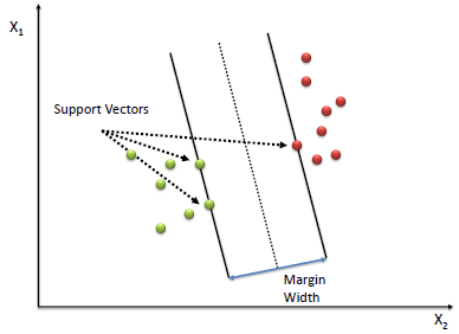
Ans 10:
In simple linear regression, try to minimize the error rate. But in SVR, we try to fit the error within a certain threshold.

Main Concepts:-

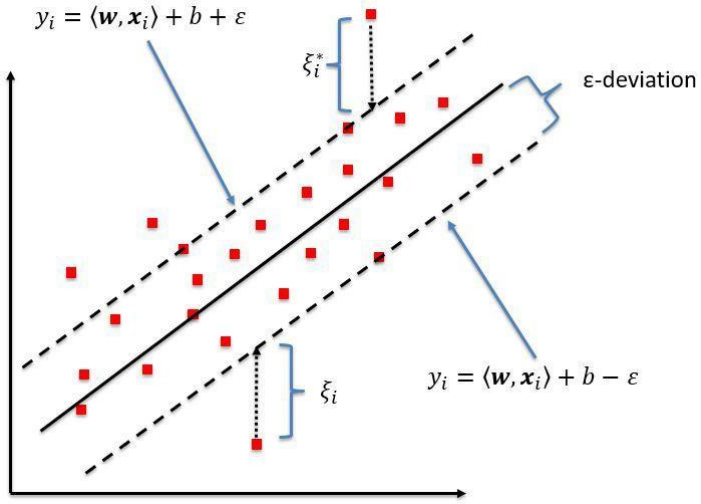
1. **Boundary**
2. **Kernel**
3. **Support Vector**
4. **Hyper Plane**



Blueline: Hyper Plane; Red Line: Boundary-Line



Our best fit line is the one where the hyperplane has the maximum number of points.
We are trying to do here is trying to decide a decision boundary at 'e' distance from the original hyperplane such that data points closest to the hyperplane or the support vectors are within that boundary line



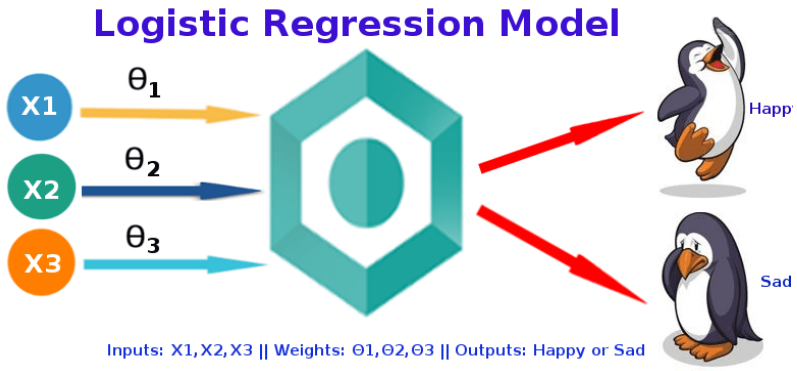
DATA SCIENCE INTERVIEW PREPARATION (30 Days of Interview Preparation)

#DAY 02

Q1. What is Logistic Regression?

Answer:

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 fail event.



Logistic Regression is used when the dependent variable (target) is categorical.

