Data Science Interview Questions and Answers

Table of Contents

STATISTICS **6**

Q1. What is the Central Limit Theorem and why is it important? 6
Q2. What is sampling? How many sampling methods do you know? 7
Q3. What is the difference between type I vs type II error? 9
Q4. What is linear regression? What do the terms p-value, coefficient, and r-squared value mean? What is the significance of each of these components? 9
Q5. What are the assumptions required for linear regression? 10
Q6. What is a statistical interaction? 10
Q7. What is selection bias? 11
Q8. What is an example of a data set with a non-Gaussian distribution? 11

DATA SCIENCE **12**

Q1. What is Data Science? List the differences between supervised and unsupervised learning. 12
Q2. What is Selection Bias? 12
Q3. What is bias-variance trade-off? 12
Q4. What is a confusion matrix? 13
Q5. What is the difference between "long" and "wide" format data? 14
Q6. What do you understand by the term Normal Distribution? 15
Q7. What is correlation and covariance in statistics? 15
Q8. What is the difference between Point Estimates and Confidence Interval? 16
Q9. What is the goal of A/B Testing? 16
Q10. What is p-value? 16
Q11. In any 15-minute interval, there is a 20% probability that you will see at least one shooting star. What is the probability that you see at least one shooting star in the period of an hour? 16
Q12. How can you generate a random number between 1 – 7 with only a die? 17
Q13. A certain couple tells you that they have two children, at least one of which is a girl. What is the probability that they have two girls? 17
Q14. A jar has 1000 coins, of which 999 are fair and 1 is double headed. Pick a coin at random and toss it 10 times. Given that you see 10 heads, what is the probability that the next toss of that coin is also a head? 17
Q15. What do you understand by statistical power of sensitivity and how do you calculate it? 18
Q16. Why is Re-sampling done? 18
Q17. What are the differences between over-fitting and under-fitting? 19
Q18. How to combat Overfitting and Underfitting? 19
Q19. What is regularization? Why is it useful? 20
Q20. What Is the Law of Large Numbers? 20
Q21. What Are Confounding Variables? 20
Q22. What Are the Types of Biases That Can Occur During Sampling? 20
Q23. What is Survivorship Bias? 20
Q24. What is Selection Bias? What is under coverage bias? 21
Q25. Explain how a ROC curve works? 21
Q26. What is TF/IDF vectorization? 22
Q27. Why we generally use Soft-max (or sigmoid) non-linearity function as last operation in-network? Why RELU in an inner layer? 22

DATA ANALYSIS **23**

Q1. Python or R – Which one would you prefer for text analytics? 23
Q2. How does data cleaning play a vital role in the analysis? 23
Q3. Differentiate between univariate, bivariate and multivariate analysis. 23
Q4. Explain Star Schema. 23
Q5. What is Cluster Sampling? 23

Q6. What is Systematic Sampling? 24
Q7. What are Eigenvectors and Eigenvalues? 24
Q8. Can you cite some examples where a false positive is important than a false negative? 24
Q9. Can you cite some examples where a false negative important than a false positive? And vice versa? 24
Q10. Can you cite some examples where both false positive and false negatives are equally important? 25
Q11. Can you explain the difference between a Validation Set and a Test Set? 25
Q12. Explain cross-validation. 25

MACHINE LEARNING 27

Q1. What is Machine Learning? 27
Q2. What is Supervised Learning? 27
Q3. What is Unsupervised learning? 27
Q4. What are the various algorithms? 27
Q5. What is 'Naive' in a Naive Bayes? 28
Q6. What is PCA? When do you use it? 29
Q7. Explain SVM algorithm in detail. 30
Q8. What are the support vectors in SVM? 31
Q9. What are the different kernels in SVM? 32
Q10. What are the most known ensemble algorithms? 32
Q11. Explain Decision Tree algorithm in detail. 32
Q12. What are Entropy and Information gain in Decision tree algorithm? 33
Gini Impurity and Information Gain - CART 34
Entropy and Information Gain – ID3 37
Q13. What is pruning in Decision Tree? 41
Q14. What is logistic regression? State an example when you have used logistic regression recently. 41
Q15. What is Linear Regression? 42
Q16. What Are the Drawbacks of the Linear Model? 43
Q17. What is the difference between Regression and classification ML techniques? 43
Q18. What are Recommender Systems? 43
Q19. What is Collaborative filtering? And a content based? 44
Q20. How can outlier values be treated? 44
Q21. What are the various steps involved in an analytics project? 45
Q22. During analysis, how do you treat missing values? 45
Q23. How will you define the number of clusters in a clustering algorithm? 45
Q24. What is Ensemble Learning? 48
Q25. Describe in brief any type of Ensemble Learning. 49
Bagging 49
Boosting 49
Q26. What is a Random Forest? How does it work? 50
Q27. How Do You Work Towards a Random Forest? 51
Q28. What cross-validation technique would you use on a time series data set? 52
Q29. What is a Box-Cox Transformation? 53
Q30. How Regularly Must an Algorithm be Updated? 53
Q31. If you are having 4GB RAM in your machine and you want to train your model on 10GB data set. How would you go about this problem? Have you ever faced this kind of problem in your machine learning/data science experience so far? 53

DEEP LEARNING 55

Q1. What do you mean by Deep Learning? 55
Q2. What is the difference between machine learning and deep learning? 55
Q3. What, in your opinion, is the reason for the popularity of Deep Learning in recent times? 56
Q4. What is reinforcement learning? 56
Q5. What are Artificial Neural Networks? 57

Q6. Describe the structure of Artificial Neural Networks? 57
Q7. How Are Weights Initialized in a Network? 57
Q8. What Is the Cost Function? 58
Q9. What Are Hyperparameters? 58
Q10. What Will Happen If the Learning Rate Is Set inaccurately (Too Low or Too High)? 58
Q11. What Is The Difference Between Epoch, Batch, and Iteration in Deep Learning? 58
Q12. What Are the Different Layers on CNN? 58
Convolution Operation *60*
Pooling Operation *62*
Classification *63*
Training *64*
Testing *65*
Q13. What Is Pooling on CNN, and How Does It Work? 65
Q14. What are Recurrent Neural Networks (RNNs)? 65
Parameter Sharing *67*
Deep RNNs *68*
Bidirectional RNNs *68*
Recursive Neural Network *69*
Encoder Decoder Sequence to Sequence RNNs *70*
LSTMs *70*
Q15. How Does an LSTM Network Work? 70
Recurrent Neural Networks *71*
The Problem of Long-Term Dependencies *72*
LSTM Networks *73*
The Core Idea Behind LSTMs *74*
Q16. What Is a Multi-layer Perceptron (MLP)? 75
Q17. Explain Gradient Descent. 76
Q18. What is exploding gradients? 77
Solutions *78*
Q19. What is vanishing gradients? 78
Solutions *79*
Q20. What is Back Propagation and Explain it Works. 79
Q21. What are the variants of Back Propagation? 79
Q22. What are the different Deep Learning Frameworks? 81
Q23. What is the role of the Activation Function? 81
Q24. Name a few Machine Learning libraries for various purposes. 81
Q25. What is an Auto-Encoder? 81
Q26. What is a Boltzmann Machine? 82
Q27. What Is Dropout and Batch Normalization? 83
Q28. Why Is TensorFlow the Most Preferred Library in Deep Learning? 83
Q29. What Do You Mean by Tensor in TensorFlow? 83
Q30. What is the Computational Graph? 83
Q31. How is logistic regression done? 83

MISCELLANEOUS 84

Q1. Explain the steps in making a decision tree. 84
Q2. How do you build a random forest model? 84
Q3. Differentiate between univariate, bivariate, and multivariate analysis. 85
Univariate *85*
Bivariate *85*
Multivariate *85*
Q4. What are the feature selection methods used to select the right variables? 86
Filter Methods *86*

Wrapper Methods *86*
Q5. In your choice of language, write a program that prints the numbers ranging from one to 50. But for multiples of three, print "Fizz" instead of the number and for the multiples of five, print "Buzz." For numbers which are multiples of both three and five, print "FizzBuzz." 86
Q6. You are given a data set consisting of variables with more than 30 percent missing values. How will you deal with them? 87
Q7. For the given points, how will you calculate the Euclidean distance in Python? 87
Q8. What are dimensionality reduction and its benefits? 87
Q9. How will you calculate eigenvalues and eigenvectors of the following 3x3 matrix? 88
Q10. How should you maintain a deployed model? 88
Q11. How can a time-series data be declared as stationery? 88
Q12. 'People who bought this also bought...' recommendations seen on Amazon are a result of which algorithm? 89
Q13. What is a Generative Adversarial Network? 89
Q14. You are given a dataset on cancer detection. You have built a classification model and achieved an accuracy of 96 percent. Why shouldn't you be happy with your model performance? What can you do about it? 90
Q15. Below are the eight actual values of the target variable in the train file. What is the entropy of the target variable? [0, 0, 0, 1, 1, 1, 1, 1] 90
Q16. We want to predict the probability of death from heart disease based on three risk factors: age, gender, and blood cholesterol level. What is the most appropriate algorithm for this case? Choose the correct option: 90
Q17. After studying the behavior of a population, you have identified four specific individual types that are valuable to your study. You would like to find all users who are most similar to each individual type. Which algorithm is most appropriate for this study? 90
Q18. You have run the association rules algorithm on your dataset, and the two rules {banana, apple} => {grape} and {apple, orange} => {grape} have been found to be relevant. What else must be true? Choose the right answer: .. 90
Q19. Your organization has a website where visitors randomly receive one of two coupons. It is also possible that visitors to the website will not receive a coupon. You have been asked to determine if offering a coupon to website visitors has any impact on their purchase decisions. Which analysis method should you use? 91
Q20. What are the feature vectors? 91
Q21. What is root cause analysis? 91
Q22. Do gradient descent methods always converge to similar points? 91
Q23. What are the most popular Cloud Services used in Data Science? 91
Q24. What is a Canary Deployment? 92
Q25. What is a Blue Green Deployment? 93

Data Science interview questions

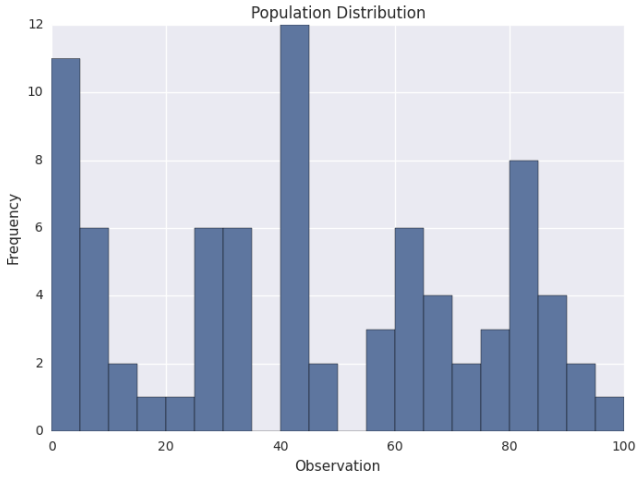
Statistics

Q1. What is the Central Limit Theorem and why is it important?

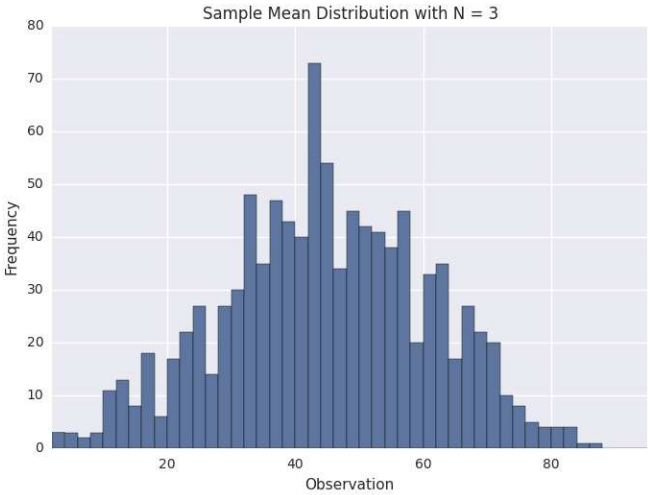
https://spin.atomicobject.com/2015/02/12/central-limit-theorem-intro/

Suppose that we are interested in estimating the average height among all people. Collecting data for every person in the world is impractical, bordering on impossible. While we can't obtain a height measurement from everyone in the population, we can still sample some people. The question now becomes, what can we say about the average height of the entire population given a single sample.
The Central Limit Theorem addresses this question exactly. Formally, it states that if we sample from a population using a sufficiently large sample size, the mean of the samples (also known as the sample population) will be normally distributed (assuming true random sampling), the mean tending to the mean of the population and variance equal to the variance of the population divided by the size of the sampling. What's especially important is that this will be true regardless of the distribution of the original population.

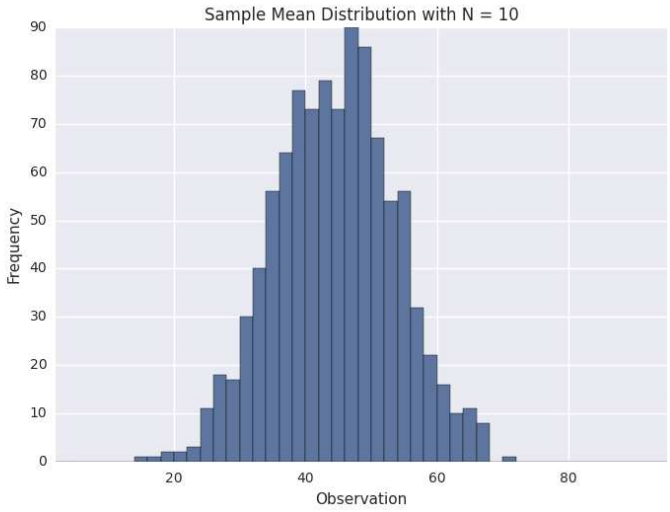
EX:



As we can see, the distribution is pretty ugly. It certainly isn't normal, uniform, or any other commonly known distribution. In order to sample from the above distribution, we need to define a sample size, referred to as N. This is the number of observations that we will sample at a time. Suppose that we choose N to be 3. This means that we will sample in groups of 3. So for the above population, we might sample groups such as [5, 20, 41], [60, 17, 82], [8, 13, 61], and so on.
Suppose that we gather 1,000 samples of 3 from the above population. For each sample, we can compute its average. If we do that, we will have 1,000 averages. This set of 1,000 averages is called a sampling distribution, and according to Central Limit Theorem, the sampling distribution will approach a normal distribution as the sample size N used to produce it increases. Here is what our sample distribution looks like for N = 3.



As we can see, it certainly looks uni-modal, though not necessarily normal. If we repeat the same process with a larger sample size, we should see the sampling distribution start to become more normal. Let's repeat the same process again with N = 10. Here is the sampling distribution for that sample size.



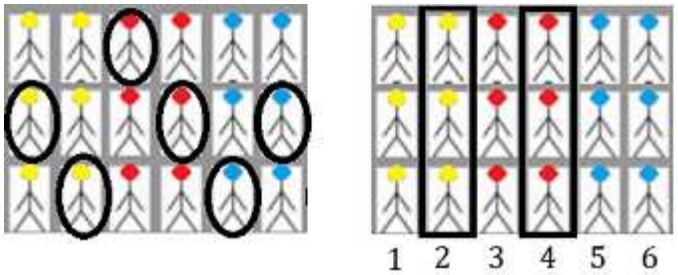
Q2. What is sampling? How many sampling methods do you know?

https://searchbusinessanalytics.techtarget.com/definition/data-sampling
https://nikolanews.com/difference-between-stratified-sampling-cluster-sampling-and-quota-sampling/

Data sampling is a statistical analysis technique used to select, manipulate and analyze a representative subset of data points to identify patterns and trends in the larger data set being examined. It enables data scientists, predictive modelers and other data analysts to work with a small, manageable amount of data about a statistical population to build and run analytical models more quickly, while still producing accurate findings.

Sampling can be particularly useful with data sets that are too large to efficiently analyze in full – for example, in big data analytics applications or surveys. Identifying and analyzing a representative sample is more efficient and cost-effective than surveying the entirety of the data or population.
An important consideration, though, is the size of the required data sample and the possibility of introducing a sampling error. In some cases, a small sample can reveal the most important information about a data set. In others, using a larger sample can increase the likelihood of accurately representing the data as a whole, even though the increased size of the sample may impede ease of manipulation and interpretation.
There are many different methods for drawing samples from data; the ideal one depends on the data set and situation. Sampling can be based on probability, an approach that uses random numbers that correspond to points in the data set to ensure that there is no correlation between points chosen for the sample. Further variations in probability sampling include:

- Simple random sampling: Software is used to randomly select subjects from the whole population.
- Stratified sampling: Subsets of the data sets or population are created based on a common factor, and samples are randomly collected from each subgroup. A sample is drawn from each strata (using a random sampling method like simple random sampling or systematic sampling).
    - *EX: In the image below, let's say you need a sample size of 6. Two members from each group (yellow, red, and blue) are selected randomly. Make sure to sample proportionally: In this simple example, 1/3 of each group (2/6 yellow, 2/6 red and 2/6 blue) has been sampled. If you have one group that's a different size, make sure to adjust your proportions. For example, if you had 9 yellow, 3 red and 3 blue, a 5-item sample would consist of 3/9 yellow (i.e. one third), 1/3 red and 1/3 blue.*
- Cluster sampling: The larger data set is divided into subsets (clusters) based on a defined factor, then a random sampling of clusters is analyzed. The sampling unit is the whole cluster; Instead of sampling individuals from within each group, a researcher will study whole clusters.
    - *EX: In the image below, the strata are natural groupings by head color (yellow, red, blue). A sample size of 6 is needed, so two of the complete strata are selected randomly (in this example, groups 2 and 4 are chosen).*

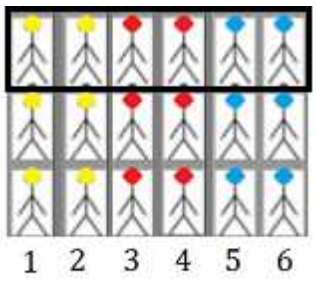


- Multistage sampling: A more complicated form of cluster sampling, this method also involves dividing the larger population into a number of clusters. Second-stage clusters are then broken out based on a secondary factor, and those clusters are then sampled and analyzed. This staging could continue as multiple subsets are identified, clustered and analyzed.
- Systematic sampling: A sample is created by setting an interval at which to extract data from the larger population – for example, selecting every 10th row in a spreadsheet of 200 items to create a sample size of 20 rows to analyze.

Sampling can also be based on non-probability, an approach in which a data sample is determined and extracted based on the judgment of the analyst. As inclusion is determined by the analyst, it can be more difficult to extrapolate whether the sample accurately represents the larger population than when probability sampling is used.

Non-probability data sampling methods include:

- Convenience sampling: Data is collected from an easily accessible and available group.
- Consecutive sampling: Data is collected from every subject that meets the criteria until the predetermined sample size is met.
- Purposive or judgmental sampling: The researcher selects the data to sample based on predefined criteria.
- Quota sampling: The researcher ensures equal representation within the sample for all subgroups in the data set or population (random sampling is not used).



Quota Sampling

Once generated, a sample can be used for predictive analytics. For example, a retail business might use data sampling to uncover patterns about customer behavior and predictive modeling to create more effective sales strategies.

Q3. What is the difference between type I vs type II error?

https://www.datasciencecentral.com/profiles/blogs/understanding-type-i-and-type-ii-errors

Is H_a true? No, H_0 is True (H_a is Negative: TN); Yes, H_0 is False (H_a is Positive: TP).
A type I error occurs when the null hypothesis is true but is rejected. A type II error occurs when the null hypothesis is false but erroneously fails to be rejected.

	No reject H_0	Reject H_0
H_0 is True	TN	FP (I error)
H_0 is False	FN (II error)	TP

Q4. What is linear regression? What do the terms p-value, coefficient, and r-squared value mean? What is the significance of each of these components?

https://www.springboard.com/blog/linear-regression-in-python-a-tutorial/
https://blog.minitab.com/blog/adventures-in-statistics-2/how-to-interpret-regression-analysis-results-p-values-and-coefficients

Imagine you want to predict the price of a house. That will depend on some factors, called independent variables, such as location, size, year of construction… if we assume there is a linear relationship between these variables and the price (our dependent variable), then our price is predicted by the following function:

$$Y = a + b\,X$$

The p-value in the table is the minimum α (the significance level) at which the coefficient is relevant. The lower the p-value, the more important is the variable in predicting the price. Usually we set a 5% level, so that we have a 95% confidentiality that our variable is relevant.
The p-value is used as an alternative to rejection points to provide the smallest level of significance at which the null hypothesis would be rejected. A smaller p-value means that there is stronger evidence in favor of the alternative hypothesis.

The coefficient value signifies how much the mean of the dependent variable changes given a one-unit shift in the independent variable while holding other variables in the model constant. This property of holding the other variables constant is crucial because it allows you to assess the effect of each variable in isolation from the others.

R squared (R^2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model.

Q5. What are the assumptions required for linear regression?

There are four major assumptions:

- There is a linear relationship between the dependent variables and the regressors, meaning the model you are creating actually fits the data,
- The errors or residuals ($y_i - \hat{y}_i$) of the data are normally distributed and independent from each other,
- There is minimal multicollinearity between explanatory variables, and
- Homoscedasticity. This means the variance around the regression line is the same for all values of the predictor variable.

Q6. What is a statistical interaction?

http://icbseverywhere.com/blog/mini-lessons-tutorials-and-support-pages/statistical-interactions/

Basically, an interaction is when the effect of one factor (input variable) on the dependent variable (output variable) differs among levels of another factor. When two or more independent variables are involved in a research design, there is more to consider than simply the "main effect" of each of the independent variables (also termed "factors"). That is, the effect of one independent variable on the dependent variable of interest may not be the same at all levels of the other independent variable. Another way to put this is that the effect of one independent variable may depend on the level of the other independent variable. In order to find an interaction, you must have a factorial design, in which the two (or more)

independent variables are "crossed" with one another so that there are observations at every combination of levels of the two independent variables. *EX: stress level and practice to memorize words: together they may have a lower performance.*

Q7. What is selection bias?

https://www.elderresearch.com/blog/selection-bias-in-analytics

Selection (or 'sampling') bias occurs when the sample data that is gathered and prepared for modeling has characteristics that are not representative of the true, future population of cases the model will see. That is, active selection bias occurs when a subset of the data is systematically (i.e., non-randomly) excluded from analysis.

Q8. What is an example of a data set with a non-Gaussian distribution?

https://www.quora.com/Most-machine-learning-datasets-are-in-Gaussian-distribution-Where-can-we-find-the-dataset-which-follows-Bernoulli-Poisson-gamma-beta-etc-distribution

The Gaussian distribution is part of the Exponential family of distributions, but there are a lot more of them, with the same sort of ease of use, in many cases, and if the person doing the machine learning has a solid grounding in statistics, they can be utilized where appropriate.

Binomial: multiple toss of a coin Bin(n,p): the binomial distribution consists of the probabilities of each of the possible numbers of successes on n trials for independent events that each have a probability of p of occurring.
Bernoulli: Bin(1,p) = Be(p)
Poisson: Pois(λ)

Data Science

Q1. What is Data Science? List the differences between supervised and unsupervised learning.

Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data. How is this different from what statisticians have been doing for years? The answer lies in the difference between explaining and predicting: statisticians work a posteriori, explaining the results and designing a plan; data scientists use historical data to make predictions.

The differences between supervised and unsupervised learning are:

Supervised	Unsupervised
Input data is labelled	Input data is unlabeled
Split in training/validation/test	No split
Used for prediction	Used for analysis
Classification and Regression	Clustering, dimension reduction, and density estimation

Q2. What is Selection Bias?

Selection bias is a kind of error that occurs when the researcher decides what has to be studied. It is associated with research where the selection of participants is not random. Therefore, some conclusions of the study may not be accurate.

The types of selection bias include:

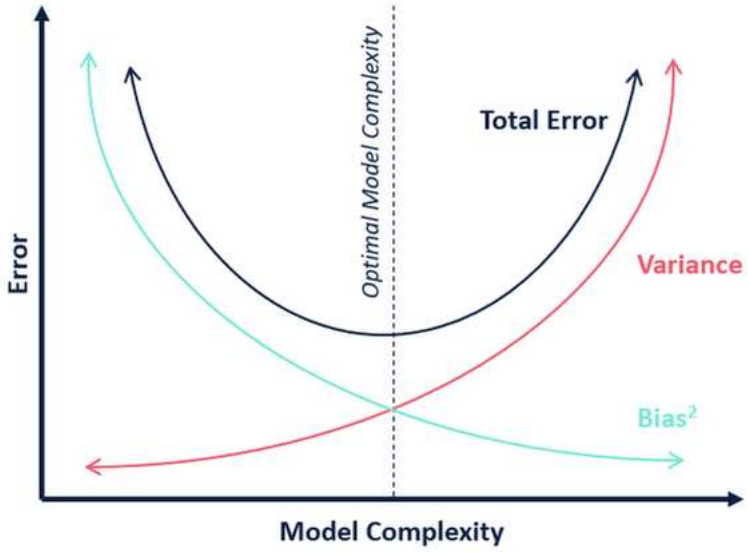
- Sampling bias: It is a systematic error due to a non-random sample of a population causing some members of the population to be less likely to be included than others resulting in a biased sample.
- Time interval: A trial may be terminated early at an extreme value (often for ethical reasons), but the extreme value is likely to be reached by the variable with the largest variance, even if all variables have a similar mean.
- Data: When specific subsets of data are chosen to support a conclusion or rejection of bad data on arbitrary grounds, instead of according to previously stated or generally agreed criteria.
- Attrition: Attrition bias is a kind of selection bias caused by attrition (loss of participants) discounting trial subjects/tests that did not run to completion.

