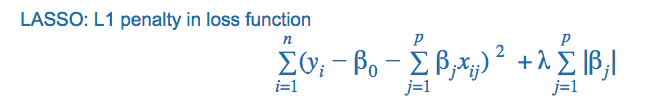
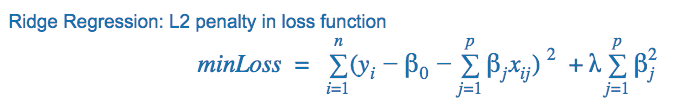
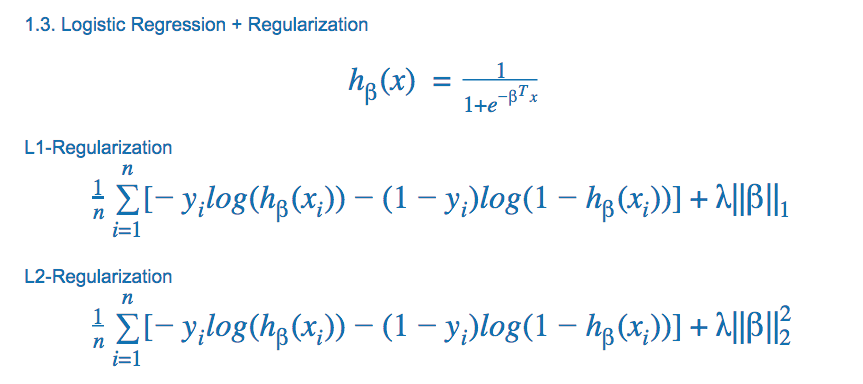
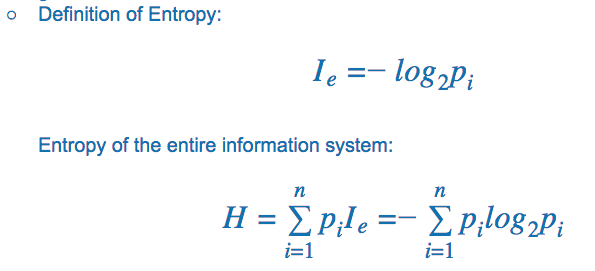
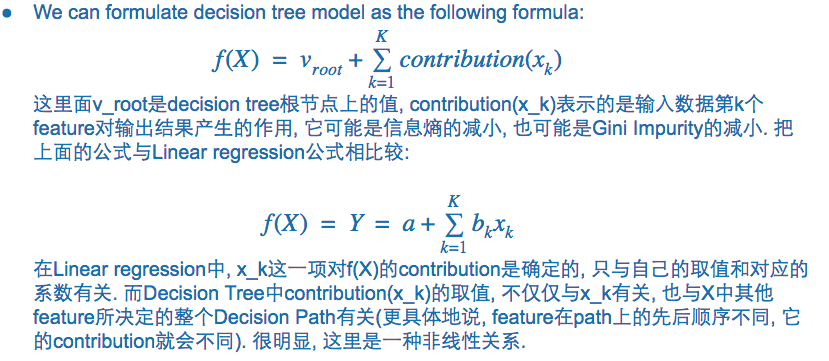
**DS Interview Questions**

**Part 1: Modeling**

1. How to solve overfitting problem?
   * Increase training data size
   * Avoid overfitting your dataset
     + Filter out features, e.g. feature reduction PCA
     + Regularization: Ridge Regression, LASSO (Least Absolute Shrinkage and Selection Operator), Logistic Regression L2, Logistic Regression L1
     + cross validation
2. What is the difference between L1 and L2?
   * Regularization: is a method for adding constraints or penalty to a model, with the goal of preventing overfitting and improving generalization
   * Lasso/L1 
     + is not very stable
     + tends to provide sparse solution
     + It only keeps one of the many highly correlated features; out of these correlated features, only one of them would have a non-zero coefficient, the others would be zero
     + Thus L1 could be used for feature selection
   * Ridge/L2 
     + is stable
     + It keeps all the correlated features; all the coefficients would be the same for correlated features
3. Could you write down the formula/loss function for Logistic Regression?
4. What is Decision Tree and how to build one?
   * Decision Tree
     + is a supervised prediction model
     + is usually a k-nary tree. Each leaf denotes one classification or continuous value of the dependent variable
     + two types: regression and classification
   * How to build
     + two steps
       1. feature prioritization
       2. feature splitting
     + Algorithm Details - ID3 Algorithm (Entropy): At each iteration, we split the feature which can give us the largest entropy gain
     + Calculate Entropy Gain - see example in class 8
5. Comparison Decision Tree with Linear Regression
   * Linear vs. Non-Linear (See formula)
6. What is KNN and how to implement a KNN?
   * K denotes the k labels that are closest to your target sample
     + The higher K is, the smoother (more robust) your model can be
   * Multiple definition choices for distances
     + City Block
     + Euclidean
     + Gaussian
   * Pros and Cons
     + Pros: No training process, making predictions directly
     + Cons: Large time and space complexity when the data is big, hard to save the model
   * Implement a KNN

Find the top K shortest distance from your returned unsorted distance array

* + - Step 1: Sorting (Merge Sort/Quick Sort)

Time: O(nlogn)

Space: O(n)

# Function to merge two arrays

def merge(a,b):

c = []

index\_a = index\_b = 0

while index\_a < len(a) and index\_b < len(b):

if a[index\_a] < b[index\_b]:

c.append(a[index\_a])

index\_a += 1

else:

c.append(b[index\_b])

index\_b += 1

if index\_a < len(a):

c.extend(a[index\_a:])

if index\_b < len(b):

c.extend(b[index\_b:])

return c

# Function to sort an array using merge sort algorithm

def merge\_sort(x):

if len(x) == 0 or len(x) == 1:

return x

mid = len(x)/2

a = merge\_sort(x[:mid])

b = merge\_sort(x[mid:])

return merge(a,b)

* + - Step 2: MaxHeap (Onlien Algo - Streaming Data)

Time: O(k+(n-k)\*logk)

Space: O(k)

import heapq

def find\_k\_closest(array, k):

if array is None or len(array) == 0:

return None

result = [-i for i in array[0:k]]

heapq.heapify(array)

for i in range(k, len(array)):

if array[i] < -res[0]:

heapq.heappop(result)

heapq.heappush(result, -array[i])

return [-i for i in result]

* + - Step 2: MinHeap

Time: O(n+k\*logn)