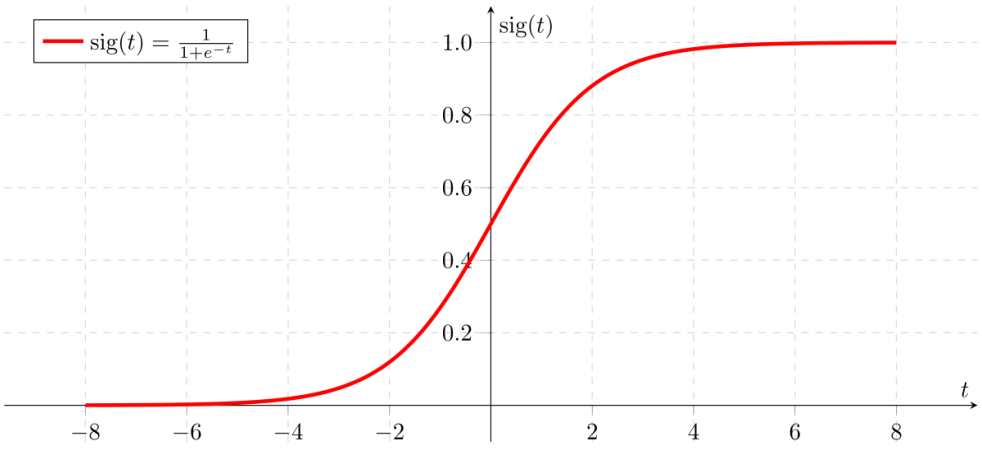
Model

Output = 0 or 1

Z = WX + B

hΘ(x) = sigmoid (Z)

hΘ(x) = log(P(X) / 1 - P(X)) = WX +B



If 'Z' goes to infinity, Y(predicted) will become 1, and if 'Z' goes to negative infinity, Y(predicted) will become 0.

The output from the hypothesis is the estimated probability. This is used to infer how confident can predicted value be actual value when given an input X.

Cost Function

$Cost(h_\Theta(x), y) = -y \log(h_\Theta(x)) - (1-y) \log(1- h_\Theta(x))$

If y = 1, (1-y) term will become zero, therefore – log ($h_\Theta(x)$) alone will be present

If y = 0, (y) term will become zero, therefore – log (1- $h_\Theta(x)$) alone will be present

$$\text{Cost}(h_\Theta(x), Y(\text{Actual})) = \begin{cases} -\log(h_\Theta(x)) & \text{if } y=1 \\ -\log(1 - h_\Theta(x)) & \text{if } y=0 \end{cases}$$

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

Q2. Difference between logistic and linear regression?

Answer:

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.

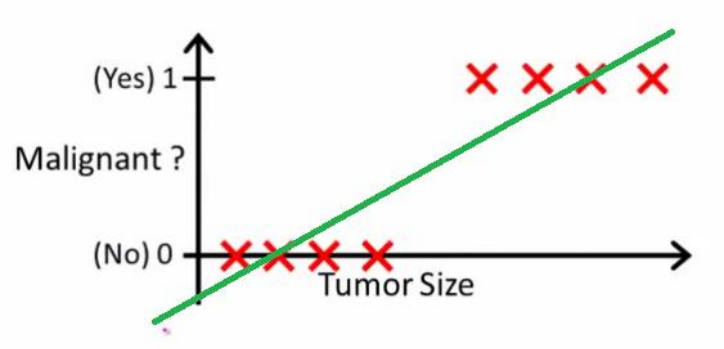
Linear regression requires to establish the linear relationship among dependent and independent variables, whereas it is not necessary for logistic regression.

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.

Q3. Why we can’t do a classification problem using Regression?

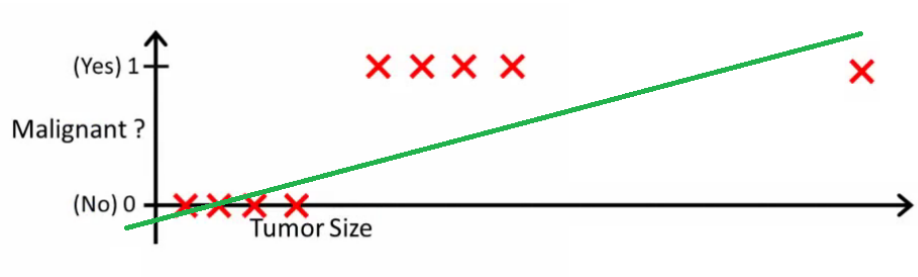
Answer:-

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 larger certain threshold are malignant. So let's add another sample with huge tumor size, and run linear regression again:



