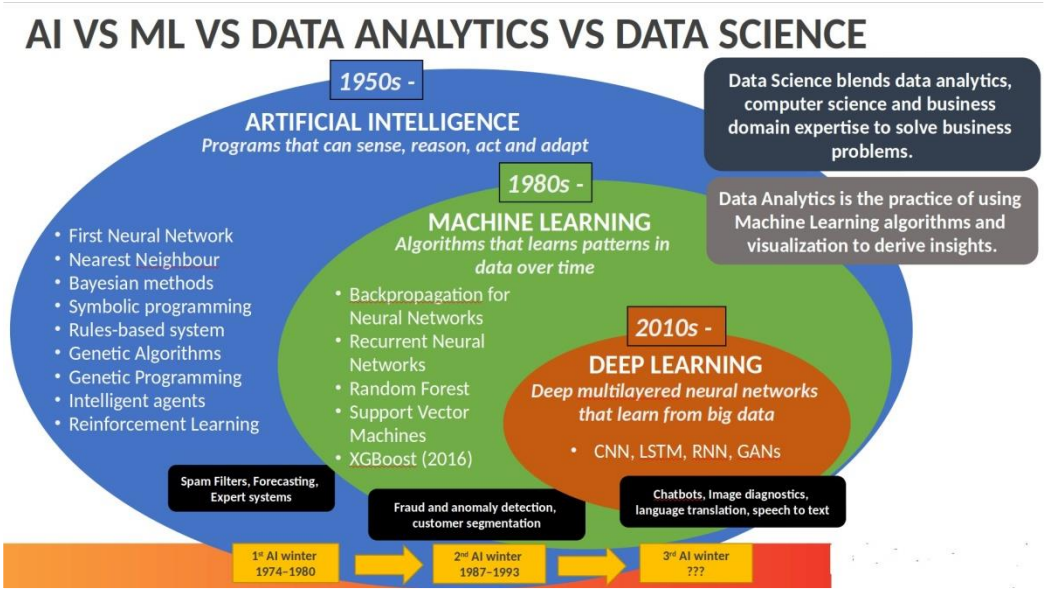
Data Science Interview Questions

(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

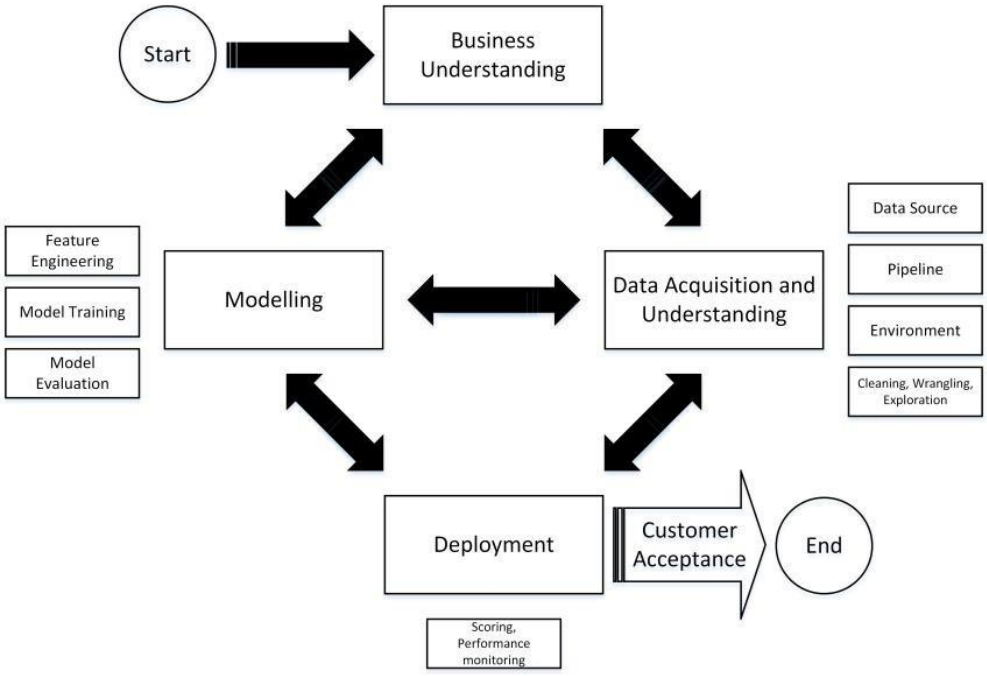
• Association

• 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!!!

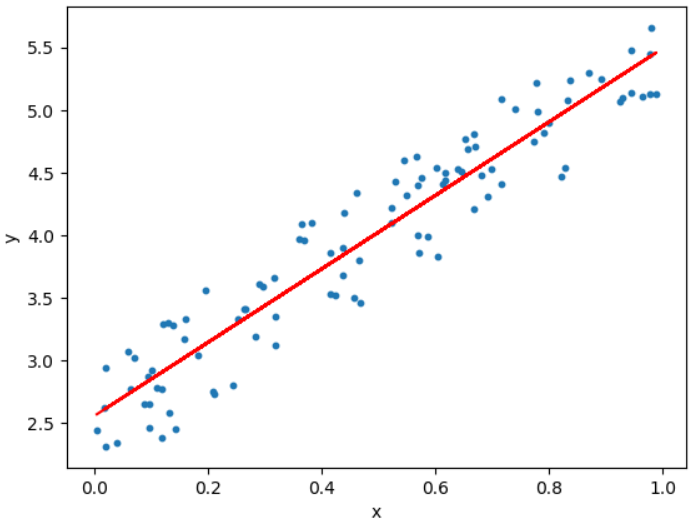
What if, after a few days, our model performs badly because of new data. In that case, we do all the process again by collecting new data and redeploy the model.

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)**

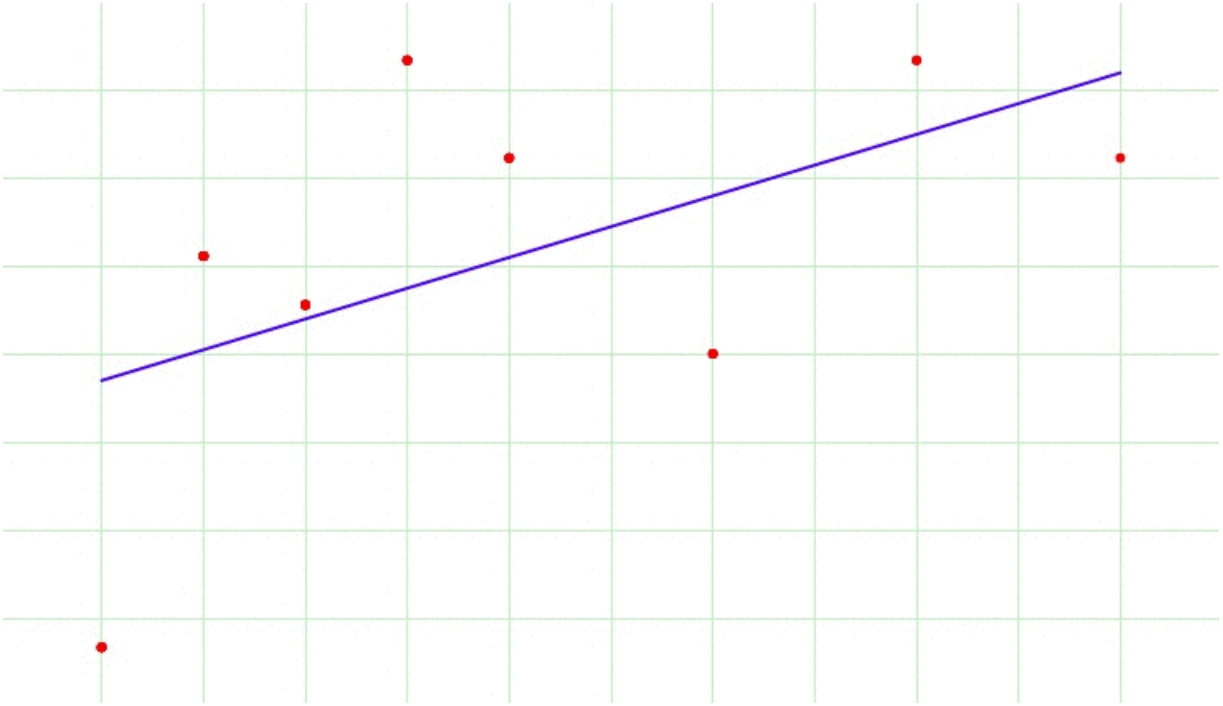
Ans 7:

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

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

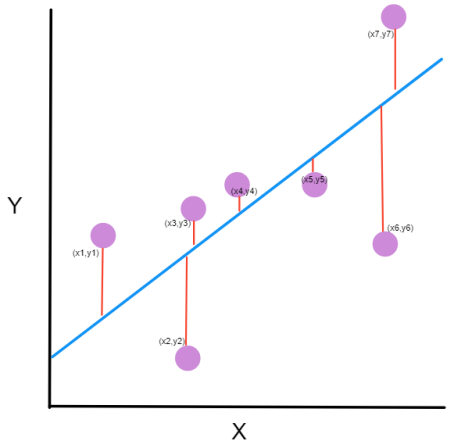
p = Number of predictors

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.

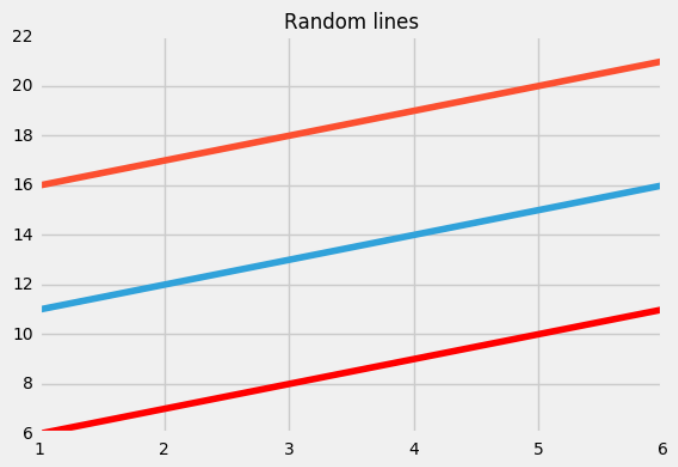
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

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

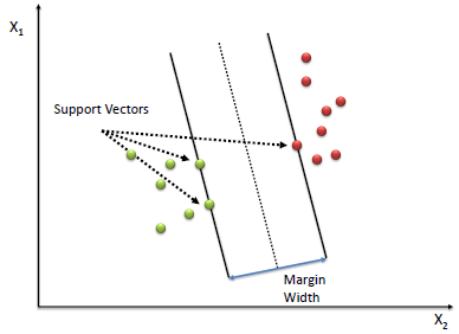
Ans 10:
In simple linear regression, try to minimize the error rate. But in SVR, we try to fit the error within a certain threshold.

Main Concepts:-

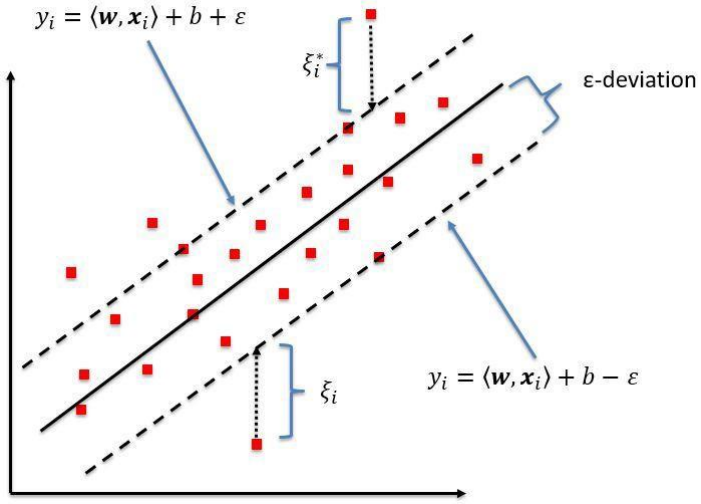
1. **Boundary**
2. **Kernel**
3. **Support Vector**
4. **Hyper Plane**



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



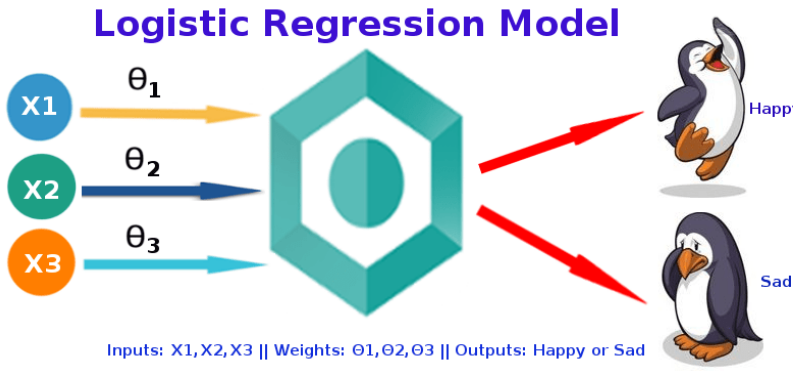
DATA SCIENCE INTERVIEW PREPARATION (30 Days of Interview Preparation)

#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.

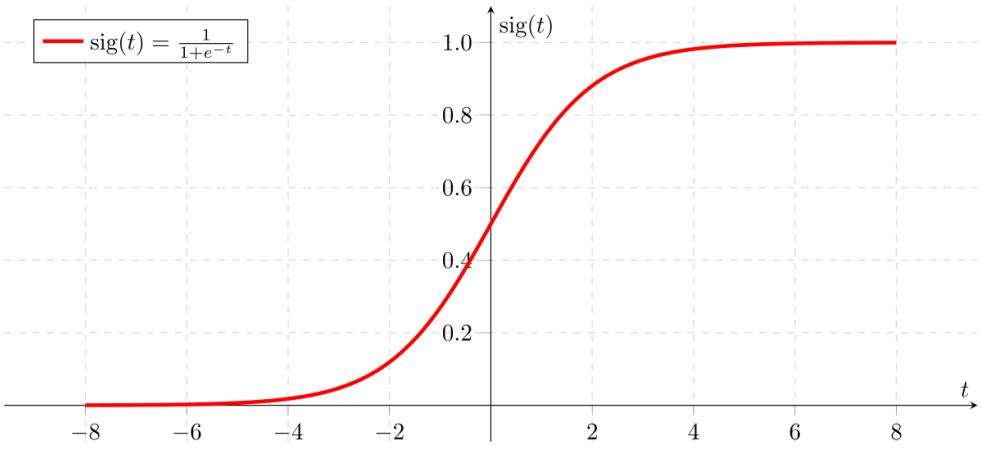
Model

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.