Space: O(n)

import heapq

def find\_k\_closest(array, k):

if array is None or len(array) == 0:

return None

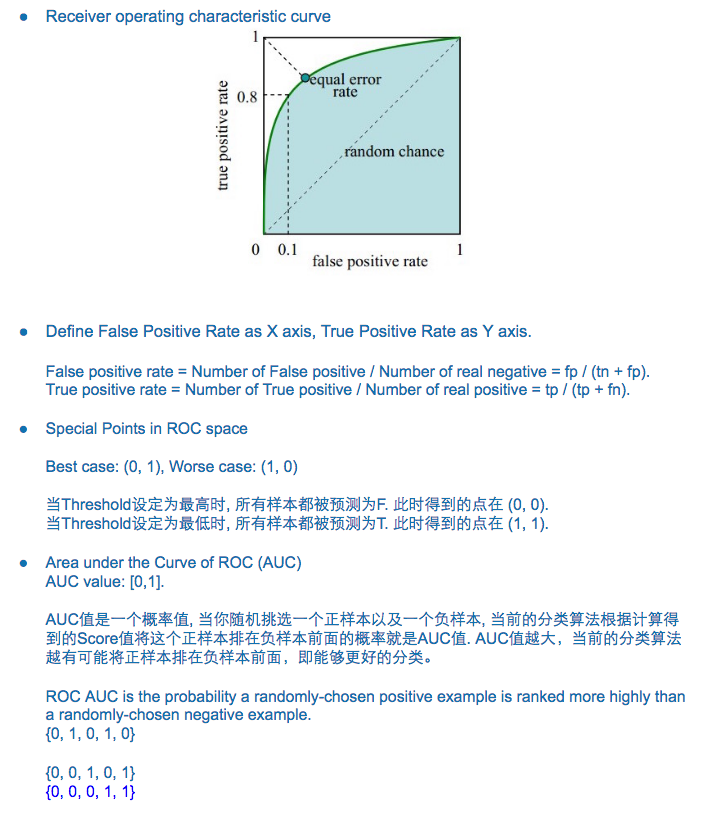
result = []

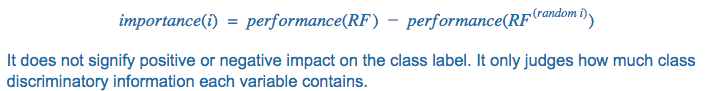
heapq.heapify(array)

for i in range(k):

result.append(heapq.heappop(array))

return result

1. What is the difference between KNN and K-Means?
   * KNN is a supervised learning algorithm - classification
   * KMeans is an unsupervised learning algorithm - clustering
2. What is cross validation?
   * Assess how your model result will generalize to another independent dataset
   * k-fold cross validation - explanation and example
   * model selection with CV - hyperparameter tuning (k in KNN, lambda in regularization)
   * CV can only validate your model selection, it cannot deal with overfitting
3. What is confusion matrix?
   * Precision: TP/(TP+FP)
   * Recall (True Positive Rate): TP/(TP+FN)
   * F1 Score: 2 / (1/p + 1/r)
   * Accuracy: (TP+TN)/(TP+TN+FP+FN)
   * precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances, while recall (also known as sensitivity) is the fraction of relevant instances that have been retrieved over the total amount of relevant instances.
4. We do we care about precision/recall?
   * Spam email - precision: we want to find out all the spams, when you think it’s spam, it better be the actual spam
   * Doctor identify illness - recall: we have to identify illness 有病的人一定要找出来 有病得治
   * Internet security, identify virus - recall has to be 100%, and then increase the precision, reduce FP rate
   * recommendation system - only cares about recall
5. What is ROC Curve/AUC?
   * FP: Type 1 error
   * FN: Type 2 error
6. What is Ensemble Learning？
   * We train multiple learners to solve the same problem
   * Ensemble can combine multiple hypotheses to form a better hypothesis
   * including boosting and bagging methods
7. What is Random Forest and its pros and cons?
   * Description of Random Forest
     + Suppose we have N data points and M features for every point
     + Everytime we take n data points (sampling with replacement) out from the entire data. Some of the sample data might be selected multiple times, while some might never get selected
     + We randomly pick k (k<M) features and use these k features to calculate a best decision tree
     + iterate the above, we get a group of completely different decision trees, and we combine them to get a random forest model, to do classification or prediction for new data
     + Random Forests train each tree independently, using a random sample of the data. This randomness helps to make the model more robust than a single decision tree, and less likely to overfit on the training data. There are typically two parameters in RF - number of trees and no. of features to be selected at each node.
   * Pros
     + Reduce overfitting - bagging method decrease variance by introducing randomness into the model framework; RF leverage bagging method (bagging for data, bagging for features) to solve overfitting problem in complete decision tree
     + Good at imbalanced data: multiple sampling, slice differently every time, original 1:10 could become 1:5, reduce the effect of imbalanced data, adding weight to reduce bias
     + Parallel implementation, faster training speed
   * Cons
     + Not easy to visually interpret
   * Others - Bagging
     + sampling with replacement
     + bagging = bootstrap aggregating
     + bagging vs. boosting: A weak learner is a classifier which is only slightly correlated with the true classification; after a weak learner is added, the data is reweighted - examples that are misclassified gain weight and examples that are classified correctly lose weight
     + RF vs. bagging: different at the feature sampling step
   * Others - feature importance value in RF



1. What is PCA?
2. What is tokenizing?
   * Tokenizing is a process of splitting a text into individual words or sequences of words (N-grams)
   * After doing document tokenization, we usually remove “stop words” from the result set. Stop words are words that don’t convey significant meaning, such as a, an, the, in.
3. What is TF-IDF and why we need it?
   * TF-IDF: Term Frequency - Inverse Document Frequency
     + is the product of two statistics: TF-IDF(wordA in docB) = TF(wordA in docB) \* IDF(wordA)
     + TF(wordA in docB) = count of wordA in docB / total words count in docB
     + IDF(wordA) = log(total number of documents in the corpus / number of documents where wordA appears + 1）
4. What is K-Means and what are some major characteristics?
   * K-Means is an unsupervised learning algorithm
5. What is curse of dimensionality and how to avoid it?
6. Difference between discriminative model and Generative model
   * Let's say you have input data x and you want to classify the data into labels y.
   * A generative model learns the joint probability distribution p(x,y)
   * a discriminative model learns the conditional probability distribution p(y|x) - which you should read as "the probability of y given x".

(x,y):

(1,0), (1,0), (2,0), (2, 1)

p(x,y) is

y=0 y=1

-----------

x=1 | 1/2 0

x=2 | 1/4 1/4

p(y|x) is

y=0 y=1

-----------

x=1 | 1 0

x=2 | 1/2 1/2

* + The distribution p(y|x) is the natural distribution for classifying a given example x into a class y, which is why algorithms that model this directly are called discriminative algorithms.
  + Generative algorithms model p(x,y), which can be transformed into p(y|x) by applying Bayes rule and then used for classification. However, the distribution p(x,y) can also be used for other purposes. For example, you could use p(x,y) to generate likely (x,y) pairs.
  + discriminative models generally outperform generative models in classification tasks.