Q3. What is bias-variance trade-off?

Bias: Bias is an error introduced in the model due to the oversimplification of the algorithm used (does not fit the data properly). It can lead to under-fitting.
Low bias machine learning algorithms — Decision Trees, k-NN and SVM
High bias machine learning algorithms — Linear Regression, Logistic Regression

Variance: Variance is error introduced in the model due to a too complex algorithm, it performs very well in the training set but poorly in the test set. It can lead to high sensitivity and overfitting.
Possible high variance – polynomial regression

Normally, as you increase the complexity of your model, you will see a reduction in error due to lower bias in the model. However, this only happens until a particular point. As you continue to make your model more complex, you end up over-fitting your model and hence your model will start suffering from high variance.



Bias-Variance trade-off: The goal of any supervised machine learning algorithm is to have low bias and low variance to achieve good prediction performance.

1. The k-nearest neighbor algorithm has low bias and high variance, but the trade-off can be changed by increasing the value of k which increases the number of neighbors that contribute to the prediction and in turn increases the bias of the model.
2. The support vector machine algorithm has low bias and high variance, but the trade-off can be changed by increasing the C parameter that influences the number of violations of the margin allowed in the training data which increases the bias but decreases the variance.
3. The decision tree has low bias and high variance, you can decrease the depth of the tree or use fewer attributes.
4. The linear regression has low variance and high bias, you can increase the number of features or use another regression that better fits the data.

There is no escaping the relationship between bias and variance in machine learning. Increasing the bias will decrease the variance. Increasing the variance will decrease bias.

Q4. What is a confusion matrix?

The confusion matrix is a 2X2 table that contains 4 outputs provided by the binary classifier.

	Predict +	Predict -
Actual +	TP	FN (II error)
Actual -	FP (I error)	TN

A data set used for performance evaluation is called a test data set. It should contain the correct labels and predicted labels. The predicted labels will exactly the same if the performance of a binary classifier is perfect. The predicted labels usually match with part of the observed labels in real-world scenarios.
A binary classifier predicts all data instances of a test data set as either positive or negative. This produces four outcomes: TP, FP, TN, FN. Basic measures derived from the confusion matrix:

1. $Error\ Rate\ = \frac{FP+FN}{P+N}$
2. $Accuracy\ = \frac{TP+TN}{P+N}$
3. $Sensitivity\ (Recall\ or\ True\ positive\ rate) = \frac{TP}{TP+FN} = \frac{TP}{P}$
4. $Specificity\ (True\ negative\ rate) = \frac{TN}{TN+FP} = \frac{TN}{N}$
5. $Precision\ (Positive\ predicted\ value) = \frac{TP}{TP+FP}$
6. $F-Score\ (Harmonic\ mean\ of\ precision\ and\ recall) = \frac{2\,TP}{(2\,TP+FP+FN)}$

Q5. What is the difference between “long” and “wide” format data?

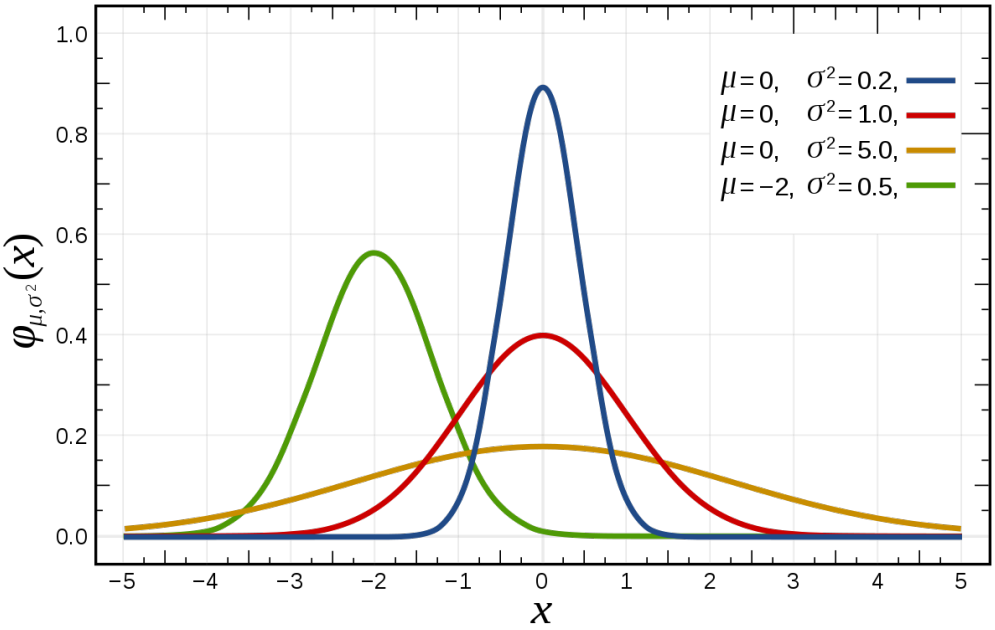
In the wide-format, a subject’s repeated responses will be in a single row, and each response is in a separate column. In the long-format, each row is a one-time point per subject. You can recognize data in wide format by the fact that columns generally represent groups (variables).

X	Y1	Y2	Y3
10	2	3	4
15	0	4	6
20	1	4	5

VarName	X	Value
Y1	10	2
Y2	10	3
Y3	10	4
Y1	15	0
Y2	15	4
Y3	15	6
Y1	20	1
Y2	20	4
Y3	20	5

Q6. What do you understand by the term Normal Distribution?

Data is usually distributed in different ways with a bias to the left or to the right or it can all be jumbled up. However, there are chances that data is distributed around a central value without any bias to the left or right and reaches normal distribution in the form of a bell-shaped curve.



The random variables are distributed in the form of a symmetrical, bell-shaped curve. Properties of Normal Distribution are as follows:

1. Unimodal (Only one mode)
2. Symmetrical (left and right halves are mirror images)
3. Bell-shaped (maximum height (mode) at the mean)
4. Mean, Mode, and Median are all located in the center
5. Asymptotic

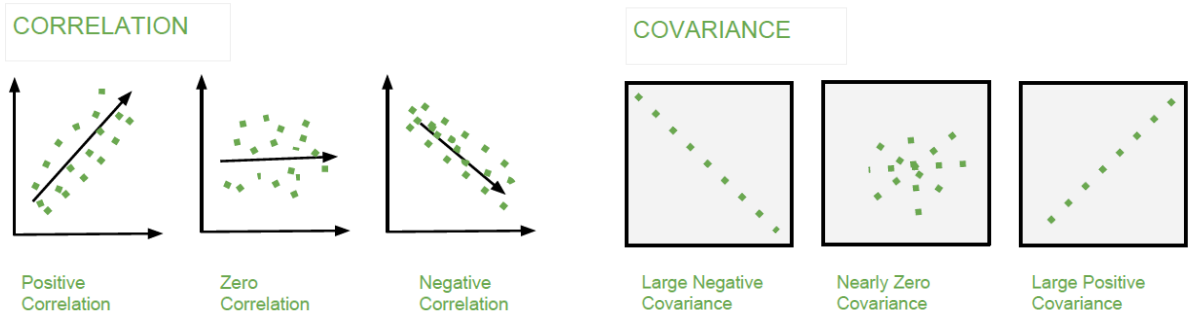
Q7. What is correlation and covariance in statistics?

Correlation is considered or described as the best technique for measuring and also for estimating the quantitative relationship between two variables. Correlation measures how strongly two variables are related. Given two random variables, it is the covariance between both divided by the product of the two standard deviations of the single variables, hence always between -1 and 1.

$$\rho = \frac{Cov(X,Y)}{\sigma(X)\,\sigma(Y)} \in [-1,1]$$

Covariance is a measure that indicates the extent to which two random variables change in cycle. It explains the systematic relation between a pair of random variables, wherein changes in one variable reciprocal by a corresponding change in another variable.

$$Cov(X,Y) = E[(X - E[X])(Y - E[Y])] = E[XY] - E[X]E[Y]$$



Q8. What is the difference between Point Estimates and Confidence Interval?

Point Estimation gives us a particular value as an estimate of a population parameter. Method of Moments and Maximum Likelihood estimator methods are used to derive Point Estimators for population parameters.
A confidence interval gives us a range of values which is likely to contain the population parameter. The confidence interval is generally preferred, as it tells us how likely this interval is to contain the population parameter. This likeliness or probability is called Confidence Level or Confidence coefficient and represented by $1 - \alpha$, where α is the level of significance.

Q9. What is the goal of A/B Testing?

It is a hypothesis testing for a randomized experiment with two variables A and B.
The goal of A/B Testing is to identify any changes to the web page to maximize or increase the outcome of interest. A/B testing is a fantastic method for figuring out the best online promotional and marketing strategies for your business. *It can be used to test everything from website copy to sales emails to search ads. An example of this could be identifying the click-through rate for a banner ad.*

Q10. What is p-value?

When you perform a hypothesis test in statistics, a p-value can help you determine the strength of your results. p-value is the minimum significance level at which you can reject the null hypothesis. The lower the p-value, the more likely you reject the null hypothesis.

Q11. In any 15-minute interval, there is a 20% probability that you will see at least one shooting star. What is the probability that you see at least one shooting star in the period of an hour?

- $Probability\ of\ not\ seeing\ any\ shooting\ star\ in\ 15\ minutes\ is\ =\ 1 - P(Seeing\ one\ shooting\ star) = 1 - 0.2 = 0.8$
- $Probability\ of\ not\ seeing\ any\ shooting\ star\ in\ the\ period\ of\ one\ hour = (0.8)^4 = 0.4096$

- $Probability\ of\ seeing\ at\ least\ one\ shooting\ star\ in\ the\ one\ hour\ =$
  $1 - P(Not\ seeing\ any\ star) = 1 - 0.4096 = 0.5904$

Q12. How can you generate a random number between 1 – 7 with only a die?

Any die has six sides from 1-6. There is no way to get seven equal outcomes from a single rolling of a die. If we roll the die twice and consider the event of two rolls, we now have 36 different outcomes. To get our 7 equal outcomes we have to reduce this 36 to a number divisible by 7. We can thus consider only 35 outcomes and exclude the other one. A simple scenario can be to exclude the combination (6,6), i.e., to roll the die again if 6 appears twice. All the remaining combinations from (1,1) till (6,5) can be divided into 7 parts of 5 each. This way all the seven sets of outcomes are equally likely.

Q13. A certain couple tells you that they have two children, at least one of which is a girl. What is the probability that they have two girls?

$$P(Having\ two\ girls\ given\ one\ girl) = \frac{1}{2}$$

Q14. A jar has 1000 coins, of which 999 are fair and 1 is double headed. Pick a coin at random and toss it 10 times. Given that you see 10 heads, what is the probability that the next toss of that coin is also a head?

There are two ways of choosing the coin. One is to pick a fair coin and the other is to pick the one with two heads.

$$Probability\ of\ selecting\ fair\ coin\ = \frac{999}{1000} = 0.999$$

$$Probability\ of\ selecting\ unfair\ coin\ = \frac{1}{1000} = 0.001$$

$$Selecting\ 10\ heads\ in\ a\ row$$
$$= Selecting\ fair\ coin\ *\ Getting\ 10\ heads\ +\ Selecting\ unfair\ coin$$
$$= P(A) + P(B)$$

$$P(A) = 0.999 * \left(\frac{1}{2}\right)^{10} = 0.999 * \left(\frac{1}{1024}\right) = 0.000976$$

$$P(B) = 0.001 * 1 = 0.001$$

$$\frac{P(A)}{P(A) + P(B)} = \frac{0.000976}{0.000976 + 0.001} = 0.4939$$

$$\frac{P(B)}{P(A) + P(B)} = \frac{0.001}{0.001976} = 0.5061$$

$$Probability\ of\ selecting\ another\ head\ = \frac{P(A)}{P(A)+P(B)} * 0.5 + \frac{P(B)}{P(A)+P(B)} * 1 =$$
$$= 0.4939 * 0.5 + 0.5061 = 0.7531$$

Q15. What do you understand by statistical power of sensitivity and how do you calculate it?

Sensitivity is commonly used to validate the accuracy of a classifier (Logistic, SVM, Random Forest etc.).

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

Q16. Why is Re-sampling done?

https://machinelearningmastery.com/statistical-sampling-and-resampling/

- Sampling is an active process of gathering observations with the intent of estimating a population variable.
- Resampling is a methodology of economically using a data sample to improve the accuracy and quantify the uncertainty of a population parameter. Resampling methods, in fact, make use of a nested resampling method.

Once we have a data sample, it can be used to estimate the population parameter. The problem is that we only have a single estimate of the population parameter, with little idea of the variability or uncertainty in the estimate. One way to address this is by estimating the population parameter multiple times from our data sample. This is called resampling. Statistical resampling methods are procedures that describe how to economically use available data to estimate a population parameter. The result can be both a more accurate estimate of the parameter (such as taking the mean of the estimates) and a quantification of the uncertainty of the estimate (such as adding a confidence interval).

Resampling methods are very easy to use, requiring little mathematical knowledge. A downside of the methods is that they can be computationally very expensive, requiring tens, hundreds, or even thousands of resamples in order to develop a robust estimate of the population parameter.

The key idea is to resample form the original data — either directly or via a fitted model — to create replicate datasets, from which the variability of the quantiles of interest can be assessed without long-winded and error-prone analytical calculation. Because this approach involves repeating the original data analysis procedure with many replicate sets of data, these are sometimes called computer-intensive methods. Each new subsample from the original data sample is used to estimate the population parameter. The sample of estimated population parameters can then be considered with statistical tools in order to quantify the expected value and variance, providing measures of the uncertainty of the estimate. Statistical sampling methods can be used in the selection of a subsample from the original sample.

A key difference is that process must be repeated multiple times. The problem with this is that there will be some relationship between the samples as observations that will be shared across multiple subsamples. This means that the subsamples and the estimated population parameters are not strictly

identical and independently distributed. This has implications for statistical tests performed on the sample of estimated population parameters downstream, i.e. paired statistical tests may be required.

Two commonly used resampling methods that you may encounter are k-fold cross-validation and the bootstrap.

- Bootstrap. Samples are drawn from the dataset with replacement (allowing the same sample to appear more than once in the sample), where those instances not drawn into the data sample may be used for the test set.
- k-fold Cross-Validation. A dataset is partitioned into k groups, where each group is given the opportunity of being used as a held out test set leaving the remaining groups as the training set. The k-fold cross-validation method specifically lends itself to use in the evaluation of predictive models that are repeatedly trained on one subset of the data and evaluated on a second held-out subset of the data.

Resampling is done in any of these cases:

- Estimating the accuracy of sample statistics by using subsets of accessible data or drawing randomly with replacement from a set of data points
- Substituting labels on data points when performing significance tests
- Validating models by using random subsets (bootstrapping, cross-validation)

Q17. What are the differences between over-fitting and under-fitting?

In statistics and machine learning, one of the most common tasks is to fit a model to a set of training data, so as to be able to make reliable predictions on general untrained data.

In overfitting, a statistical model describes random error or noise instead of the underlying relationship. Overfitting occurs when a model is excessively complex, such as having too many parameters relative to the number of observations. A model that has been overfitted, has poor predictive performance, as it overreacts to minor fluctuations in the training data.

Underfitting occurs when a statistical model or machine learning algorithm cannot capture the underlying trend of the data. Underfitting would occur, for example, when fitting a linear model to non-linear data. Such a model too would have poor predictive performance.

Q18. How to combat Overfitting and Underfitting?

To combat overfitting:

1. Add noise
2. Feature selection
3. Increase training set

4. L2 (ridge) or L1 (lasso) regularization; L1 drops weights, L2 no
5. Use cross-validation techniques, such as k folds cross-validation
6. Boosting and bagging

7. Dropout technique

8. Perform early stopping

9. Remove inner layers

To combat underfitting:

1. Add features
2. Increase time of training

Q19. What is regularization? Why is it useful?

Regularization is the process of adding tuning parameter (penalty term) to a model to induce smoothness in order to prevent overfitting. This is most often done by adding a constant multiple to an existing weight vector. This constant is often the L1 (Lasso - $|\alpha|$) or L2 (Ridge - α^2). The model predictions should then minimize the loss function calculated on the regularized training set.

Q20. What Is the Law of Large Numbers?

It is a theorem that describes the result of performing the same experiment a large number of times. This theorem forms the basis of frequency-style thinking. It says that the sample means, the sample variance and the sample standard deviation converge to what they are trying to estimate. According to the law, the average of the results obtained from a large number of trials should be close to the expected value and will tend to become closer to the expected value as more trials are performed.

Q21. What Are Confounding Variables?

In statistics, a confounder is a variable that influences both the dependent variable and independent variable.

If you are researching whether a lack of exercise leads to weight gain:
lack of exercise = independent variable
weight gain = dependent variable
A confounding variable here would be any other variable that affects both of these variables, such as the age of the subject.

Q22. What Are the Types of Biases That Can Occur During Sampling?

a. Selection bias
b. Under coverage bias
c. Survivorship bias

Q23. What is Survivorship Bias?

It is the logical error of focusing aspects that support surviving some process and casually overlooking those that did not work because of their lack of prominence. This can lead to wrong conclusions in numerous different means. *For example, during a recession you look just at the survived businesses, noting*

that they are performing poorly. However, they perform better than the rest, which is failed, thus being removed from the time series.

Q24. What is Selection Bias? What is under coverage bias?

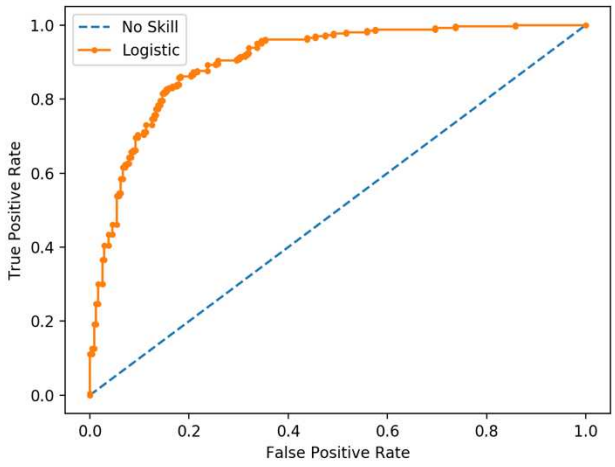
https://stattrek.com/survey-research/survey-bias.aspx

Selection bias occurs when the sample obtained is not representative of the population intended to be analyzed. *For instance, you select only Asians to perform a study on the world population height.*
Under coverage bias occurs when some members of the population are inadequately represented in the sample. *A classic example of under coverage is the Literary Digest voter survey, which predicted that Alfred Landon would beat Franklin Roosevelt in the 1936 presidential election. The survey sample suffered from under coverage of low-income voters, who tended to be Democrats.*
How did this happen? The survey relied on a convenience sample, drawn from telephone directories and car registration lists. In 1936, people who owned cars and telephones tended to be more affluent. Under coverage is often a problem with convenience samples.

Q25. Explain how a ROC curve works?

The ROC curve is a graphical representation of the contrast between true positive rates and false positive rates at various thresholds. It is often used as a proxy for the trade-off between the sensitivity (true positive rate) and false positive rate.

- $TPR = \frac{TP}{P} = \frac{TP}{TP+FN}$
- $TNR = \frac{TN}{TN+FP} = \frac{TN}{N}$
- $FPR = \frac{FP}{TN+FP}$
- $FNR = \frac{FN}{FN+TP}$



Q26. What is TF/IDF vectorization?

TF-IDF is short for term frequency-inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining.

- $TF = \frac{\#\ 'word'\ in\ doc}{tot\ \#\ words\ in\ doc}$
- $IDF = log\left(\frac{\#\ docs\ with\ 'word'\ in\ it}{tot\ docs\ in\ collection}\right)$

The TF-IDF value increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

Q27. Why we generally use Soft-max (or sigmoid) non-linearity function as last operation in-network? Why RELU in an inner layer?

It is because it takes in a vector of real numbers and returns a probability distribution. Its definition is as follows. Let x be a vector of real numbers (positive, negative, whatever, there are no constraints).
Then the i-eth component of soft-max(x) is:

It should be clear that the output is a probability distribution: each element is non-negative and the sum over all components is 1.

RELU because it avoids the vanishing gradient descent issue.

Data Analysis

Q1. Python or R – Which one would you prefer for text analytics?

We will prefer Python because of the following reasons:

- Python would be the best option because it has Pandas library that provides easy to use data structures and high-performance data analysis tools.
- R is more suitable for machine learning than just text analysis.
- Python performs faster for all types of text analytics.

Q2. How does data cleaning play a vital role in the analysis?

Data cleaning can help in analysis because:

- Cleaning data from multiple sources helps transform it into a format that data analysts or data scientists can work with.
- Data Cleaning helps increase the accuracy of the model in machine learning.
- It is a cumbersome process because as the number of data sources increases, the time taken to clean the data increases exponentially due to the number of sources and the volume of data generated by these sources.
- It might take up to 80% of the time for just cleaning data making it a critical part of the analysis task.

Q3. Differentiate between univariate, bivariate and multivariate analysis.

Univariate analyses are descriptive statistical analysis techniques which can be differentiated based on one variable involved at a given point of time. *For example, the pie charts of sales based on territory involve only one variable and can the analysis can be referred to as univariate analysis.*

The bivariate analysis attempts to understand the difference between two variables at a time as in a scatterplot. *For example, analyzing the volume of sale and spending can be considered as an example of bivariate analysis.*

Multivariate analysis deals with the study of more than two variables to understand the effect of variables on the responses.

Q4. Explain Star Schema.

It is a traditional database schema with a central table. Satellite tables map IDs to physical names or descriptions and can be connected to the central fact table using the ID fields; these tables are known as lookup tables and are principally useful in real-time applications, as they save a lot of memory. Sometimes star schemas involve several layers of summarization to recover information faster.

Q5. What is Cluster Sampling?

Cluster sampling is a technique used when it becomes difficult to study the target population spread across a wide area and simple random sampling cannot be applied. Cluster Sample is a probability sample where each sampling unit is a collection or cluster of elements.
For example, a researcher wants to survey the academic performance of high school students in Japan. He can divide the entire population of Japan into different clusters (cities). Then the researcher selects a number of clusters depending on his research through simple or systematic random sampling.

Q6. What is Systematic Sampling?

Systematic sampling is a statistical technique where elements are selected from an ordered sampling frame. In systematic sampling, the list is progressed in a circular manner so once you reach the end of the list, it is progressed from the top again. The best example of systematic sampling is equal probability method.

Q7. What are Eigenvectors and Eigenvalues?

Eigenvectors are used for understanding linear transformations. In data analysis, we usually calculate the eigenvectors for a correlation or covariance matrix. Eigenvectors are the directions along which a particular linear transformation acts by flipping, compressing or stretching.
Eigenvalue can be referred to as the strength of the transformation in the direction of eigenvector or the factor by which the compression occurs.

Q8. Can you cite some examples where a false positive is important than a false negative?

Let us first understand what false positives and false negatives are

- False Positives are the cases where you wrongly classified a non-event as an event a.k.a Type I error.
- False Negatives are the cases where you wrongly classify events as non-events, a.k.a Type II error.

Example 1: In the medical field, assume you have to give chemotherapy to patients. Assume a patient comes to that hospital and he is tested positive for cancer, based on the lab prediction but he actually doesn't have cancer. This is a case of false positive. Here it is of utmost danger to start chemotherapy on this patient when he actually does not have cancer. In the absence of cancerous cell, chemotherapy will do certain damage to his normal healthy cells and might lead to severe diseases, even cancer.

Example 2: Let's say an e-commerce company decided to give $1000 Gift voucher to the customers whom they assume to purchase at least $10,000 worth of items. They send free voucher mail directly to 100 customers without any minimum purchase condition because they assume to make at least 20% profit on sold items above $10,000. Now the issue is if we send the $1000 gift vouchers to customers who have not actually purchased anything but are marked as having made $10,000 worth of purchase.

Q9. Can you cite some examples where a false negative important than a false positive? And vice versa?

Example 1 FN: What if Jury or judge decides to make a criminal go free?

Example 2 FN: Fraud detection.

Example 3 FP: customer voucher use promo evaluation: if many used it and actually if was not true, promo sucks.

Q10. Can you cite some examples where both false positive and false negatives are equally important?

In the Banking industry giving loans is the primary source of making money but at the same time if your repayment rate is not good you will not make any profit, rather you will risk huge losses.
Banks don't want to lose good customers and at the same point in time, they don't want to acquire bad customers. In this scenario, both the false positives and false negatives become very important to measure.

Q11. Can you explain the difference between a Validation Set and a Test Set?

A Training Set:

- to fit the parameters i.e. weights

A Validation set:

- part of the training set
- for parameter selection
- to avoid overfitting

A Test set:

- for testing or evaluating the performance of a trained machine learning model, i.e. evaluating the predictive power and generalization.