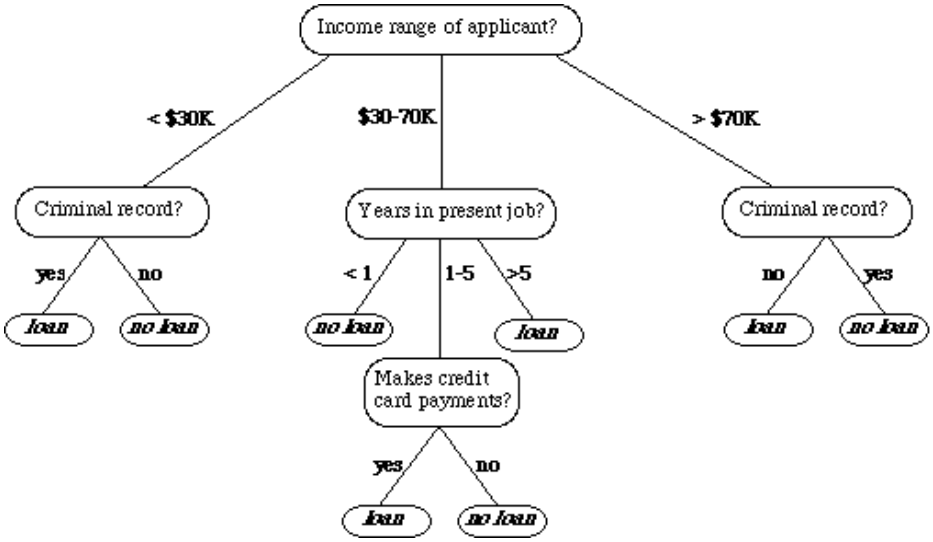
Now our h(x)>0.5⟶malignant doesn't work anymore. To keep making correct predictions, we need to change it to h(x)>0.2 or something - but that not how the algorithm should work.

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.

Q4. What is 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 root node or internal node is done by using a statistical approach.



Q5. Entropy, Information Gain, Gini Index, Reducing Impurity?

Answer:

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:

2.1 Calculate entropy for all categorical values.

2.2 Take average information entropy for the attribute.

2.3 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 - $\sum$ (Sb/S)*Entropy (Sb)

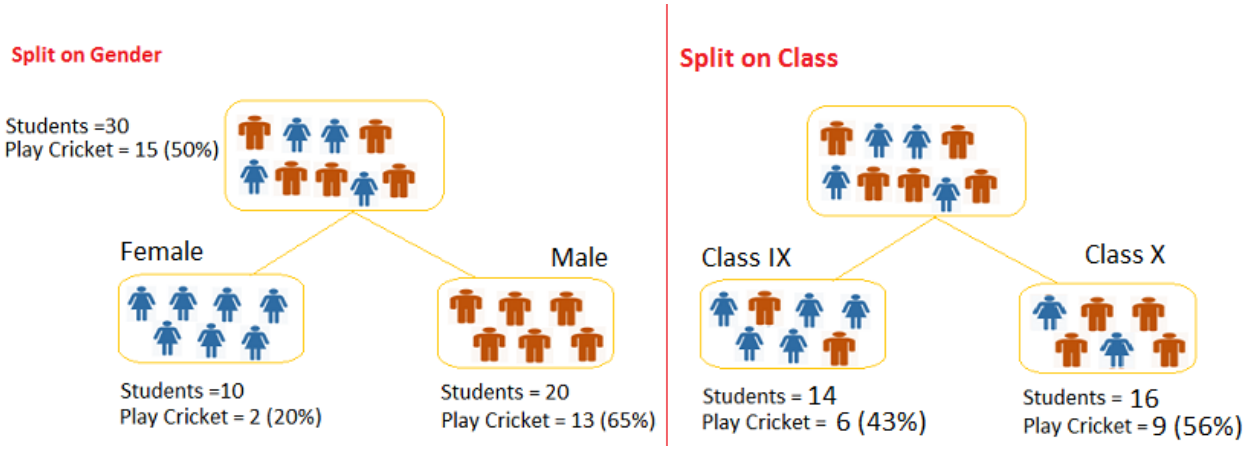
Sb - Subset, S - entire data

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 weighted Gini score of each node of that split.

Choose the split based on higher Gini value



Split on Gender:

Gini for sub-node Female = (0.2)*(0.2)+(0.8)*(0.8)=0.68

Gini for sub-node Male = (0.65)*(0.65)+(0.35)*(0.35)=0.55

Weighted Gini for Split Gender = (10/30)*0.68+(20/30)*0.55 = 0.59

Similar for Split on Class:

Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51

Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51

Weighted Gini for Split Class = (14/30)*0.51+(16/30)*0.51 = 0.51

Here Weighted Gini is high for gender, so we consider splitting based on gender

Q6. How to control leaf height and Pruning?

Answer:

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.