**Part 2: Statistics**

1. What is p-value?

* The probability of obtaining as or more extreme results than the current observation, under the null hypothesis.
* When p-value is very small, it means that it would be almost impossible to see the current results if H0 was true. But we actually did, so H0 must be wrong.
* (Because a conditional statement is logically equivalent to its contrapositive.)
* P-value is used to determine the significance of results after a hypothesis test in statistics. P-value helps the readers to draw conclusions and is always between 0 and 1.
  + P- Value > 0.05 denotes weak evidence against the null hypothesis which means the null hypothesis cannot be rejected.
  + P-value <= 0.05 denotes strong evidence against the null hypothesis which means the null hypothesis can be rejected.
  + P-value=0.05 is the marginal value indicating it is possible to go either way.

1. What is Hypothesis Testing?

* Hypothesis Testing is the procedure to determine whether the observed/estimated difference/relation is a statistically meaningful difference/relation.
* The idea is to use the observed data/evidence to challenge the claim in H0.
* If the evidence is strong enough
  + that is, the existence of the observed evidence (or stronger evidence) would be almost impossible if H0 were true
  + but we actually see it, so we reject H0, and conclude H1 is true
* Otherwise, there’s no enough evidence to reject H0, so we hold no conclusion on whether H0 is true

1. What is significance level?

The probability of rejecting H0, when H0 is true. Must be pre-defined. This is Type I error.

1. What is a confidence interval and how do you interpret it?
   * For example, 95% confidence interval is an interval that when constructed for a set of samples each sampled in the same way, the constructed intervals include the true mean 95% of the time.
   * if confidence intervals are constructed using a given confidence level in an infinite number of independent experiments, the proportion of those intervals that contain the true value of the parameter will match the confidence level.
2. What is Type I/Type II error?

* Type I error: When H0 is true, but got rejected
* Type II error: When H0 is not true/H1 is true, but H0 doesn’t get rejected
* Power: The probability of rejecting H0, when H0 is not true/H1 is true;

Power = 1-Type II error rate

1. With the dataset fixed, if you increase the significance level, what will happen to Type I and Type II error rate? Why?

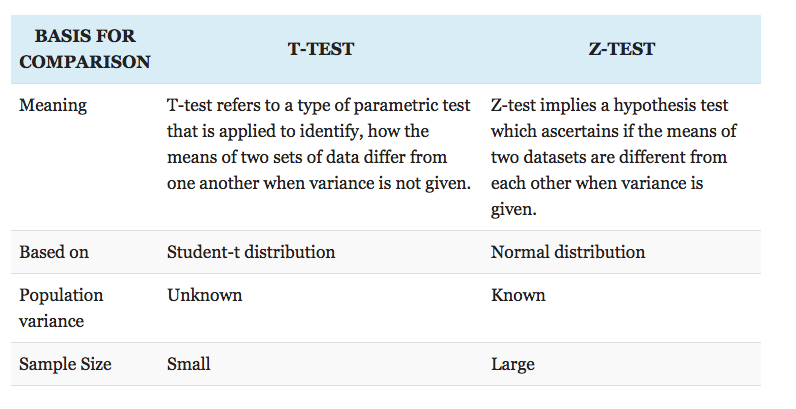
* Type I error will increase (this is by definition), and Type II error will decrease.
* Increasing significance level inevitably increases the chance of making rejections. Among these rejected H0’s, there could be true H0’s as well as true H1’s, that is, a higher chance of rejecting H0 when H0 is not true/H1 is true.

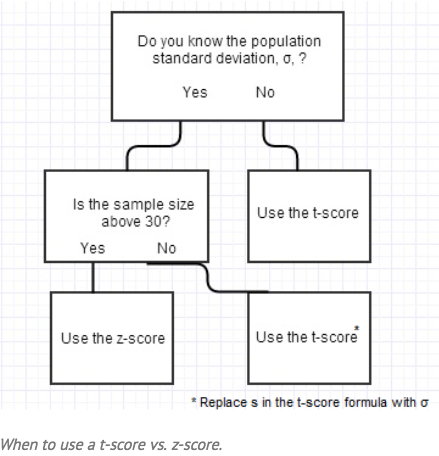
→ power = (1-b) increases

1. How do you trade off between Type I and Type II error?

* In theory, it depends on what your H0 and H1 are, and how you value them. In other words, which type of error is more serious. e.g. convicting an innocent man (type I) vs. releasing a criminal (type II).
* In practical, Type I error is often associated with a larger risk than Type II error is. That’s why we usually fix a small significance level, then try to maximize the testing power.

1. What is T-test? Z-test? F-test? ANOVA? the difference? When should we use them?





A t-test is used for testing the mean of one population against a standard, or comparing the means of two populations if you don’t know the populations’ standard deviation and when you have a limited sample (n < 30).

*Example:Measuring the average diameter of shafts from a certain machine when you have a small sample.*

A z-test is used for testing the mean of a population versus a standard, or comparing the means of two populations, with large (n ≥ 30) samples whether you know the population standard deviation or not.   
*Example:Comparing the average engineering salaries of men versus women.  
Example: Comparing the fraction defectives from 2 production lines.*  
An F-test is used to compare 2 populations’ variances. The samples can be any size. It is the basis of ANOVA.  
*Example: Comparing the variability of bolt diameters from two machines.*  
  
Matched pair test is used to compare the means before and after something is done to the samples. A t-test is often used because the samples are often small. However, a z-test is used when the samples are large. The variable is the difference between the before and after measurements.  
*Example: The average weight of subjects before and after following a diet for 6 weeks*

1. What is Central Limit Theorem and why is it important
   * Central Limit Theorem means that the sampling distribution of the sample means approaches a normal distribution as the sample size gets larger — no matter what the shape of the population distribution. As you take more samples, especially large ones, your graph of the sample means will look more like a normal distribution.
   * This theorem enables you to measure how much the means of various samples vary without having to use other sample means as a comparison.
   * The law of large numbers is another different theorem from statistics. It is simpler in that it states that as the size of a sample is increased, the more accurate of an estimate the sample mean will be of the population mean.
2. What is maximum likelihood estimation? Could there be any case where it doesn’t exist?  
   A method for parameter optimization (fitting a model). We choose parameters so as to maximize the likelihood function (how likely the outcome would happen given the current data and our model).  
   maximum likelihood estimation (MLE) is a method of estimating the parameters of a statistical model given observations, by finding the parameter values that maximize the likelihood of making the observations given the parameters. MLE can be seen as a special case of the maximum a posteriori estimation (MAP) that assumes a uniform prior distribution of the parameters, or as a variant of the MAP that ignores the prior and which therefore is unregularized.  
   for gaussian mixtures, non parametric models, it doesn’t exist