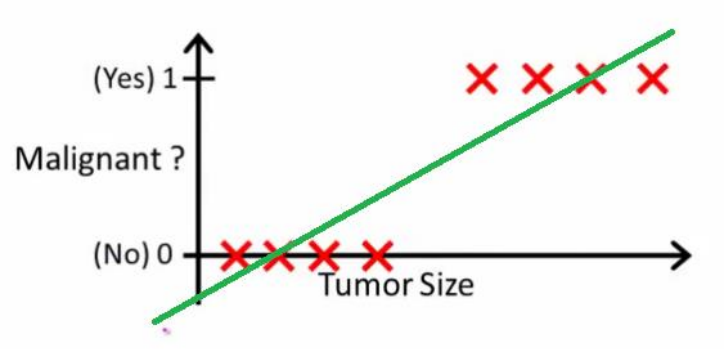
Linear regression requires to establish the linear relationship among dependent and independent variables, whereas it is not necessary for logistic regression.

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?

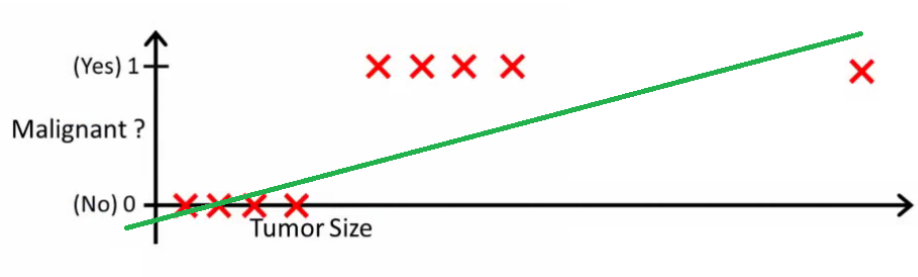
Answer:-

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:



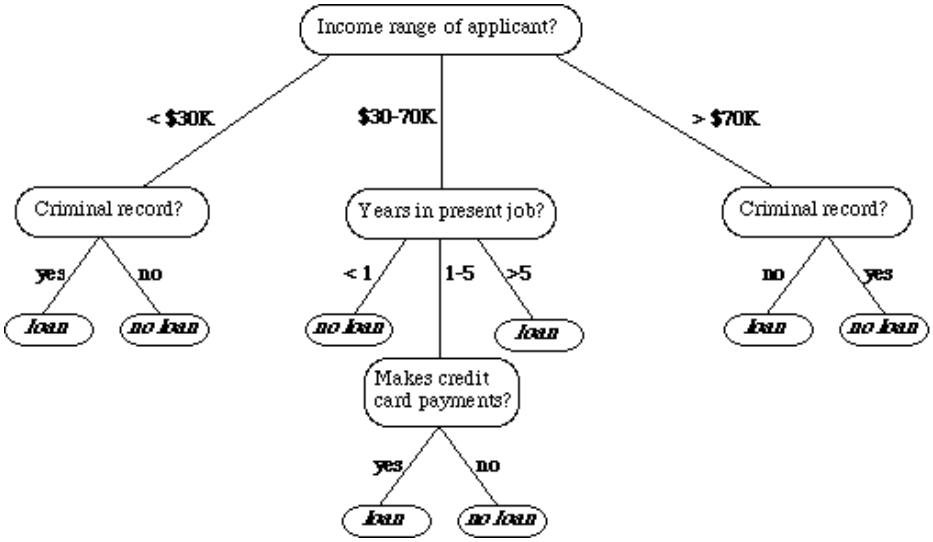
Now our h(x)>0.5⟶malignant doesn't work anymore. To keep making correct predictions, we need to change it to h(x)>0.2 or something - but that not how the algorithm should work.

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)

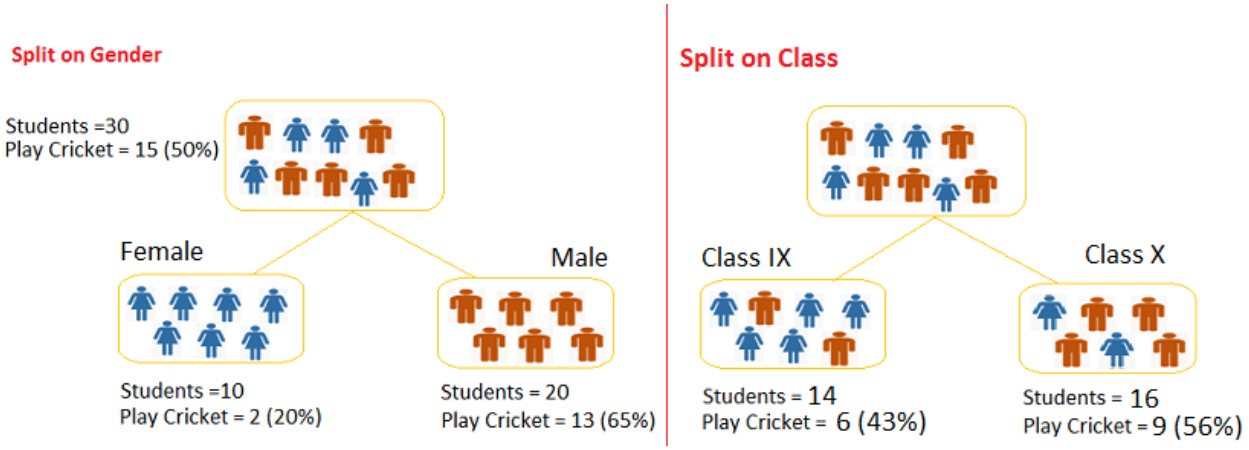
Sb - Subset, S - entire data

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.

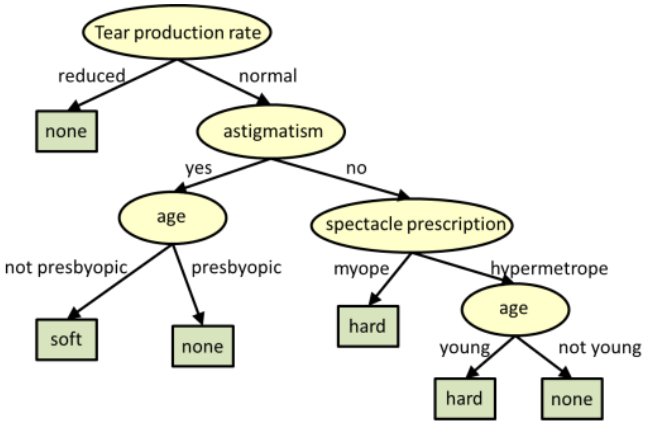
Pruning is mostly done to reduce the chances of overfitting the tree to the training data and reduce the overall complexity of the tree.

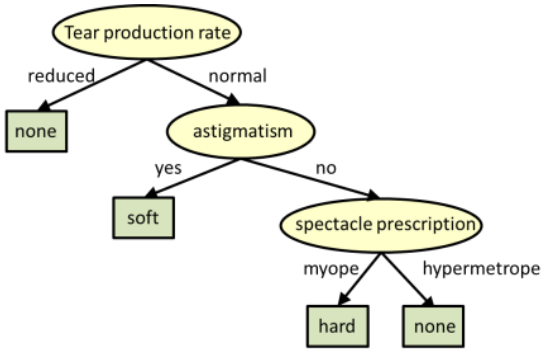
There are two types of pruning: **Pre-pruning** and **Post-pruning**.

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:

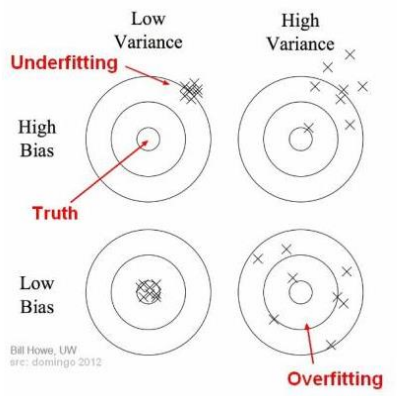
In predicting models, the prediction error is composed of two different errors

1. Bias
2. Variance

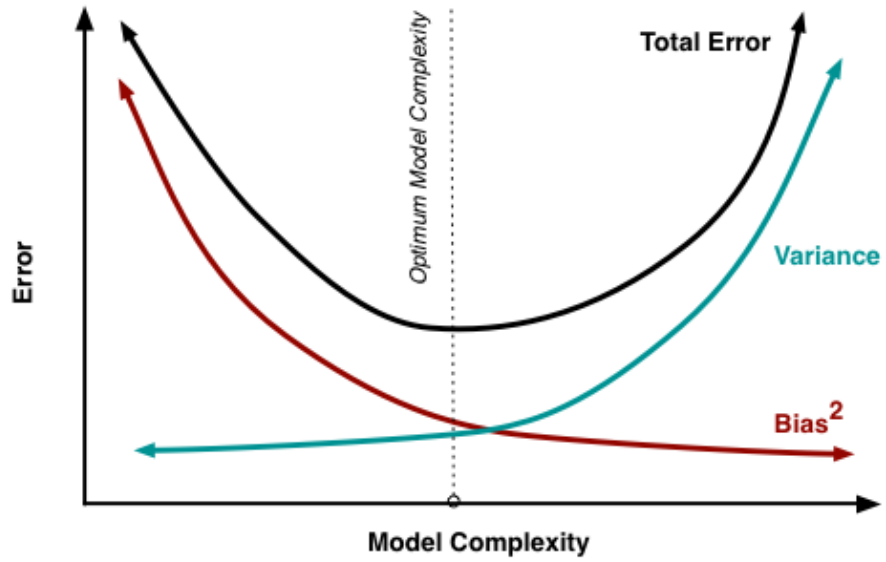
It is important to understand the variance and bias trade-off which tells about to minimize the Bias and Variance in the prediction and avoids overfitting & under fitting of the model.

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**

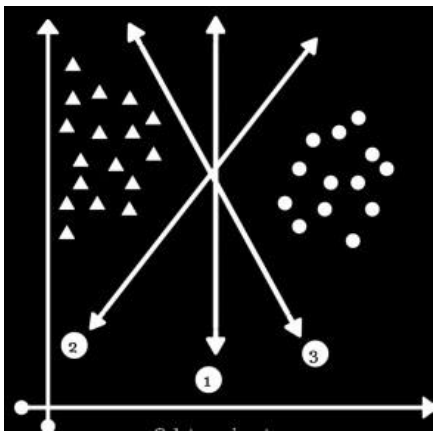
Q11. What is SVM Classification?

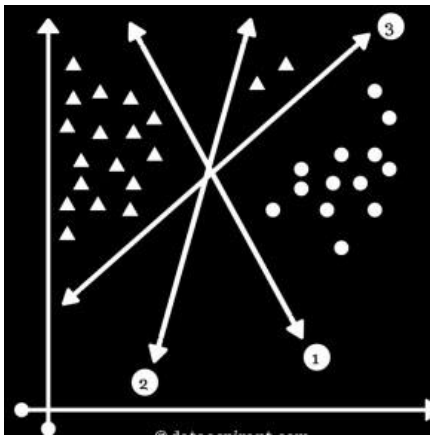
Answer:

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

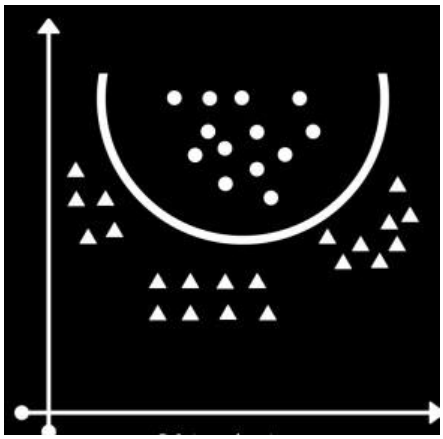
We have two types of SVM classifiers:

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.

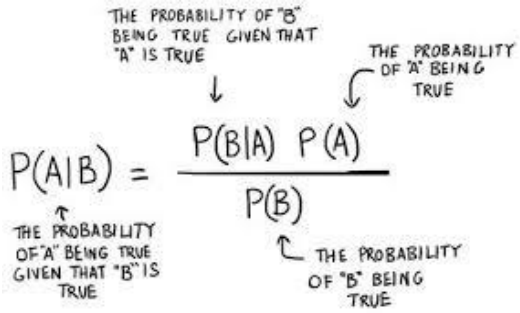
Disadvantages of SVM classifier:

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)

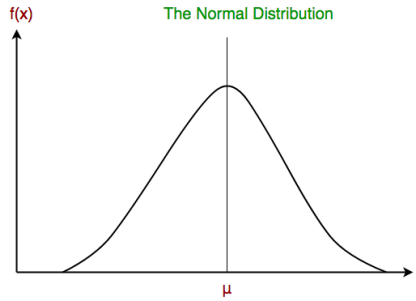
X = (Rainy, Hot, High, False) y = No So basically, P(X|y) here means, the probability of "Not playing golf" given that the weather conditions are "Rainy outlook", "Temperature is hot", "high humidity" and "no wind".

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:-

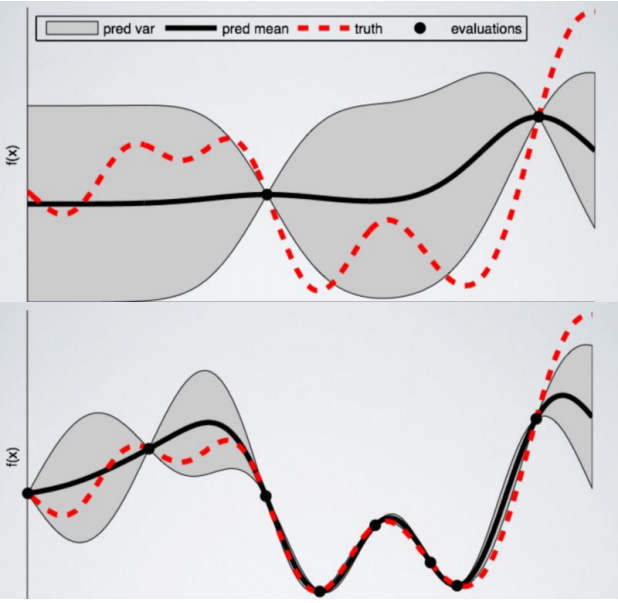
from sklearn.model_selection import GridSearchCV from sklearn.svm import SVR gsc = GridSearchCV(estimator=SVR(kernel='rbf'), param_grid={ 'C': [0.1, 1, 100, 1000], 'epsilon': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10], 'gamma': [0.0001, 0.001, 0.005, 0.1, 1, 3, 5] }, cv=5, scoring='neg_mean_squared_error',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

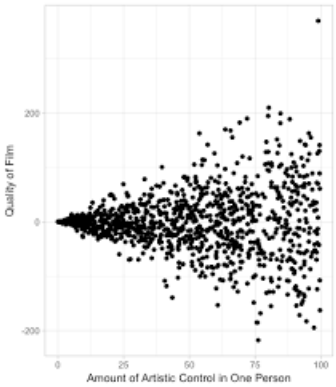
In some image analysis applications, especially when working with images of the color and tiny typ e, it is frequently interesting to apply some whitening to the data before, e.g. training a classifier.