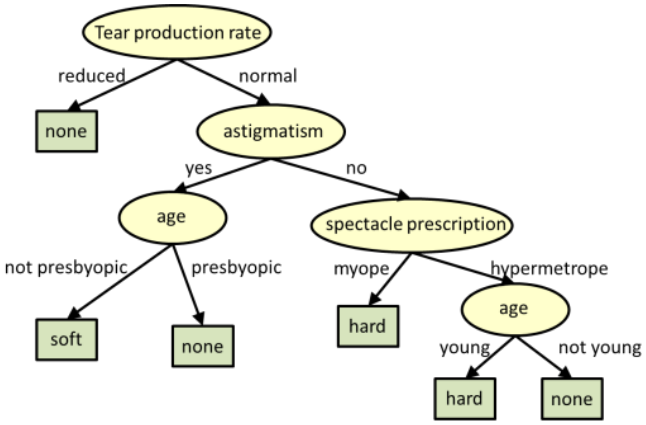
Pruning is mostly done to reduce the chances of overfitting the tree to the training data and reduce the overall complexity of the tree.

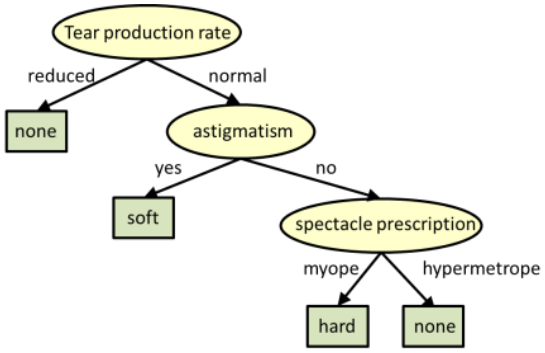
There are two types of pruning: **Pre-pruning** and **Post-pruning**.

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.





Q7. How to handle a decision tree for numerical and categorical data?

Answer:

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.**

Q8. What is the Random Forest Algorithm?

Answer:

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:

1. Random subsets are created from the original dataset (**bootstrapping**).
2. At each node in the decision tree, only a random set of features are considered to decide the best split.
3. A decision tree model is fitted on each of the subsets.
4. 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.**

Q9. What is Variance and Bias tradeoff?

Answer:

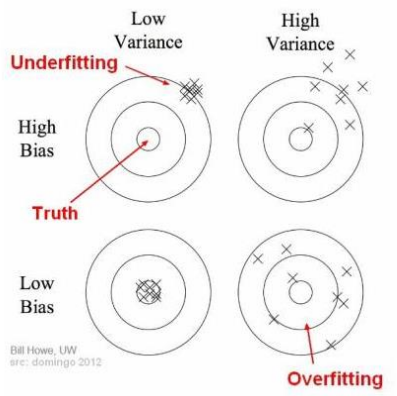
In predicting models, the prediction error is composed of two different errors

1. Bias
2. Variance

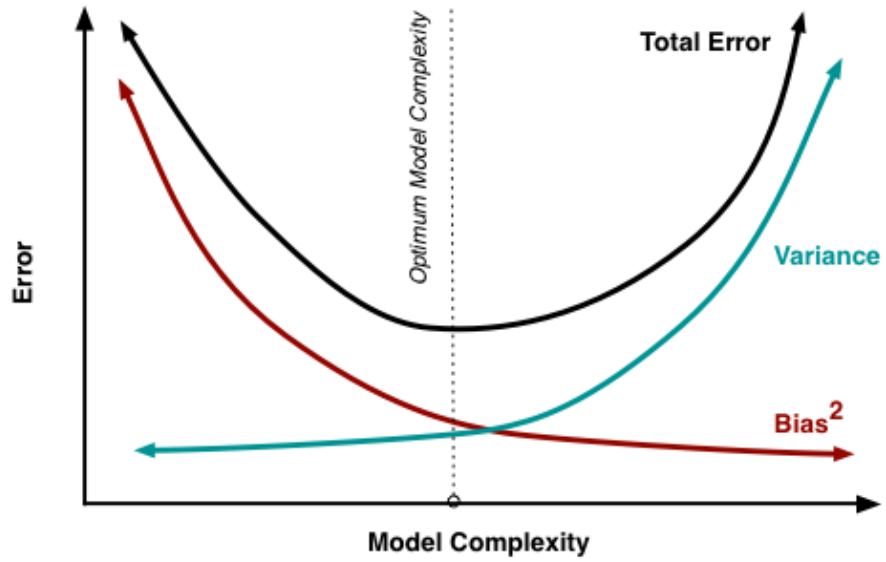
It is important to understand the variance and bias trade-off which tells about to minimize the Bias and Variance in the prediction and avoids overfitting & under fitting of the model.