Q12. Explain cross-validation.

https://machinelearningmastery.com/k-fold-cross-validation/

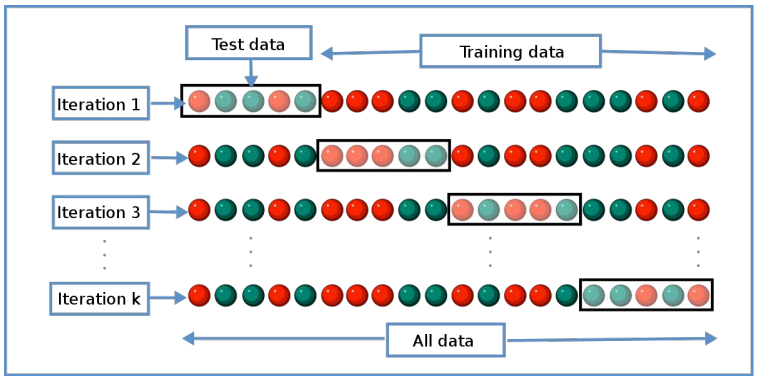
Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation. Mainly used in backgrounds where the objective is forecast, and one wants to estimate how accurately a model will accomplish in practice.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

The general procedure is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:
    a. Take the group as a hold out or test data set
    b. Take the remaining groups as a training data set
    c. Fit a model on the training set and evaluate it on the test set
    d. Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores



There is an alternative in Scikit-Learn called Stratified k fold, in which the split is shuffled to make it sure you have a representative sample of each class and a k fold in which you may not have the assurance of it (not good with a very unbalanced dataset).

Machine Learning

Q1. What is Machine Learning?

Machine learning is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine Learning explores the study and construction of algorithms that can learn from and make predictions on data. You select a model to train and then manually perform feature extraction. Used to devise complex models and algorithms that lend themselves to a prediction which in commercial use is known as predictive analytics.

Q2. What is Supervised Learning?

Supervised learning is the machine learning task of inferring a function from labeled training data. The training data consist of a set of training examples.

Algorithms: Support Vector Machines, Regression, Naive Bayes, Decision Trees, K-nearest Neighbor Algorithm and Neural Networks

E.g. If you built a fruit classifier, the labels will be "this is an orange, this is an apple and this is a banana", based on showing the classifier examples of apples, oranges and bananas.

Q3. What is Unsupervised learning?

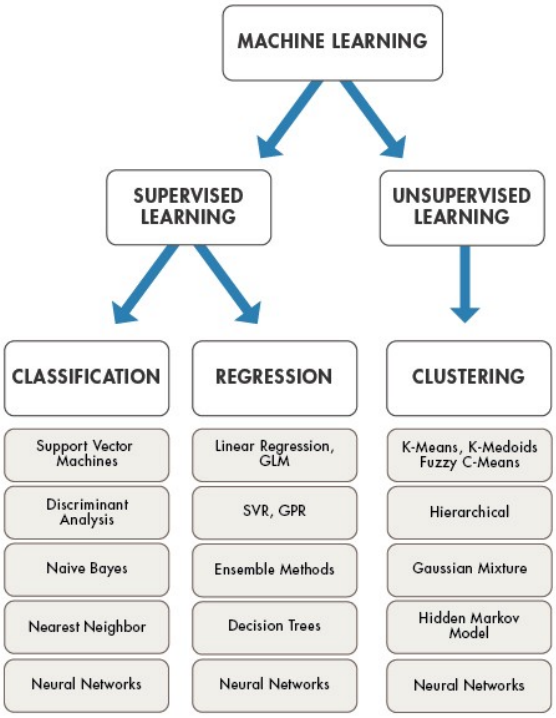
Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labelled responses.

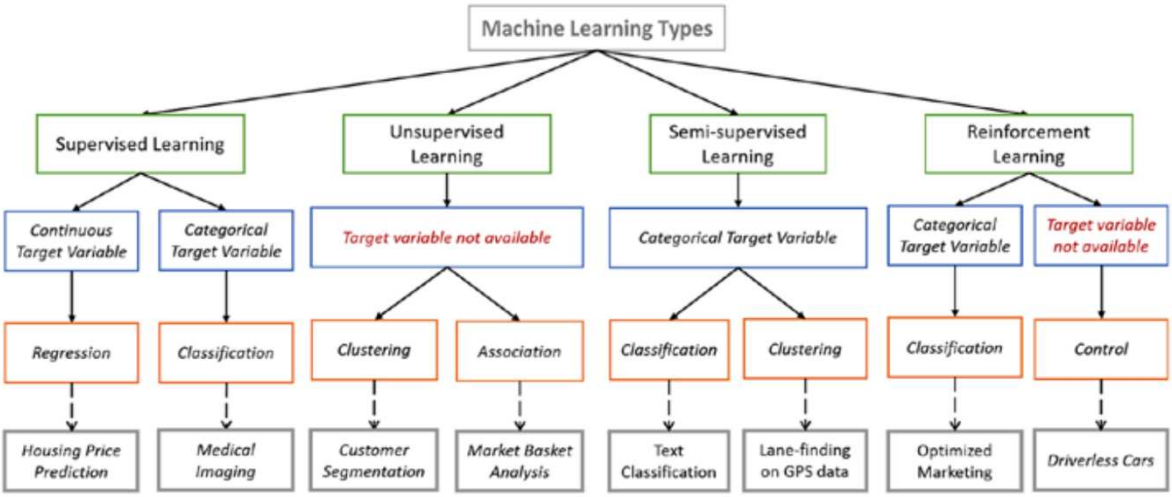
Algorithms: Clustering, Anomaly Detection, Neural Networks and Latent Variable Models

E.g. In the same example, a fruit clustering will categorize as "fruits with soft skin and lots of dimples", "fruits with shiny hard skin" and "elongated yellow fruits".

Q4. What are the various algorithms?

There are various algorithms. Here is a list.





Q5. What is 'Naive' in a Naive Bayes?

https://en.wikipedia.org/wiki/Naive_Bayes_classifier

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Bayes' theorem states the following relationship, given class variable y and dependent feature vector $x_1$through x_n:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}$$

Using the naive conditional independence assumption that each x_i is independent:
for all i, this relationship is simplified to:

$$P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y),$$

Since $P(x_1, \ldots, x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(y|x_i)$; the former is then the relative frequency of class y in the training set.

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(y|x_i)$: can be Bernoulli, Binomial, Gaussian, and so on.

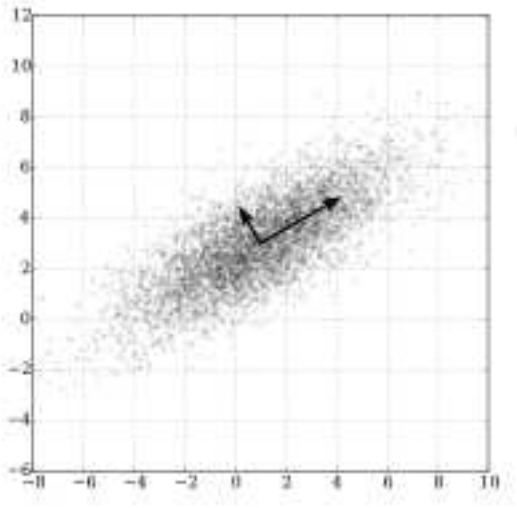
Q6. What is PCA? When do you use it?

https://en.wikipedia.org/wiki/Principal_component_analysis
https://blog.umetrics.com/what-is-principal-component-analysis-pca-and-how-it-is-used
https://blog.umetrics.com/why-preprocesing-data-creates-better-data-analytics-models

Principal component analysis (PCA) is a statistical method used in Machine Learning. It consists in projecting data in a higher dimensional space into a lower dimensional space by maximizing the variance of each dimension.

The process works as following. We define a matrix A with n rows (the single observations of a dataset – in a tabular format, each single row) and p columns, our features. For this matrix we construct a variable space with as many dimensions as there are features. Each feature represents one coordinate axis. For

each feature, the length has been standardized according to a scaling criterion, normally by scaling to unit variance. It is determinant to scale the features to a common scale, otherwise the features with a greater magnitude will weigh more in determining the principal components. Once plotted all the observations and computed the mean of each variable, that mean will be represented by a point in the center of our plot (the center of gravity). Then, we subtract each observation with the mean, shifting the coordinate system with the center in the origin. The best fitting line resulting is the line that best accounts for the shape of the point swarm. It represents the maximum variance direction in the data. Each observation may be projected onto this line in order to get a coordinate value along the PC-line. This value is known as a score. The next best-fitting line can be similarly chosen from directions perpendicular to the first. Repeating this process yields an orthogonal basis in which different individual dimensions of the data are uncorrelated. These basis vectors are called **principal components**.



PCA is mostly used as a tool in exploratory data analysis and for making predictive models. It is often used to visualize genetic distance and relatedness between populations.

Q7. Explain SVM algorithm in detail.

https://en.wikipedia.org/wiki/Support_vector_machine

Classifying data is a common task in machine learning. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new data point will be in. In the case of support-vector machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a $(p-1)$-dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So, we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum-margin classifier; or equivalently, the perceptron of optimal stability. The best hyper plane that divides the data is H_3.

We have n data $(x_1, y_1), \dots, (x_n, y_n)$ and p different features $x_i = (x_i^1, \dots, x_i^p)$ and y_i is either 1 or -1.

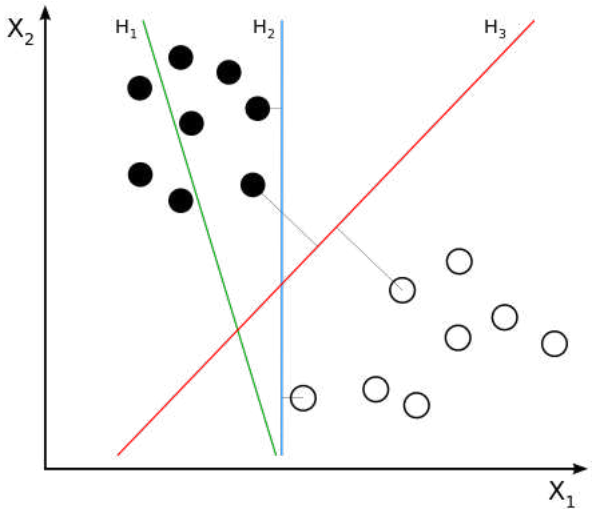
The equation of the hyperplane H_3 is as the set of points x satisfying:

$$w \cdot x - b = 0$$

where w is the (not necessarily normalized) normal vector to the hyperplane. The parameter $\frac{b}{\|w\|}$ determines the offset of the hyperplane from the origin along the normal vector w.

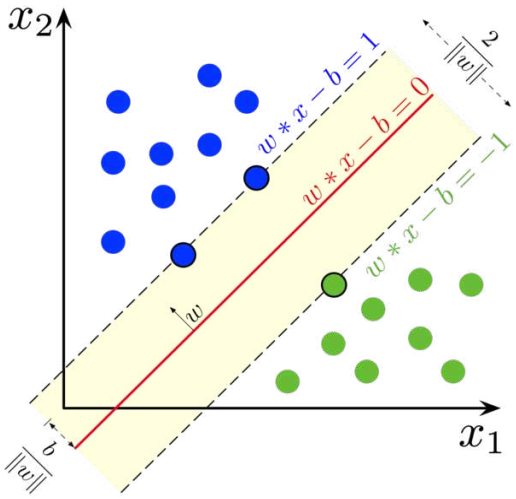
So, for each i, either x_i is in the hyperplane of 1 or -1. Basically, x_i satisfies:

$$w \cdot x_i - b \geq 1 \quad or \quad w \cdot x_i - b \leq -1$$



- *SVMs are helpful in text and hypertext categorization, as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings. Some methods for shallow semantic parsing are based on support vector machines.*
- *Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback.*
- *Classification of satellite data like SAR data using supervised SVM.*
- *Hand-written characters can be recognized using SVM.*

Q8. What are the support vectors in SVM?



In the diagram, we see that the sketched lines mark the distance from the classifier (the hyper plane) to the closest data points called the support vectors (darkened data points). The distance between the two thin lines is called the margin.

To extend SVM to cases in which the data are not linearly separable, we introduce the *hinge loss* function,

$$\max(0, 1 - y_i(w \cdot x_i - b))$$

This function is zero if x lies on the correct side of the margin. For data on the wrong side of the margin, the function's value is proportional to the distance from the margin.

Q9. What are the different kernels in SVM?

There are four types of kernels in SVM.

1. LinearKernel
2. Polynomial kernel
3. Radial basis kernel
4. Sigmoid kernel

Q10. What are the most known ensemble algorithms?

https://towardsdatascience.com/the-ultimate-guide-to-adaboost-random-forests-and-xgboost-7f9327061c4f

The most popular trees are: AdaBoost, Random Forest, and eXtreme Gradient Boosting (XGBoost).

AdaBoost is **best used** in a dataset with low noise, when computational complexity or timeliness of results is not a main concern and when there are not enough resources for broader hyperparameter tuning due to lack of time and knowledge of the user.

Random forests should not be used when dealing with time series data or any other data where look-ahead bias should be avoided, and the order and continuity of the samples need to be ensured. This algorithm can handle noise relatively well, but more knowledge from the user is required to adequately tune the algorithm compared to AdaBoost.

The main advantages of XGBoost is its lightning speed compared to other algorithms, such as AdaBoost, and its regularization parameter that successfully reduces variance. But even aside from the regularization parameter, this algorithm leverages a learning rate (shrinkage) and subsamples from the features like random forests, which increases its ability to generalize even further. However, XGBoost is more difficult to understand, visualize and to tune compared to AdaBoost and random forests. There is a multitude of hyperparameters that can be tuned to increase performance.

Q11. Explain Decision Tree algorithm in detail.

https://en.wikipedia.org/wiki/Decision_tree_learning
https://www.kdnuggets.com/2019/02/decision-trees-introduction.html/2
https://medium.com/@naeemsunesara/giniscore-entropy-and-information-gain-in-decision-trees-cbc08589852d

A decision tree is a supervised machine learning algorithm mainly used for Regression and Classification. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision tree can handle both categorical and numerical data. The term Classification and Regression Tree (CART) analysis is an umbrella term used to refer to both of the above procedures.

Some techniques, often called *ensemble* methods, construct more than one decision tree:

- **Boosted trees** Incrementally building an ensemble by training each new instance to emphasize the training instances previously mis-modeled. A typical example is AdaBoost. These can be used for regression-type and classification-type problems.
- **Bootstrap aggregated** (or bagged) decision trees, an early ensemble method, builds multiple decision trees by repeatedly resampling training data with replacement, and voting the trees for a consensus prediction.
    - A **random forest** classifier is a specific type of bootstrap aggregating.
- **Rotation forest** – in which every decision tree is trained by first applying principal component analysis (PCA) on a random subset of the input features.

A special case of a decision tree is a decision list, which is a one-sided decision tree, so that every internal node has exactly 1 leaf node and exactly 1 internal node as a child (except for the bottommost node, whose only child is a single leaf node). While less expressive, decision lists are arguably easier to understand than general decision trees due to their added sparsity, permit non-greedy learning methods and monotonic constraints to be imposed.

Notable decision tree algorithms include:

- ID3 (Iterative Dichotomiser 3)
- C4.5 (successor of ID3)
- CART (Classification and Regression Tree)
- Chi-square automatic interaction detection (CHAID). Performs multi-level splits when computing classification trees.
- MARS: extends decision trees to handle numerical data better.
- Conditional Inference Trees. Statistics-based approach that uses non-parametric tests as splitting criteria, corrected for multiple testing to avoid overfitting. This approach results in unbiased predictor selection and does not require pruning.

Q12. What are Entropy and Information gain in Decision tree algorithm?

https://www.saedsayad.com/decision_tree.htm
https://medium.com/@naeemsunesara/giniscore-entropy-and-information-gain-in-decision-trees-cbc08589852d

There are a lot of algorithms which are employed to build a decision tree, ID3 (Iterative Dichotomiser 3), C4.5, C5.0, CART (Classification and Regression Trees) to name a few but at their core all of them tell us what questions to ask and when.

The below table has color and diameter of a fruit and the label tells the name of the fruit. How do we build a decision tree to classify the fruits?

Color	Diameter	Label
Green	3	Apple
Yellow	3	Apple
Red	1	Grape
Red	1	Grape
Yellow	3	Lemon

Here is how we will build the tree. We will start with a node which will ask a true or false question to split the data into two. The two resulting nodes will each ask a true or false question again to split the data further and so on.
There are 2 main things to consider with the above approach:

- Which is the best question to ask at each node
- When do we stop splitting the data further?

Let's start building the tree with the first or the topmost node. There is a list of possible questions which can be asked. The first node can ask the following questions:

- *Is the color green?*
- *Is the color yellow?*
- *Is the color red?*
- *Is the diameter ≥ 3?*
- *Is the diameter ≥ 1?*

Of these possible set of questions, which one is the best to ask so that our data is split into two sets after the first node? Remember we are trying to split or classify our data into separate classes. Our question should be such that our data is partitioned into as unmixed or pure classes as possible. An impure set or class here refers to one which has many different types of objects *for example if we ask the question for the above data, "Is the color green?" our data will be split into two sets one of which will be pure the other will have a mixed set of labels.* If we assign a label to a mixed set, we have higher chances of being incorrect. But how do we measure this impurity?

Color	Diameter	Label
Green	3	Apple
Yellow	3	Apple
Red	1	Grape
Red	1	Grape
Yellow	3	Lemon

Gini Impurity and Information Gain - CART

CART (Classification and Regression Trees) → uses ***Gini Index (Classification)*** as metric.

The Gini Impurity (GI) metric measures the homogeneity of a set of items. The lowest possible value of GI is 0.0. The maximum value of GI depends on the particular problem being investigated but gets close to 1.0.

Suppose for example you have 12 items — apples, grapes, lemons. If there are 0 apples, 0 grapes, 12 lemons, then you have minimal impurity (this is good for decision trees) and GI = 0.0. But if you have 4 apples, 4 grapes, 4 lemons, you have maximum impurity and it turns out that GI = 0.667.
I'll show example calculations.
Maximum GI: Apples, Grapes, Lemons

```
count = 4 4 4
p = 4/12 4/12 4/12
= 1/3 1/3 1/3

GI = 1 - [ (1/3)^2 + (1/3)^2 + (1/3)^2 ]
= 1 - [ 1/9 + 1/9 + 1/9 ]
= 1 - 1/3
= 2/3
= 0.667
```

When the number of items is evenly distributed, as in the example above, you have maximum GI but the exact value depends on how many items there are. A bit less than maximum GI:

```
count = 3 3 6
p = 3/12 3/12 6/12
= 1/4 1/4 1/2

GI = 1 - [ (1/4)^2 + (1/4)^2 + (1/2)^2 ]
= 1 - [ 1/16 + 1/16 + 1/4 ]
= 1 - 6/16
= 10/16
= 0.625
```

In the example above, the items are not quite evenly distributed, and the GI is slightly less (which is better when used for decision trees). Minimum GI:

```
count = 0 12 0
p = 0/12 12/12 0/12
= 0 1 0

GI = 1 - [ 0^2 + 1^2 + 0^2 ]
= 1 - [ 0 + 1 + 0 ]
= 1 - 1
= 0.00
```

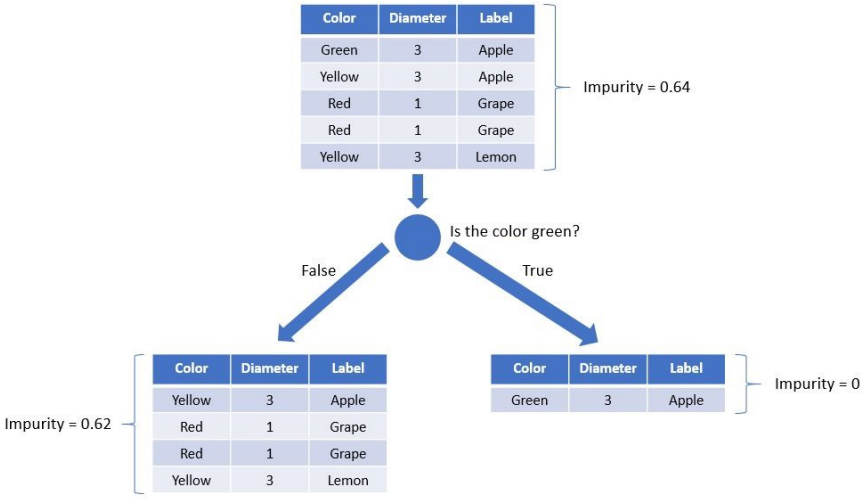
J classes $\{1, 2, \ldots J\}$, p_i fraction of elements of class i:

Gini impurity: $I_G(p) = 1 - \sum_{i=1}^{J} p_i^2$

The ***Gini index*** is not at all the same as a different metric called the ***Gini coefficient***. The Gini impurity metric can be used when creating a decision tree but there are alternatives, including ***Entropy Information gain***. The advantage of GI is its simplicity.

Information Gain

Information gain is another metric which tells us how much a question *unmixes* the labels at a node. ***"Mathematically it is just a difference between impurity values before splitting the data at a node and the weighted average of the impurity after the split"***. *For instance, if we go back to our data of apples, lemons and grapes and ask the question "Is the color Green?"*

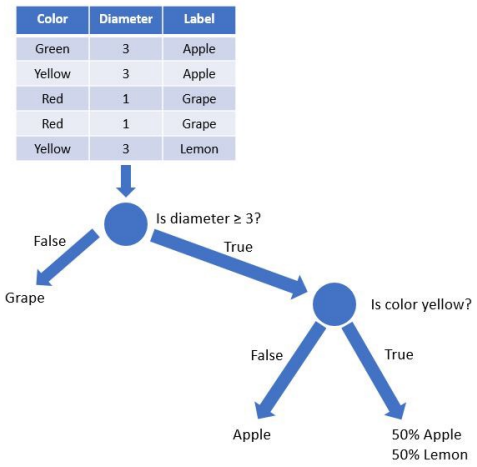


$$Information\ Gain = 0.64 - \left(\frac{4}{5} * 0.62 + \frac{1}{5} * 0\right) = 0.144$$

The information gain by asking this question is 0.144. Similarly, we can ask another question from the set of possible questions split the data and compute information gain. This is also called ***(Recursive Binary Splitting).***

Question	Information Gain
Is the color green?	0.14
Is diameter ≥ 3?	0.37
Is the color yellow?	0.17
Is the color red?	0.37
Is diameter ≥ 1?	0

The question where we have the highest information gain *"Is diameter ≥ 3?"* is the best question to ask. Note that the information gain is same for the question *"Is the color red?"* we just picked the first one at random.
Repeating the same method at the child node we can complete the tree. Note that no further questions can be asked which would increase the information gain.

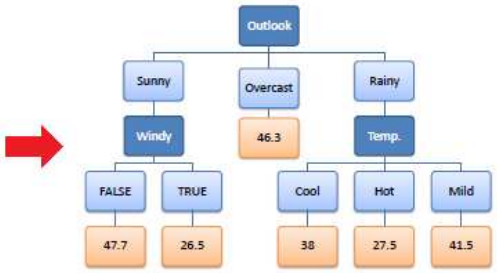


Also note that the rightmost leaf which says 50% Apple & 50% lemon means that this class cannot be divided further, and this branch can tell an apple or a lemon with 50% probability. For the grape and apple branches we stop asking further questions since the Gini Impurity is 0 for those.

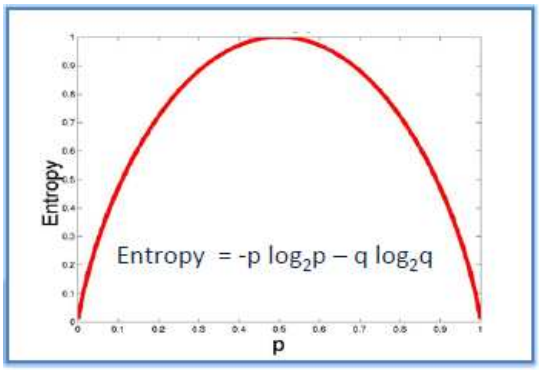
Entropy and Information Gain – ID3

ID3 (Iterative Dichotomiser 3) → uses ***Entropy function*** and ***Information gain*** as metrics.

Predictors				Target
Outlook	Temp	Humidity	Windy	Hours Played
Rainy	Hot	High	False	25
Rainy	Hot	High	True	30
Overcast	Hot	High	False	48
Sunny	Mild	High	False	46
Sunny	Cool	Normal	False	62
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	36
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	48
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30



If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.



Entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

To build a decision tree, we need to calculate two types of entropy using frequency tables as follows:

a) Entropy using the frequency table of one attribute:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

b) Entropy using the frequency table of two attributes:

$$E(T,X) = \sum_{c \in X} P(c)E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)*__E__(3,2) + **P**(Overcast)*__E__(4,0) + **P**(Rainy)*__E__(2,3)

= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971

= 0.693

Information Gain

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

Step 1: Calculate entropy of the target.

Entropy(PlayGolf) = Entropy (5,9)

= Entropy (0.36, 0.64)

= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)

= 0.94

Step 2: The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain or decrease in entropy.