DATA SCIENCE INTERVIEW PREPARATION (30 Days of Interview Preparation)

DAY 03

Q1. How do you treat heteroscedasticity in regression?

Heteroscedasticity means unequal scattered distribution. In regression analysis, we generally talk about the heteroscedasticity in the context of the error term. Heteroscedasticity is the systematic change in the spread of the residuals or errors over the range of measured values. Heteroscedasticity is the problem because *Ordinary least squares (OLS)* regression assumes that all residuals are drawn from a random population that has a constant variance.



What causes Heteroscedasticity?
Heteroscedasticity occurs more often in datasets, where we have a large range between the largest and the smallest observed values. There are many reasons why heteroscedasticity can exist, and a generic explanation is that the error variance changes proportionally with a factor.

We can categorize Heteroscedasticity into two general types:-

Pure heteroscedasticity:- It refers to cases where we specify the correct model and let us observe the non-constant variance in residual plots.

Impure heteroscedasticity:- It refers to cases where you incorrectly specify the model, and that causes the non-constant variance. When you leave an important variable out of a model, the omitted effect is absorbed into the error term. If the effect of the omitted variable varies throughout the observed range of data, it can produce the telltale signs of heteroscedasticity in the residual plots.

How to Fix Heteroscedasticity

Redefining the variables:
If your model is a cross-sectional model that includes large differences between the sizes of the observations, you can find different ways to specify the model that reduces the impact of the size

differential. To do this, change the model from using the raw measure to using rates and per capita values. Of course, this type of model answers a slightly different kind of question. You'll need to determine whether this approach is suitable for both your data and what you need to learn.

Weighted regression:
It is a method that assigns each data point to a weight based on the variance of its fitted value. The idea is to give small weights to observations associated with higher variances to shrink their squared residuals. Weighted regression minimizes the sum of the weighted squared residuals. When you use the correct weights, heteroscedasticity is replaced by homoscedasticity.

Q2. What is multicollinearity, and how do you treat it?

Multicollinearity means independent variables are highly correlated to each other. In regression analysis, it's an important assumption that the regression model should not be faced with a problem of multicollinearity.
If two explanatory variables are highly correlated, it's hard to tell, which affects the dependent variable. Let's say Y is regressed against X1 and X2 and where X1 and X2 are highly correlated. Then the effect of X1 on Y is hard to distinguish from the effect of X2 on Y because any increase in X1 tends to be associated with an increase in X2.
Another way to look at the multicollinearity problem is: Individual t-test P values can be misleading. It means a P-value can be high, which means the variable is not important, even though the variable is important.

Correcting Multicollinearity:
1) Remove one of the highly correlated independent variables from the model. If you have two or more factors with a high VIF, remove one from the model.
2) Principle Component Analysis (PCA) - It cut the number of interdependent variables to a smaller set of uncorrelated components. Instead of using highly correlated variables, use components in the model that have eigenvalue greater than 1.
3) Run PROC VARCLUS and choose the variable that has a minimum (1-R2) ratio within a cluster.
4) Ridge Regression - It is a technique for analyzing multiple regression data that suffer from multicollinearity.
5) If you include an interaction term (the product of two independent variables), you can also reduce multicollinearity by "centering" the variables. By "centering," it means subtracting the mean from the values of the independent variable before creating the products.

When is multicollinearity not a problem?

1) If your goal is to predict Y from a set of X variables, then multicollinearity is not a problem. The predictions will still be accurate, and the overall R2 (or adjusted R2) quantifies how well the model predicts the Y values.
2) Multiple dummy (binary) variables that represent a categorical variable with three or more categories.

Q3. What is market basket analysis? How would you do it in Python?

Market basket analysis is the study of items that are purchased or grouped in a single transaction or multiple, sequential transactions. Understanding the relationships and the strength of those relationships is valuable information that can be used to make recommendations, cross-sell, up-sell, offer coupons, etc.
Market Basket Analysis is one of the key techniques used by large retailers to uncover associations between items. It works by looking for combinations of items that occur together frequently in transactions. To put it another way, it allows retailers to identify relationships between the items that people buy.

Q4. What is Association Analysis? Where is it used?

Association analysis uses a set of transactions to discover rules that indicate the likely occurrence of an item based on the occurrences of other items in the transaction.

The technique of association rules is widely used for retail basket analysis. It can also be used for classification by using rules with class labels on the right-hand side. It is even used for outlier detection with rules indicating infrequent/abnormal association.

Association analysis also helps us to identify cross-selling opportunities, for example, we can use the rules resulting from the analysis to place associated products together in a catalog, in the supermarket, or the Webshop, or apply them when targeting a marketing campaign for product B at customers who have already purchased product A.

Association rules are given in the form as below:

A=>B[Support,Confidence] The part before => is referred to as if (Antecedent) and the part after => is referred to as then (Consequent).
Where A and B are sets of items in the transaction data, a and B are disjoint sets.
Computer=>Anti−virusSoftware[Support=20%,confidence=60%] Above rule says:

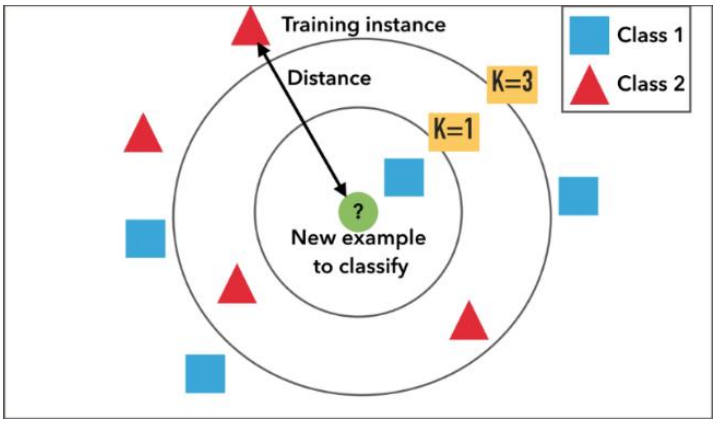
1. 20% transaction show Anti-virus software is bought with purchase of a Computer
2. 60% of customers who purchase Anti-virus software is bought with purchase of a Computer

An example of Association Rules * Assume there are 100 customers

1. 10 of them bought milk, 8 bought butter and 6 bought both of them 2 .bought milk => bought butter
2. support = P(Milk & Butter) = 6/100 = 0.06
3. confidence = support/P(Butter) = 0.06/0.08 = 0.75
4. lift = confidence/P(Milk) = 0.75/0.10 = 7.5

Q5. What is KNN Classifier ?

KNN means **K-Nearest Neighbour** Algorithm. It can be used for both classification and regression.



It is the simplest machine learning algorithm. Also known as **lazy learning** (why? Because it does not create a generalized model during the time of training, so the testing phase is very important where it does the actual job. Hence Testing is very costly - in terms of time & money). Also called an instance-based or memory-based learning
In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is assigned to the class of that single nearest neighbor.

In **k-NN regression**, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

Distance functions

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

All three distance measures are only valid for continuous variables. In the instance of categorical variables, the Hamming distance must be used.

Hamming Distance

$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

How to choose the value of K: K value is a hyperparameter which needs to choose during the time of model building

Also, a small number of neighbors are most flexible fit, which will have a low bias, but the high variance and a large number of neighbors will have a smoother decision boundary, which means lower variance but higher bias.

We should choose an odd number if the number of classes is even. It is said the most common values are to be 3 & 5.

Q6. What is Pipeline in sklearn ?

A pipeline is what chains several steps together, once the initial exploration is done. For example, some codes are meant to transform features—normalize numerically, or turn text into vectors, or fill up missing data, and they are transformers; other codes are meant to predict variables by fitting an algorithm,

such as random forest or support vector machine, they are estimators. Pipeline chains all these together, which can then be applied to training data in block.

Example of a pipeline that imputes data with the most frequent value of each column, and then fit a decision tree classifier.

```
From sklearn.pipeline import Pipeline
 steps = [('imputation', Imputer(missing_values='NaN', strategy = 'most_frequent', axis=0)),
        ('clf', DecisionTreeClassifier())]
 pipeline = Pipeline(steps)
 clf = pipeline.fit(X_train,y_train)```
```

Instead of fitting to one model, it can be looped over several models to find the best one.

```
classifiers = [ KNeighborsClassifier(5), RandomForestClassifier(), GradientBoostingClassifier()]
for clf in classifiers:
steps = [('imputation', Imputer(missing_values='NaN', strategy = 'most_frequent', axis=0)),
     ('clf', clf)]

pipeline = Pipeline(steps)
```

I also learned the pipeline itself can be used as an estimator and passed to cross-validation or grid search.

```
from sklearn.model_selection import KFold
 from sklearn.model_selection import cross_val_score
 kfold = KFold(n_splits=10, random_state=seed)
 results = cross_val_score(pipeline, X_train, y_train, cv=kfold)
 print(results.mean())
```

Q7. What is Principal Component Analysis(PCA), and why we do?

The main idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. The same is done by transforming the variables to a new set of variables, which are known as the principal components (or simply, the PCs) and are orthogonal, ordered such that the retention of variation present in the original variables decreases as we move down in the order. So, in this way, the 1st principal component retains maximum variation that was present in the original components. The principal components are the eigenvectors of a covariance matrix, and hence they are orthogonal.

Main important points to be considered:

1. Normalize the data
2. Calculate the covariance matrix
3. Calculate the eigenvalues and eigenvectors
4. Choosing components and forming a feature vector
5. Forming Principal Components

Q8. What is t-SNE?

(t-SNE) t-Distributed Stochastic Neighbor Embedding is a non-linear dimensionality reduction algorithm used for exploring high-dimensional data. It maps multi-dimensional data to two or more dimensions suitable for human observation. With the help of the t-SNE algorithms, you may have to plot fewer exploratory data analysis plots next time you work with high dimensional data.

Q9. VIF(Variation Inflation Factor),Weight of Evidence & Information Value. Why and when to use?

Variation Inflation Factor

It provides an index that measures how much the variance (the square of the estimate's standard deviation) of an estimated regression coefficient is increased because of collinearity.

VIF = 1 / (1-R-Square of j-th variable) where R2 of jth variable is the coefficient of determination of the model that includes all independent variables except the jth predictor.

Where R-Square of j-th variable is the multiple R2 for the regression of Xj on the other independent variables (a regression that does not involve the dependent variable Y).

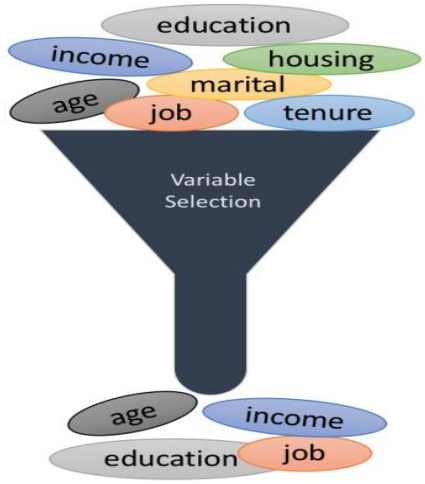
If VIF > 5, then there is a problem with multicollinearity.

Understanding VIF

If the variance inflation factor of a predictor variable is 5 this means that variance for the coefficient of that predictor variable is 5 times as large as it would be if that predictor variable were uncorrelated with the other predictor variables.

In other words, if the variance inflation factor of a predictor variable is 5 this means that the standard error for the coefficient of that predictor variable is 2.23 times ($\sqrt{5} = 2.23$) as large as it would be if that predictor variable were uncorrelated with the other predictor variables.

Weight of evidence (WOE) and information value (IV) are simple, yet powerful techniques to perform variable transformation and selection.



The formula to create WOE and IV is

$$WOE = \ln\left(\frac{\text{Event\%}}{\text{Non Event\%}}\right)$$

$$IV = \sum(\text{Event\%} - \text{Non Event\%}) * \ln\left(\frac{\text{Event\%}}{\text{Non Event\%}}\right)$$

Here is a simple table that shows how to calculate these values.

Variable Name	Min. Value	Max. Value	Count	# Event	# Non Event	Event%	Non event%	WOE	Event% - Non event%	IV
Age	10	20	1200	150	1050	28.3%	19.0%	0.3992	9.3%	0.03718
Age	21	30	900	120	780	22.6%	14.1%	0.4733	8.5%	0.04040
Age	31	40	1090	110	980	20.8%	17.7%	0.1580	3.0%	0.00479
Age	41	50	1460	100	1360	18.9%	24.6%	-0.2650	-5.7%	0.01517
Age	50	inf	1410	50	1360	9.4%	24.6%	-0.9582	-15.2%	0.14525
Total			**6060**	**530**	**5530**					**0.24279**

The IV value can be used to select variables quickly.

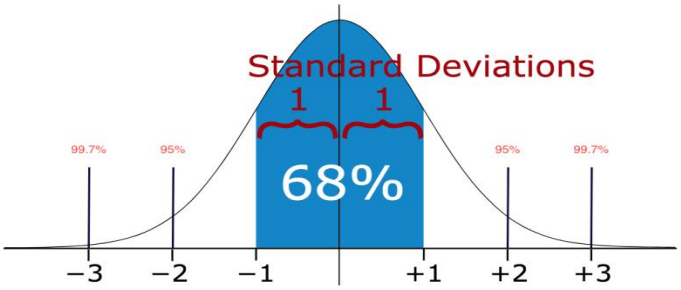
Information Value (IV)	Predictive Power
< 0.02	useless for prediction
0.02 to 0.1	weak predictor
0.1 to 0.3	medium predictor
0.3 to 0.5	strong predictor
> 0.5	suspicious or too good to be true

Q10: How to evaluate that data does not have any outliers ?

In statistics, outliers are data points that don't belong to a certain population. It is an abnormal observation that lies far away from other values. An outlier is an observation that diverges from otherwise well-structured data.

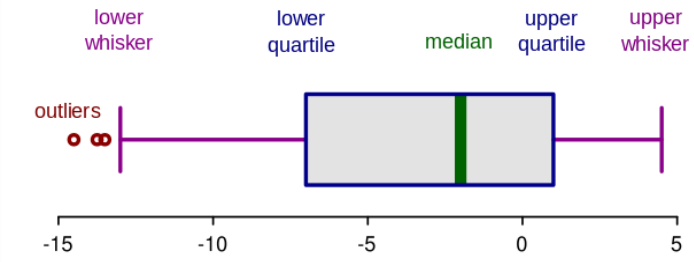
Detection:

Method 1 — Standard Deviation: In statistics, If a data distribution is approximately normal, then about 68% of the data values lie within one standard deviation of the mean, and about 95% are within two standard deviations, and about 99.7% lie within three standard deviations.