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 by different prediction for all the models. So, bias is something which measures how far these model prediction 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 scenario's: 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 center 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



Q10. What are Ensemble Methods?

Answer

1. **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 analyzing 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**

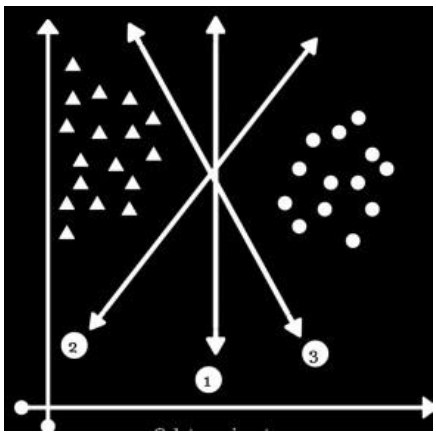
Q11. What is SVM Classification?

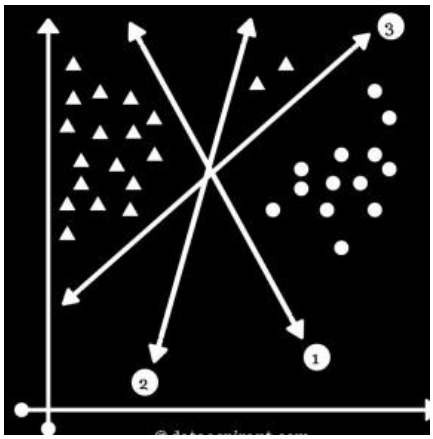
Answer:

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

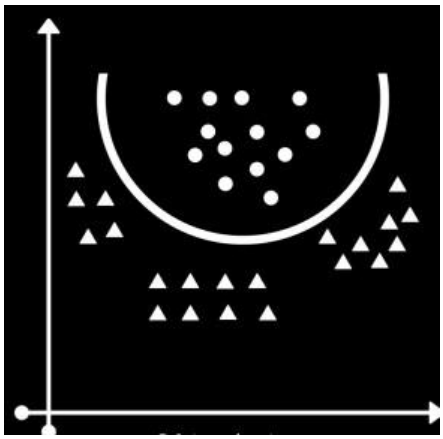
We have two types of SVM classifiers:

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:

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

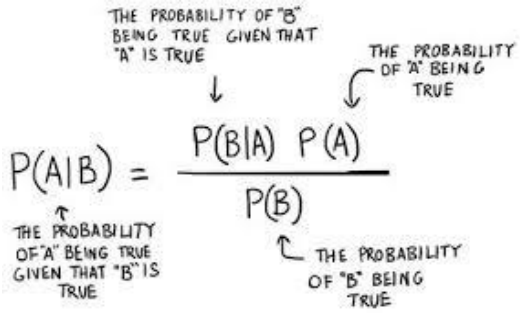
Disadvantages of SVM classifier:

1) 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.
2) With a greater number of samples, it starts giving poor performances.
3) SVMs have good generalization performance, but they can be extremely slow in the test phase.
4) SVMs have high algorithmic complexity and extensive memory requirements due to the use of quadratic programming.

Q11. What is Naive Bayes Classification and Gaussian Naive Bayes

Answer:

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)

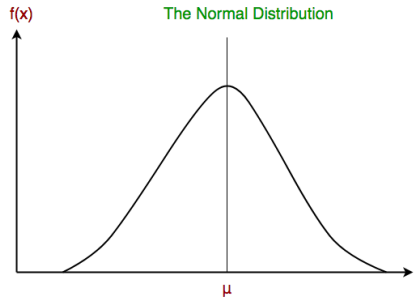
X = (Rainy, Hot, High, False) y = No So basically, P(X|y) here means, the probability of "Not playing golf" given that the weather conditions are "Rainy outlook", "Temperature is hot", "high humidity" and "no wind".

Naive Bayes Classification:

1. 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.
2. 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.

Q12. What is the Confusion Matrix?

Answer:

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.

Q13. What is Accuracy and Misclassification Rate?

Answer:

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.