# **Machine Learning Algorithms Pros and Cons**

### **Naive Bayes**

* super simple, just doing a bunch of counts.
* if the NB conditional independence assumption actually holds, a Naive Bayes classifier will converge quicker than discriminative models like logistic regression, so you need less training data. And even if the NB assumption doesn't hold, a NB classifier still often performs surprisingly well in practice.
* a good bet if you want to do some kind of semi-supervised learning, or want something embarrassingly simple that performs pretty well.
* no distribution requirements,
* good for few categories variables
* compute the multiplication of independent distributions
* suffer multicollinearity

### **Logistic Regression**

Logistic regression is still the most widely used

[Learn More](https://www.hackingnote.com/en/machine-learning/logistic-regression)

* a pretty well-behaved classification algorithm that can be trained as long as you expect your features to be roughly linear and the problem to be linearly separable.
* can do some feature engineering to turn most non-linear features into linear pretty easily.
* it is also pretty robust to noise and you can avoid overfitting and even do feature selection by using l2 or l1 regularization.
* logistic regression can also be used in Big Data scenarios since it is pretty efficient and can be distributed using, for example, ADMM (see logreg).
* **the output can be interpreted as a probability: you can use it for ranking instead of classification.**
* run a simple l2-regularized LR to come up with a baseline
* no distribution requirement
* perform well with few categories categorical variables
* compute the logistic distribution
* good for few categories variables
* easy to interpret
* compute CI
* suffer multicollinearity
* lots of ways to regularize your model
* no need to worry about features being correlated, like in Naive Bayes.
* easily update the model to take in new data (using an online gradient descent method)
* use it if you want a probabilistic framework (e.g., to easily adjust classification thresholds, to say when you're unsure, or to get confidence intervals) or if you expect to receive more training data in the future that you want to be able to quickly incorporate into your model.

Lasso

* no distribution requirement
* compute L1 loss
* variable selection
* suffer multicollinearity

Ridge

* no distribution requirement
* compute L2 loss
* no variable selection
* not suffer multicollinearity

When not to use

* if the variables are normally distributed and the categorical variables all have 5+ categories: use Linear discriminant analysis
* if the correlations are mostly nonlinear: use SVM
* if sparsity and multicollinearity are a concern: Adaptive Lasso with Ridge(weights) + Lasso

### **Linear discriminant analysis**

LDA: Linear discriminant analysis, not latent Dirichlet allocation

* require normal distrbution
* not good for few categories variables
* compute the addition of Multivariate distribution
* compute CI
* suffer multicollinearity

### **Support Vector Machines**

SVM vs LR:

* Support Vector Machines (SVMs) use a different loss function (Hinge) from LR.
* They are also interpreted differently (maximum-margin).
* However, in practice, an SVM with a linear kernel is not very different from a Logistic Regression (If you are curious, you can see how Andrew Ng derives SVMs from Logistic Regression in his Coursera Machine Learning Course).
* The main reason you would want to use an SVM instead of a Logistic Regression is because your problem might not be linearly separable. In that case, you will have to use an SVM with a non linear kernel (e.g. RBF).
* The truth is that a Logistic Regression can also be used with a different kernel, but at that point you might be better off going for SVMs for practical reasons.
* Another related reason to use SVMs is if you are in a highly dimensional space. For example, SVMs have been reported to work better for text classification.
* High accuracy, nice theoretical guarantees regarding overfitting
* with an appropriate kernel they can work well even if you're data isn't linearly separable in the base feature space.
* Especially popular in text classification problems where very high-dimensional spaces are the norm.
* no distribution requirement
* compute hinge loss
* flexible selection of kernels for nonlinear correlation
* not suffer multicollinearity
* hard to interpret

Cons:

* can be painfully inefficient to train. not recommend for any problem that have many training examples. not recommend SVMs for most "industry scale" applications. Anything beyond a toy/lab problem might be better approached with a different algorithm. Memory-intensive and kind of annoying to run and tune, though, so I think random forests are starting to steal the crown.

### **Decision Tree**

* Easy to interpret and explain
* Non-parametric, so you don't have to worry about outliers or whether the data is linearly separable (e.g., decision trees easily take care of cases where you have class A at the low end of some feature x, class B in the mid-range of feature x, and A again at the high end). Their main disadvantage is that they easily overfit, but that's where ensemble methods like random forests (or boosted trees) come in.
* Plus, random forests are often the winner for lots of problems in classification (usually slightly ahead of SVMs, I believe), they're fast and scalable, and you don't have to worry about tuning a bunch of parameters like you do with SVMs, so they seem to be quite popular these days.
* no distribution requirement
* heuristic
* good for few categories variables
* not suffer multicollinearity (by choosing one of them)

Bagging, boosting, ensemble methods generally outperform single algorithm.

Tree Ensembles: Random Forests and Gradient Boosted Trees.

Tree Ensembles vs LR.

* they do not expect linear features or even features that interact linearly. Something I did not mention in LR is that it can hardly handle categorical (binary) features. Tree Ensembles, because they are nothing more than a bunch of Decision Trees combined, can handle this very well. The other main advantage is that, because of how they are constructed (using bagging or boosting) these algorithms handle very well high dimensional spaces as well as large number of training examples.
* both are fast and scalable, random forests tend to beat out logistic regression in terms of accuracy, but logistic regression can be updated online and gives you useful probabilities.

### **Random Forests**

Random Forests train each tree independently, using a random sample of the data. This randomness helps to make the model more robust than a single decision tree, and less likely to overfit on the training data. There are typically two parameters in RF - number of trees and no. of features to be selected at each node.

* RF is good for parallel or distributed computing.
* Almost always have lower classification error and better f-scores than decision trees.
* Almost always perform as well as or better than SVMs, but are far easier for humans to understand.
* Deal really well with uneven data sets that have missing variables.
* Give you a really good idea of which features in your data set are the most important for free.
* Generally train faster than SVMs (though this obviously depends on your implementation).

### **Gradient Boosted Decision Trees**

GBTs build trees one at a time, where each new tree helps to correct errors made by previously trained tree. With each tree added, the model becomes even more expressive. There are typically three parameters - number of trees, depth of trees and learning rate, and the each tree built is generally shallow.

* prone to overfitting
* GBDTs will usually perform better than RF, but they are harder to get right. More concretely, GBDTs have more hyper-parameters to tune and are also more prone to overfitting. RFs can almost work "out of the box" and that is one reason why they are very popular.
* GBDT training generally takes longer because of the fact that trees are built sequentially

### **Neural Network**

Pros

* good to model the non-linear data with large number of input features
* widely used in industry
* many open source implementations

Cons

* NNs are useable only for numerical inputs, vectors with constant number of values, and datasets with non-missing data.
* The classification boundaries are hard to understand intuitively and ANNs are computationally expensive.
* black box, makes them difficult to work with, it’s like trying interrogate the human unconscious for the reasons behind our conscious actions.
* difficult to train: the training outcome can be nondeterministic and depend crucially on the choice of initial parameters
* It makes them difficult to troubleshoot when they don't work as you expect, and when they do work, you will never really feel confident that they will generalize well to data not included in your training set because, fundamentally, you don't understand how your network is solving the problem
* multi-layer neural networks are usually hard to train, and require tuning lots of parameters
* Neural networks are not probabilistic, unlike their more statistical or Bayesian counterparts. A neural network might give you a continuous number as its output (e.g. a score) but translating that into a probability is often difficult. Approaches with stronger theoretical foundations usually give you those probabilities directly.