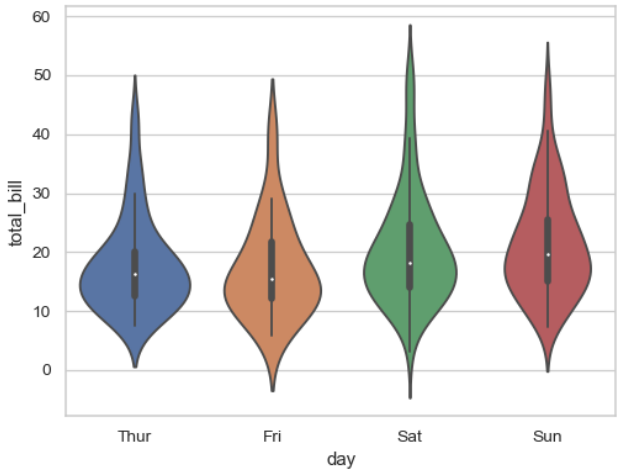


Therefore, if you have any data point that is more than 3 times the standard deviation, then those points are very likely to be anomalous or outliers.

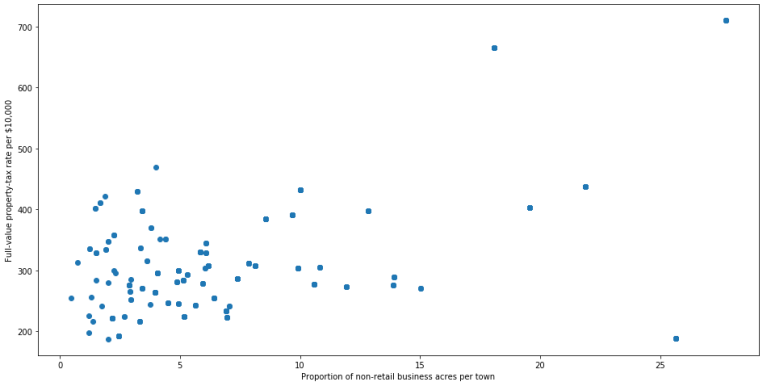
Method 2 — Boxplots: Box plots are a graphical depiction of numerical data through their quantiles. It is a very simple but effective way to visualize outliers. Think about the lower and upper whiskers as the boundaries of the data distribution. Any data points that show above or below the whiskers can be considered outliers or anomalous.



Method 3 - Violin Plots: Violin plots are similar to box plots, except that they also show the probability density of the data at different values, usually smoothed by a kernel density estimator. Typically a violin plot will include all the data that is in a box plot: a marker for the median of the data, a box or marker indicating the interquartile range, and possibly all sample points if the number of samples is not too high.



Method 4 - Scatter Plots: A scatter plot is a type of plot or mathematical diagram using Cartesian coordinates to display values for typically two variables for a set of data. The data are displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of the other variable determining the position on the vertical axis.



The points which are very far away from the general spread of data and have a very few neighbors are considered to be outliers

Q11: What you do if there are outliers?

Following are the approaches to handle the outliers:

1. Drop the outlier records
2. Assign a new value: If an outlier seems to be due to a mistake in your data, you try imputing a value.
3. If percentage-wise the number of outliers is less, but when we see numbers, there are several, then, in that case, dropping them might cause a loss in insight. We should group them in that case and run our analysis separately on them.

Q12: What are the encoding techniques you have applied with Examples ?

In many practical data science activities, the data set will contain categorical variables. These variables are typically stored as text values". Since machine learning is based on mathematical equations, it would cause a problem when we keep categorical variables as is.

Let's consider the following dataset of fruit names and their weights.
Some of the common encoding techniques are:
Label encoding: In label encoding, we map each category to a number or a label. The labels chosen for the categories have no relationship. So categories that have some ties or are close to each other lose such information after encoding.
One - hot encoding: In this method, we map each category to a vector that contains 1 and 0 denoting the presence of the feature or not. The number of vectors depends on the categories which we want to keep. For high cardinality features, this method produces a lot of columns that slows down the learning significantly.

Q13: Tradeoff between bias and variances, the relationship between them.

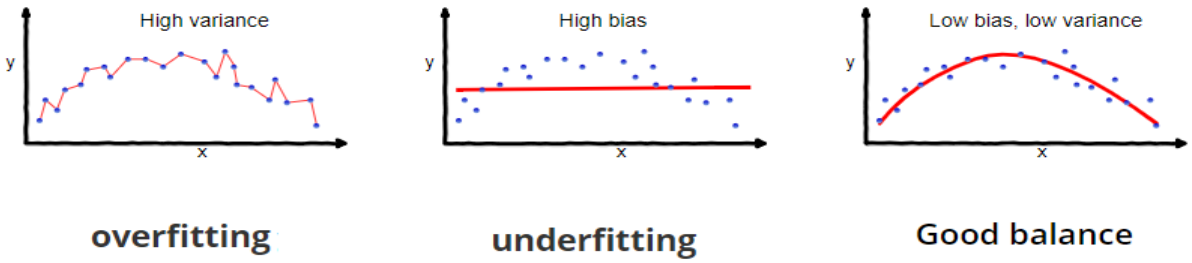
Whenever we discuss model prediction, it's important to understand prediction errors (bias and variance). The prediction error for any machine learning algorithm can be broken down into three parts:

- Bias Error
- Variance Error
- Irreducible Error

The irreducible error cannot be reduced regardless of what algorithm is used. It is the error introduced from the chosen framing of the problem and may be caused by factors like unknown variables that influence the mapping of the input variables to the output variable.

Bias: Bias means that the model favors one result more than the others. Bias is the simplifying assumptions made by a model to make the target function easier to learn. The model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to a high error in training and test data.

Variance: Variance is the amount that the estimate of the target function will change if different training data was used. The model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but have high error rates on test data.



So, the end goal is to come up with a model that balances both Bias and Variance. This is called *Bias Variance Trade-off*. To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.

Q14: What is the difference between Type 1 and Type 2 error and severity of the error?

Type I Error

A Type I error is often referred to as a "false positive" and is the incorrect rejection of the true null hypothesis in favor of the alternative.

In the example above, the null hypothesis refers to the natural state of things or the absence of the tested effect or phenomenon, i.e., stating that the patient is HIV negative. The alternative hypothesis states that the patient is HIV positive. Many medical tests will have the disease they are testing for as the alternative hypothesis and the lack of that disease as the null hypothesis.

A Type I error would thus occur when the patient doesn't have the virus, but the test shows that they do. In other words, the test incorrectly rejects the true null hypothesis that the patient is HIV negative.

Type II Error

A Type II error is the inverse of a Type I error and is the false acceptance of a null hypothesis that is not true, i.e., a false negative. A Type II error would entail the test telling the patient they are free of HIV when they are not.

Considering this HIV example, which error type do you think is more acceptable? In other words, would you rather have a test that was more prone to Type I or Types II error? With HIV, the momentary stress of a false positive is likely better than feeling relieved at a false negative and then failing to take steps to treat the disease. Pregnancy tests, blood tests, and any diagnostic tool that has serious consequences for the health of a patient are usually overly sensitive for this reason – they should err on the side of a false positive.

But in most fields of science, Type II errors are seen as less serious than Type I errors. With the Type II error, a chance to reject the null hypothesis was lost, and no conclusion is inferred from a non-rejected null. But the Type I error is more serious because you have wrongly rejected the null hypothesis and ultimately made a claim that is not true. In science, finding a phenomenon where there is none is more egregious than failing to find a phenomenon where there is.

Q15: What is binomial distribution and polynomial distribution?

Binomial Distribution: A binomial distribution can be thought of as simply the probability of a SUCCESS or FAILURE outcome in an experiment or survey that is repeated multiple times. The binomial is a type of distribution that has two possible outcomes (the prefix "bi" means two, or twice). For example, a coin toss has only two possible outcomes: heads or tails, and taking a test could have two possible outcomes: pass or fail.

Multimonial/Polynomial Distribution: Multi or Poly means many. In probability theory, the multinomial distribution is a generalization of the binomial distribution. For example, it models the probability of counts of each side for rolling a k-sided die n times. For n independent trials each of which leads to success for exactly one of k categories, with each category having a given fixed success probability, the multinomial distribution gives the probability of any particular combination of numbers of successes for the various categories

Q16: What is the Mean Median Mode standard deviation for the sample and population?

Mean It is an important technique in statistics. Arithmetic Mean can also be called an average. It is the number of the quantity obtained by summing two or more numbers/variables and then dividing the sum by the number of numbers/variables.

Mode The mode is also one of the types for finding the average. A mode is a number that occurs most frequently in a group of numbers. Some series might not have any mode; some might have two modes, which is called a bimodal series.
In the study of statistics, the three most common 'averages' in statistics are mean, median, and mode.

Median is also a way of finding the average of a group of data points. It's the middle number of a set of numbers. There are two possibilities, the data points can be an odd number group, or it can be an even number group.
If the group is odd, arrange the numbers in the group from smallest to largest. The median will be the one which is exactly sitting in the middle, with an equal number on either side of it. If the group is even, arrange the numbers in order and pick the two middle numbers and add them then divide by 2. It will be the median number of that set.

Standard Deviation (Sigma) Standard Deviation is a measure of how much your data is spread out in statistics.

Q17: What is Mean Absolute Error ?

What is Absolute Error? Absolute Error is the amount of error in your measurements. It is the difference between the measured value and the "true" value. For example, if a scale states 90 pounds, but you know your true weight is 89 pounds, then the scale has an absolute error of 90 lbs – 89 lbs = 1 lbs.
This can be caused by your scale, not measuring the exact amount you are trying to measure. For example, your scale may be accurate to the nearest pound. If you weigh 89.6 lbs, the scale may "round up" and give you 90 lbs. In this case the absolute error is 90 lbs – 89.6 lbs = .4 lbs.

Mean Absolute Error The Mean Absolute Error(MAE) is the average of all absolute errors. The formula is: mean absolute error

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |x_i - x|$$

Where,

n = the number of errors, Σ = summation symbol (which means “add them all up”), |xi – x| = the absolute errors. The formula may look a little daunting, but the steps are easy:

Find all of your absolute errors, xi – x. Add them all up. Divide by the number of errors. For example, if you had 10 measurements, divide by 10.

Q18: What is the difference between long data and wide data?

There are many different ways that you can present the same dataset to the world. Let's take a look at one of the most important and fundamental distinctions, whether a dataset is wide or long.

The difference between wide and long datasets boils down to whether we prefer to have more columns in our dataset or more rows.

Wide Data A dataset that emphasizes putting additional data about a single subject in columns is called a wide dataset because, as we add more columns, the dataset becomes wider.

Long Data Similarly, a dataset that emphasizes including additional data about a subject in rows is called a long dataset because, as we add more rows, the dataset becomes longer. It's important to point out that there's nothing inherently good or bad about wide or long data.

In the world of data wrangling, we sometimes need to make a long dataset wider, and we sometimes need to make a wide dataset longer. However, it is true that, as a general rule, data scientists who embrace the concept of tidy data usually prefer longer datasets over wider ones.

Q19: What are the data normalization method you have applied, and why?

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges.

In simple words, when multiple attributes are there, but attributes have values on different scales, this may lead to poor data models while performing data mining operations. So they are normalized to bring all the attributes on the same scale, usually something between (0,1).

It is not always a good idea to normalize the data since we might lose information about maximum and minimum values. Sometimes it is a good idea to do so.

For example, ML algorithms such as Linear Regression or Support Vector Machines typically converge faster on normalized data. But on algorithms like K-means or K Nearest Neighbours, normalization could

be a good choice or a bad depending on the use case since the distance between the points plays a key role here.

person_name	Salary	Year_of_ experience	Expected Position Level
Aman	100000	10	2
Abhinav	78000	7	4
Ashutosh	32000	5	8
Dishi	55000	6	7
Abhishek	92000	8	3
Avantika	120000	15	1
Ayushi	65750	7	5

The attributes salary and year_of_experience are on different scale and hence attribute salary can take high priority over attribute year_of_experience in the model.

Types of Normalisation :

1 Min-Max Normalization: In most cases, standardization is used feature-wise

$$\hat{X}[:,i] = \frac{X[:,i] - \min(X[:,i])}{\max(X[:,i]) - \min(X[:,i])}$$

2 Z-score normalization In this technique, values are normalized based on a mean and standard deviation of the data

$$v' = \frac{v - \overline{A}}{\sigma_A}$$

v', v is new and old of each entry in data respectively. σA, A is the standard deviation and mean of A respectively.

standardization (or Z-score normalization) is that the features will be rescaled so that they'll have the properties of a standard normal distribution with

μ=0 and σ=1 where μ is the mean (average) and σ is the standard deviation from the mean; standard scores (also called z scores) of the samples are calculated as follows:

z=(x−μ)/σ

Q20: What is the difference between normalization and Standardization with example?

In ML, every practitioner knows that feature scaling is an important issue. The two most discussed scaling methods are **Normalization** and **Standardization**. Normalization typically means it rescales the values into a range of [0,1].

It is an alternative approach to Z-score normalization (or standardization) is the so-called Min-Max scaling (often also called "normalization" - a common cause for ambiguities). In this approach, the data is scaled to a fixed range - **usually 0 to 1**. Scikit-Learn provides a transformer called **MinMaxScaler** for this. A Min-Max scaling is typically done via the following equation:

Xnorm = X-Xmin/Xmax-Xmin

Example with sample data: Before Normalization: Attribute Price in Dollars Storage Space Camera

- Attribute Price in Dollars Storage Space Camera
- Mobile 1 250 16 12
- Mobile 2 200 16 8
- Mobile 3 300 32 16
- Mobile 4 275 32 8
- Mobile 5 225 16 16

After Normalization: (Values ranges from 0-1 which is working as expected)

- Attribute Price in Dollars Storage Space Camera
- Mobile 1 0.5 0 0.5
- Mobile 2 0 0 0
- Mobile 3 1 1 1
- Mobile 4 0.75 1 0
- Mobile 5 0.25 0 1

Standardization (or Z-score normalization) typically means rescales data to have a mean of 0 and a standard deviation of 1 (unit variance) Formula: **Z or X_new=(x−μ)/σ** where μ is the mean (average), and σ is the standard deviation from the mean; standard scores (also called z scores) Scikit-Learn provides a transformer called StandardScaler for standardization **Example:** Let's take an approximately normally distributed set of numbers: 1, 2, 2, 3, 3, 3, 4, 4, and 5. Its mean is 3, and its standard deviation: 1.22. Now, let's subtract the mean from all data points. we get a new data set of: -2, -1, -1, 0, 0, 0, 1, 1, and 2. Now, let's divide each data point by 1.22. As you can see in the picture below, we get: -1.6, -0.82, -0.82, 0, 0, 0, 0.82, 0.82, and 1.63

DATA SCIENCE INTERVIEW PREPARATION (30 Days of Interview Preparation)

DAY 04

Q1. 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 data sets 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 down sampling 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 Data Frame 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.

Q2. What is the statistical test for data validation with an example,

Chi-square, ANOVA test, Z statics, T statics, F statics,

Hypothesis Testing?

Before discussing the different statistical test, we need to get a clear understanding of what a null hypothesis is. A null hypothesis proposes that has no significant difference exists in the set of a given observation.