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

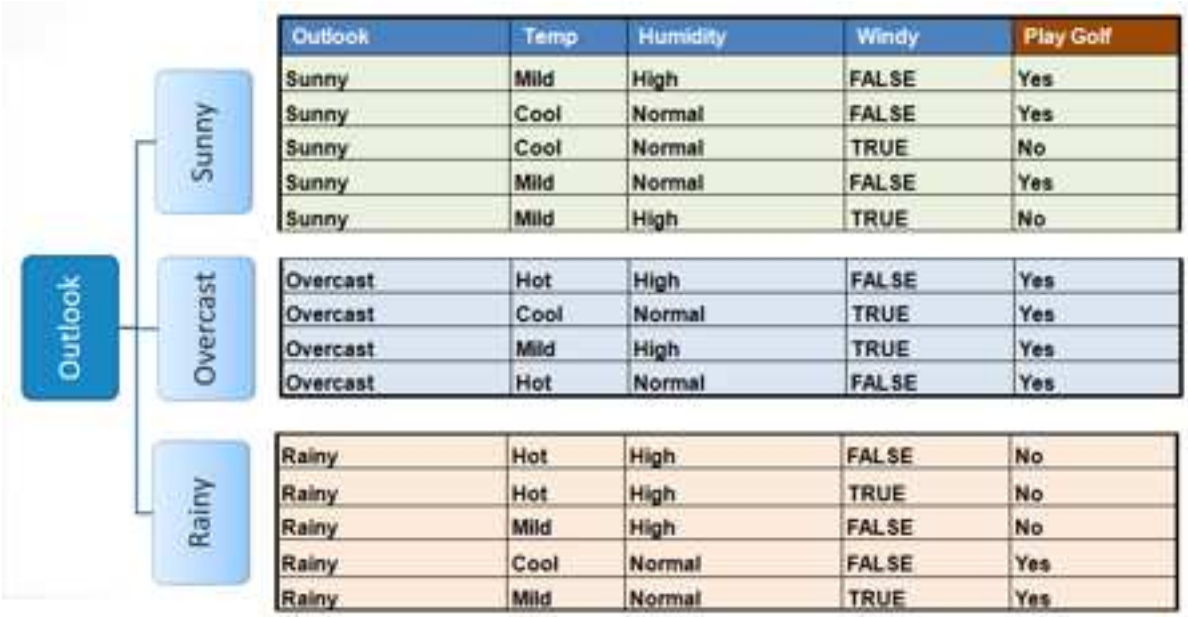
$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

= 0.940 – 0.693 = 0.247

Step 3: Choose attribute with the largest information gain as the decision node, divide the dataset by its branches and repeat the same process on every branch.

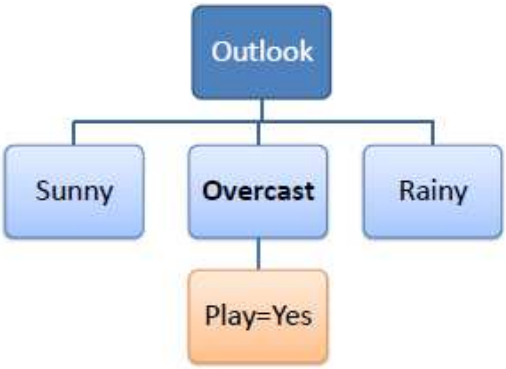
		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			



Outlook	Temp	Humidity	Windy	Play Golf
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Sunny	Mild	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Overcast	Cool	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes

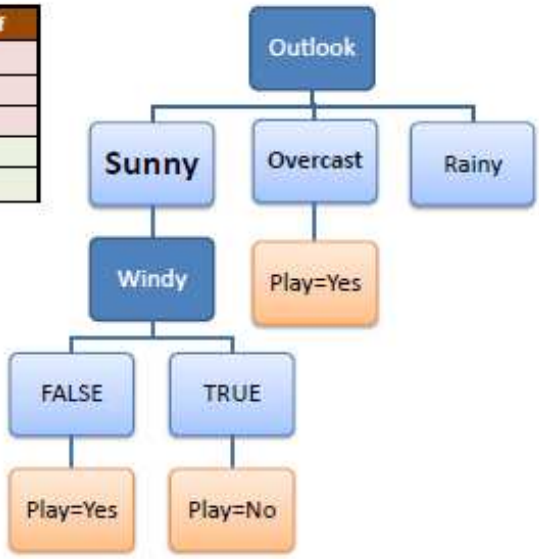
Step 4a: A branch with entropy of 0 is a leaf node.

Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes



Step 4b: A branch with entropy more than 0 needs further splitting.

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Step 5: The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.

Q13. What is pruning in Decision Tree?

Pruning is a technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances. So, when we remove sub-nodes of a decision node, this process is called pruning or opposite process of splitting.

Q14. What is logistic regression? State an example when you have used logistic regression recently.

Logistic Regression often referred to as the logit model is a technique to predict the binary outcome from a linear combination of predictor variables. Since we are interested in a probability outcome, a line does not fit the model. Logistic Regression is a classification algorithm that works by trying to learn a function

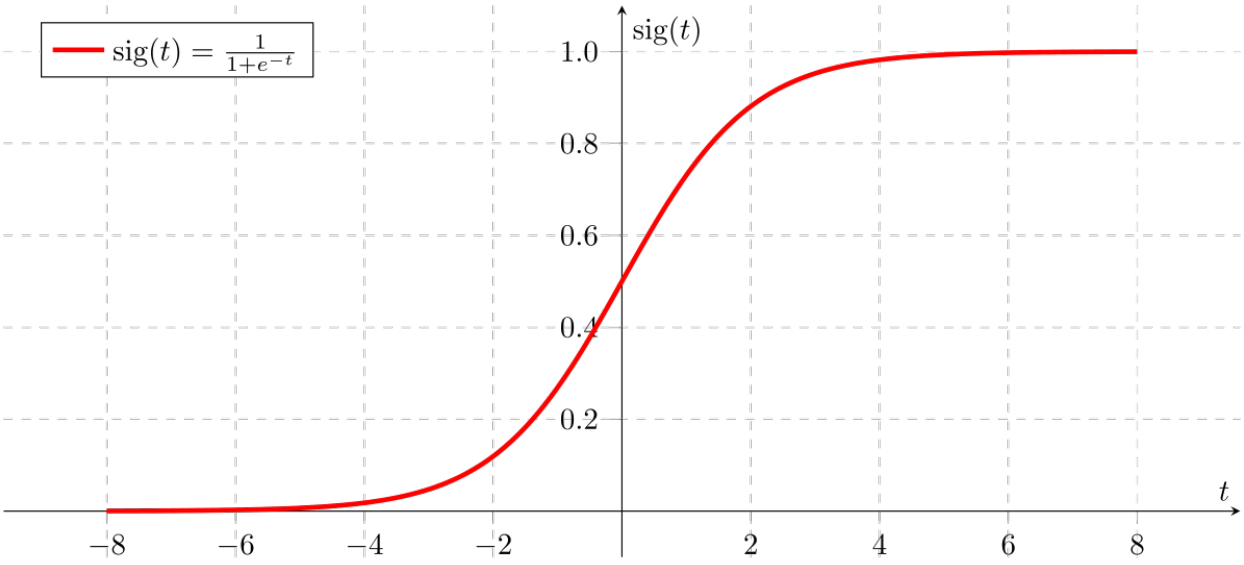
that approximates $P(Y|X)$. It makes the central assumption that $P(X|Y)$ can be approximated as a sigmoid function applied to a linear combination of input features.

- $P(Y=1|X) = p \Rightarrow$ assume $\log \frac{p}{1-p} = b_0 + \sum_{i=1}^{p} b_i X_i \Leftrightarrow \frac{p}{1-p} = e^{b_0 + b^T X}$

$$\Leftrightarrow p = \frac{e^{b_0 + b^T X}}{1 + e^{b_0 + b^T X}}$$

$$\Leftrightarrow p = \frac{1}{1 + e^{-(b_0 + b^T X)}} \equiv sig(z), \quad z = b_0 + b^T X$$

- $P(Y=0|X) = 1 - sig(z)$

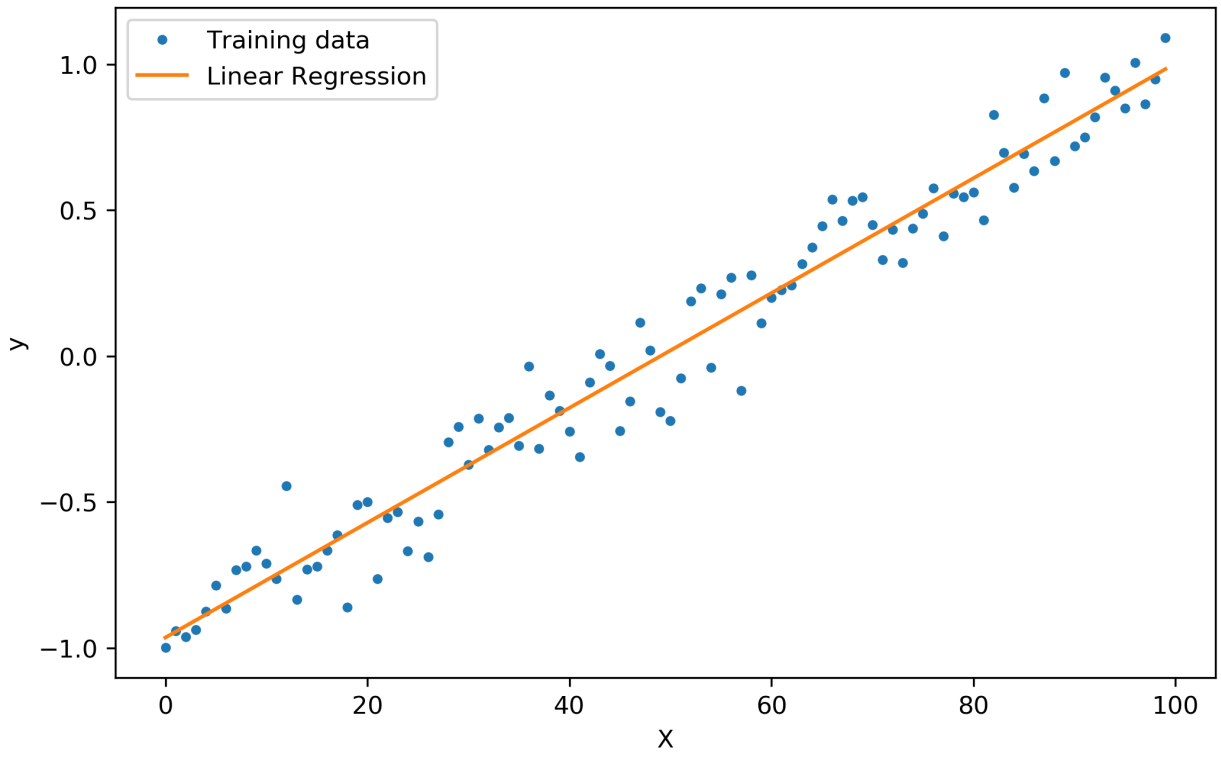


For example, if you want to predict whether a particular political leader will win the election or not. In this case, the outcome of prediction is binary i.e. 0 or 1 (Win/Lose). The predictor variables here would be the amount of money spent for election campaigning of a particular candidate, the amount of time spent in campaigning, etc.

Q15. What is Linear Regression?

Linear regression is a statistical technique where the score of a variable Y is predicted from the score of a second variable X. X is referred to as the predictor variable and Y as the criterion variable.

$$Y = b_0 + b_1 X_1 + \cdots + b_p X_p$$



Q16. What Are the Drawbacks of the Linear Model?

Some drawbacks of the linear model are:

- The assumption of linearity of the model
- It can't be used for count outcomes or binary outcomes.
- There are overfitting or underfitting problems that it can't solve.

Q17. What is the difference between Regression and classification ML techniques?

Both Regression and classification machine learning techniques come under Supervised machine learning algorithms. In Supervised machine learning algorithm, we have to train the model using labelled data set, while training we have to explicitly provide the correct labels and algorithm tries to learn the pattern from input to output. If our labels are discrete values then it will a classification problem, but if our labels are continuous values then it will be a regression problem.

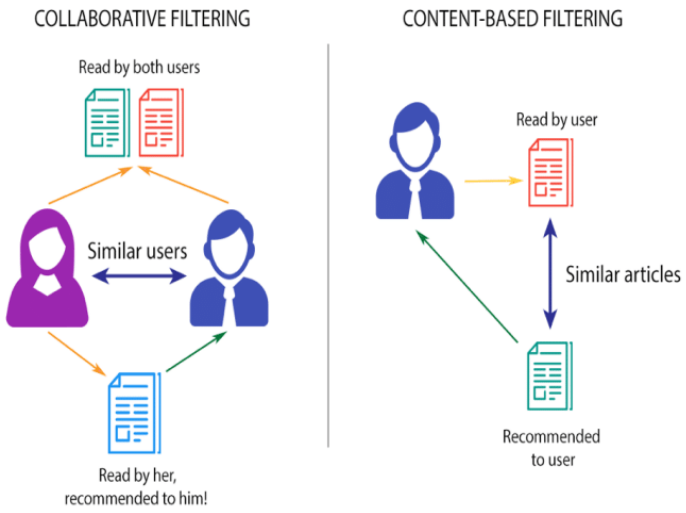
Q18. What are Recommender Systems?

https://en.wikipedia.org/wiki/Recommender_system

Recommender Systems are a subclass of information filtering systems that are meant to predict the preferences or ratings that a user would give to a product. Recommender systems are widely used in movies, news, research articles, products, social tags, music, etc.
Examples include movie recommenders in IMDB, Netflix & BookMyShow, product recommenders in e-commerce sites like Amazon, eBay & Flipkart, YouTube video recommendations and game recommendations in Xbox.

Q19. What is Collaborative filtering? And a content based?

The process of filtering used by most of the recommender systems to find patterns or information by collaborating viewpoints, various data sources and multiple agents. Collaborative filtering is a technique that can filter out items that a user might like on the basis of reactions by similar users. It works by searching a large group of people and finding a smaller set of users with tastes similar to a particular user. It looks at the items they like (usually based on rating) and combines them to create a ranked list of suggestions. Similar users are those with similar rating and on the based on that they get recommendations. In content based, we look only at the item level, recommending on similar items sold.



An example of collaborative filtering can be to predict the rating of a particular user based on his/her ratings for other movies and others' ratings for all movies. This concept is widely used in recommending movies in IMDB, Netflix & BookMyShow, product recommenders in e-commerce sites like Amazon, eBay & Flipkart, YouTube video recommendations and game recommendations in Xbox.

Q20. How can outlier values be treated?

Outlier values can be identified by using univariate or any other graphical analysis method. If the number of outlier values is few then they can be assessed individually but for a large number of outliers, the values can be substituted with either the 99th or the 1st percentile values.
All extreme values are not outlier values. The most common ways to treat outlier values:

1. Change it with a mean or median
2. Standardize the feature, changing the distribution but smoothing the outliers
3. Log transform the feature (with many outliers)
4. Drop the value

5. First/third quartile value if more than 2σ

Q21. What are the various steps involved in an analytics project?

The following are the various steps involved in an analytics project:

1. Understand the Business problem
2. Explore the data and become familiar with it
3. Prepare the data for modeling by detecting outliers, treating missing values, transforming variables, etc.
4. After data preparation, start running the model, analyze the result and tweak the approach. This is an iterative step until the best possible outcome is achieved.
5. Validate the model using a new data set.
6. Start implementing the model and track the result to analyze the performance of the model over the period of time.

Q22. During analysis, how do you treat missing values?

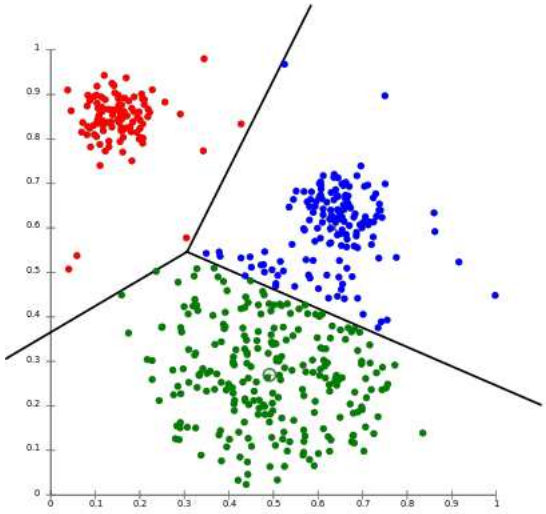
The extent of the missing values is identified after identifying the variables with missing values. If any patterns are identified the analyst has to concentrate on them as it could lead to interesting and meaningful business insights.
If there are no patterns identified, then the missing values can be substituted with mean or median values (imputation) or they can simply be ignored. Assigning a default value which can be mean, minimum or maximum value. Getting into the data is important.
If it is a categorical variable, the default value is assigned. The missing value is assigned a default value. If you have a distribution of data coming, for normal distribution give the mean value.
If 80% of the values for a variable are missing, then you can answer that you would be dropping the variable instead of treating the missing values.

Q23. How will you define the number of clusters in a clustering algorithm?

https://stackabuse.com/hierarchical-clustering-with-python-and-scikit-learn/

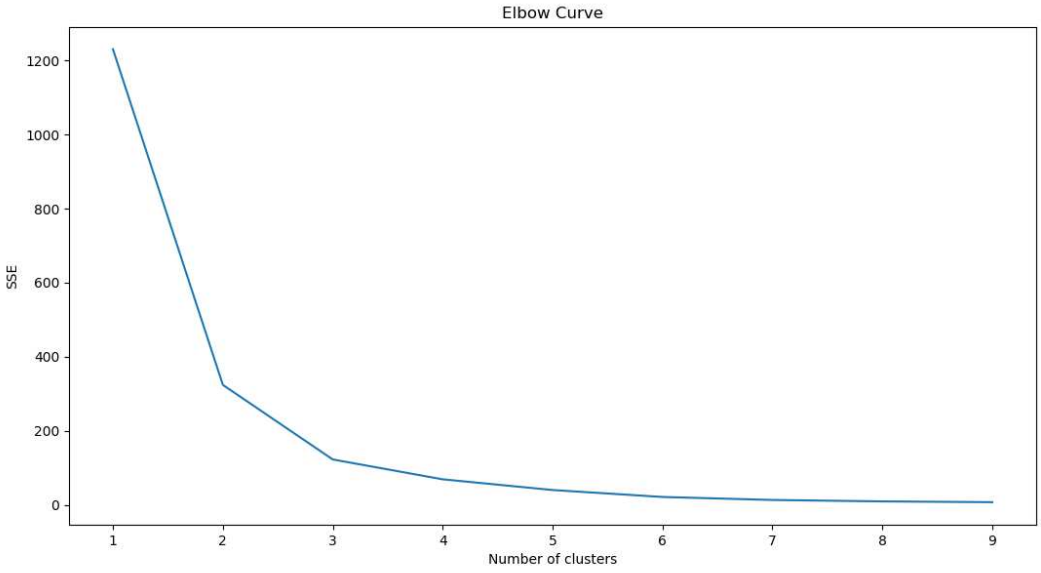
Though the Clustering Algorithm is not specified, this question is mostly in reference to K-Means clustering where “K” defines the number of clusters. The objective of clustering is to group similar entities in a way that the entities within a group are similar to each other, but the groups are different from each other.

For example, the following image shows three different groups.

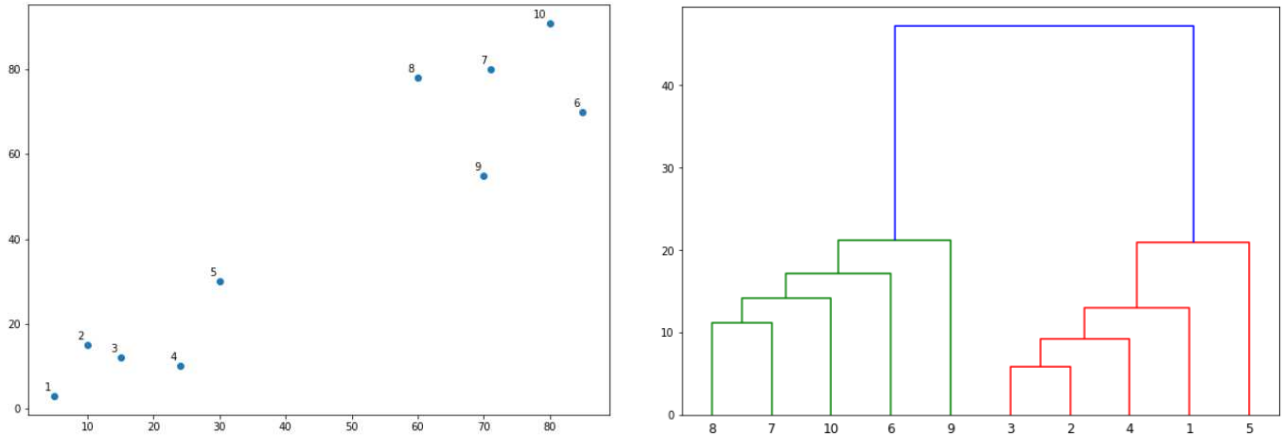


Within Sum of squares is generally used to explain the homogeneity within a cluster. If you plot WSS (as the sum of the squared distance between each member of the cluster and its centroid) for a range of number of clusters, you will get the plot shown below.

- The Graph is generally known as Elbow Curve.
- Red circled a point in above graph i.e. Number of Cluster = 3 is the point after which you don't see any decrement in WSS.
- This point is known as the bending point and taken as K in K – Means.

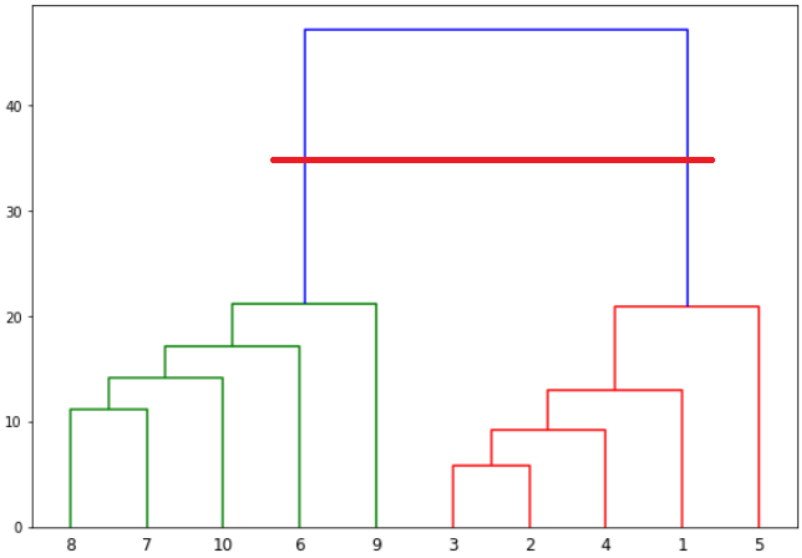


This is the widely used approach but few data scientists also use Hierarchical clustering first to create dendrograms and identify the distinct groups from there.

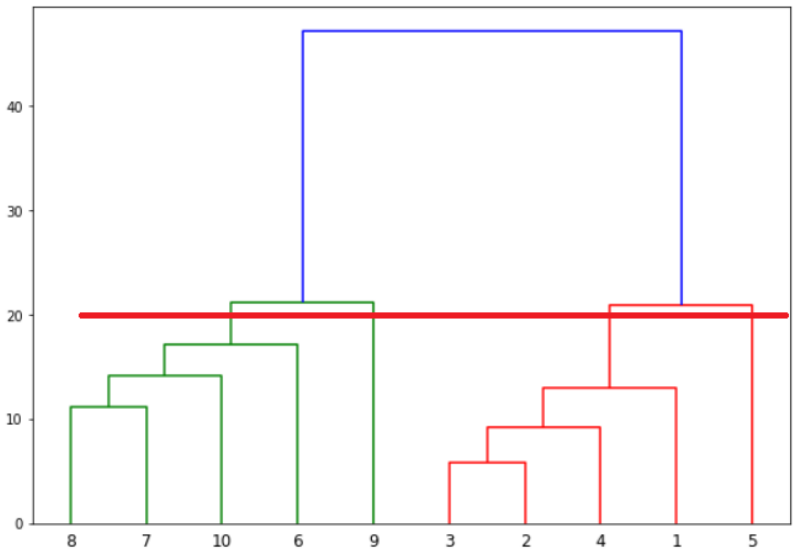


The algorithm starts by finding the two points that are closest to each other on the basis of Euclidean distance. If we look back at Graph1, we can see that points 2 and 3 are closest to each other while points 7 and 8 are closes to each other. Therefore a cluster will be formed between these two points first. In Graph2, you can see that the dendograms have been created joining points 2 with 3, and 8 with 7. The vertical height of the dendogram shows the Euclidean distances between points. From Graph2, it can be seen that Euclidean distance between points 8 and 7 is greater than the distance between point 2 and 3. The next step is to join the cluster formed by joining two points to the next nearest cluster or point which in turn results in another cluster. If you look at Graph1, point 4 is closest to cluster of point 2 and 3, therefore in Graph2 dendrogram is generated by joining point 4 with dendrogram of point 2 and 3. This process continues until all the points are joined together to form one big cluster.

Once one big cluster is formed, the longest vertical distance without any horizontal line passing through it is selected and a horizontal line is drawn through it. The number of vertical lines this newly created horizontal line passes is equal to number of clusters. Take a look at the following plot:



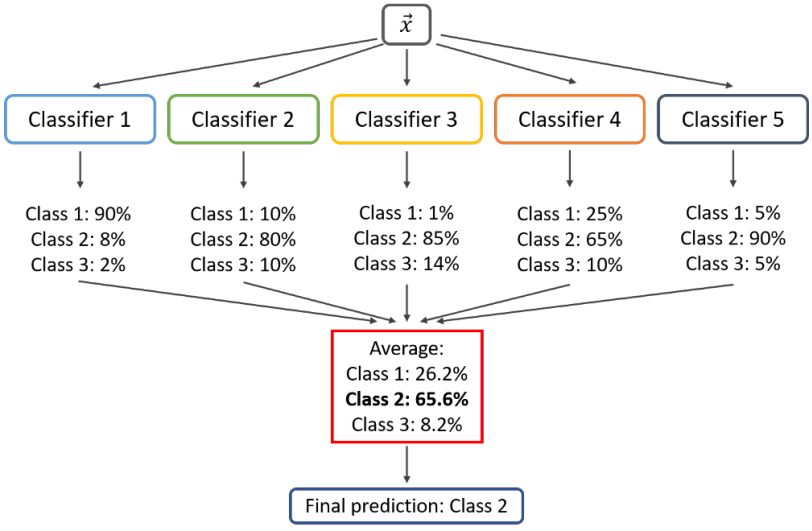
We can see that the largest vertical distance without any horizontal line passing through it is represented by blue line. So we draw a new horizontal red line that passes through the blue line. Since it crosses the blue line at two points, therefore the number of clusters will be 2. Basically the horizontal line is a threshold, which defines the minimum distance required to be a separate cluster. If we draw a line further down, the threshold required to be a new cluster will be decreased and more clusters will be formed as see in the image below:



In the above plot, the horizontal line passes through four vertical lines resulting in four clusters: cluster of points 6,7,8 and 10, cluster of points 3,2,4 and points 9 and 5 will be treated as single point clusters.