	Actual Positive	Actual Negative
Predicted Positive	True Positive(TP)	False Positive(FP) (Type 1 Error)
Predicted Negative	False Negative(FN) (Type 2 Error)	True Negative(TN)

$$\text{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total Population}}$$

$$\text{Error Rate/Misclassification rate} = \frac{\text{False Positive + False Negative}}{\text{Total Population}}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{Predicted Positive(TP+FP)}}$$

$$\text{Sensitivity/Recall} = \frac{\text{True Positive}}{\text{Actual Positive(TP+FN)}}$$

$$\text{Specificity} = \frac{\text{True Negative}}{\text{Actual Negative(FP+TN)}}$$

$$\text{F1 Score} = \frac{2 * (\text{Recall} * \text{Precision})}{\text{Recall + Precision}}$$

Q14. True Positive Rate & True Negative Rate

Answer:

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.

$$SN = \frac{TP}{TPFN} = \frac{TP}{P}$$

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.

$$SN = \frac{TP}{TPFN} = \frac{TP}{P}$$

Q15. 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.

$$SN = \frac{TP}{TPFN} = \frac{TP}{P}$$

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.

Name	Formula	Explanation
True Positive Rate (TP rate)	TP / (TP + FP)	The closer to 1, the better. TP rate = 1 when FP = 0. (No false positives)
True Negative Rate (TN rate)	TN / (TN + FN)	The closer to 1, the better. TN rate = 1 when FN = 0. (No false negatives)
False Positive Rate (FP rate)	FP / (FP + TN)	The closer to 0, the better. FP rate = 0 when FP = 0. (No false positives)
False Negative Rate (FN rate)	FN / (FN + TP)	The closer to 0, the better. FN rate = 0 when FN = 0. (No false negatives)

Q16. What are F1 Score, precision and recall?

Recall:-
Recall can be defined as the ratio of the total number of correctly classified positive examples divide 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.

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

Q17. What is RandomizedSearchCV?

Answer:
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='2*n_jobs', random_state=None, error_score='raise-deprecating', return_train_score='warn')

Q18. What is GridSearchCV?

Answer:

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

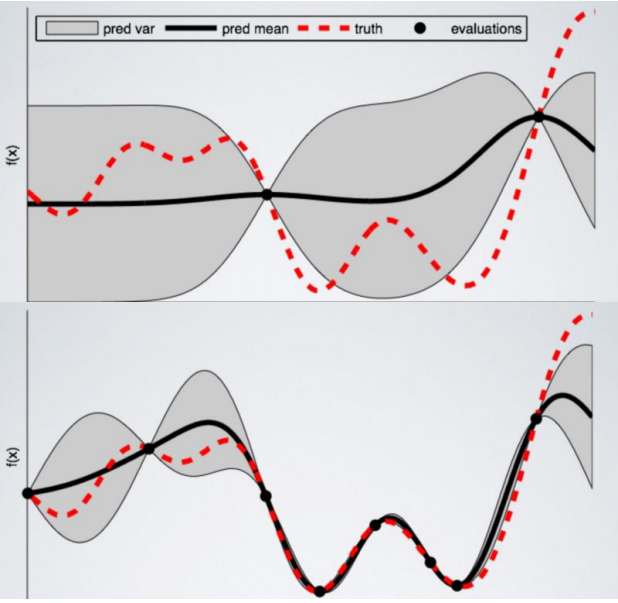
from sklearn.model_selection import GridSearchCV from sklearn.svm import SVR gsc = GridSearchCV(estimator=SVR(kernel='rbf'), param_grid={ 'C': [0.1, 1, 100, 1000], 'epsilon': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10], 'gamma': [0.0001, 0.001, 0.005, 0.1, 1, 3, 5] }, cv=5, scoring='neg_mean_squared_error', verbose=0, n_jobs=-1)

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

Q19. What is BaysianSearchCV?

Answer:
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.

$$P(score \mid hyperparameters)$$



Code:

```
from skopt import BayesSearchCV
opt = BayesSearchCV(
   SVC(),
```

```
  {
    'C': (1e-6, 1e+6, 'log-uniform'),
    'gamma': (1e-6, 1e+1, 'log-uniform'),
    'degree': (1, 8),  # integer valued parameter
    'kernel': ['linear', 'poly', 'rbf']
  },
  n_iter=32,
  cv=3)
```

Q20. What is ZCA Whitening?

Answer:

Zero Component Analysis:

Making the co-variance 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 color and tiny typ e, it is frequently interesting to apply some whitening to the data before, e.g. training a classifier.