Null: Two samples mean are equal. Alternate: Two samples mean are not equal.

For rejecting the null hypothesis, a test is calculated. Then the test statistic is compared with a critical value, and if found to be greater than the critical value, the hypothesis will be rejected.

Critical Value:-

Critical values are the point beyond which we reject the null hypothesis. Critical value tells us, what is the probability of N number of samples, belonging to the same distribution. Higher, the critical value which means lower the probability of N number of samples belonging to the same distribution.

Critical values can be used to do hypothesis testing in the following way.

1. Calculate test statistic
2. Calculate critical values based on the significance level alpha
3. Compare test statistics with critical values.

IMP-If the test statistic is lower than the critical value, accept the hypothesis or else reject the hypothesis.

Chi-Square Test:-

A chi-square test is used if there is a relationship between two categorical variables.

Chi-Square test is used to determine whether there is a significant difference between the expected frequency and the observed frequency in one or more categories. Chi-square is also called the non-parametric test as it will not use any parameter

2-Anova test:-

ANOVA, also called an analysis of variance, is used to compare multiples (three or more) samples with a single test.

Useful when there are more than three populations. Anova compares the variance within and between the groups of the population. If the variation is much larger than the within variation, the means of different samples will not be equal. If the between and within variations are approximately the same size, then there will be no significant difference between sample means. Assumptions of ANOVA: 1-All populations involved follow a normal distribution. 2-All populations have the same variance (or standard deviation). 3-The samples are randomly selected and independent of one another.

ANOVA uses the mean of the samples or the population to reject or support the null hypothesis. Hence it is called parametric testing.

3-Z Statics:-

In a z-test, the samples are assumed to be normal distributed. A z score is calculated with population parameters as “population mean” and “population standard deviation” and it is used to validate a hypothesis that the sample drawn belongs to the same population.

The statistics used for this hypothesis testing is called z-statistic, the score for which is calculated as $z = (x — \mu) / (\sigma / \sqrt{n})$, where x= sample mean μ = population mean $\sigma / \sqrt{n}$ = population standard deviation If the test statistic is lower than the critical value, accept the hypothesis or else reject the hypothesis

4- T Statics:-

A t-test used to compare the mean of the given samples. Like z-test, t-test also assumed a normal distribution of the samples. A t-test is used when the population parameters (mean and standard deviation) are unknown.

There are three versions of t-test

1. Independent samples t-test which compare means for two groups
2. Paired sample t-test which compares mean from the same group at different times
3. Sample t-test, which tests the mean of the single group against the known mean. The statistic for hypothesis testing is called t-statistic, the score for which is calculated as t = (x1 — x2) / (σ / √n1 + σ / √n2), where

   x1 = It is mean of sample A, x2 = mean of sample B,

   n1 = size of sample 1 n2 = size of sample 2

5- F Statics:-

The F-test is designed to test if the two population variances are equal. It compares the ratio of the two variances. Therefore, if the variances are equal, then the ratio of the variances will be 1.

The F-distribution is the ratio of two independent chi-square variables divided by their respective degrees of freedom.

F = s1^2 / s2^2 and where s1^2 > s2^2.

If the null hypothesis is true, then the F test-statistic given above can be simplified. This ratio of sample variances will be tested statistic used. If the null hypothesis is false, then we will reject the null hypothesis that the ratio was equal to 1 and our assumption that they were equal.

Q3. 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 consider 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 a uniformly distributed data:

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

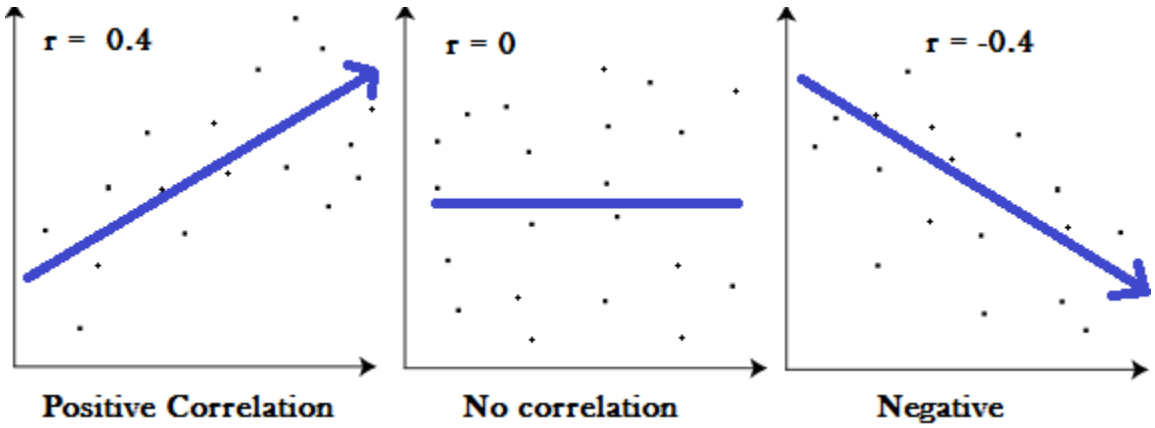
The advantage of CLT is that we need not worry about the actual data since the means of it will always be normally distributed. With this, we can create component intervals, perform T-tests and ANOVA tests from the given samples.

Q4. What is the correlation and coefficient?

What is the Correlation Coefficient?

The correlation coefficient is a statistical measure that calculates the strength of the relationship between the relative movements of two

variables. We use it to measure both the strength and direction of a linear relationship between two variables the values range between -1.0 and 1.0. A calculated number greater than 1.0 or less than -1.0 means that there was an error in the correlation measurement. A correlation of -1.0 shows a perfect negative correlation, while a correlation of 1.0 shows a perfect positive correlation.



Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

1 indicates a strong positive relationship. -1 indicates a strong negative relationship. A result of zero indicates no relationship at all.

Meaning

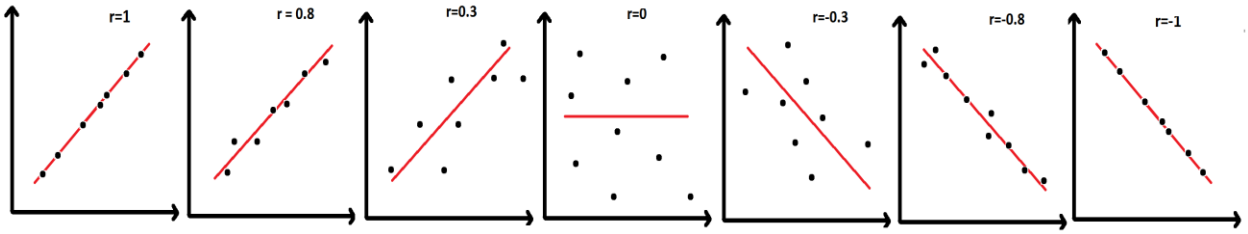
1. A correlation coefficient of 1 means that for every positive increase in one variable, there is a positive increase in a fixed proportion in the other. For example, shoe sizes go up in (almost) perfect correlation with foot length.
2. A correlation coefficient of -1 means that for every positive increase in one variable, there is a negative decrease of a fixed proportion in the other. For example, the amount of gas in a tank decreases in (almost) perfect correlation with speed.
3. Zero means that for every increase, there isn't a positive or negative increase. The two just aren't related.

What is a Negative Correlation?

Negative correlation is a relationship between two variables in which one variable increases as the other decreases, and vice versa. In statistics, a perfect negative correlation is represented by the value -1. Negative correlation or inverse correlation is a relationship between two variables whereby they move in opposite directions. If variables X and Y have a negative correlation (or are negatively correlated), as X increases in value, Y will decrease; similarly, if X decreases in value, Y will increase.

What Is Positive Correlation?

Positive correlation is a relationship between two variables in which both variables move in tandem—that is, in the same direction. A positive correlation exists when one variable decreases as the other variable decreases or one variable increases while the other increases.



We use the correlation coefficient to measure the strength and direction of the linear relationship between two numerical variables X and Y. The correlation coefficient for a sample of data is denoted by r.

Pearson Correlation Coefficient

Pearson is the most widely used correlation coefficient. Pearson correlation measures the linear association between continuous variables. In other words, this coefficient quantifies the degree to which a relationship between two variables can be described by a line. Formula developed by Karl Pearson over 120 years ago is still the most widely used today. The formula for the correlation (r) is

Correlation Coefficient Formula

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[\, n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

Where n is the number of pairs of data;

Are the sample means of all the x-values and all the y-values, respectively; and sx and sy are the sample standard deviations of all the x- and y-values, respectively.

1. Find the mean of all the x-values and mean of all y-values.
2. Find the standard deviation of all the x-values (call it sx) and the standard deviation of all the y-values (call it sy). For example, to find sx, you would use the following equation:
3. For each of the n pairs (x, y) in the data set, take
4. Add up the n results from Step 3.
5. Divide the sum by sx ∗ sy.
6. Divide the result by n – 1, where n is the number of (x, y) pairs. (It's the same as multiplying by 1 over n – 1.) This gives you the correlation, r.

Q5: What is the difference between machine learning and deep learning?

Machine Learning | deep learning

Machine Learning is a technique to learn from that data and then apply wha t has been learnt to make an informed decision | The main difference betwe en deep and machine learning is, machine learning models become better progressively but the model still needs some guidance. If a machine-learning model returns an inaccurate prediction then the programmer need s to fix that problem explicitly but in the case of deep learning, the model do es it by himself.

>Machine Learning can perform well with small size data also | Deep Learning does not perform as good with smaller datasets.

>Machine learning can work on some low-end machines also | Deep Learning involves many matrix multiplication operations which are better suited for GPUs

>Features need to be identified and extracted as per the domain before pushing them to the algorithm | Deep learning algorithms try to learn high-level features from data.

>It is generally recommended to break the problem into smaller chunks, solve them and then combine the results | It generally focusses on solving the problem end to end

>Training time is comparatively less | Training time is comparatively more

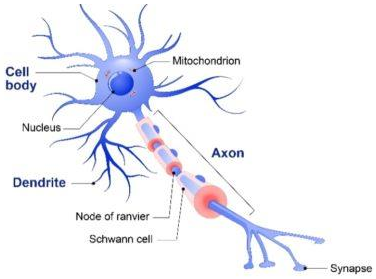
>Results are more interpretable | Results Maybe more accurate but less interpretable

> No use of Neural networks | uses neural networks

> Solves comparatively less complex problems | Solves more complex problems.

Q6: What is perceptron and how it is related to human neurons?

If we focus on the structure of a biological neuron, it has dendrites, which are used to receive inputs. These inputs are summed in the cell body and using the Axon it is passed on to the next biological neuron as shown below.

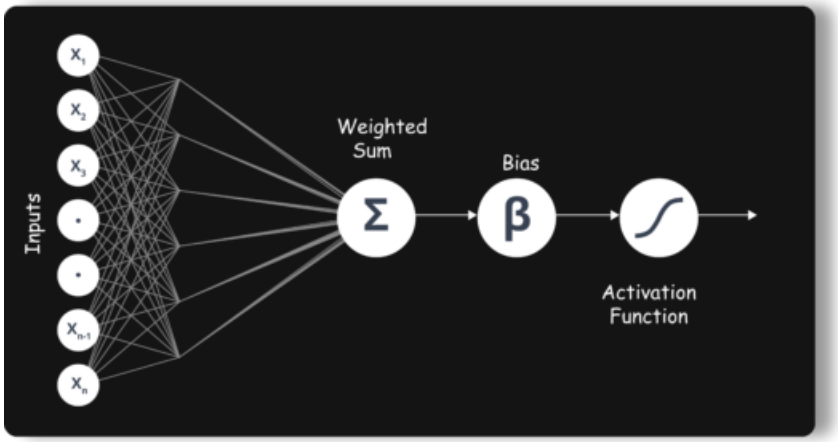


Dendrite: Receives signals from other neurons

Cell Body: Sums all the inputs

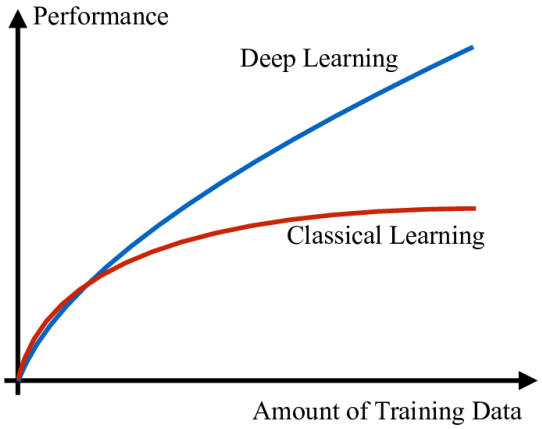
Axon: It is used to transmit signals to the other cells

Similarly, a perceptron receives multiple inputs, applies various transformations and functions and provides an output. A Perceptron is a linear model used for binary classification. It models a neuron, which has a set of inputs, each of which is given a specific weight. The neuron computes some function on these weighted inputs and gives the output.



Q7: Why deep learning is better than machine learning?

Though traditional ML algorithms solve a lot of our cases, they are not useful while working with high dimensional data that is where we have a large number of inputs and outputs. For example, in the case of handwriting recognition, we have a large amount of input where we will have different types of inputs associated with different types of handwriting.



The second major challenge is to tell the computer what are the features it should look for that will play an important role in predicting the outcome as well as to achieve better accuracy while doing so.

Q8: What kind of problem can be solved by using deep learning?

Deep Learning is a branch of Machine Learning, which is used to solve problems in a way that mimics the human way of solving problems. Examples:

- Image recognition
- Object Detection
- Natural Language processing- Translation, Sentence formations, text to speech, speech to text
- understand the semantics of actions

Q9: List down all the activation function using mathematical Expression and example. What is the activation function?

Activation functions are very important for an Artificial Neural Network to learn and make sense of something complicated and the Non-linear complex functional mappings between the inputs and response variable. They introduce non-linear properties to our Network. Their main purposes are to convert an input signal of a node in an A-NN to an output signal.

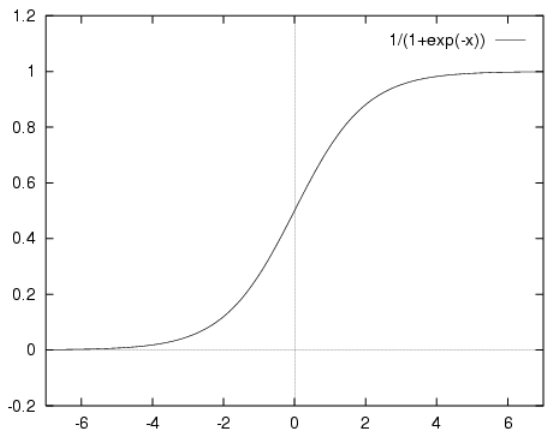
So why do we need Non-Linearities?