Q24. What is Ensemble Learning?

In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. Ensembles are a divide-and-conquer approach used to improve performance. The main principle behind ensemble methods is that a group of "weak learners" can come together to form a "strong



learner". Each classifier, individually, is a "weak learner," while all the classifiers taken together are a "strong learner".

Q25. Describe in brief any type of Ensemble Learning.

https://medium.com/@ruhi3929/bagging-and-boosting-method-c036236376eb

Ensemble learning has many types but two more popular ensemble learning techniques are mentioned below.

Bagging

Bagging tries to implement similar learners on small sample populations and then takes a mean of all the predictions. In generalized bagging, you can use different learners on different population. As you expect this helps us to reduce the variance error.

Pros

- Bagging method helps when we face variance or overfitting in the model. It provides an environment to deal with variance by using N learners of same size on same algorithm.
- During the sampling of train data, there are many observations which overlaps. So, the combination of these learners helps in overcoming the high variance.
- Bagging uses Bootstrap sampling method (Bootstrapping is any test or metric that uses random sampling with replacement and falls under the broader class of resampling methods.)

Cons

- Bagging is not helpful in case of bias or underfitting in the data.
- Bagging ignores the value with the highest and the lowest result which may have a wide difference and provides an average result.

Boosting

Boosting is an iterative technique which adjusts the weight of an observation based on the last classification. If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa. Boosting in general decreases the bias error and builds strong predictive models. However, they may over fit on the training data.

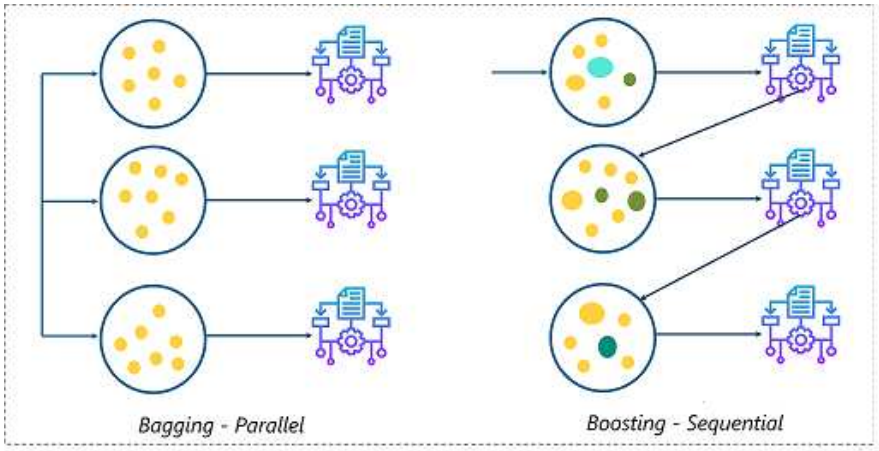
Pros

- Boosting technique takes care of the weightage of the higher accuracy sample and lower accuracy sample and then gives the combined results.
- Net error is evaluated in each learning steps. It works good with interactions.
- Boosting technique helps when we are dealing with bias or underfitting in the data set.
- Multiple boosting techniques are available. *For example: AdaBoost, LPBoost, XGBoost, GradientBoost, BrownBoost*

Cons

- Boosting technique often ignores overfitting or variance issues in the data set.

- It increases the complexity of the classification.
- Time and computation can be a bit expensive.



There are multiple areas where Bagging and Boosting technique is used to boost the accuracy.

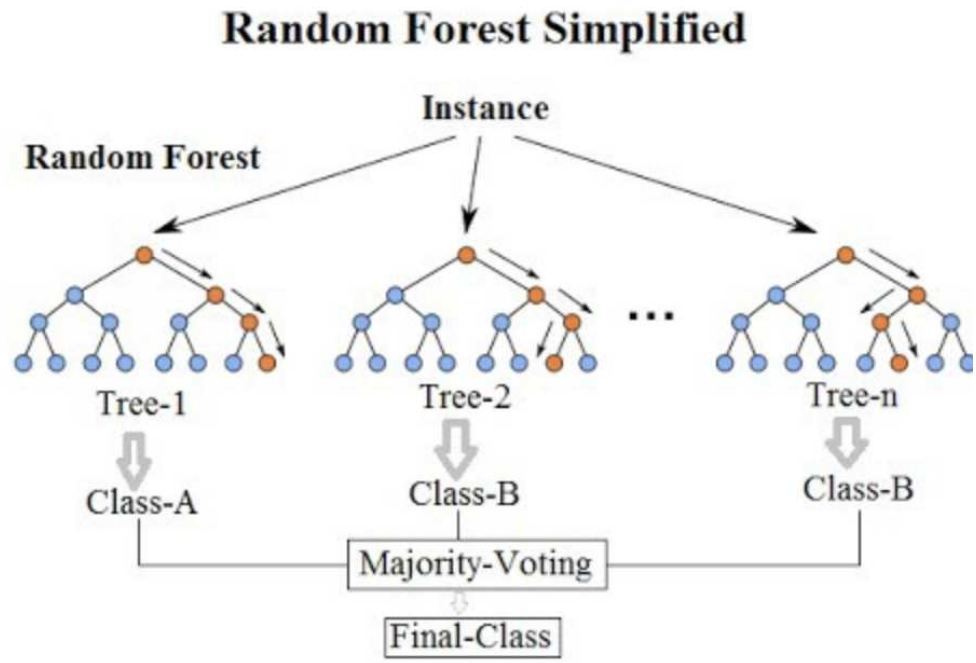
- Banking: Loan defaulter prediction, fraud transaction
- Credit risks
- Kaggle competitions
- Fraud detection
- Recommender system for Netflix
- Malware
- Wildlife conservations and so on.

Q26. What is a Random Forest? How does it work?

Random forest is a versatile machine learning method capable of performing:

- regression
- classification
- dimensionality reduction
- treat missing values
- outlier values

It is a type of ensemble learning method, where a group of weak models combine to form a powerful model. The random forest starts with a standard machine learning technique called a "decision tree" which, in ensemble terms, corresponds to our weak learner. In a decision tree, an input is entered at the top and as it traverses down the tree the data gets bucketed into smaller and smaller sets.



In Random Forest, we grow multiple trees as opposed to a single tree. To classify a new object based on attributes, each tree gives a classification. The forest chooses the classification having the most votes (Over all the trees in the forest) and in case of regression, it takes the average of outputs by different trees.

Q27. How Do You Work Towards a Random Forest?

https://blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics

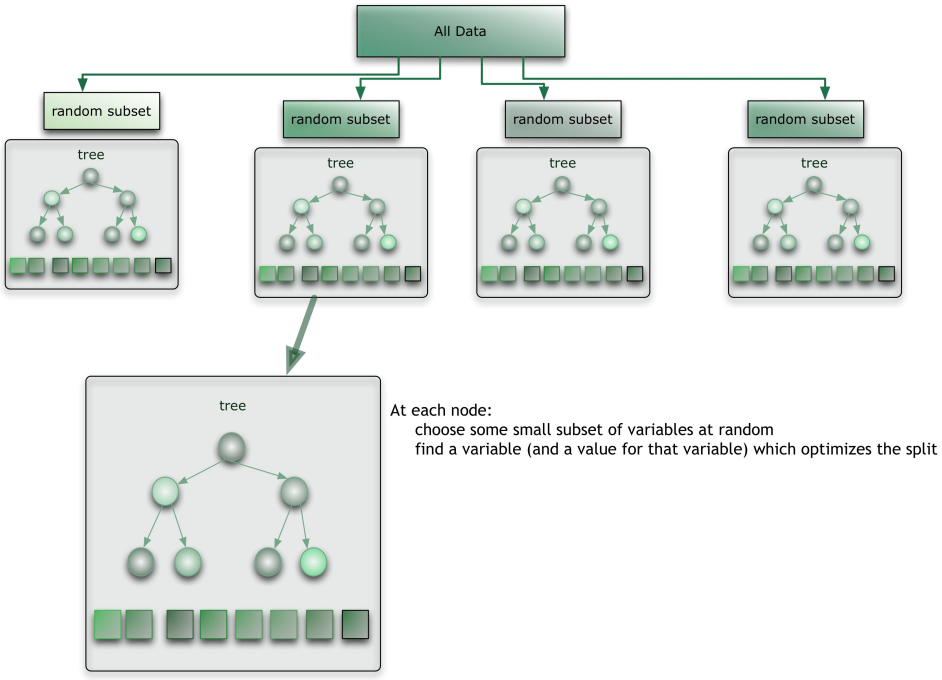
The underlying principle of this technique is that several weak learners combined to provide a keen learner. Here is how such a system is trained for some number of trees *T*:

1. Sample *N* cases at random with replacement to create a subset of the data. The subset should be about 66% of the total set.
2. At each node:
    a. For some number *m* (see below), *m* predictor variables are selected at random from all the predictor variables.
    b. The predictor variable that provides the best split, according to some objective function, is used to do a binary split on that node.
    c. At the next node, choose another *m* variables at random from all predictor variables and do the same.

Depending upon the value of *m*, there are three slightly different systems:

- Random splitter selection: $m = 1$
- Breiman's bagger: m = total number of predictor variables (p)
- Random forest: $m \ll$ number of predictor variables.
    - Brieman suggests three possible values for m: $\frac{1}{2}\sqrt{p},\ \sqrt{p},\ 2\sqrt{p}$

-



When a new input is entered into the system, it is run down all of the trees. The result may either be an average or weighted average of all of the terminal nodes that are reached, or, in the case of categorical variables, a voting majority.

Note that:

- With a large number of predictors ($p \gg 0$), the eligible predictor set (m) will be quite different from node to node.
- The greater the inter-tree correlation, the greater the random forest error rate, so one pressure on the model is to have the trees as uncorrelated as possible.
- As m goes down, both inter-tree correlation and the strength of individual trees go down. So some optimal value of m must be discovered.
- Strengths: Random forest runtimes are quite fast, and they are able to deal with unbalanced and missing data.
- Weaknesses: Random Forest used for regression cannot predict beyond the range in the training data, and that they may over-fit data sets that are particularly noisy. Of course, the best test of any algorithm is how well it works upon your own data set.

Q28. What cross-validation technique would you use on a time series data set?

Instead of using k-fold cross-validation, you should be aware of the fact that a time series is not randomly distributed data — It is inherently ordered by chronological order.
In case of time series data, you should use techniques like forward=chaining — Where you will be model on past data then look at forward-facing data.

fold 1: training[1], test[2]
fold 2: training[1 2], test[3]
fold 3: training[1 2 3], test[4]
fold 4: training[1 2 3 4], test[5]

Q29. What is a Box-Cox Transformation?

The dependent variable for a regression analysis might not satisfy one or more assumptions of an ordinary least squares regression. The residuals could either curve as the prediction increases or follow the skewed distribution. In such scenarios, it is necessary to transform the response variable so that the data meets the required assumptions. A Box-Cox transformation is a statistical technique to transform non-normal dependent variables into a normal shape. If the given data is not normal then most of the statistical techniques assume normality. Applying a Box-Cox transformation means that you can run a broader number of tests.
A Box-Cox transformation is a way to transform non-normal dependent variables into a normal shape. Normality is an important assumption for many statistical techniques, if your data isn't normal, applying a Box-Cox means that you are able to run a broader number of tests. The Box-Cox transformation is named after statisticians George Box and Sir David Roxbee Cox who collaborated on a 1964 paper and developed the technique.

Q30. How Regularly Must an Algorithm be Updated?

You will want to update an algorithm when:

- You want the model to evolve as data streams through infrastructure
- The underlying data source is changing
- There is a case of non-stationarity (mean, variance change over the time)
- The algorithm underperforms/results lack accuracy

Q31. If you are having 4GB RAM in your machine and you want to train your model on 10GB data set. How would you go about this problem? Have you ever faced this kind of problem in your machine learning/data science experience so far?

First of all, you have to ask which ML model you want to train.
For Neural networks: Batch size with Numpy array will work. Steps:

1. Load the whole data in the Numpy array. Numpy array has a property to create a mapping of the complete data set, it doesn't load complete data set in memory.
2. You can pass an index to Numpy array to get required data.
3. Use this data to pass to the Neural network.
4. Have a small batch size.

For SVM: Partial fit will work. Steps:

1. Divide one big data set in small size data sets.

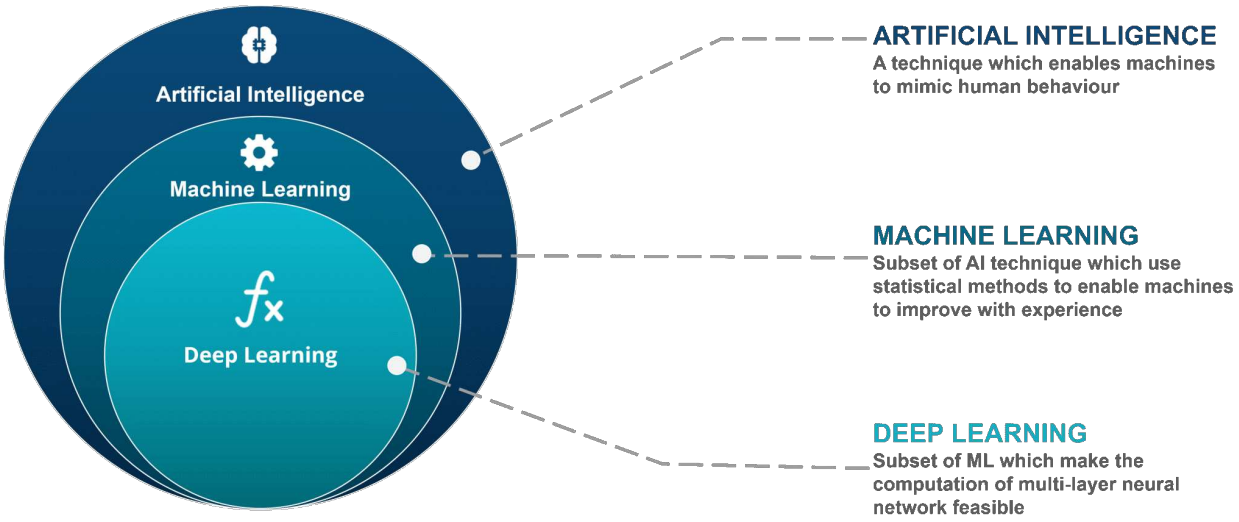
2. Use a partial fit method of SVM, it requires a subset of the complete data set.
3. Repeat step 2 for other subsets.

However, you could actually face such an issue in reality. So, you could check out the best laptop for Machine Learning to prevent that. Having said that, let's move on to some questions on deep learning.

Deep Learning

Q1. What do you mean by Deep Learning?

Deep Learning is nothing but a paradigm of machine learning which has shown incredible promise in recent years. This is because of the fact that Deep Learning shows a great analogy with the functioning of the neurons in the human brain.



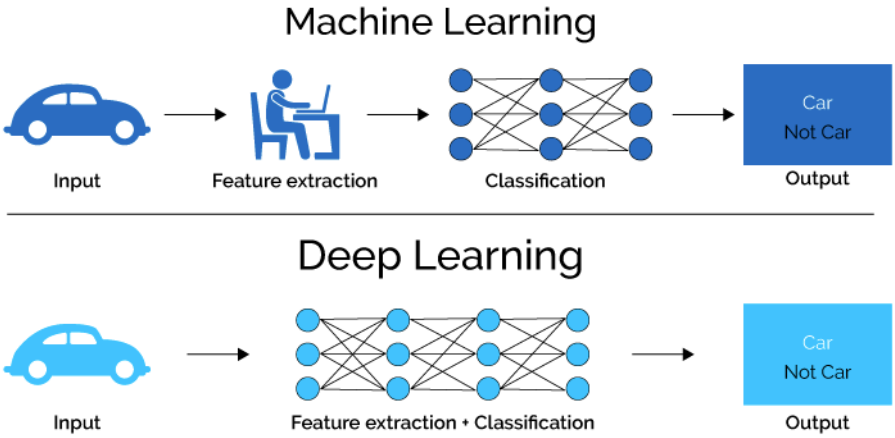
Q2. What is the difference between machine learning and deep learning?

https://parsers.me/deep-learning-machine-learning-whats-the-difference/

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed. Machine learning can be categorized in the following four categories.

1. Supervised machine learning,
2. Semi-supervised machine learning,
3. Unsupervised machine learning,
4. Reinforcement learning.

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.



- The main difference between deep learning and machine learning is due to the way data is presented in the system. Machine learning algorithms almost always require structured data, while deep learning networks rely on layers of ANN (artificial neural networks).
- Machine learning algorithms are designed to "learn" to act by understanding labeled data and then use it to produce new results with more datasets. However, when the result is incorrect, there is a need to "teach them". Because machine learning algorithms require bulleted data, they are not suitable for solving complex queries that involve a huge amount of data.
- Deep learning networks do not require human intervention, as multilevel layers in neural networks place data in a hierarchy of different concepts, which ultimately learn from their own mistakes. However, even they can be wrong if the data quality is not good enough.
- Data decides everything. It is the quality of the data that ultimately determines the quality of the result.
- Both of these subsets of AI are somehow connected to data, which makes it possible to represent a certain form of "intelligence." However, you should be aware that deep learning requires much more data than a traditional machine learning algorithm. The reason for this is that deep learning networks can identify different elements in neural network layers only when more than a million data points interact. Machine learning algorithms, on the other hand, are capable of learning by pre-programmed criteria.

Q3. What, in your opinion, is the reason for the popularity of Deep Learning in recent times?

Now although Deep Learning has been around for many years, the major breakthroughs from these techniques came just in recent years. This is because of two main reasons:

- The increase in the amount of data generated through various sources
- The growth in hardware resources required to run these models

GPUs are multiple times faster and they help us build bigger and deeper deep learning models in comparatively less time than we required previously.

Q4. What is reinforcement learning?

Reinforcement Learning allows to take actions to max cumulative reward. It learns by trial and error through reward/penalty system. Environment rewards agent so by time agent makes better decisions.
Ex: robot=agent, maze=environment. Used for complex tasks (self-driving cars, game AI).

RL is a series of time steps in a Markov Decision Process:

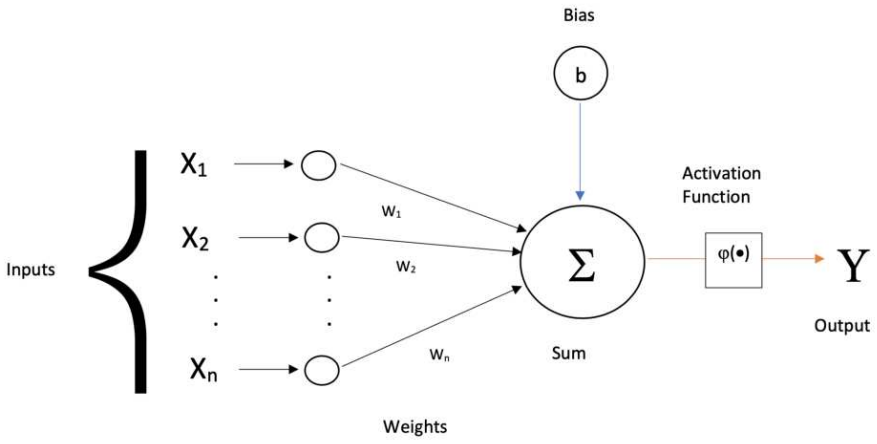
1. Environment: space in which RL operates
2. State: data related to past action RL took
3. Action: action taken
4. Reward: number taken by agent after last action
5. Observation: data related to environment: can be visible or partially shadowed

Q5. What are Artificial Neural Networks?

Artificial Neural networks are a specific set of algorithms that have revolutionized machine learning. They are inspired by biological neural networks. Neural Networks can adapt to changing the input, so the network generates the best possible result without needing to redesign the output criteria.

Q6. Describe the structure of Artificial Neural Networks?

Artificial Neural Networks works on the same principle as a biological Neural Network. It consists of inputs which get processed with weighted sums and Bias, with the help of Activation Functions.



Q7. How Are Weights Initialized in a Network?

There are two methods here: we can either initialize the weights to zero or assign them randomly.

Initializing all weights to 0: This makes your model similar to a linear model. All the neurons and every layer perform the same operation, giving the same output and making the deep net useless.

Initializing all weights randomly: Here, the weights are assigned randomly by initializing them very close to 0. It gives better accuracy to the model since every neuron performs different computations. This is the most commonly used method.

Q8. What Is the Cost Function?

Also referred to as "loss" or "error," cost function is a measure to evaluate how good your model's performance is. It's used to compute the error of the output layer during backpropagation. We push that error backwards through the neural network and use that during the different training functions.
The most known one is the mean sum of squared errors.

$$\text{MSE} = \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{test set}} \underbrace{(y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i)^2}_{\text{actual value}}$$

$$\hat{y}_i = \phi(\sum(w_i x_i) + b)$$

Q9. What Are Hyperparameters?

With neural networks, you're usually working with hyperparameters once the data is formatted correctly. A hyperparameter is a parameter whose value is set before the learning process begins. It determines how a network is trained and the structure of the network (such as the number of hidden units, the learning rate, epochs, batches, etc.).

Q10. What Will Happen If the Learning Rate Is Set inaccurately (Too Low or Too High)?

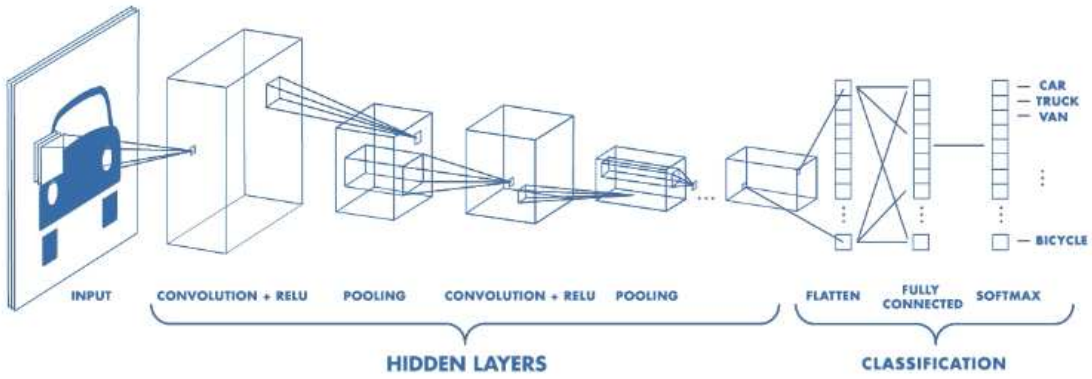
When your learning rate is too low, training of the model will progress very slowly as we are making minimal updates to the weights. It will take many updates before reaching the minimum point.
If the learning rate is set too high, this causes undesirable divergent behavior to the loss function due to drastic updates in weights. It may fail to converge (model can give a good output) or even diverge (data is too chaotic for the network to train).

Q11. What Is The Difference Between Epoch, Batch, and Iteration in Deep Learning?

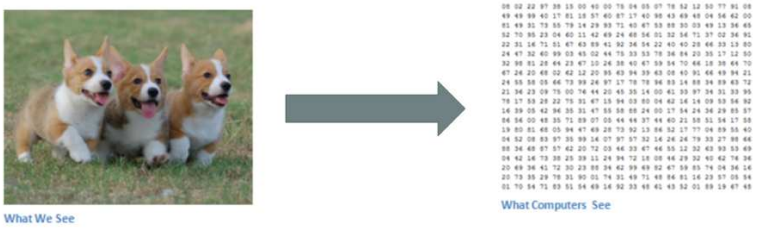
- Epoch – Represents one iteration over the entire dataset (everything put into the training model).
- Batch – Refers to when we cannot pass the entire dataset into the neural network at once, so we divide the dataset into several batches.
- Iteration – if we have 10,000 images as data and a batch size of 200. then an epoch should run 50 iterations (10,000 divided by 50).

Q12. What Are the Different Layers on CNN?

https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add



The Convolutional neural networks are regularized versions of multilayer perceptron (MLP). They were developed based on the working of the neurons of the animal visual cortex.



Let's say we have a color image in JPG form and its size is 480 x 480. The representative array will be 480 x 480 x 3. Each of these numbers is given a value from 0 to 255 which describes the pixel intensity at that point. RGB intensity values of the image are visualized by the computer for processing.

The objective of using the CNN:

The idea is that you give the computer this array of numbers and it will output numbers that describe the probability of the image being a certain class (.80 for a cat, .15 for a dog, .05 for a bird, etc.). It works similar to how our brain works. When we look at a picture of a dog, we can classify it as such if the picture has identifiable features such as paws or 4 legs. In a similar way, the computer is able to perform image classification by looking for low-level features such as edges and curves and then building up to more abstract concepts through a series of convolutional layers. The computer uses low-level features obtained at the initial levels to generate high-level features such as paws or eyes to identify the object.