### **Deep Learning**

* not a general-purpose technique for classification.
* good in image classification, video, audio, text.

## **Summary**

Factors to Consider

* Number of training examples, (how large is your training set?)
  + If your training set is small, high bias/low variance classifiers (e.g., Naive Bayes) have an advantage over low bias/high variance classifiers (e.g., kNN or logistic regression), since the latter will overfit. But low bias/high variance classifiers start to win out as your training set grows (they have lower asymptotic error), since high bias classifiers aren't powerful enough to provide accurate models. You can also think of this as a generative model vs. discriminative model distinction.
* Dimensionality of the feature space
* Do I expect the problem to be linearly separable?
* Are features independent?
* Are features expected to linearly dependent with the target variable?
* Is overfitting expected to be a problem?
* What are the system's requirement in terms of speed/performance/memory usage...?
* Does it require variables to be normally distributed?
* Does it suffer multicollinearity issue?
* Dose it do as well with categorical variables as continuous variables?
* Does it calculate CI without CV?
* Does it conduct variables selection without stepwise?
* Does it apply to sparse data?

Start with something simple like Logistic Regression to set a baseline and only make it more complicated if you need to. At that point, tree ensembles, and in particular Random Forests since they are easy to tune, might be the right way to go. If you feel there is still room for improvement, try GBDT or get even fancier and go for Deep Learning.

### 

## **Machine Learning Interview Questions: Algorithms/Theory**

### <https://www.springboard.com/blog/machine-learning-interview-questions/>

These algorithms questions will test your grasp of the theory behind machine learning.

Q1- What’s the trade-off between bias and variance?

More reading: [Bias-Variance Tradeoff (Wikipedia)](https://en.wikipedia.org/wiki/Bias-variance_tradeoff)

Bias

* error due to erroneous or overly simplistic assumptions
* leads to underfitting, making it hard for it to have high predictive accuracy and for you to generalize your knowledge from the training set to the test set.

Variance

* error due to too much complexity
* leads to overfitting, the algorithm being highly sensitive to high degrees of variation in your training data
* carrying too much noise from your training data for your model to be very useful for your test data

The bias-variance decomposition essentially decomposes

* learning error = bias + variance + irreducible error due to noise in the data
* if you make the model more complex and add more variables, you’ll lose bias but gain some variance — in order to get the optimally reduced amount of error, you’ll have to tradeoff bias and variance. You don’t want either high bias or high variance in your model.

Q2- What is the difference between supervised and unsupervised machine learning?

More reading: [What is the difference between supervised and unsupervised machine learning? (Quora)](https://www.quora.com/What-is-the-difference-between-supervised-and-unsupervised-learning-algorithms)

Supervised learning requires training labeled data. For example, in order to do classification (a supervised learning task), you’ll need to first label the data you’ll use to train the model to classify data into your labeled groups. Unsupervised learning, in contrast, does not require labeling data explicitly.

Q3- How is KNN different from k-means clustering?

More reading: [How is the k-nearest neighbor algorithm different from k-means clustering? (Quora)](https://www.quora.com/How-is-the-k-nearest-neighbor-algorithm-different-from-k-means-clustering)

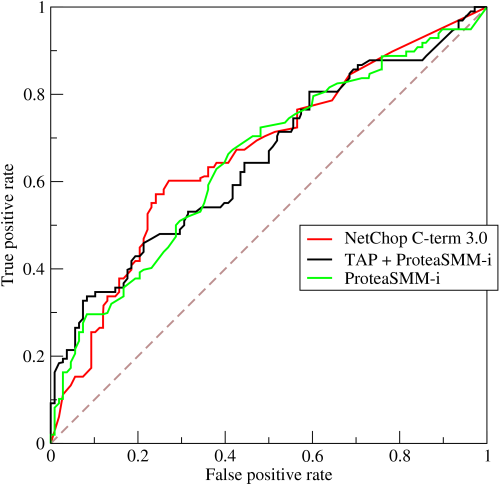
K-Nearest Neighbors is a supervised classification algorithm, while k-means clustering is an unsupervised clustering algorithm. While the mechanisms may seem similar at first, what this really means is that in order for K-Nearest Neighbors to work, you need labeled data you want to classify an unlabeled point into (thus the nearest neighbor part). K-means clustering requires only a set of unlabeled points and a threshold: the algorithm will take unlabeled points and gradually learn how to cluster them into groups by computing the mean of the distance between different points.

The critical difference here is that KNN needs labeled points and is thus supervised learning, while k-means doesn’t — and is thus unsupervised learning.

Q4- Explain how a ROC curve works.

More reading: [Receiver operating characteristic (Wikipedia)](https://en.wikipedia.org/wiki/Receiver_operating_characteristic)

The ROC curve is a graphical representation of the contrast between true positive rates and the false positive rate at various thresholds. It’s often used as a proxy for the trade-off between the sensitivity of the model (true positives) vs the fall-out or the probability it will trigger a false alarm (false positives).



Q5- Define precision and recall.

More reading: [Precision and recall (Wikipedia)](https://en.wikipedia.org/wiki/Precision_and_recall)

Recall is also known as the true positive rate: the amount of positives your model claims compared to the actual number of positives there are throughout the data. Precision is also known as the positive predictive value, and it is a measure of the amount of accurate positives your model claims compared to the number of positives it actually claims. It can be easier to think of recall and precision in the context of a case where you’ve predicted that there were 10 apples and 5 oranges in a case of 10 apples. You’d have perfect recall (there are actually 10 apples, and you predicted there would be 10) but 66.7% precision because out of the 15 events you predicted, only 10 (the apples) are correct.