Non-linear functions are those, which have a degree more than one, and they have a curvature when we plot a Non-Linear function. Now we need a Neural Network Model to learn and represent almost anything and any arbitrary complex function, which maps inputs to outputs. Neural-Networks are considered Universal Function Approximations. It means that they can compute and learn any function at all.

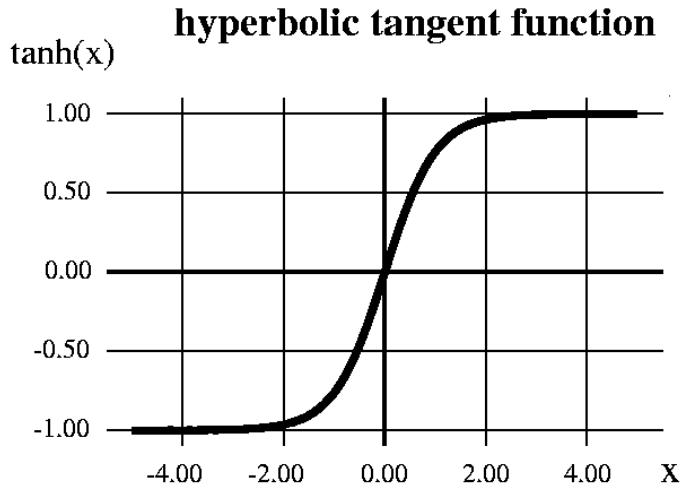
Most popular types of Activation functions -

- Sigmoid or Logistic
- Tanh — Hyperbolic tangent
- ReLu -Rectified linear units

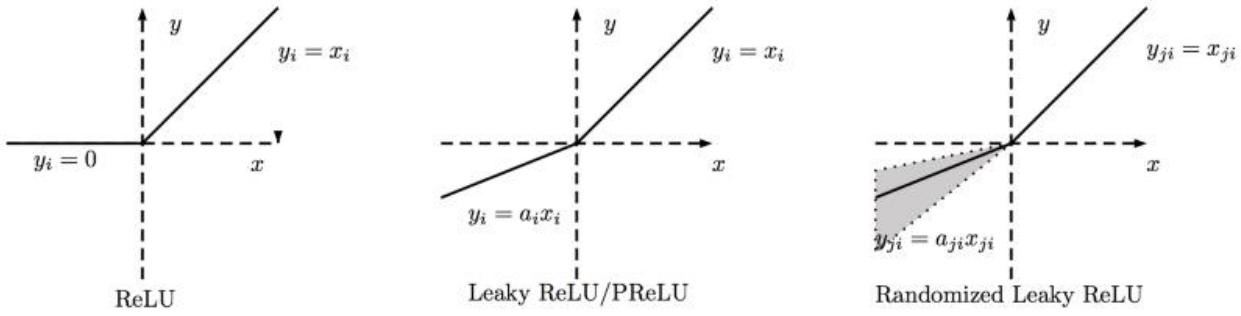
S**igmoid Activation function:** It is a activation function of form f(x) = 1 / 1 + exp(-x) . Its Range is between 0 and 1. It is an S-shaped curve. It is easy to understand.



Hyperbolic Tangent function- Tanh : It's mathematical formula is f(x) = 1 — exp(-2x) / 1 + exp(-2x). Now it's the output is zero centred because its range in between -1 to 1 i.e. -1 < output < 1 . Hence optimisation is easier in this method; Hence in practice, it is always preferred over Sigmoid function.

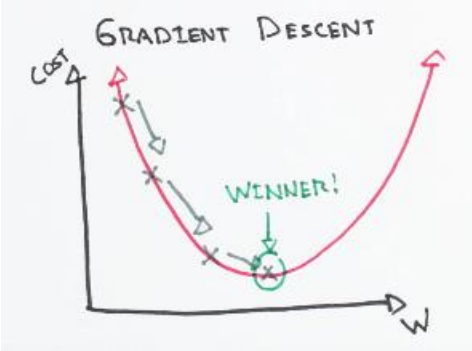


ReLu- Rectified Linear units: It has become more popular in the past couple of years. It was recently proved that it has six times improvement in convergence from Tanh function. It's R(x) = max (0,x) i.e. if x < 0 , R(x) = 0 and if x >= 0 , R(x) = x. Hence as seen that mathematical form of this function, we can see that it is very simple and efficient. Many times in Machine learning and computer science we notice that most simple and consistent techniques and methods are only preferred and are the best. Hence, it avoids and rectifies the vanishing gradient problem. Almost all the deep learning Models use ReLu nowadays.



Q10: Detail explanation about gradient decent using example and Mathematical expression?

Gradient descent is an optimisation algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by negative of the gradient. In machine learning, we used gradient descent to update the parameters of our model. Parameters refer to coefficients in the Linear Regression and weights in neural networks.



The size of these steps called the learning rate. With the high learning rate, we can cover more ground each step, but we risk overshooting the lower point since the slope of the hill is constantly changing. With a very lower learning rate, we can confidently move in the direction of the negative gradient because we are recalculating it so frequently. The Lower learning rate is more precise, but calculating the gradient is time-consuming, so it will take a very large time to get to the bottom.

Math

Now let's run gradient descent using new cost function. There are two parameters in cost function we can control: m (weight) and b (bias). Since we need to consider that the impact each one has on the final prediction, we need to use partial derivatives. We calculate the partial derivative of the cost function concerning each parameter and store the results in a gradient.

Math

Given the cost function:

$$f(m,b) = \frac{1}{N}\sum_{i=1}^{n}(y_i - (mx_i + b))^2$$

The gradient can be calculated as:

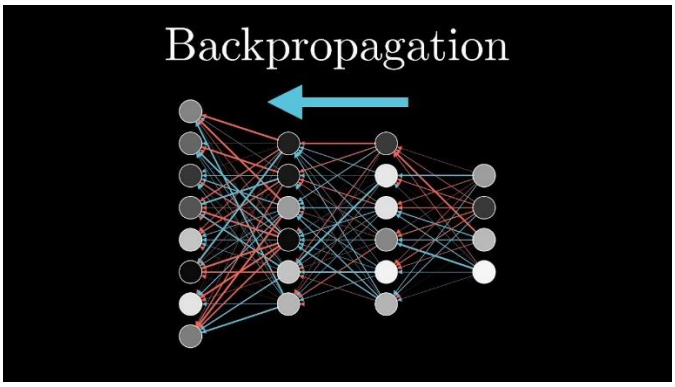
$$f'(m,b) = \begin{bmatrix} \frac{df}{dm} \\ \frac{df}{db} \end{bmatrix} = \begin{bmatrix} \frac{1}{N}\sum -2x_i(y_i - (mx_i + b)) \\ \frac{1}{N}\sum -2(y_i - (mx_i + b)) \end{bmatrix}$$

To solve for the gradient, we iterate by our data points using our new m and b values and compute the partial derivatives. This new gradient tells us about the slope of the cost function at our current position (current parameter values) and the directions we should move to update our parameters. The learning rate controls the size of our update.

Q11: What is backward propagation?

Back-propagation is the essence of the neural net training and this method of fine-tuning the weights of a neural net based on the errors rate obtained in the previous epoch. Proper tuning of the weights allows us to reduce error rates and to make the model reliable by increasing its generalisation.

Backpropagation is a short form of "backward propagation of errors." This is the standard method of training artificial neural networks. This helps to calculate the gradient of a loss function with respects to all the weights in the network.



Most prominent advantages of Backpropagation are:

- Backpropagation is the fast, simple and easy to program.
- It has no parameters to tune apart from the numbers of input.
- It is the flexible method as it does not require prior knowledge about the network
- It is the standard method that generally works well.
- It does not need any special mentions of the features of the function to be learned**.**

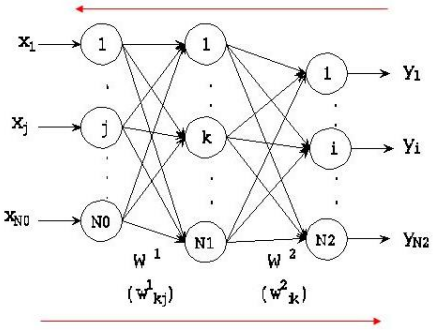
The BackPropagation Algorithm

Main idea:

For each example in the training set:

- compute the output signal
- compute the error corresponding to the output level
- propagate the error back into the network and store the corresponding delta values for each layer
- adjust each weight by using the error signal and input signal for each layer

Computation of the error signal (BACKWARD)



Computation of the output signal (FORWARD)

Q12: How we assign weights in deep learning?

We already know that in a neural network, weights are usually initialised randomly and that kind of initialisation takes a fair/significant amount of repetitions to converge to the least loss and reach the ideal weight matrix. The problem is, that kind of initialisation is prone to vanishing or exploding gradient problems.

General ways to make it initialise better weights:

ReLu activation function in the deep nets.

1. Generate a random sample of weights from a Gaussian distribution having mean 0 and a standard deviation of 1.
2. Multiply the sample with the square root of (2/ni). Where ni is the number of input units for that layer.

b) Likewise, if you're using Tanh activation function :

1. Generate a random sample of weights from a Gaussian distribution having mean 0 and a standard deviation of 1.
2. Multiply the sample with the square root of (1/ni) where ni is several input units for that layer.

Q13: What is optimiser is deep learning, and which one is the best?

Deep learning is an iterative process. With so many hyperparameters to tune or methods to try, it is important to be able to train models fast, to quickly complete the iterative cycle. This is the key to increase the speed and efficiency of a machine learning team.

Hence the importance of optimisation algorithms such as stochastic gradient descent, min-batch gradient descent, gradient descent with momentum and the Adam optimiser.

Adam optimiser is the best one.

Given an algorithm *f(x),* it helps in either minimisation or maximisation of the value of *f(x).* In this context of deep learning, we use optimisation algorithms to train the neural network by optimising the cost function *J.*

The cost function is defined as:

$$J(W,b) = \sum_{i=1}^{m} L(y'^{i}, y^{i})$$

The value of the cost function *J* is the mean of the loss *L* between the predicted value *y'* and actual value *y*. The value *y"* is obtained during the forward propagation step and makes use of the Weights *W* and biases *b* of the network. With the help of optimisation algorithms, we minimise the value of Cost Function *J* by updating the values of trainable parameters *W* and *b*.

Q14: What is gradient descent, mini-batch gradient descent, batch gradient decent, stochastic gradient decent and adam?

Gradient Descent

it is an iterative machine learning optimisation algorithm to reduce the cost function, and help models to make accurate predictions.

Gradient indicates the direction of increase. As we want to find the minimum points in the valley, we need to go in the opposite direction of the gradient. We update the parameters in the negative gradient direction to minimise the loss.

$$\theta = \theta - \eta \nabla J(\theta; x, y)$$

Where θ is the weight parameter, η is the learning rate, and ∇J(θ;x,y) is the gradient of weight parameter θ

Types of Gradient Descent

Different types of Gradient descents are

- Batch Gradient Descent or Vanilla Gradient Descent
- Stochastic Gradient Descent

- Mini batch Gradient Descent

Batch Gradient Descent

In the batch gradient, we use the entire dataset to compute the gradient of the cost function for each iteration for gradient descent and then update the weights.

Stochastic Gradient descent

Stochastic gradient descent, we use a single data point or example to calculate the gradient and update the weights with every iteration.

We first need to shuffle the datasets so that we get a completely randomised dataset. As the datasets are random and weights, are updated for every single example, an update of the weights and the cost functions will be noisy jumping all over the place

Mini Batch Gradient descent

Mini-batch gradients is a variation of stochastic gradient descent where instead of a single training example, a mini-batch of samples are used.

Mini -batch gradient descent is widely used and converges faster and is more stable**.**

The batch size can vary depending upon the dataset.

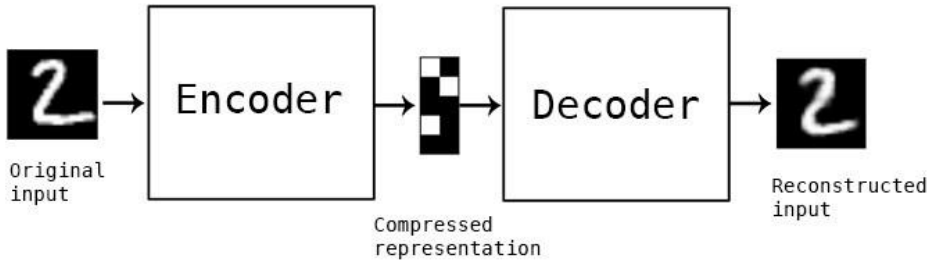
As we take batches with different samples, it reduces the noise which is a variance of the weights updates, and that helps to have a more stable converge faster**.**

Q15: What are autoencoders?

An **autoencoder,** neural networks that have three layers:

An input layer, a hidden layer which is also known as encoding layer, and a decoding layer. This network is trained to reconstruct its inputs, which forces the hidden layer to try to learn good representations of the inputs.

An autoencoder neural network is an unsupervised Machine-learning algorithm that applies backpropagation, setting the target values to be equal to the inputs. An autoencoder is trained to attempts to copy its input to its output. Internally, it has a hidden layer which describes a code used to represent the input.



Autoencoder Components:

Autoencoders consists of 4 main parts:

1- Encoder: In this, the model learns how to reduce the input dimensions and compress the input data into an encoded representation.

2- Bottleneck: In this, the layer that contains the compressed representation of the input data. This is the lowest possible dimension of the input data.

3- Decoder: In this, the model learns how to reconstruct the data from the encod represented to be as close to the original inputs as possible.

4- Reconstruction Loss: In this method that measures measure how well the decoder is performing and how closed the output is related to the original input.

Types of Autoencoders :

1. Denoising auto encoder

2. Sparse auto encoder

3. Variational auto encoder (VAE)

4. Contractive auto encoder (CAE)

Q16: What is CNN?

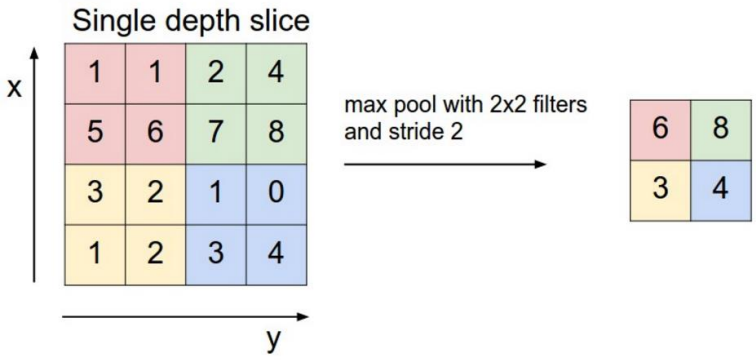
This is the simple application of a filter to an input that results in inactivation. Repeated application of the same filter to input results in a map of activations called a feature map, indicating the locations and strength of a detected feature in input, such as an image.

Convolutional layers are the major building blocks which are used in convolutional neural networks.