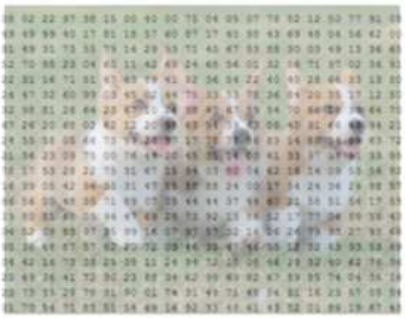
There are four layers in CNN:

1. **Convolutional Layer –** the layer that performs a convolutional operation, creating several smaller picture windows to go over the data.
2. **Activation Layer (ReLU Layer)** – it brings non-linearity to the network and converts all the negative pixels to zero. The output is a rectified feature map. It follows each convolutional layer.
3. **Pooling Layer –** pooling is a down-sampling operation that reduces the dimensionality of the feature map. Stride = how much you slide, and you get the max of the $n \times n$ matrix
4. **Fully Connected Layer –** this layer recognizes and classifies the objects in the image.

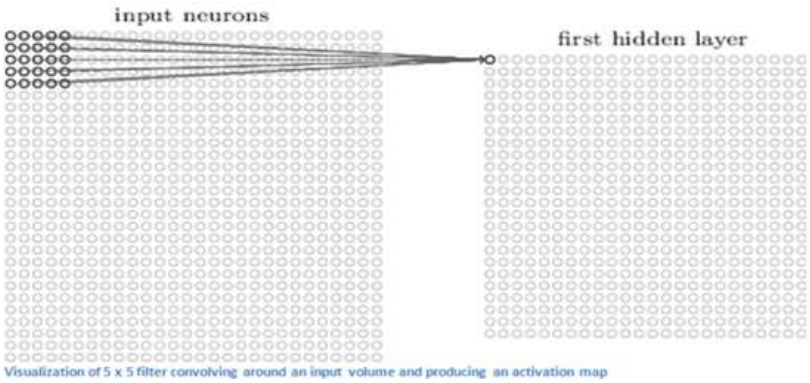
Convolution Operation

First Layer:

1. **Input to a convolutional layer**
   The image is resized to an optimal size and is fed as input to the convolutional layer.
   Let us consider the input as 32x32x3 array of pixel values.



2. **There exists a filter or neuron or kernel which lays over some of the pixels of the input image depending on the dimensions of the Kernel size.**
   Let the dimensions of the kernel of the filter be 5x5x3.



Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

3. **The Kernel actually slides over the input image; thus, it is multiplying the values in the filter with the original pixel values of the image (aka computing element-wise multiplications).**
   The multiplications are summed up generating a single number for that particular receptive field and hence for sliding the kernel a total of 784 numbers are mapped to 28x28 array known as the feature map.

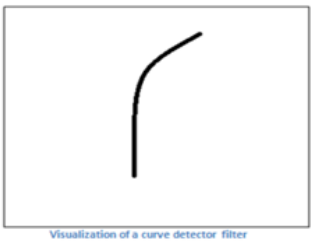
   ***Now if we consider two kernels of the same dimension then the obtained first layer feature map will be (28x28x2).*

High-level Perspective

- Let us take a kernel of size (7x7x3) for understanding. Each of the kernels is considered to be a feature identifier, hence say that our filter will be a curve detector.

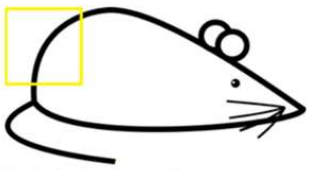
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter



Visualization of a curve detector filter

- The original image and the visualization of the kernel on the image.



Visualization of the filter on the image

Original image

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

$*$

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

The sum of the multiplication value that is generated is $= 4 * (50 * 30) + (20 * 30) = 6600$ *(large number).*

- Now when the kernel moves to the other part of the image.

Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

$*$

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

The sum of the multiplication value that is generated is = 0 (small number).

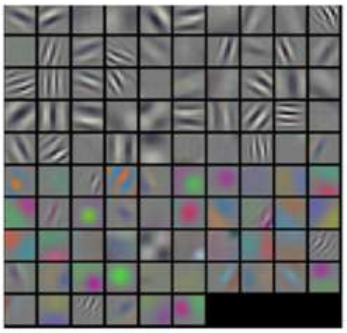
The use of the small and the large value

1. The value is much lower! This is because there wasn't anything in the image section that responded to the curve detector filter. Remember, the output of this convolution layer is an activation map. So, in the simple case of a one filter convolution (and if that filter is a curve detector), the activation map will show the areas in which there at most likely to be curved in the picture.

2. *In the previous example, the top-left value of our 26 x 26 x 1 activation map (26 because of the 7x7 filter instead of 5x5) will be 6600. This high value means that it is likely that there is some sort of curve in the input volume that caused the filter to activate. The top right value in our activation map will be 0 because there wasn't anything in the input volume that caused the filter to activate. This is just for one filter.*

3. This is just a filter that is going to detect lines that curve outward and to the right. We can have other filters for lines that curve to the left or for straight edges. The more filters, the greater the depth of the activation map, and the more information we have about the input volume.

   In the picture, we can see some examples of actual visualizations of the filters of the first conv. layer of a trained network. Nonetheless, the main argument remains the same. The filters on the first layer convolve around the input image and "activate" (or compute high values) when the specific feature it is looking for is in the input volume.



Visualizations of filters

Sequential convolutional layers after the first one

1. When we go through another conv. layer, the output of the first conv. layer becomes the input of the 2nd conv. layer.

2. However, when we're talking about the 2nd conv. layer, the input is the activation map(s) that result from the first layer. So, each layer of the input is basically describing the locations in the original image for where certain low-level features appear.

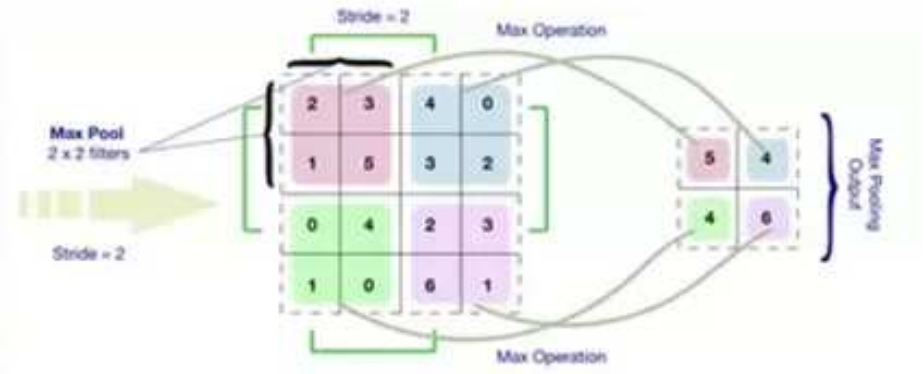
3. Now when you apply a set of filters on top of that (pass it through the 2nd conv. layer), the output will be activations that represent higher-level features. Types of these features could be semicircles (a combination of a curve and straight edge) or squares (a combination of several straight edges). As you go through the network and go through more convolutional layers, you get activation maps that represent more and more complex features.

4. By the end of the network, you may have some filters that activate when there is handwriting in the image, filters that activate when they see pink objects, etc.

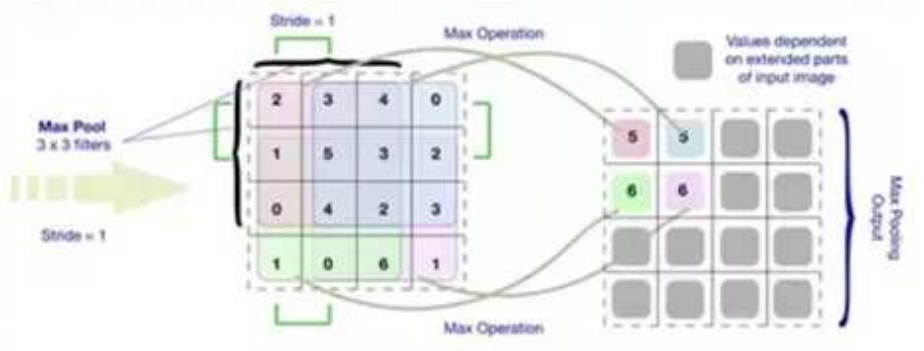
Pooling Operation

It consists in getting the largest number out of a matrix to get the most important number and reduce the dimention.

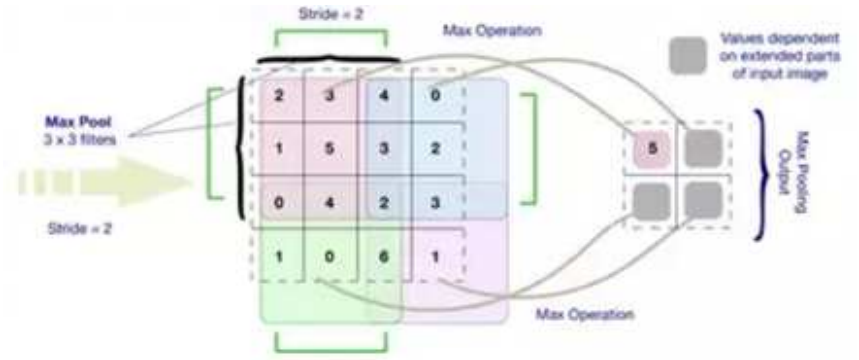
Max Pooling example



2x2 filters with stride = 2 (maximum value) is considered



3x3 filters with stride = 1 (maximum value) is considered



3x3 filters with stride = 2 (maximum value) is considered

Classification

1. **Flatten:** The pooled matrix is converted to a vector**.**

2. **Fully Connected layer:** The way this fully connected layer works is that it looks at the output of the previous layer (which as we remember should represent the activation maps of high-level features) and the number of classes p (10 for digit classification). *For example, if the program is predicting that some image is a dog, it will have high values in the activation maps that represent high-level features like a paw or 4 legs, etc. Basically, an FC layer looks at what high level features most strongly correlate to a particular class and has particular weights so that when you compute the products between the weights and the previous layer, you get the correct probabilities for the different classes.*
3. **Soft-max approach:** The output of a fully connected layer is as follows [0 .1 .1 .75 0 0 0 0 0 .05], then this represents a 10% probability that the image is a 1, a 10% probability that the image is a 2, a 75% probability that the image is a 3, and a 5% probability that the image is a 9 (SoftMax approach) for digit classification.

Training

§We know kernels also known as feature identifiers, used for identification of specific features. But how the kernels are initialized with the specific weights or how do the filters know what values to have.

Hence comes the important step of training. The training process is also known as backpropagation, which is further separated into 4 distinct sections or processes.

- Forward Pass
- Loss Function
- Backward Pass
- Weight Update

The Forward Pass

For the first epoch or iteration of the training the initial kernels of the first convolutional layer are initialized with random values. Thus, after the first iteration output will be something like [.1.1.1.1.1.1.1.1.1.1], which does not give preference to any class as the kernels don't have specific weights.

The Loss Function

The training involves images along with labels, hence the label for the digit 3 will be [0 0 0 1 0 0 0 0 0 0], whereas the output after a first epoch is very different, hence we will calculate loss (MSE — Mean Squared Error)

$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

The objective is to minimize the loss, which is an optimization problem in calculus. It involves trying to adjust the weights to reduce the loss.

The Backward Pass

It involves determining which weights contributed most to the loss and finding ways to adjust them so that the loss decreases. It is computed using $\frac{\partial L}{\partial W}$ (or $\nabla_w L$), where L is the loss and the W is the weights of the corresponding kernel.

The weights update

This is where the weights of the kernel are updated using the following equation.

$$w = w_i - \eta \frac{dL}{dW}$$

w = Weight
w_i = Initial Weight
η = Learning Rate

Here the Learning Rate is chosen by the programmer. Larger value of the learning rate indicates much larger steps towards optimization of steps and larger time to convolve to an optimized weight.

Testing

Finally, to see whether or not our CNN works, we have a different set of images and labels (can't double dip between training and test!) and pass the images through the CNN. We compare the outputs to the ground truth and see if our network works!

Q13. What Is Pooling on CNN, and How Does It Work?

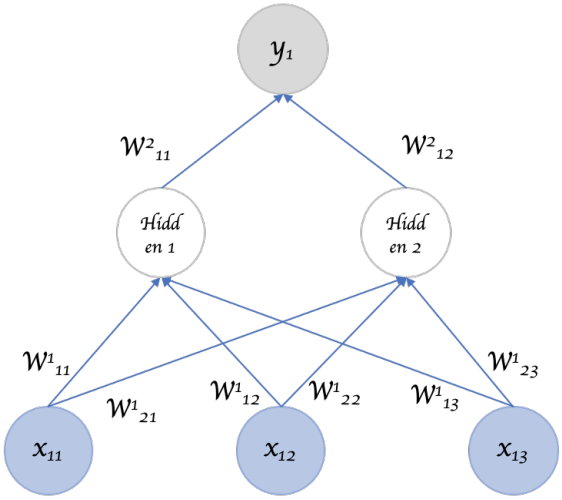
Pooling is used to reduce the spatial dimensions of a CNN. It performs down-sampling operations to reduce the dimensionality and creates a pooled feature map by sliding a filter matrix over the input matrix.

Q14. What are Recurrent Neural Networks (RNNs)?

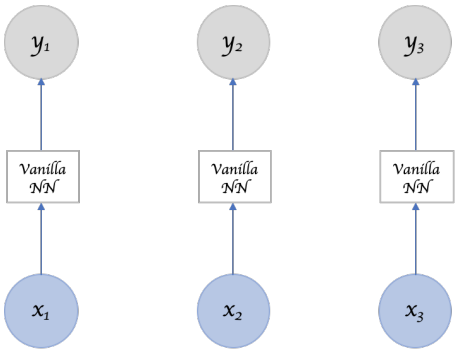
https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce

RNNs are a type of artificial neural networks designed to recognize the pattern from the sequence of data such as Time series, stock market and government agencies etc.

Recurrent Neural Networks (RNNs) add an interesting twist to basic neural networks. A vanilla neural network takes in a fixed size vector as input which limits its usage in situations that involve a 'series' type input with no predetermined size.

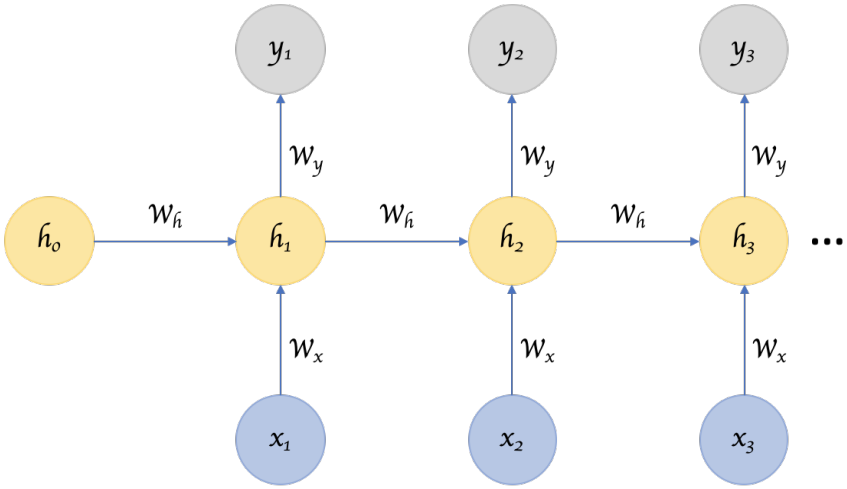


RNNs are designed to take a series of input with no predetermined limit on size. One could ask what's the big deal, I can call a regular NN repeatedly too?



Sure can, but the 'series' part of the input means something. A single input item from the series is related to others and likely has an influence on its neighbors. Otherwise it's just "many" inputs, not a "series" input (duh!).

Recurrent Neural Network remembers the past and its decisions are influenced by what it has learnt from the past. Note: Basic feed forward networks "remember" things too, but they remember things they learnt during training. For example, an image classifier learns what a "1" looks like during training and then uses that knowledge to classify things in production.
While RNNs learn similarly while training, in addition, they remember things learnt from prior input(s) while generating output(s). RNNs can take one or more input vectors and produce one or more output vectors and the output(s) are influenced not just by weights applied on inputs like a regular NN, but also by a "hidden" state vector representing the context based on prior input(s)/output(s). So, the same input could produce a different output depending on previous inputs in the series.



In summary, in a vanilla neural network, a fixed size input vector is transformed into a fixed size output vector. Such a network becomes "recurrent" when you repeatedly apply the transformations to a series of given input and produce a series of output vectors. There is no pre-set limitation to the size of the vector. And, in addition to generating the output which is a function of the input and hidden state, we update the hidden state itself based on the input and use it in processing the next input.

Parameter Sharing

You might have noticed another key difference between Figure 1 and Figure 3. In the earlier, multiple different weights are applied to the different parts of an input item generating a hidden layer neuron, which in turn is transformed using further weights to produce an output. There seems to be a lot of weights in play here. Whereas in Figure 3, we seem to be applying the same weights over and over again to different items in the input series.

I am sure you are quick to point out that we are kind of comparing apples and oranges here. The first figure deals with "a" single input whereas the second figure represents multiple inputs from a series. But nevertheless, intuitively speaking, as the number of inputs increase, shouldn't the number of weights in play increase as well? Are we losing some versatility and depth in Figure 3?

Perhaps we are. We are sharing parameters across inputs in Figure 3. If we don't share parameters across inputs, then it becomes like a vanilla neural network where each input node requires weights of their own. This introduces the constraint that the length of the input has to be fixed and that makes it impossible to leverage a series type input where the lengths differ and is not always known.

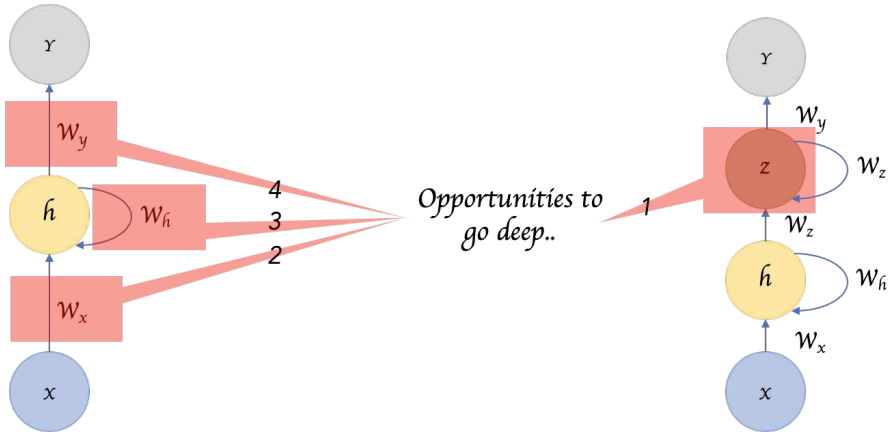
But what we seemingly lose in value here, we gain back by introducing the "hidden state" that links one input to the next. The hidden state captures the relationship that neighbors might have with each other in a serial input and it keeps changing in every step, and thus effectively every input undergoes a different transition!

Image classifying CNNs have become so successful because the 2D convolutions are an effective form of parameter sharing where each convolutional filter basically extracts the presence or absence of a feature in an image which is a function of not just one pixel but also of its surrounding neighbor pixels.

In other words, the success of CNNs and RNNs can be attributed to the concept of "parameter sharing" which is fundamentally an effective way of leveraging the relationship between one input item and its surrounding neighbors in a more intrinsic fashion compared to a vanilla neural network.

Deep RNNs

While it's good that the introduction of hidden state enabled us to effectively identify the relationship between the inputs, is there a way we can make an RNN "deep" and gain the multi-level abstractions and representations we gain through "depth" in a typical neural network?

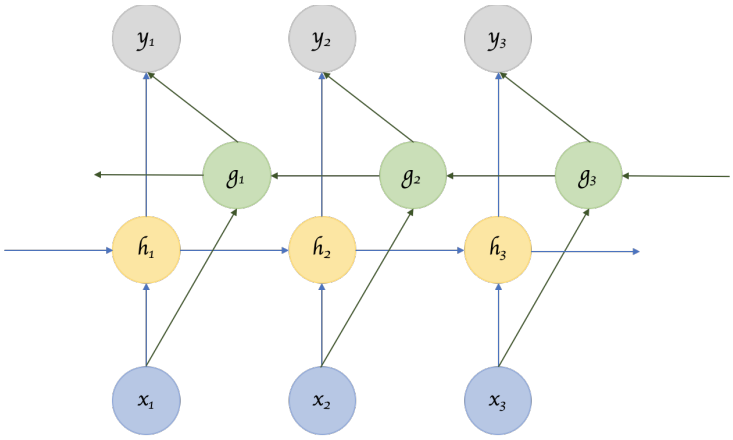


Here are four possible ways to add depth.

1) We can add hidden states, one on top of another, feeding the output of one to the next.
2) We can also add additional nonlinear hidden layers between input to hidden state.
3) We can increase depth in the hidden to hidden transition.
4) We can increase depth in the hidden to output transition.

Bidirectional RNNs

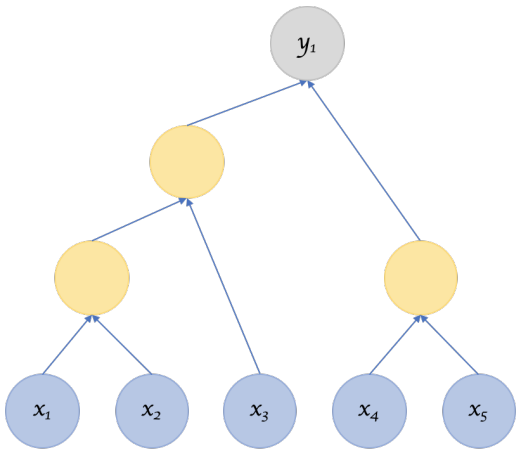
Sometimes it's not just about learning from the past to predict the future, but we also need to look into the future to fix the past. In speech recognition and handwriting recognition tasks, where there could be considerable ambiguity given just one part of the input, we often need to know what's coming next to better understand the context and detect the present.



This does introduce the obvious challenge of how much into the future we need to look into, because if we have to wait to see all inputs then the entire operation will become costly. And in cases like speech recognition, waiting till an entire sentence is spoken might make for a less compelling use case. Whereas for NLP tasks, where the inputs tend to be available, we can likely consider entire sentences all at once. Also, depending on the application, if the sensitivity to immediate and closer neighbors is higher than inputs that come further away, a variant that looks only into a limited future/past can be modeled.

Recursive Neural Network

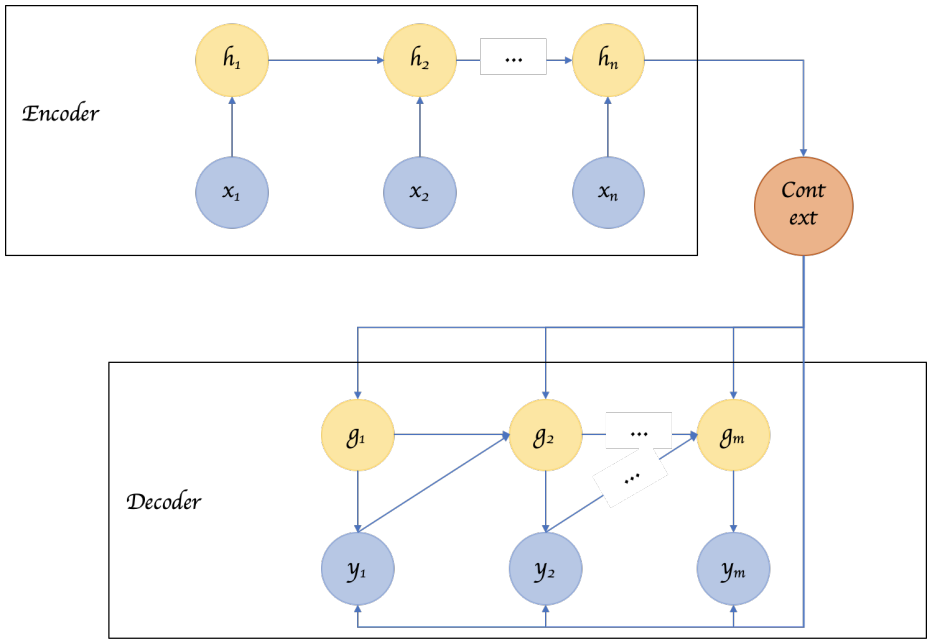
A recurrent neural network parses the inputs in a sequential fashion. A recursive neural network is similar to the extent that the transitions are repeatedly applied to inputs, but not necessarily in a sequential fashion. Recursive Neural Networks are a more general form of Recurrent Neural Networks. It can operate on any hierarchical tree structure. Parsing through input nodes, combining child nodes into parent nodes and combining them with other child/parent nodes to create a tree like structure. Recurrent Neural Networks do the same, but the structure there is strictly linear. i.e. weights are applied on the first input node, then the second, third and so on.



But this raises questions pertaining to the structure. How do we decide that? If the structure is fixed like in Recurrent Neural Networks then the process of training, backprop, makes sense in that they are similar to a regular neural network. But if the structure isn't fixed, is that learnt as well?

Encoder Decoder Sequence to Sequence RNNs

Encoder Decoder or Sequence to Sequence RNNs are used a lot in translation services. The basic idea is that there are two RNNs, one an encoder that keeps updating its hidden state and produces a final single "Context" output. This is then fed to the decoder, which translates this context to a sequence of outputs. Another key difference in this arrangement is that the length of the input sequence and the length of the output sequence need not necessarily be the same.