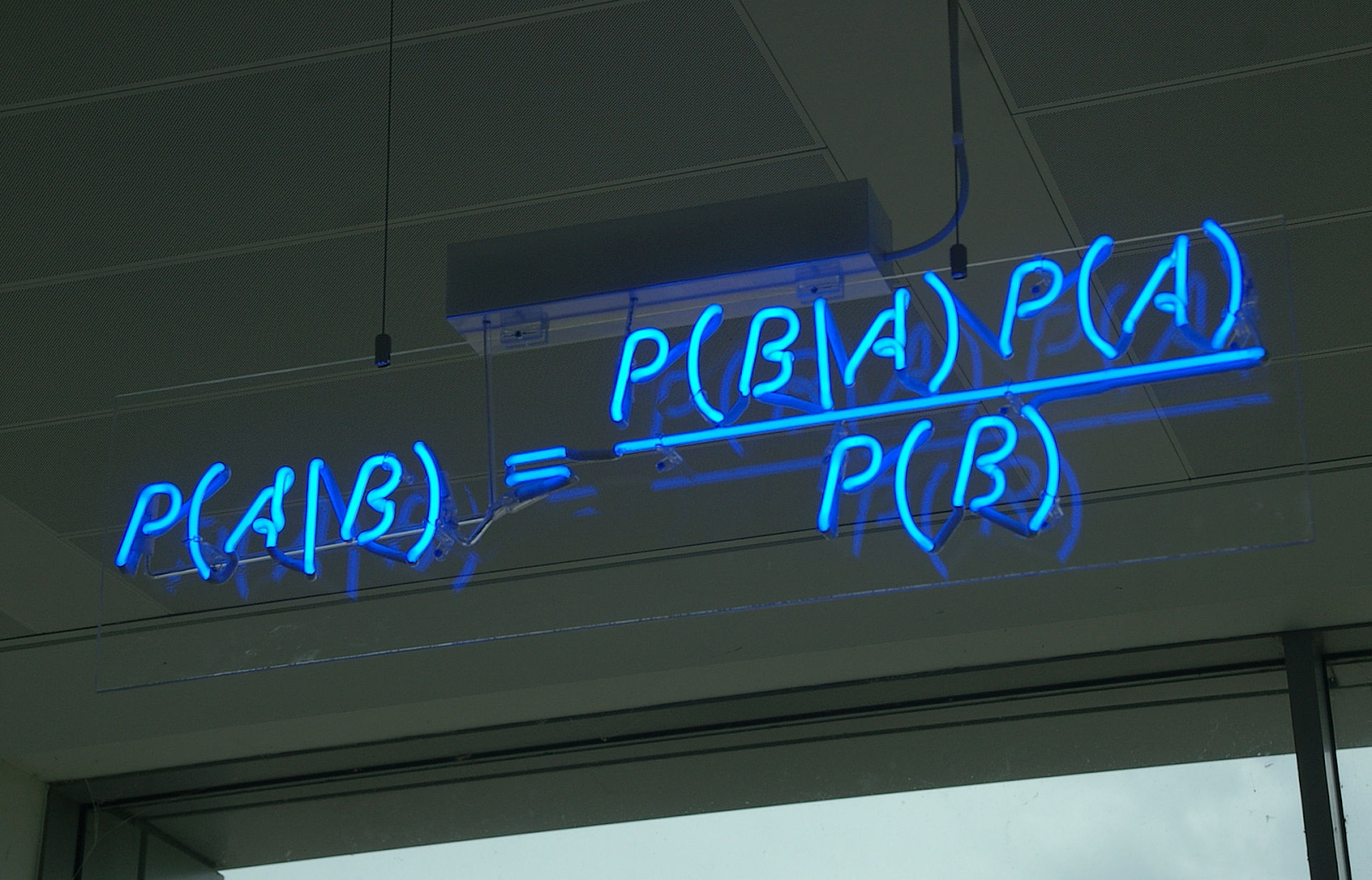
Q6- What is Bayes’ Theorem? How is it useful in a machine learning context?

More reading: [An Intuitive (and Short) Explanation of Bayes’ Theorem (BetterExplained)](https://betterexplained.com/articles/an-intuitive-and-short-explanation-of-bayes-theorem/)

Bayes’ Theorem gives you the posterior probability of an event given what is known as prior knowledge.

Mathematically, it’s expressed as the true positive rate of a condition sample divided by the sum of the false positive rate of the population and the true positive rate of a condition. Say you had a 60% chance of actually having the flu after a flu test, but out of people who had the flu, the test will be false 50% of the time, and the overall population only has a 5% chance of having the flu. Would you actually have a 60% chance of having the flu after having a positive test?

Bayes’ Theorem says no. It says that you have a (.6 \* 0.05) (True Positive Rate of a Condition Sample) / (.6\*0.05)(True Positive Rate of a Condition Sample) + (.5\*0.95) (False Positive Rate of a Population) = 0.0594 or 5.94% chance of getting a flu.



Bayes’ Theorem is the basis behind a branch of machine learning that most notably includes the Naive Bayes classifier. That’s something important to consider when you’re faced with machine learning interview questions.

Q7- Why is “Naive” Bayes naive?

More reading: [Why is “naive Bayes” naive? (Quora)](https://www.quora.com/Why-is-naive-Bayes-naive?share=1)

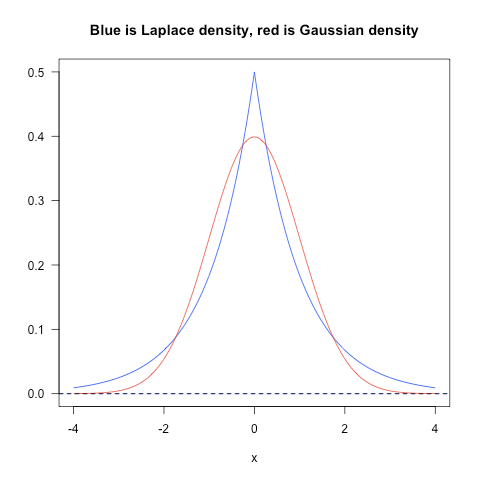
Despite its practical applications, especially in text mining, Naive Bayes is considered “Naive” because it makes an assumption that is virtually impossible to see in real-life data: the conditional probability is calculated as the pure product of the individual probabilities of components. This implies the absolute independence of features — a condition probably never met in real life.

As a Quora commenter put it whimsically, a Naive Bayes classifier that figured out that you liked pickles and ice cream would probably naively recommend you a pickle ice cream.

Q8- Explain the difference between L1 and L2 regularization.

More reading: [What is the difference between L1 and L2 regularization? (Quora)](https://www.quora.com/What-is-the-difference-between-L1-and-L2-regularization)

L2 regularization tends to spread error among all the terms, while L1 is more binary/sparse, with many variables either being assigned a 1 or 0 in weighting. L1 corresponds to setting a Laplacean prior on the terms, while L2 corresponds to a Gaussian prior.



Q9- What’s your favorite algorithm, and can you explain it to me in less than a minute?

This type of question tests your understanding of how to communicate complex and technical nuances with poise and the ability to summarize quickly and efficiently. Make sure you have a choice and make sure you can explain different algorithms so simply and effectively that a five-year-old could grasp the basics!

Q10- What’s the difference between Type I and Type II error?

More reading: [Type I and type II errors (Wikipedia)](https://en.wikipedia.org/wiki/Type_I_and_type_II_errors)

Don’t think that this is a trick question! Many machine learning interview questions will be an attempt to lob basic questions at you just to make sure you’re on top of your game and you’ve prepared all of your bases.