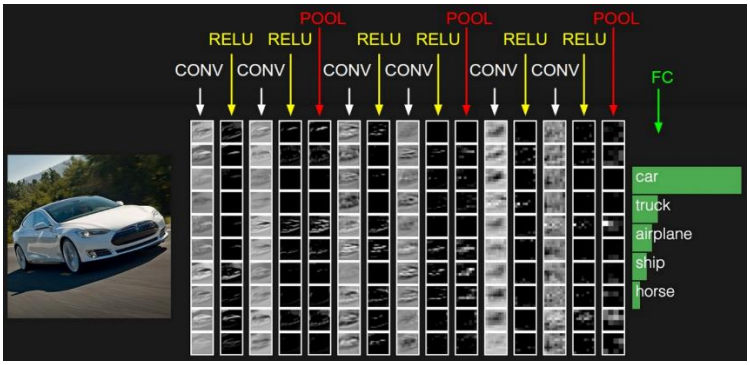
A covnets is the sequence of layers, and every layer transforms one volume to another through differentiable functions.
Different types of layers in CNN:
Let's take an example by running a covnets on of image of dimensions **32 x 32 x 3.**

1. Input Layer: It holds the raw input of image with width 32, height 32 and depth 3.
2. Convolution Layer: It computes the output volume by computing dot products between all filters and image patches. Suppose we use a total of 12 filters for this layer we'll get output volume of dimension 32 x 32 x 12.
3. Activation Function Layer: This layer will apply the element-wise activation function to the output of the convolution layer. Some activation functions are RELU: max(0, x), Sigmoid: 1/(1+e^-x), Tanh, Leaky RELU, etc. So the volume remains unchanged. Hence output volume will have dimensions 32 x 32 x 12.
4. Pool Layer: This layer is periodically inserted within the covnets, and its main function is to reduce the size of volume which makes the

computation fast reduces memory and also prevents overfitting. Two common types of pooling layers are max pooling and average pooling. If we use a max pool with 2 x 2 filters and stride 2, the resultant volume will be of dimension 16x16x12.



5. Fully-Connected Layer: This layer is a regular neural network layer that takes input from the previous layer and computes the class scores and outputs the 1-D array of size equal to the number of classes.

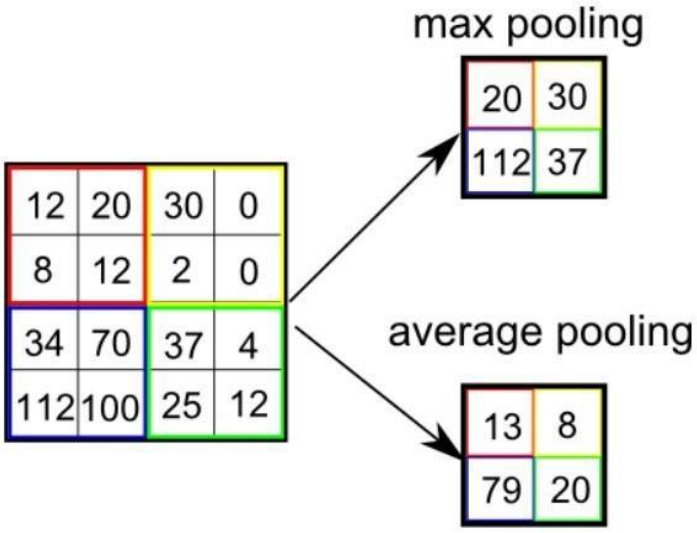


Q17: What is pooling, padding, filtering operations on CNN?

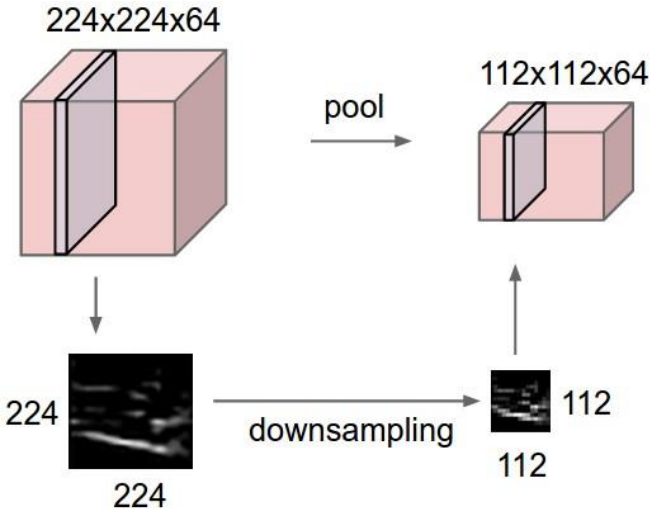
Pooling Layer

It is commonly used to periodically insert a Pooling layer in-between successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also

control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation.

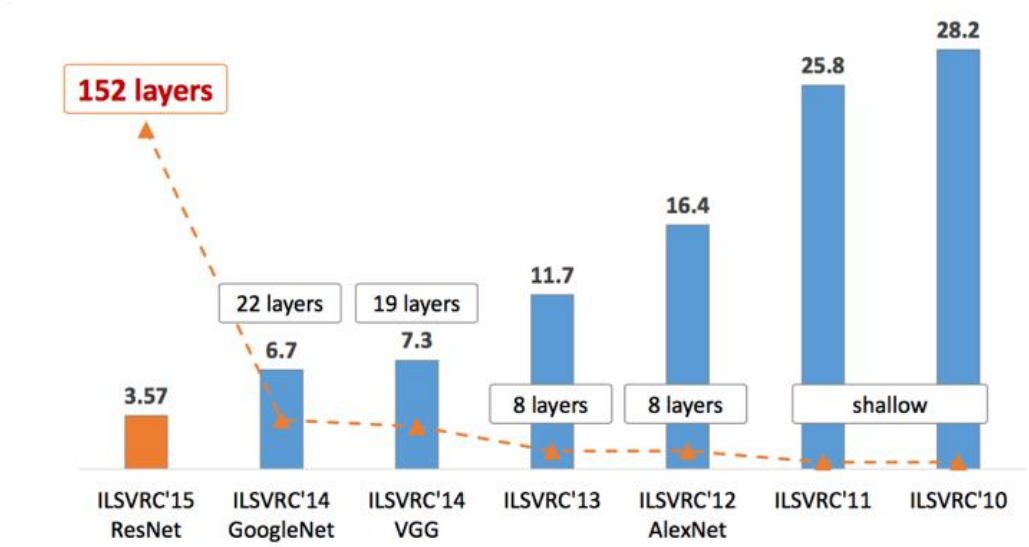


The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by two along both width and height, discarding 75% of the activations. Every MAX operation would, in this case, be taking a max over four numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged.



Q18: What is the Evolution technique of CNN?

It all started with LeNet in 1998 and eventually, after nearly 15 years, lead to groundbreaking models winning the ImageNet Large Scale Visual Recognition Challenge which includes AlexNet in 2012 to Google Net in 2014 to ResNet in 2015 to an ensemble of previous models in 2016. In the last two years, no significant progress has been made, and the new models are an ensemble of previous groundbreaking models.



LeNet in 1998

LeNet is a 7-level convolutional network by LeCun in 1998 that classifies digits and used by several banks to recognise the hand-written numbers on cheques digitised in 32x32 pixel greyscale input images.

AlexNet in 2012

AlexNet: It is considered to be the first paper/ model, which rose the interest in CNNs when it won the ImageNet challenge in the year 2012. It is a deep CNN trained on ImageNet and outperformed all the entries that year.

VGG in 2014

VGG was submitted in the year 2013, and it became a runner up in the ImageNet contest in 2014. It is widely used as a simple architecture compared to AlexNet.

GoogleNet in 2014

In 2014, several great models were developed like VGG, but the winner of the ImageNet contest was GoogleNet.

GoogLeNet proposed a module called the inception modules that includes skipping connections in the network, forming a mini-module, and this module is repeated throughout the network.

ResNet in 2015

There are 152 layers in the Microsoft ResNet. The authors showed empirically that if you keep on adding layers, the error rate should keep on decreasing in contrast to “plain nets” we're adding a few layers resulted in higher training and test errors.

Q19: How to initialise biases in deep learning?

It is possible and common to initialise the biases to be zero since the random numbers in the weights provide the asymmetry braking. For ReLU non-linearities, some people like to use small constant value such as 0.01 for all biases because this ensures that all ReLU units fire in the beginning, therefore obtain, and propagate some gradient. However, it is unclear if this provides a consistent improvement (in fact some results seem to indicates that this performs worst) and it is more commonly used to use 0 bias initialisation.

Q20: What is learning Rate?

Learning Rate

The learning rate controls how much we should adjust the weights concerning the loss gradient. Learning rates are randomly initialised.

Lower the values of the learning rate slower will be the convergence to global minima.

Higher values for the learning rate will not allow the gradient descent to converge

Since our goal is to minimise the function cost to find the optimised value for weights, we run multiples iteration with different weights and calculate the cost to arrive at a minimum cost

---

DATA SCIENCE INTERVIEW PREPARATION (30 Days of Interview Preparation)

Day-5

Q1: What are Epochs?

One Epoch is an ENTIRE dataset is passed forwards and backwards through the neural network.

Since one epoch is too large to feed to the computer at once, we divide it into several smaller batches.

We always use more than one Epoch because *one epoch leads to underfitting.*

As the number of epochs increases, several times the weight are changed in the neural network and the curve goes from underfitting up to optimal to overfitting curve.

Q2. What is the batch size?

Batch Size

The total number of training and examples present in a single batch.

Unlike the learning rate hyperparameter where its value doesn't affect computational time, the batch sizes must be examined in conjunctions with the execution time of training. The batch size is limited by hardware's memory, while the learning rate is not. Leslie recommends using a batch size that fits in hardware's memory and enables using larger learning rate.

If our server has multiple GPUs, the total batch size is the batch size on a GPU multiplied by the numbers of GPU. If the architectures are small or your hardware permits very large batch sizes, then you might compare the performance of different batch sizes. Also, recall that small batch sizes add regularization while large batch sizes add less, so utilize this while balancing the proper amount of regularization. It is often better to use large batch sizes so a larger learning rate can be used.

Q3: What is dropout in Neural network?

Dropout refers to ignoring units during the training phase of a certain set of neurons which is chosen randomly. These units are not considered during the particular forward or backward pass.

More technically, at each training stage, individual nodes are either dropped out of the net with probability *1-p* or kept with probability *p*, so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed.

We need Dropout *to prevent over-fitting*

A dropout is an approach to regularization in neural networks which helps to reduce interdependent learning amongst the neurons.

Where to use

Dropout is implemented per-layer in a neural network.

It can be used with most types of layers, such as dense fully connected layers, convolutional layers, and recurrent layers such as the long short-term memory network layer.

Dropout may be implemented on any or all hidden layers in the network as well as the visible or input layer. It is not used on the output layer.

Benefits:-

1. Dropout forces a neural network to learn more robust features that are very useful in conjunction with different random subsets of the other neurons.
2. Dropout generally doubles the number of iterations required to converge. However, the training time for each epoch is less.

Q4: List down hyperparameter tuning in deep learning.

The process of setting the hyper-parameters requires expertise and extensive trial and error. There are no simple and easy ways to set hyper-parameters — specifically, learning rate, batch size, momentum, and weight decay.

Approaches to searching for the best configuration:

- Grid Search
- Random Search

Approach

1. Observe and understand the clues available during training by monitoring validation/test loss early in training, tune your architecture and hyper-parameters with short runs of a few epochs.
2. Signs of *underfitting* or *overfitting* of the test or validation loss early in the training process are useful for tuning the hyper-parameters.

Tools for Optimizing Hyperparameters

- Sage Maker
- Comet.ml
- Weights &Biases
- Deep Cognition
- Azure ML

Q5: What do you understand by activation function and error functions?

Error functions

In most learning networks, an error is calculated as the difference between the predicted output and the actual output.

$$J(w) = p - \hat{p}$$

The function that is used to compute this error is known as Loss Function J(.). Different loss functions will give different errors for the same prediction, and thus have a considerable effect on the performance of the model. One of the most widely used loss function is mean square error, which calculates the square of the difference between the actual values and predicted value. Different loss functions are used to deals with a different type of tasks, i.e. regression and classification.

Regressive loss functions:

Mean Square Error

Absolute error

Smooth Absolute Error

Classification loss functions:

1. Binary Cross-Entropy
2. Negative Log-Likelihood
3. Margin Classifier
4. Soft Margin Classifier

Activation functions decide whether a neuron should be activated or not by calculating a weighted sum and adding bias with it. The purpose of the activation function is to introduce non-linearity into the output of a neuron.

In a neural network, we would update the weights and biases of the neurons based on the error at the outputs. This process is known as back-propagation. Activation function makes the back-propagation possible since the gradients are supplied along with the errors to update the weights and biases.

Q6: Why do we need Non-linear activation functions?

A neural network without activation functions is essentially a linear regression model. The activation functions do the non-linear transformation to the input, making it capable of learning and performing more complex tasks.

1. Identity
2. Binary Step
3. Sigmoid
4. Tanh
5. ReLU
6. Leaky ReLU

7. Softmax

The activation functions do the non-linear transformation to the input, making it capable of learning and performing more complex tasks.

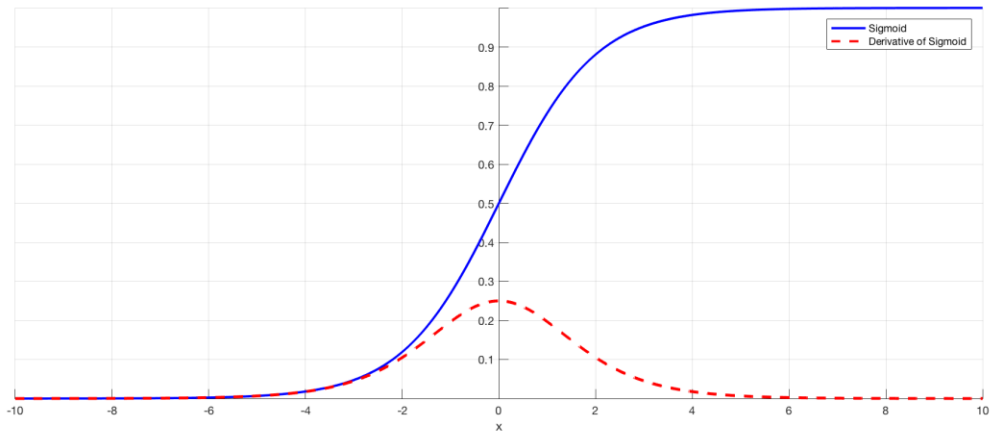
Q7: What do you under by vanishing gradient problem and how can Do we solve that?

The problem:

As more layers using certain activation function are added to neural networks, the gradients of the loss function approach zero, making the networks tougher to train.

Why:

Certain activation functions, like the sigmoid function, squishes a large input space into a small input space between 0 and 1. Therefore, a large change in the input of the sigmoid function will cause a small change in the output. Hence, the derivative becomes small.



For shallow networks with only a few layers that use these activations, this isn't a big problem. However, when more layers are used, it can cause the gradient to be too small for training to work effectively.