LSTMs

LSTM is not a different variant of RNN architecture, but rather it introduces changes to how we compute outputs and hidden state using the inputs.
In a vanilla RNN, the input and the hidden state are simply passed through a single tanh layer. LSTM (Long-Short-Term Memory) networks improve on this simple transformation and introduces additional gates and a cell state, such that it fundamentally addresses the problem of keeping or resetting context, across sentences and regardless of the distance between such context resets. There are variants of LSTMs including GRUs that utilize the gates in different manners to address the problem of long-term dependencies.

Q15. How Does an LSTM Network Work?

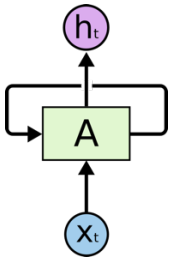
http://karpathy.github.io/2015/05/21/rnn-effectiveness
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long-Short-Term Memory (LSTM) is a special kind of recurrent neural network capable of learning long-term dependencies, remembering information for long periods as its default behavior. There are three steps in an LSTM network:

- Step 1: The network decides what to forget and what to remember.
- Step 2: It selectively updates cell state values.
- Step 3: The network decides what part of the current state makes it to the output.

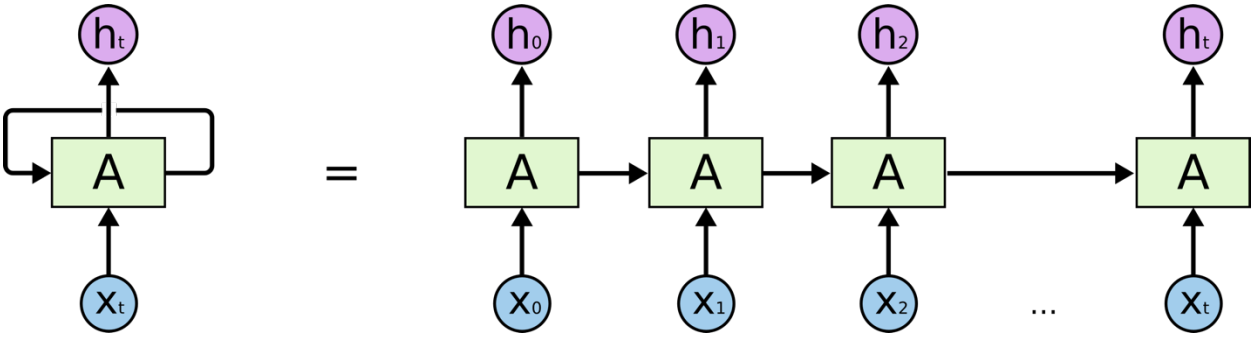
Recurrent Neural Networks

Humans don't start their thinking from scratch every second. As you read this essay, you understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Your thoughts have persistence.
Traditional neural networks can't do this, and it seems like a major shortcoming. *For example, imagine you want to classify what kind of event is happening at every point in a movie. It's unclear how a traditional neural network could use its reasoning about previous events in the film to inform later ones.*
Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist.



Recurrent Neural Networks have loops.

In the above diagram, a chunk of neural network, A, looks at some input x_t and outputs a value h_t. A loop allows information to be passed from one step of the network to the next.
These loops make recurrent neural networks seem kind of mysterious. However, if you think a bit more, it turns out that they aren't all that different than a normal neural network. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Consider what happens if we unroll the loop:



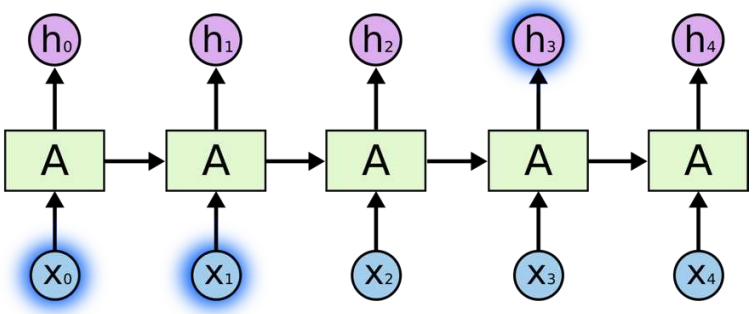
An unrolled recurrent neural network.

This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists. They're the natural architecture of neural network to use for such data.
And they certainly are used! In the last few years, there have been incredible success applying RNNs to a variety of problems: speech recognition, language modeling, translation, image captioning…

Essential to these successes is the use of "LSTMs," a very special kind of recurrent neural network which works, for many tasks, much better than the standard version. Almost all exciting results based on recurrent neural networks are achieved with them.

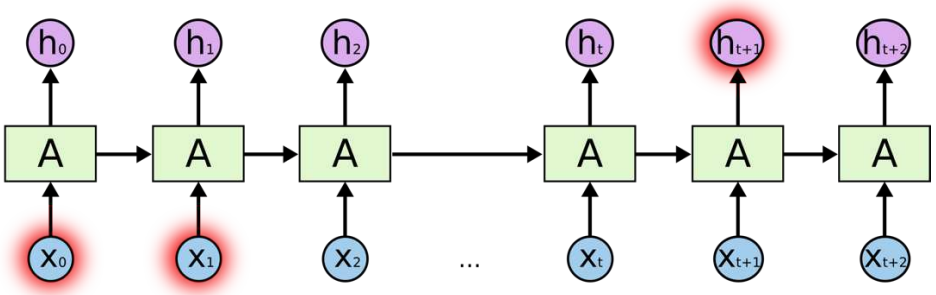
The Problem of Long-Term Dependencies

One of the appeals of RNNs is the idea that they might be able to connect previous information to the present task, such as using previous video frames might inform the understanding of the present frame. If RNNs could do this, they'd be extremely useful. But can they? It depends.
Sometimes, we only need to look at recent information to perform the present task. *For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in "the clouds are in the* sky*," we don't need any further context – it's pretty obvious the next word is going to be sky. In such cases, where the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information.*



But there are also cases where we need more context. Consider trying to predict the last word in the text "I grew up in France… I speak fluent French*." Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of France, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large.*

Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.



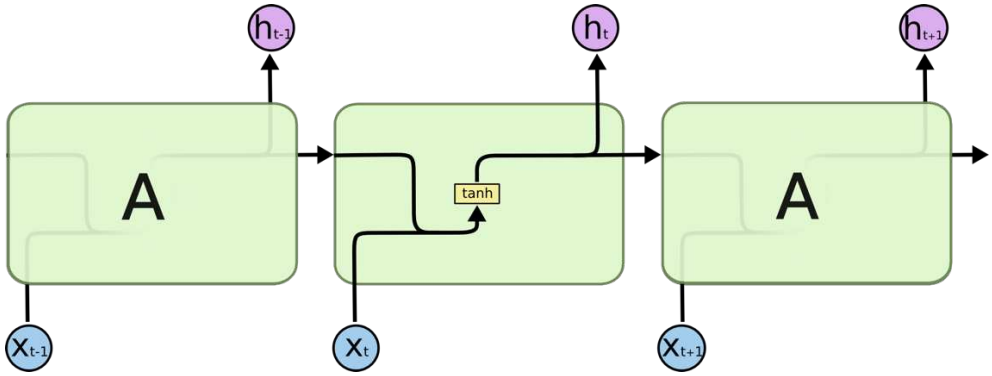
In theory, RNNs are absolutely capable of handling such "long-term dependencies." A human could carefully pick parameters for them to solve toy problems of this form. Sadly, in practice, RNNs don't seem to be able to learn them. Thankfully, LSTMs don't have this problem!

LSTM Networks

Long Short-Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies. They work tremendously well on a large variety of problems and are now widely used.

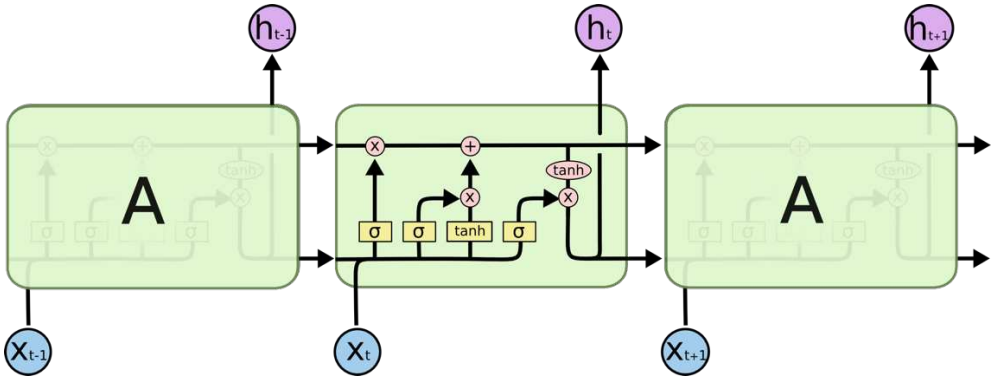
LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

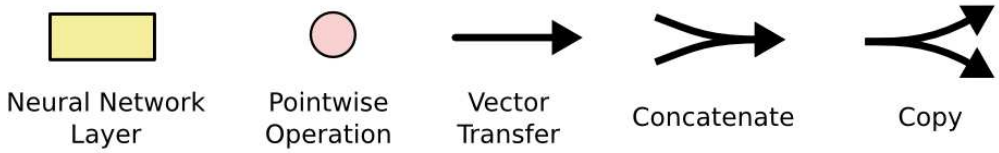


The repeating module in a standard RNN contains a single layer.

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.



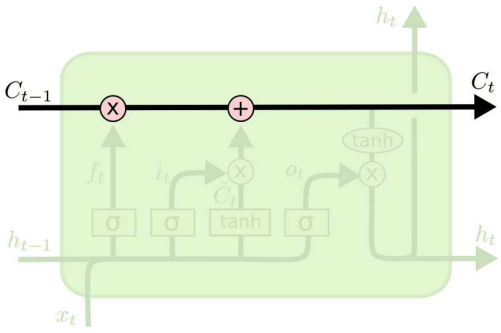
The repeating module in an LSTM contains four interacting layers.



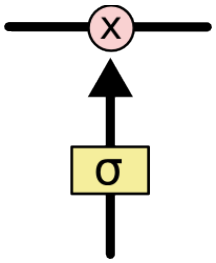
In the above diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denotes its content being copied and the copies going to different locations.

The Core Idea Behind LSTMs

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.
The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged.



The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.
Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.



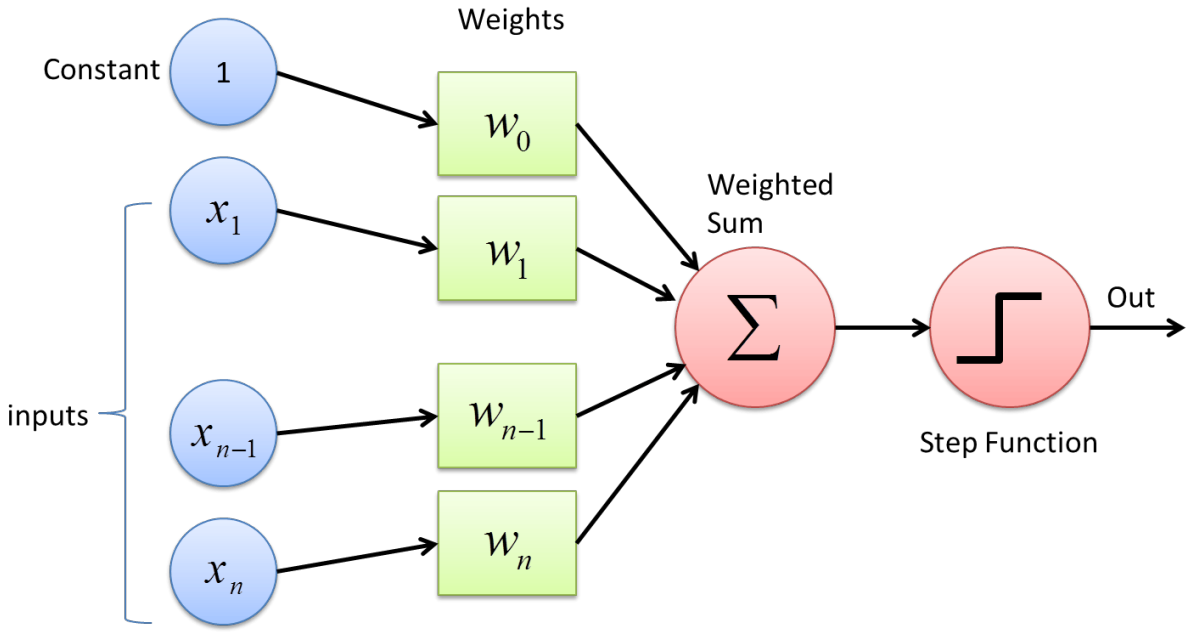
The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means “let everything through!”
An LSTM has three of these gates, to protect and control the cell state.

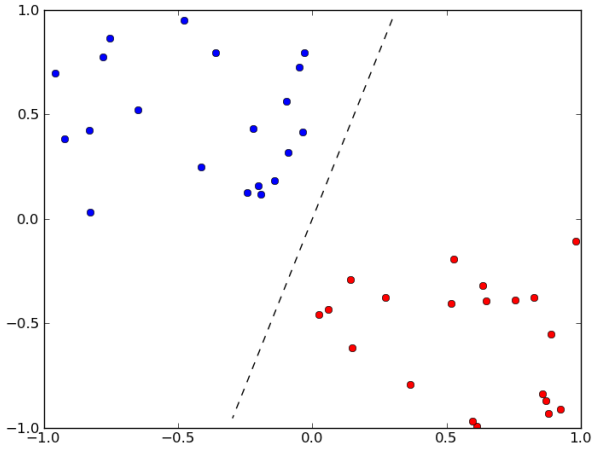
Q16. What Is a Multi-layer Perceptron (MLP)?

https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53

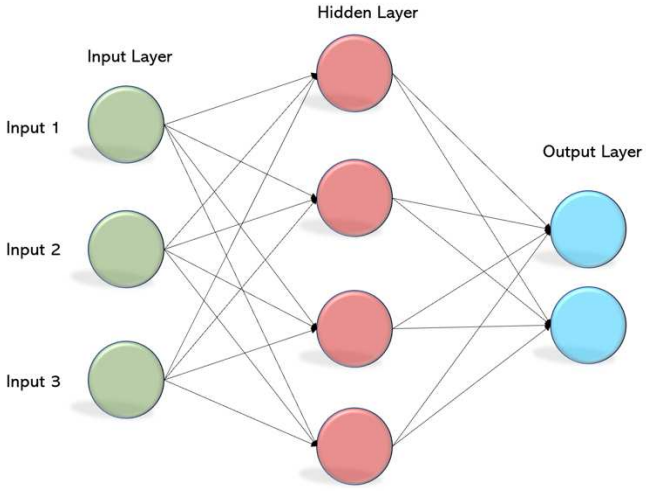
As in Neural Networks, MLPs have an input layer, a hidden layer, and an output layer. It has the same structure as a single layer perceptron with one or more hidden layers.

Perceptron is a single layer neural network and a multi-layer perceptron is called Neural Networks.
A (single layer) perceptron is a single layer neural network that works as a linear binary classifier. Being a single layer neural network, it can be trained without the use of more advanced algorithms like back propagation and instead can be trained by "stepping towards" your error in steps specified by a learning rate. When someone says perceptron, I usually think of the single layer version.





A single layer perceptron can classify only linear separable classes with binary output {0,1} or {-1,1}, but MLP can classify nonlinear classes. **The activation functions are used to map the input between the required values like {0, 1} or {-1, 1}**.



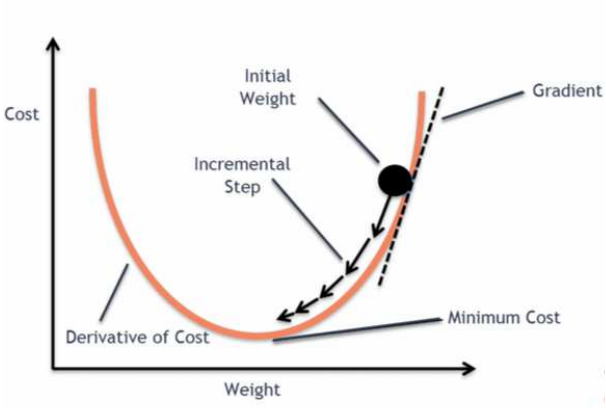
Except for the input layer, each node in the other layers uses a nonlinear activation function. This means the input layers, the data coming in, and the activation function is based upon all nodes and weights being added together, producing the output. MLP uses a supervised learning method called "backpropagation." In backpropagation, the neural network calculates the error with the help of cost function. It propagates this error backward from where it came (adjusts the weights to train the model more accurately).

Usually, RELU is in hidden layers (it does not classify), and Soft-max or tanh is in output layers.

Q17. Explain Gradient Descent.

Let's first explain what a gradient is. A gradient is a mathematical function. When calculated on a point of a function, it gives the hyperplane (or slope) of the directions in which the function increases more. The gradient vector can be interpreted as the "direction and rate of fastest increase". If the gradient of a

function is non-zero at a point *p*, the direction of the gradient is the direction in which the function increases most quickly from *p*, and the magnitude of the gradient is the rate of increase in that direction. Further, the gradient is the zero vector at a point if and only if it is a stationary point (where the derivative vanishes).
In DS, it simply measures the change in all weights with regard to the change in error, as we are partially derivating by w the loss function.



Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function.

The goal of the gradient descent is to minimize a given function which, in our case, is the loss function of the neural network. To achieve this goal, it performs two steps iteratively.

1. Compute the slope (gradient) that is the first-order derivative of the function at the current point
2. Move-in the opposite direction of the slope increase from the current point by the computed amount

So, the idea is to pass the training set through the hidden layers of the neural network and then update the parameters of the layers by computing the gradients using the training samples from the training dataset.

Think of it like this. Suppose a man is at top of the valley and he wants to get to the bottom of the valley. So, he goes down the slope. He decides his next position based on his current position and stops when he gets to the bottom of the valley which was his goal.

Q18. What is exploding gradients?

https://machinelearningmastery.com/exploding-gradients-in-neural-networks/

While training an RNN, if you see exponentially growing (very large) error gradients which accumulate and result in very large updates to neural network model weights during training, they're known as exploding gradients. At an extreme, the values of weights can become so large as to overflow and result in NaN values. The explosion occurs through exponential growth by repeatedly multiplying gradients through the network layers that have values larger than 1.0.

This has the effect of your model is unstable and unable to learn from your training data.

There are some subtle signs that you may be suffering from exploding gradients during the training of your network, such as:

- The model is unable to get traction on your training data (e.g. poor loss).
- The model is unstable, resulting in large changes in loss from update to update.
- The model loss goes to NaN during training.
- The model weights quickly become very large during training.
- The error gradient values are consistently above 1.0 for each node and layer during training.

Solutions

1. **Re-Design the Network Model:**
    a. In deep neural networks, exploding gradients may be addressed by redesigning the network to have fewer layers. There may also be some benefit in using a smaller batch size while training the network.
    b. In RNNs, updating across fewer prior time steps during training, called truncated Backpropagation through time, may reduce the exploding gradient problem.

2. **Use Long Short-Term Memory Networks:** In RNNs, exploding gradients can be reduced by using the Long Short-Term Memory (LSTM) memory units and perhaps related gated-type neuron structures. Adopting LSTM memory units is a new best practice for recurrent neural networks for sequence prediction.

3. **Use Gradient Clipping:** Exploding gradients can still occur in very deep Multilayer Perceptron networks with a large batch size and LSTMs with very long input sequence lengths. If exploding gradients are still occurring, you can check for and limit the size of gradients during the training of your network. This is called gradient clipping. Specifically, the values of the error gradient are checked against a threshold value and clipped or set to that threshold value if the error gradient exceeds the threshold.

4. **Use Weight Regularization:** another approach, if exploding gradients are still occurring, is to check the size of network weights and apply a penalty to the networks loss function for large weight values. This is called weight regularization and often an L1 (absolute weights) or an L2 (squared weights) penalty can be used.

Q19. What is vanishing gradients?

While training an RNN, your slope can become either too small; this makes the training difficult. When the slope is too small, the problem is known as a Vanishing Gradient. It leads to long training times, poor performance, and low accuracy.

- Hyperbolic tangent and Sigmoid/Soft-max suffer vanishing gradient.
- RNNs suffer vanishing gradient, LSTM no (so it is perfect to predict stock prices). In fact, the propagation of error through previous layers makes the gradient get smaller so the weights are not updated.

Solutions

1. **Choose RELU**
2. **Use LSTM (for RNNs)**
3. **Use ResNet (Residual Network)** → after some layers, add x again: $F(x) \to \cdots \to F(x) + x$
4. **Multi-level hierarchy:** pre-train one layer at the time through unsupervised learning, then fine-tune via backpropagation
5. **Gradient checking:** debugging strategy used to numerically track and assess gradients during training.

Q20. What is Back Propagation and Explain it Works.

Backpropagation is a training algorithm used for neural network. In this method, we update the weights of each layer from the last layer recursively, with the formula:

$$w_{previous\ layer} = w_{layer} - \eta \nabla_w L(w)$$

It has the following steps:

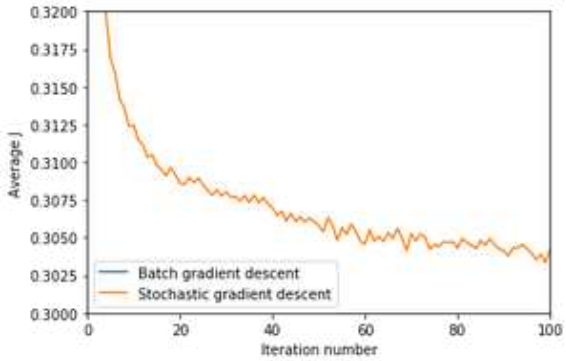
- Forward Propagation of Training Data (initializing weights with random or pre-assigned values)
- Gradients are computed using output weights and target
- Back Propagate for computing gradients of error from output activation
- Update the Weights

Q21. What are the variants of Back Propagation?

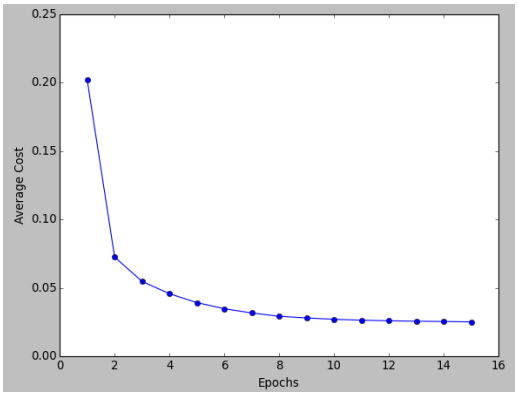
https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a

- **Stochastic Gradient Descent:** In Batch Gradient Descent we were considering all the examples for every step of Gradient Descent. But what if our dataset is very huge. Deep learning models crave for data. The more the data the more chances of a model to be good. *Suppose our dataset has 5 million examples, then just to take one step the model will have to calculate the gradients of all the 5 million examples. This does not seem an efficient way.* To tackle this problem, we have Stochastic Gradient Descent. In Stochastic Gradient Descent (SGD), we consider just one example at a time to take a single step. We do the following steps in **one epoch** for SGD:
    1. Take an example
    2. Feed it to Neural Network
    3. Calculate its gradient
    4. Use the gradient we calculated in step 3 to update the weights
    5. Repeat steps 1–4 for all the examples in training dataset

    Since we are considering just one example at a time the cost will fluctuate over the training examples and it will **not** necessarily decrease. But in the long run, you will see the cost decreasing with fluctuations. Also, because the cost is so fluctuating, it will never reach the minimum, but it will keep dancing around it. SGD can be used for larger datasets. It converges faster when the dataset is large as it causes updates to the parameters more frequently.



- **Batch Gradient Descent:** all the training data is taken into consideration to take a single step. We take the average of the gradients of all the training examples and then use that mean gradient to update our parameters. So that's just one step of gradient descent in one epoch. Batch Gradient Descent is great for convex or relatively smooth error manifolds. In this case, we move somewhat directly towards an optimum solution. The graph of cost vs epochs is also quite smooth because we are averaging over all the gradients of training data for a single step. The cost keeps on decreasing over the epochs.



- **Mini-batch Gradient Descent:** It's one of the most popular optimization algorithms. It's a variant of Stochastic Gradient Descent and here instead of single training example, mini batch of samples is used. Batch Gradient Descent can be used for smoother curves. SGD can be used when the dataset is large. Batch Gradient Descent converges directly to minima. SGD converges faster for larger datasets. But, since in SGD we use only one example at a time, we cannot implement the vectorized implementation on it. This can slow down the computations. To tackle this problem, a mixture of Batch Gradient Descent and SGD is used. Neither we use all the dataset all at once nor we use the single example at a time. We use a batch of a fixed number of training examples which is less than the actual dataset and call it a mini-batch. Doing this helps us achieve the advantages of both the former variants we saw. So, after creating the mini-batches of fixed size, we do the following steps in **one epoch:**
    1. Pick a mini-batch
    2. Feed it to Neural Network
    3. Calculate the mean gradient of the mini-batch
    4. Use the mean gradient we calculated in step 3 to update the weights
    5. Repeat steps 1–4 for the mini-batches we created

Just like SGD, the average cost over the epochs in mini-batch gradient descent fluctuates because we are averaging a small number of examples at a time. So, when we are using the mini-batch gradient descent we are updating our parameters frequently as well as we can use vectorized implementation for faster computations.