Type I error is a false positive, while Type II error is a false negative. Briefly stated, Type I error means claiming something has happened when it hasn’t, while Type II error means that you claim nothing is happening when in fact something is.

A clever way to think about this is to think of Type I error as telling a man he is pregnant, while Type II error means you tell a pregnant woman she isn’t carrying a baby.

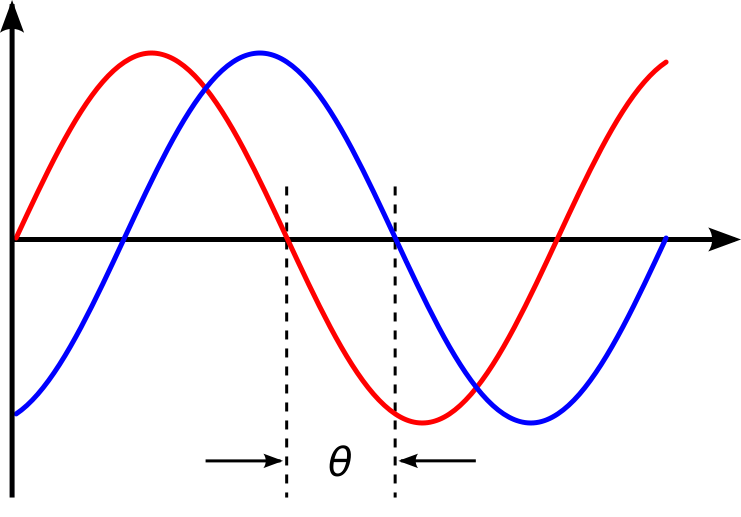
Q11- What’s a Fourier transform?

More reading: [Fourier transform (Wikipedia)](https://en.wikipedia.org/wiki/Fourier_transform)

A Fourier transform is a generic method to decompose generic functions into a superposition of symmetric functions. Or as this [more intuitive tutorial](https://betterexplained.com/articles/an-interactive-guide-to-the-fourier-transform/) puts it, given a smoothie, it’s how we find the recipe. The Fourier transform finds the set of cycle speeds, amplitudes and phases to match any time signal. A Fourier transform converts a signal from time to frequency domain — it’s a very common way to extract features from audio signals or other time series such as sensor data.

Q12- What’s the difference between probability and likelihood?

More reading: [What is the difference between “likelihood” and “probability”? (Cross Validated)](https://stats.stackexchange.com/questions/2641/what-is-the-difference-between-likelihood-and-probability#2647)

[](https://stats.stackexchange.com/questions/2641/what-is-the-difference-between-likelihood-and-probability#2647)

Q13- What is deep learning, and how does it contrast with other machine learning algorithms?

More reading: [Deep learning (Wikipedia)](https://en.wikipedia.org/wiki/Deep_learning)

Deep learning is a subset of machine learning that is concerned with neural networks: how to use backpropagation and certain principles from neuroscience to more accurately model large sets of unlabelled or semi-structured data. In that sense, deep learning represents an unsupervised learning algorithm that learns representations of data through the use of neural nets.

Q14- What’s the difference between a generative and discriminative model?

More reading: [What is the difference between a Generative and Discriminative Algorithm? (Stack Overflow)](https://stackoverflow.com/questions/879432/what-is-the-difference-between-a-generative-and-discriminative-algorithm)

A generative model will learn categories of data while a discriminative model will simply learn the distinction between different categories of data. Discriminative models will generally outperform generative models on classification tasks.

Q15- What cross-validation technique would you use on a time series dataset?

More reading: [Using k-fold cross-validation for time-series model selection (CrossValidated)](https://stats.stackexchange.com/questions/14099/using-k-fold-cross-validation-for-time-series-model-selection)

Instead of using standard k-folds cross-validation, you have to pay attention to the fact that a time series is not randomly distributed data — it is inherently ordered by chronological order. If a pattern emerges in later time periods for example, your model may still pick up on it even if that effect doesn’t hold in earlier years!

You’ll want to do something like forward chaining where you’ll be able to 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]
* fold 5 : training [1 2 3 4 5], test [6]

Q16- How is a decision tree pruned?

More reading: [Pruning (decision trees)](https://en.wikipedia.org/wiki/Pruning_%28decision_trees%29)

Pruning is what happens in decision trees when branches that have weak predictive power are removed in order to reduce the complexity of the model and increase the predictive accuracy of a decision tree model. Pruning can happen bottom-up and top-down, with approaches such as reduced error pruning and cost complexity pruning.

Reduced error pruning is perhaps the simplest version: replace each node. If it doesn’t decrease predictive accuracy, keep it pruned. While simple, this heuristic actually comes pretty close to an approach that would optimize for maximum accuracy.

Q17- Which is more important to you– model accuracy, or model performance?

More reading: [Accuracy paradox (Wikipedia)](https://en.wikipedia.org/wiki/Accuracy_paradox)

This question tests your grasp of the nuances of machine learning model performance! Machine learning interview questions often look towards the details. There are models with higher accuracy that can perform worse in predictive power — how does that make sense?

Well, it has everything to do with how model accuracy is only a subset of model performance, and at that, a sometimes misleading one. For example, if you wanted to detect fraud in a massive dataset with a sample of millions, a more accurate model would most likely predict no fraud at all if only a vast minority of cases were fraud. However, this would be useless for a predictive model — a model designed to find fraud that asserted there was no fraud at all! Questions like this help you demonstrate that you understand model accuracy isn’t the be-all and end-all of model performance.

Q18- What’s the F1 score? How would you use it?

More reading: [F1 score (Wikipedia)](https://en.wikipedia.org/wiki/F1_score)

The F1 score is a measure of a model’s performance. It is a weighted average of the precision and recall of a model, with results tending to 1 being the best, and those tending to 0 being the worst. You would use it in classification tests where true negatives don’t matter much.

Q19- How would you handle an imbalanced dataset?

More reading: [8 Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset (Machine Learning Mastery)](http://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/)

An imbalanced dataset is when you have, for example, a classification test and 90% of the data is in one class. That leads to problems: an accuracy of 90% can be skewed if you have no predictive power on the other category of data! Here are a few tactics to get over the hump:

1- Collect more data to even the imbalances in the dataset.

2- Resample the dataset to correct for imbalances.

3- Try a different algorithm altogether on your dataset.

What’s important here is that you have a keen sense for what damage an unbalanced dataset can cause, and how to balance that.

Q20- When should you use classification over regression?

More reading: [Regression vs Classification (Math StackExchange)](https://math.stackexchange.com/questions/141381/regression-vs-classification)

Classification produces discrete values and dataset to strict categories, while regression gives you continuous results that allow you to better distinguish differences between individual points. You would use classification over regression if you wanted your results to reflect the belongingness of data points in your dataset to certain explicit categories (ex: If you wanted to know whether a name was male or female rather than just how correlated they were with male and female names.)