However, when *n* hidden layers use an activation like the sigmoid function, *n* small derivatives are multiplied together. Thus, the gradient decreases exponentially as we propagate down to the initial layers.

Solutions:

The simplest solution is to use other activation functions, such as ReLU, which doesn't cause a small derivative.

Residual networks are another solution, as they provide residual connections straight to earlier layers.

Finally, batch normalization layers can also resolve the issue.

Q8: What is Transfer learning in deep learning ?

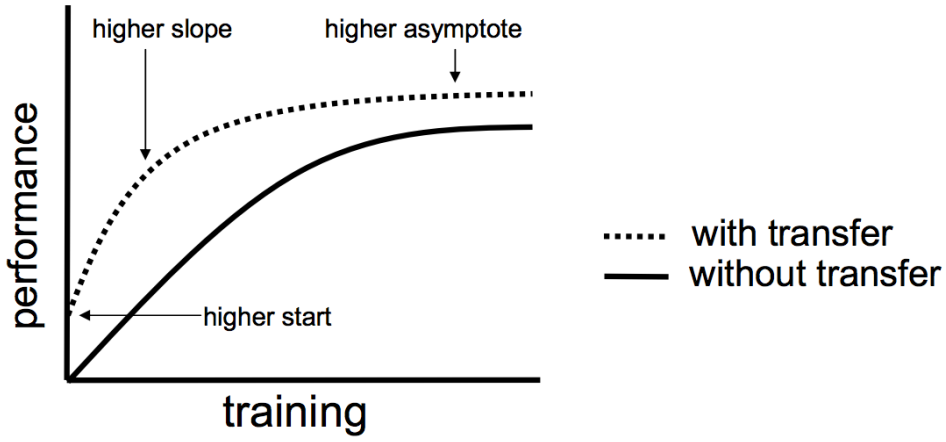
Transfer learning: It is a machine learning method where a model is developed for the task is again used as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task.

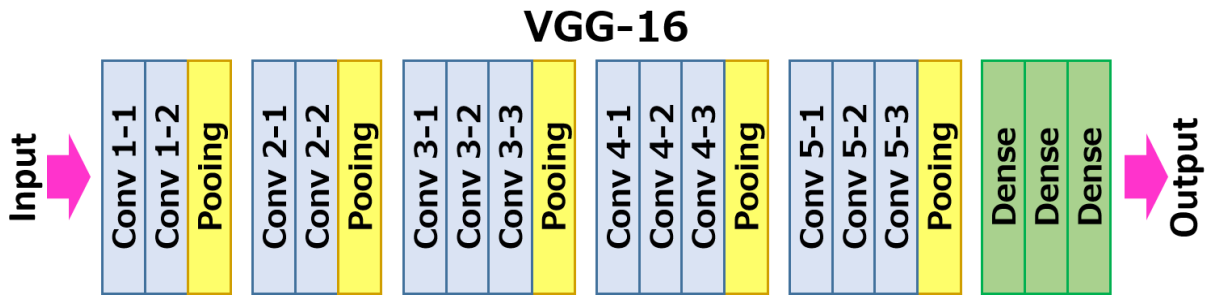
Transfer learning is an optimization that allows rapid progress or improved performance when modelling the second task.

Transfer learning only works in deep learning if the model features learned from the first task are general.



Q9: What is VGG16 and explain the architecture of VGG16?

VGG-16 is a simpler architecture model since it's not using many hyperparameters. It always uses 3 x 3 filters with the stride of 1 in convolution layer and uses SAME padding in pooling layers 2 x 2 with a stride of 2.

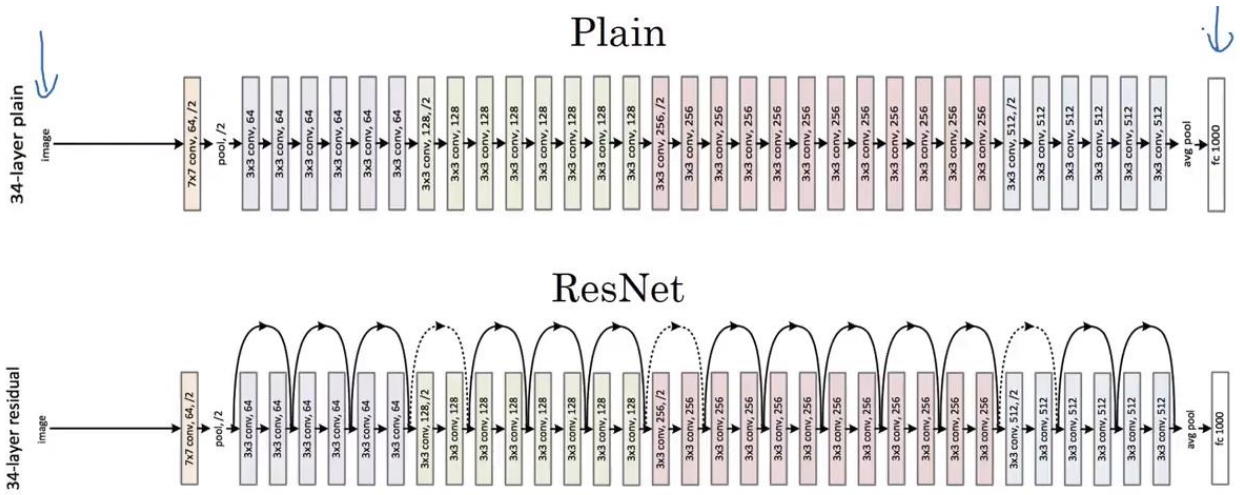


This architecture is from the VGG group, Oxford. It improves AlexNet by replacing the large kernel-sized filter with multiple 3X3 kernel-sized filters one after another. With a given receptive field(the effective area size of input image on which output depends), multiple stacked smaller size kernel is better than the one with a larger size kernel because multiple non-linear layers increases the depth of the network which enables it to learn more complex features, and that too at a lower cost.

Three fully connected layers follow the VGG convolutional layers. The width of the networks starts at the small value of 64 and increases by a factor of 2 after every sub-sampling/pooling layer. It achieves the top-5 accuracy of 92.3 % on ImageNet.

Q10: What is RESNET?

The winner of ILSRVC 2015, it also called as Residual Neural Network (ResNet) by Kaiming. This architecture introduced a concept called "skip connections". Typically, the input matrix calculates in two linear transformations with ReLU activation function. In Residual network, it directly copies the input matrix to the second transformation output and sums the output in final ReLU function.



Skip Connection

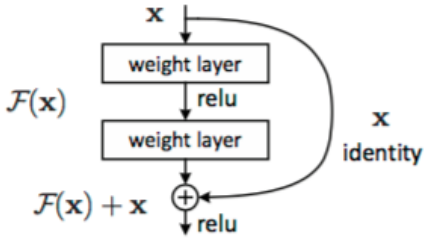


Figure 2. Residual learning: a building block.

Experiments in paper four can judge the power of the residual network. The plain 34 layer network had high validation error than the 18 layers plain network. This is where we realize the degradation problems. And the same 34 layers network when converted to the residual network has much less training error than the 18 layers residual network.

Q11: What is ImageNet?

ImageNet is a project aimed at (manually) labelling and categorizing images into almost 22,000 separate object categories for computer vision researches.

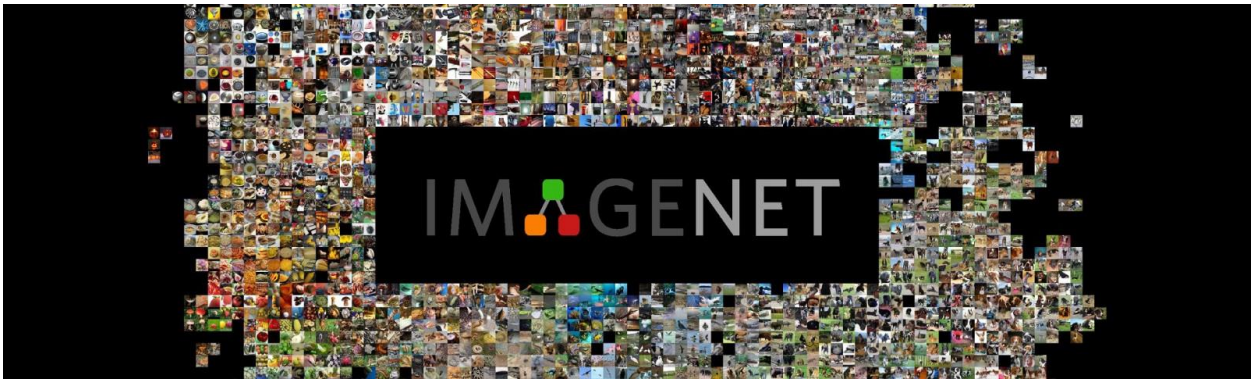
When we hear the about *"ImageNet"* in the context of deep learning and Convolutional Neural Network, we are referring to *ImageNet Large Scale Visual Recognition Challenge*.

The main aim of this image classification challenge is to train the model that can correctly classify an input image into the 1,000 separate objects category.

Models are trained on the ~1.2 million training images with another 50,000 images for validation and 100,000 images for testing.

These 1,000 image categories represent object classes that we encounter in our day-to-day lives, such as species of dogs, cats, various household objects, vehicle types, and much more.

When it comes to the image classification, the ImageNet challenge is the "de facto " benchmark for computer vision classification algorithms — and the leaderboard for this challenge has been dominated by Convolutional Neural Networks and Deep learning techniques since 2012.



Q12: What is DarkNet?

DarkNet is a framework used to train neural networks; it is open source and written in C/CUDA and serves as the basis for YOLO. Darknet is also used as the framework for training YOLO, meaning it sets the architecture of the network.

Clone the repo locally, and you have it. To compile it, run a make. But first, if you intend to use the GPU capability, you need to edit the **Makefile** in the first two lines, where you tell it to compile for GPU usage with CUDA drivers.

Q13: What is YOLO and explain the architecture of YOLO (you only Look Once). One use case?

YOLO v1

The first YOLO You only look once (YOLO) version came about May 2016 and sets the core of the algorithm, the following versions are improvements that fix some drawbacks.

In short, YOLO is a network “inspired by” Google Net. It has 24 convolutional layers working as the feature extractors and two dense layers for making the predictions. The architecture works upon is called Darknet, a neural network framework created by the first author of the YOLO paper.

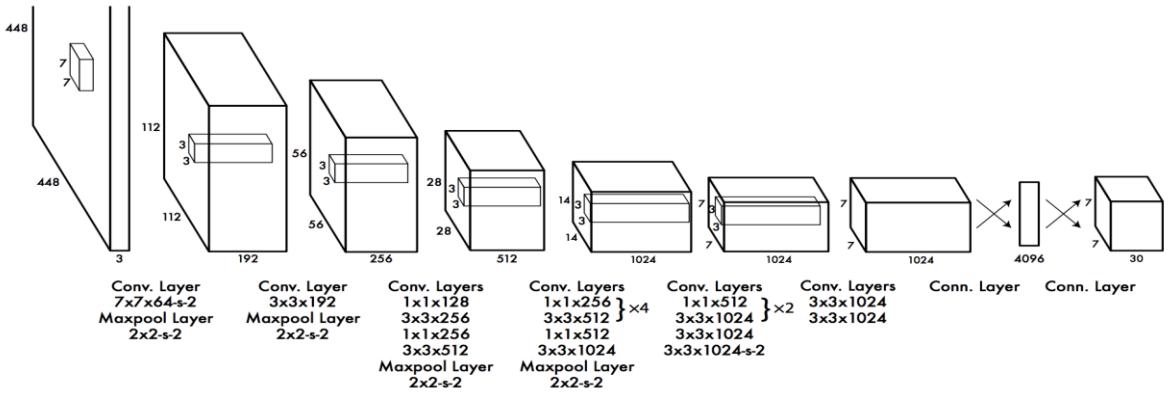
Core Concept:-

The algorithm works off by dividing the image into the grid of the cells, for each cell bounding boxes and their scores are predicted, alongside class probabilities. The confidence is given in terms of IOU (*intersection over union*), metric, which is measuring how much the detected object overlaps with the ground truth as a fraction of the total area spanned by the two together (the union).

YOLO v2-

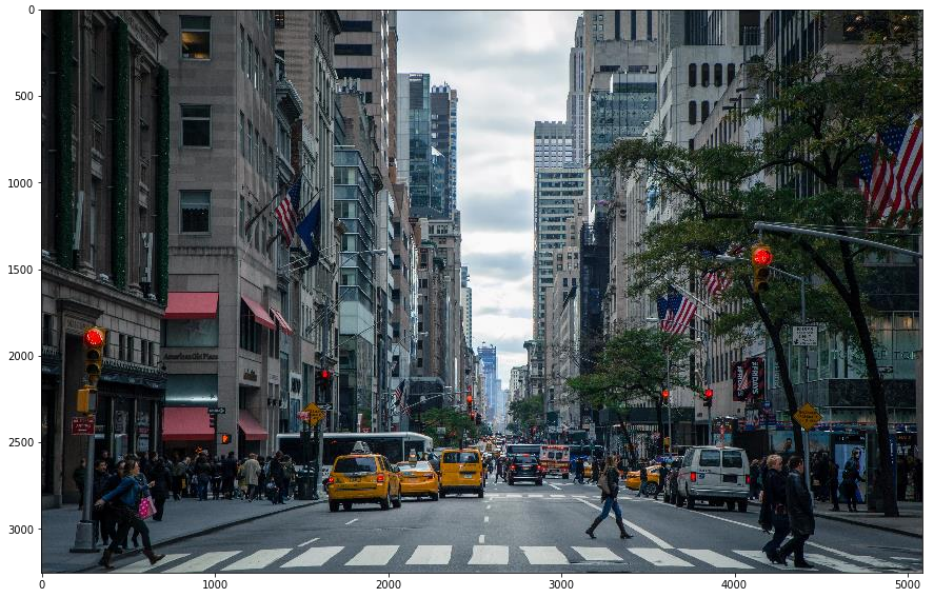
This improves on some of the shortcomings of the first version, namely the fact that it is not very good at detecting objects that are very near and tends to make some of the mistakes on localization.

It introduces a few newer things: Which are *anchor boxes* (pre-determined sets of boxes such that the network moves from predicting the bounding boxes to predicting the offsets from these) and the use of features that are more fine-grained so smaller objects can be predicted better.



YOLO v3-

YOLOv3 came about April 2018, and it adds small improvements, including the fact that bounding boxes get predicted at the different scales. The underlying meaty part of the YOLO network, Darknet, is expanded in this version to have 53 convolutional layers


0
500
1000
1500
2000
2500
3000
0
1000
2000
3000
4000
5000


traffic light: 1.0
traffic light: 1.0
traffic light: 1.0
traffic light: 1.0
traffic light: 1.0
bus: 1.0
car: 1.0
truck: 0.7
truck: 0.9
person: 1.0
umbrella: 1.0