Q22. What are the different Deep Learning Frameworks?

- PyTorch: PyTorch is an open source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab. It is free and open-source software released under the Modified BSD license.
- TensorFlow: TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. Licensed by Apache License 2.0. Developed by Google Brain Team.
- Microsoft Cognitive Toolkit: Microsoft Cognitive Toolkit describes neural networks as a series of computational steps via a directed graph.
- Keras: Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. Licensed by MIT.

Q23. What is the role of the Activation Function?

The Activation function is used to introduce non-linearity into the neural network helping it to learn more complex function. Without which the neural network would be only able to learn linear function which is a linear combination of its input data. An activation function is a function in an artificial neuron that delivers an output based on inputs.

Q24. Name a few Machine Learning libraries for various purposes.

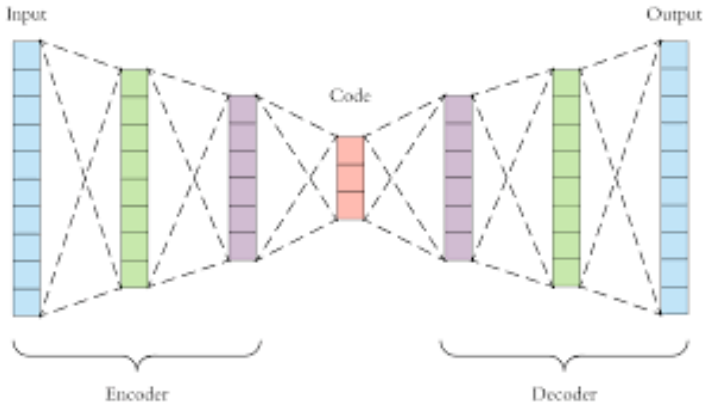
Purpose	Libraries
Scientific Computation	Numpy
Tabular Data	Pandas, GeoPandas
Data Modelling & Preprocessing	Scikit Learn
Time-Series Analysis	Statsmodels
Text processing	NTLK, Regular Expressions
Deep Learning	TensorFlow, Pytorch
Visualization	Bokeh, Seaborn
Plotting	Matplot

Q25. What is an Auto-Encoder?

https://www.quora.com/What-is-an-autoencoder-What-are-its-applications

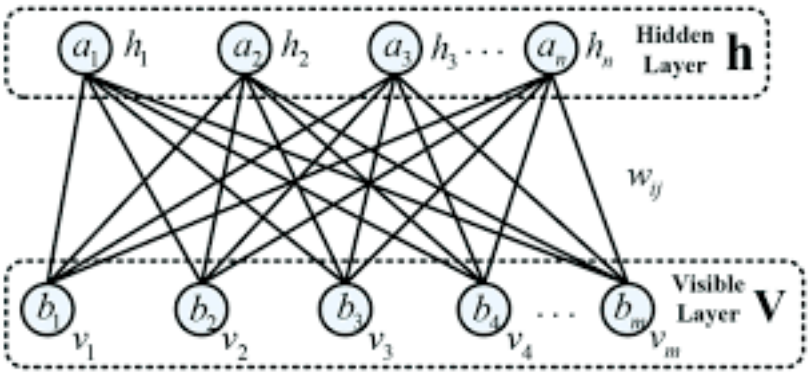
Auto-encoders are simple learning networks that aim to transform inputs into outputs with the minimum possible error. This means that we want the output to be as close to input as possible. We add a couple of layers between the input and the output, and the sizes of these layers are smaller than the input layer. The auto-encoder receives unlabeled input which is then encoded to reconstruct the input.

An **autoencoder** is a type of artificial neural network used to learn efficient data coding in an unsupervised manner. The aim of an **autoencoder** is to learn a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore signal "noise". Along with the reduction side, a reconstructing side is learnt, where the autoencoder tries to generate from the reduced encoding a representation as close as possible to its original input, hence its name. Several variants exist to the basic model, with the aim of forcing the learned representations of the input to assume useful properties. Autoencoders are effectively used for solving many applied problems, from face recognition to acquiring the semantic meaning of words.



Q26. What is a Boltzmann Machine?

Boltzmann machines have a simple learning algorithm that allows them to discover interesting features that represent complex regularities in the training data. The Boltzmann machine is basically used to optimize the weights and the quantity for the given problem. The learning algorithm is very slow in networks with many layers of feature detectors. "Restricted Boltzmann Machines" algorithm has a single layer of feature detectors which makes it faster than the rest.



Q27. What Is Dropout and Batch Normalization?

Dropout is a technique of dropping out hidden and visible nodes of a network randomly to prevent overfitting of data (typically dropping 20 per cent of the nodes). It doubles the number of iterations needed to converge the network. It used to avoid overfitting, as it increases the capacity of generalization.

Batch normalization is the technique to improve the performance and stability of neural networks by normalizing the inputs in every layer so that they have mean output activation of zero and standard deviation of one.

Q28. Why Is TensorFlow the Most Preferred Library in Deep Learning?

TensorFlow provides both C++ and Python APIs, making it easier to work on and has a faster compilation time compared to other Deep Learning libraries like Keras and PyTorch. TensorFlow supports both CPU and GPU computing devices.

Q29. What Do You Mean by Tensor in TensorFlow?

A tensor is a mathematical object represented as arrays of higher dimensions. Think of a n-D matrix. These arrays of data with different dimensions and ranks fed as input to the neural network are called “Tensors.”

Q30. What is the Computational Graph?

Everything in a TensorFlow is based on creating a computational graph. It has a network of nodes where each node operates. Nodes represent mathematical operations, and edges represent tensors. Since data flows in the form of a graph, it is also called a “DataFlow Graph.”

Q31. How is logistic regression done?

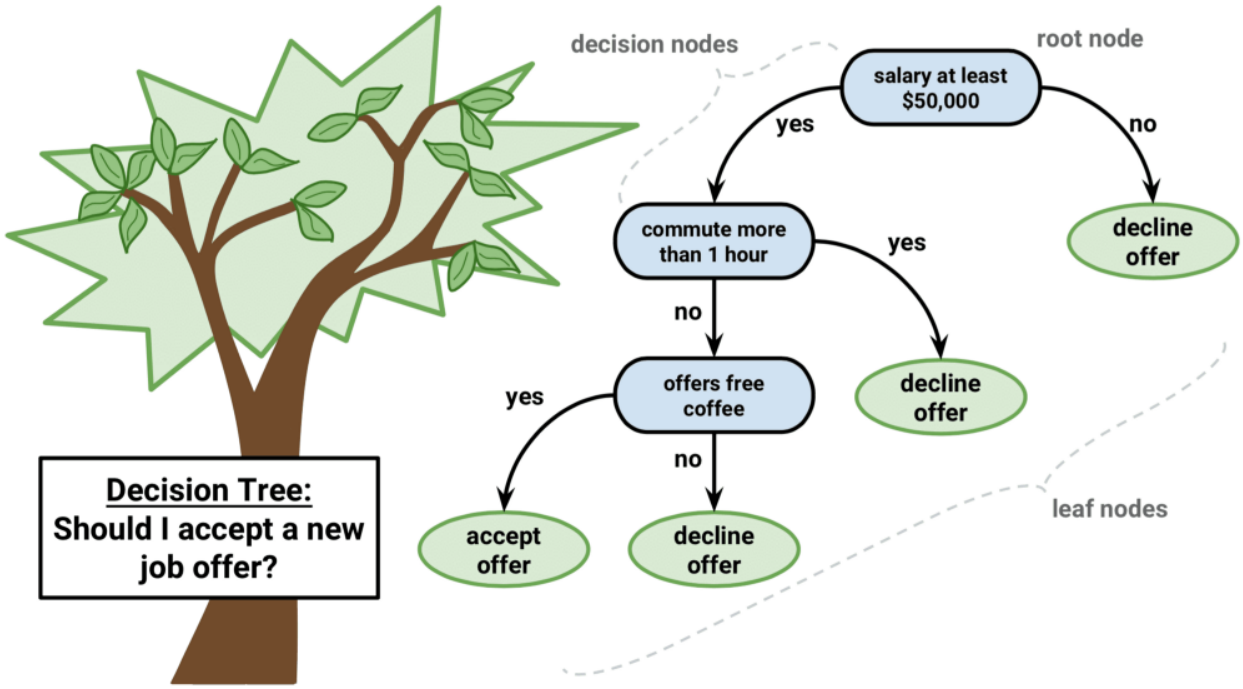
Logistic regression measures the relationship between the dependent variable (our label of what we want to predict) and one or more independent variables (our features) by estimating probability using its underlying logistic function (sigmoid).

Miscellaneous

Q1. Explain the steps in making a decision tree.

1. Take the entire data set as input
2. Calculate entropy of the target variable, as well as the predictor attributes
3. Calculate your information gain of all attributes (we gain information on sorting different objects from each other)
4. Choose the attribute with the highest information gain as the root node
5. Repeat the same procedure on every branch until the decision node of each branch is finalized

For example, let's say you want to build a decision tree to decide whether you should accept or decline a job offer. The decision tree for this case is as shown:



It is clear from the decision tree that an offer is accepted if:

- Salary is greater than $50,000
- The commute is less than an hour
- Coffee is offered

Q2. How do you build a random forest model?

A random forest is built up of a number of decision trees. If you split the data into different packages and make a decision tree in each of the different groups of data, the random forest brings all those trees together.

Steps to build a random forest model:

1. Randomly select k features from a total of m features where $k << m$
2. Among the k features, calculate the node D using the best split point
3. Split the node into daughter nodes using the best split
4. Repeat steps two and three until leaf nodes are finalized
5. Build forest by repeating steps one to four for n times to create n number of trees

Q3. Differentiate between univariate, bivariate, and multivariate analysis.

Univariate

Univariate data contains only one variable. The purpose of the univariate analysis is to describe the data and find patterns that exist within it.

Example: height of students

Height (in cm)
164
167.3
170
174.2
178
180

The patterns can be studied by drawing conclusions using mean, median, mode, dispersion or range, minimum, maximum, etc.

Bivariate

Bivariate data involves two different variables. The analysis of this type of data deals with causes and relationships and the analysis is done to determine the relationship between the two variables.

Example: temperature and ice cream sales in the summer season

Temperature (in Celsius)	**Sales (in K $)**
20	2.0
25	2.1
26	2.3
28	2.7
30	3.1

Here, the relationship is visible from the table that temperature and sales are directly proportional to each other. The hotter the temperature, the better the sales.

Multivariate

Multivariate data involves three or more variables, it is categorized under multivariate. It is similar to a bivariate but contains more than one dependent variable.

Example: data for house price prediction

The patterns can be studied by drawing conclusions using mean, median, and mode, dispersion or range, minimum, maximum, etc. You can start describing the data and using it to guess what the price of the house will be.

Q4. What are the feature selection methods used to select the right variables?

There are two main methods for feature selection.

Filter Methods

This involves:

- Linear discrimination analysis
- ANOVA
- Chi-Square

The best analogy for selecting features is "bad data in, bad answer out." When we're limiting or selecting the features, it's all about cleaning up the data coming in.

Wrapper Methods

This involves:

- Forward Selection: We test one feature at a time and keep adding them until we get a good fit
- Backward Selection: We test all the features and start removing them to see what works better
- Recursive Feature Elimination: Recursively looks through all the different features and how they pair together

Wrapper methods are very labor-intensive, and high-end computers are needed if a lot of data analysis is performed with the wrapper method.

Q5. In your choice of language, write a program that prints the numbers ranging from one to 50. But for multiples of three, print "Fizz" instead of the number and for the multiples of five, print "Buzz." For numbers which are multiples of both three and five, print "FizzBuzz."

The code is shown below:

```
for x in range(51):

      if x % 3 == 0 and x % 5 == 0:
            print('fizzbuzz')

      elif x % 3 == 0:
            print('fizz')

      elif x % 5 == 0:
            print('buzz')

      else:
            print('fizzbuzz')
```

Q6. You are given a data set consisting of variables with more than 30 percent missing values. How will you deal with them?

If the data set is large, we can just simply remove the rows with missing data values. It is the quickest way; we use the rest of the data to predict the values.

For smaller data sets, we can impute missing values with the mean, median, or average of the rest of the data using pandas data frame in python. There are different ways to do so, such as:

```
df.mean(), df.fillna(mean)
```

Other option of imputation is using KNN for numeric or classification values (as KNN just uses k closest values to impute the missing value).

Q7. For the given points, how will you calculate the Euclidean distance in Python?

```
plot1 = [1,3]
plot2 = [2,5]
```

The Euclidean distance can be calculated as follows:

```
euclidean_distance = sqrt((plot1[0]-plot2[0])**2 + (plot1[1]-plot2[1])**2)
```

Q8. What are dimensionality reduction and its benefits?

Dimensionality reduction refers to the process of converting a data set with vast dimensions into data with fewer dimensions (fields) to convey similar information concisely.

This reduction helps in compressing data and reducing storage space. It also reduces computation time as fewer dimensions lead to less computing. It removes redundant features; for example, there's no point in storing a value in two different units (meters and inches).

Q9. How will you calculate eigenvalues and eigenvectors of the following 3x3 matrix?

Determinant of $A - \lambda 1$ and solve to find λ.

Q10. How should you maintain a deployed model?

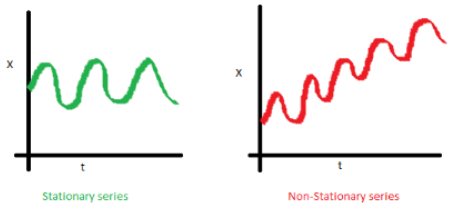
The steps to maintain a deployed model are (CREM):

1. **Monitor:** constant monitoring of all models is needed to determine their performance accuracy. When you change something, you want to figure out how your changes are going to affect things. This needs to be monitored to ensure it's doing what it's supposed to do.
2. **Evaluate:** evaluation metrics of the current model are calculated to determine if a new algorithm is needed.
3. **Compare:** the new models are compared to each other to determine which model performs the best.
4. **Rebuild:** the best performing model is re-built on the current state of data.

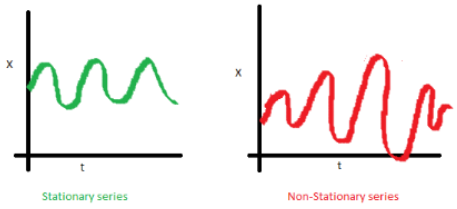
Q11. How can a time-series data be declared as stationery?

What does it mean for data to be stationary?

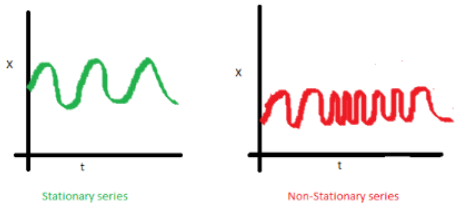
1. The mean of the series should not be a function of time. The red graph below is not stationary because the mean increases over time.



2. The variance of the series should not be a function of time. This property is known as homoscedasticity. Notice in the red graph the varying spread of data over time.



3. Finally, the covariance of the i th term and the (i + m) th term should not be a function of time. In the following graph, you will notice the spread becomes closer as the time increases. Hence, the covariance is not constant with time for the 'red series'.



Q12. 'People who bought this also bought...' recommendations seen on Amazon are a result of which algorithm?

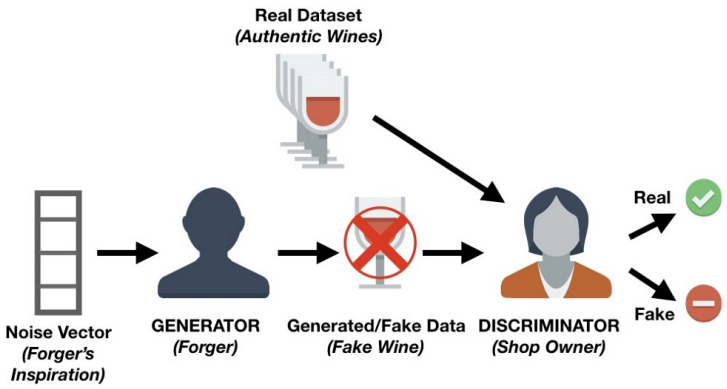
The recommendation engine is accomplished with collaborative filtering. Collaborative filtering explains the behavior of other users and their purchase history in terms of ratings, selection, etc.
The engine makes predictions on what might interest a person based on the preferences of other users.

In this algorithm, item features are unknown.

For example, a sales page shows that a certain number of people buy a new phone and also buy tempered glass at the same time. Next time, when a person buys a phone, he or she may see a recommendation to buy tempered glass as well.

Q13. What is a Generative Adversarial Network?

Suppose there is a wine shop purchasing wine from dealers, which they resell later. But some dealers sell fake wine. In this case, the shop owner should be able to distinguish between fake and authentic wine. The forger will try different techniques to sell fake wine and make sure specific techniques go past the shop owner's check. The shop owner would probably get some feedback from wine experts that some of the wine is not original. The owner would have to improve how he determines whether a wine is fake or authentic.
The forger's goal is to create wines that are indistinguishable from the authentic ones while the shop owner intends to tell if the wine is real or not accurately.



- *There is a noise vector coming into the forger who is generating fake wine.*
- *Here the forger acts as a Generator.*
- *The shop owner acts as a Discriminator.*
- *The Discriminator gets two inputs; one is the fake wine, while the other is the real authentic wine. The shop owner has to figure out whether it is real or fake.*

So, there are two primary components of Generative Adversarial Network (GAN) named:

1. Generator
2. Discriminator

The generator is a CNN that keeps keys producing images and is closer in appearance to the real images while the discriminator tries to determine the difference between real and fake images. The ultimate aim is to make the discriminator learn to identify real and fake images.

Q14. You are given a dataset on cancer detection. You have built a classification model and achieved an accuracy of 96 percent. Why shouldn't you be happy with your model performance? What can you do about it?

Cancer detection results in imbalanced data. In an imbalanced dataset, accuracy should not be based as a measure of performance. It is important to focus on the remaining four percent, which represents the patients who were wrongly diagnosed. Early diagnosis is crucial when it comes to cancer detection and can greatly improve a patient's prognosis.

Hence, to evaluate model performance, we should use Sensitivity (True Positive Rate), Specificity (True Negative Rate), F measure to determine the class wise performance of the classifier.

Q15. Below are the eight actual values of the target variable in the train file. What is the entropy of the target variable? [0, 0, 0, 1, 1, 1, 1, 1]

The target variable, in this case, is 1 (the last)

The formula for calculating the entropy is, putting $p = 5$ and $n = 8$, we get:

$$Entropy = -\left(\frac{5}{8}\, log\left(\frac{5}{8}\right) + \frac{3}{8}\log\left(\frac{3}{8}\right)\right)$$

Q16. We want to predict the probability of death from heart disease based on three risk factors: age, gender, and blood cholesterol level. What is the most appropriate algorithm for this case? Choose the correct option:

The most appropriate algorithm for this case is logistic regression.

Q17. After studying the behavior of a population, you have identified four specific individual types that are valuable to your study. You would like to find all users who are most similar to each individual type. Which algorithm is most appropriate for this study?

As we are looking for grouping people together specifically by four different similarities, it indicates the value of k. Therefore, K-means clustering is the most appropriate algorithm for this study.

Q18. You have run the association rules algorithm on your dataset, and the two rules {banana, apple} => {grape} and {apple, orange} => {grape} have been found to be relevant. What else must be true? Choose the right answer:

The answer is A: {grape, apple} must be a frequent itemset.

Q19. Your organization has a website where visitors randomly receive one of two coupons. It is also possible that visitors to the website will not receive a coupon. You have been asked to determine if offering a coupon to website visitors has any impact on their purchase decisions. Which analysis method should you use?

One-way ANOVA: in statistics, one-way analysis of variance is a technique that can be used to compare means of two or more samples. This technique can be used only for numerical response data, the "Y", usually one variable, and numerical or categorical input data, the "X", always one variable, hence "one-way".
The ANOVA tests the null hypothesis, which states that samples in all groups are drawn from populations with the same mean values. To do this, two estimates are made of the population variance. The ANOVA produces an F-statistic, the ratio of the variance calculated among the means to the variance within the samples. If the group means are drawn from populations with the same mean values, the variance between the group means should be lower than the variance of the samples, following the central limit theorem. A higher ratio therefore implies that the samples were drawn from populations with different mean values.

Q20. What are the feature vectors?

A feature vector is an n-dimensional vector of numerical features that represent an object. In machine learning, feature vectors are used to represent numeric or symbolic characteristics (called features) of an object in a mathematical way that's easy to analyze.

Q21. What is root cause analysis?

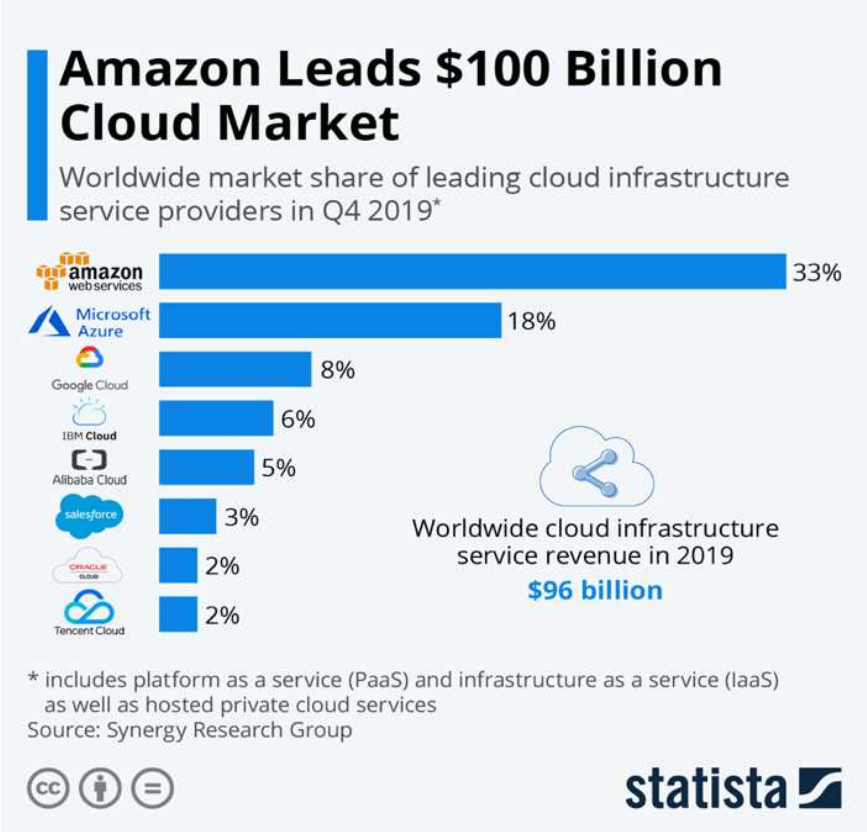
Root cause analysis was initially developed to analyze industrial accidents but is now widely used in other areas. It is a problem-solving technique used for isolating the root causes of faults or problems. A factor is called a root cause if its deduction from the problem-fault-sequence averts the final undesirable event from recurring.

Q22. Do gradient descent methods always converge to similar points?

They do not, because in some cases, they reach a local minimum or a local optimum point. You would not reach the global optimum point. This is governed by the data and the starting conditions.

Q23. What are the most popular Cloud Services used in Data Science?

https://www.zdnet.com/article/the-top-cloud-providers-of-2020-aws-microsoft-azure-google-cloud-hybrid-saas/



Q24. What is a Canary Deployment?

https://www.split.io/glossary/canary-deployment/

A canary deployment, or canary release, allows you to rollout your features to only a subset of users as an initial test to make sure nothing else in your system broke.

The initial steps for implementing canary deployment are:

1. create two clones of the production environment,
2. have a load balancer that initially sends all traffic to one version,
3. create new functionality in the other version.

When you deploy the new software version, you shift some percentage – say, 10% – of your user base to the new version while maintaining 90% of users on the old version. If that 10% reports no errors, you can roll it out to gradually more users, until the new version is being used by everyone. If the 10% has problems, though, you can roll it right back, and 90% of your users will have never even seen the problem.

Canary deployment benefits include zero downtime, easy rollout and quick rollback – plus the added safety from the gradual rollout process. It also has some drawbacks – the expense of maintaining multiple server instances, the difficult clone-or-don't-clone database decision.

Typically, software development teams implement blue/green deployment when they're sure the new version will work properly and want a simple, fast strategy to deploy it. Conversely, canary deployment is most useful when the development team isn't as sure about the new version and they don't mind a slower rollout if it means they'll be able to catch the bugs.

Q25. What is a Blue Green Deployment?

https://docs.cloudfoundry.org/devguide/deploy-apps/blue-green.html

Blue-green deployment is a technique that reduces downtime and risk by running two identical production environments called Blue and Green.

At any time, only one of the environments is live, with the live environment serving all production traffic. For this example, Blue is currently live, and Green is idle.

As you prepare a new version of your model, deployment and the final stage of testing takes place in the environment that is not live: in this example, Green. Once you have deployed and fully tested the model in Green, you switch the router, so all incoming requests now go to Green instead of Blue. Green is now live, and Blue is idle.

This technique can eliminate downtime due to app deployment and reduces risk: if something unexpected happens with your new version on Green, you can immediately roll back to the last version by switching back to Blue.

This document is based on the original document by Steve Nouri (LinkedIn).

Reviewed and corrected by Davide Callegaro (LinkedIn).

Original credits to kdnuggets, Simplilearn, Edureka, Guru99, Hackernoon, Datacamp, Nitin Panwar, Michael Rundell.

Below some questions the reader shall view the link of the original article.