Q21- Name an example where ensemble techniques might be useful.

More reading: [Ensemble learning (Wikipedia)](https://en.wikipedia.org/wiki/Ensemble_learning)

Ensemble techniques use a combination of learning algorithms to optimize better predictive performance. They typically reduce overfitting in models and make the model more robust (unlikely to be influenced by small changes in the training data).

You could list some examples of ensemble methods, from bagging to boosting to a “bucket of models” method and demonstrate how they could increase predictive power.

Q22- How do you ensure you’re not overfitting with a model?

More reading: [How can I avoid overfitting? (Quora)](https://www.quora.com/How-can-I-avoid-overfitting)

This is a simple restatement of a fundamental problem in machine learning: the possibility of overfitting training data and carrying the noise of that data through to the test set, thereby providing inaccurate generalizations.

There are three main methods to avoid overfitting:

1- Keep the model simpler: reduce variance by taking into account fewer variables and parameters, thereby removing some of the noise in the training data.

2- Use cross-validation techniques such as k-folds cross-validation.

3- Use regularization techniques such as LASSO that penalize certain model parameters if they’re likely to cause overfitting.

Q23- What evaluation approaches would you work to gauge the effectiveness of a machine learning model?

More reading: [How to Evaluate Machine Learning Algorithms (Machine Learning Mastery)](http://machinelearningmastery.com/how-to-evaluate-machine-learning-algorithms/)

You would first split the dataset into training and test sets, or perhaps use cross-validation techniques to further segment the dataset into composite sets of training and test sets within the data. You should then implement a choice selection of performance metrics: here is a fairly [comprehensive list](http://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/). You could use measures such as the F1 score, the accuracy, and the confusion matrix. What’s important here is to demonstrate that you understand the nuances of how a model is measured and how to choose the right performance measures for the right situations.

Q24- How would you evaluate a logistic regression model?

More reading: [Evaluating a logistic regression (CrossValidated)](https://stats.stackexchange.com/questions/71517/evaluating-a-logistic-regression#71522)

A subsection of the question above. You have to demonstrate an understanding of what the typical goals of a logistic regression are (classification, prediction etc.) and bring up a few examples and use cases.

Q25- What’s the “kernel trick” and how is it useful?

More reading: [Kernel method (Wikipedia)](https://en.wikipedia.org/wiki/Kernel_method)

The Kernel trick involves kernel functions that can enable in higher-dimension spaces without explicitly calculating the coordinates of points within that dimension: instead, kernel functions compute the inner products between the images of all pairs of data in a feature space. This allows them the very useful attribute of calculating the coordinates of higher dimensions while being computationally cheaper than the explicit calculation of said coordinates. Many algorithms can be expressed in terms of inner products. Using the kernel trick enables us effectively run algorithms in a high-dimensional space with lower-dimensional data.

### Machine Learning Interview Questions: Programming

These machine learning interview questions test your knowledge of programming principles you need to implement machine learning principles in practice. Machine learning interview questions tend to be technical questions that test your logic and programming skills: this section focuses more on the latter.

Q26- How do you handle missing or corrupted data in a dataset?

More reading: [Handling missing data (O’Reilly)](https://www.oreilly.com/learning/handling-missing-data)

You could find missing/corrupted data in a dataset and either drop those rows or columns, or decide to replace them with another value.

In Pandas, there are two very useful methods: isnull() and dropna() that will help you find columns of data with missing or corrupted data and drop those values. If you want to fill the invalid values with a placeholder value (for example, 0), you could use the fillna() method.

Q27- Do you have experience with Spark or big data tools for machine learning?

More reading: [50 Top Open Source Tools for Big Data (Datamation)](http://www.datamation.com/data-center/50-top-open-source-tools-for-big-data-1.html)

You’ll want to get familiar with the meaning of big data for different companies and the different tools they’ll want. Spark is the big data tool most in demand now, able to handle immense datasets with speed. Be honest if you don’t have experience with the tools demanded, but also take a look at job descriptions and see what tools pop up: you’ll want to invest in familiarizing yourself with them.

Q28- Pick an algorithm. Write the psuedo-code for a parallel implementation.

More reading: [Writing pseudocode for parallel programming (Stack Overflow)](https://stackoverflow.com/questions/5583257/writing-pseudocode-for-parallel-programming)

This kind of question demonstrates your ability to think in parallelism and how you could handle concurrency in programming implementations dealing with big data. Take a look at pseudocode frameworks such as [Peril-L](http://www.eng.utah.edu/~cs4960-01/lecture4.pdf) and visualization tools such as [Web Sequence Diagrams](https://www.websequencediagrams.com/) to help you demonstrate your ability to write code that reflects parallelism.

Q29- What are some differences between a linked list and an array?

More reading: [Array versus linked list (Stack Overflow)](https://stackoverflow.com/questions/166884/array-versus-linked-list#167016)

An array is an ordered collection of objects. A linked list is a series of objects with pointers that direct how to process them sequentially. An array assumes that every element has the same size, unlike the linked list. A linked list can more easily grow organically: an array has to be pre-defined or re-defined for organic growth. Shuffling a linked list involves changing which points direct where — meanwhile, shuffling an array is more complex and takes more memory.

Q30- Describe a hash table.

More reading: [Hash table (Wikipedia)](https://en.wikipedia.org/wiki/Hash_table)

A hash table is a data structure that produces an associative array. A key is mapped to certain values through the use of a hash function. They are often used for tasks such as database indexing.



Q31- Which data visualization libraries do you use? What are your thoughts on the best data visualization tools?

More reading: [31 Free Data Visualization Tools (Springboard)](https://www.springboard.com/blog/31-free-data-visualization-tools/)

What’s important here is to define your views on how to properly visualize data and your personal preferences when it comes to tools. Popular tools include R’s ggplot, Python’s seaborn and matplotlib, and tools such as Plot.ly and Tableau.

### Machine Learning Interview Questions: Company/Industry Specific

These machine learning interview questions deal with how to implement your general machine learning knowledge to a specific company’s requirements. You’ll be asked to create case studies and extend your knowledge of the company and industry you’re applying for with your machine learning skills.

Q32- How would you implement a recommendation system for our company’s users?

More reading: [How to Implement A Recommendation System? (Stack Overflow)](https://stackoverflow.com/questions/6302184/how-to-implement-a-recommendation-system#6302223)

A lot of machine learning interview questions of this type will involve implementation of machine learning models to a company’s problems. You’ll have to research the company and its industry in-depth, especially the revenue drivers the company has, and the types of users the company takes on in the context of the industry it’s in.

Q33- How can we use your machine learning skills to generate revenue?

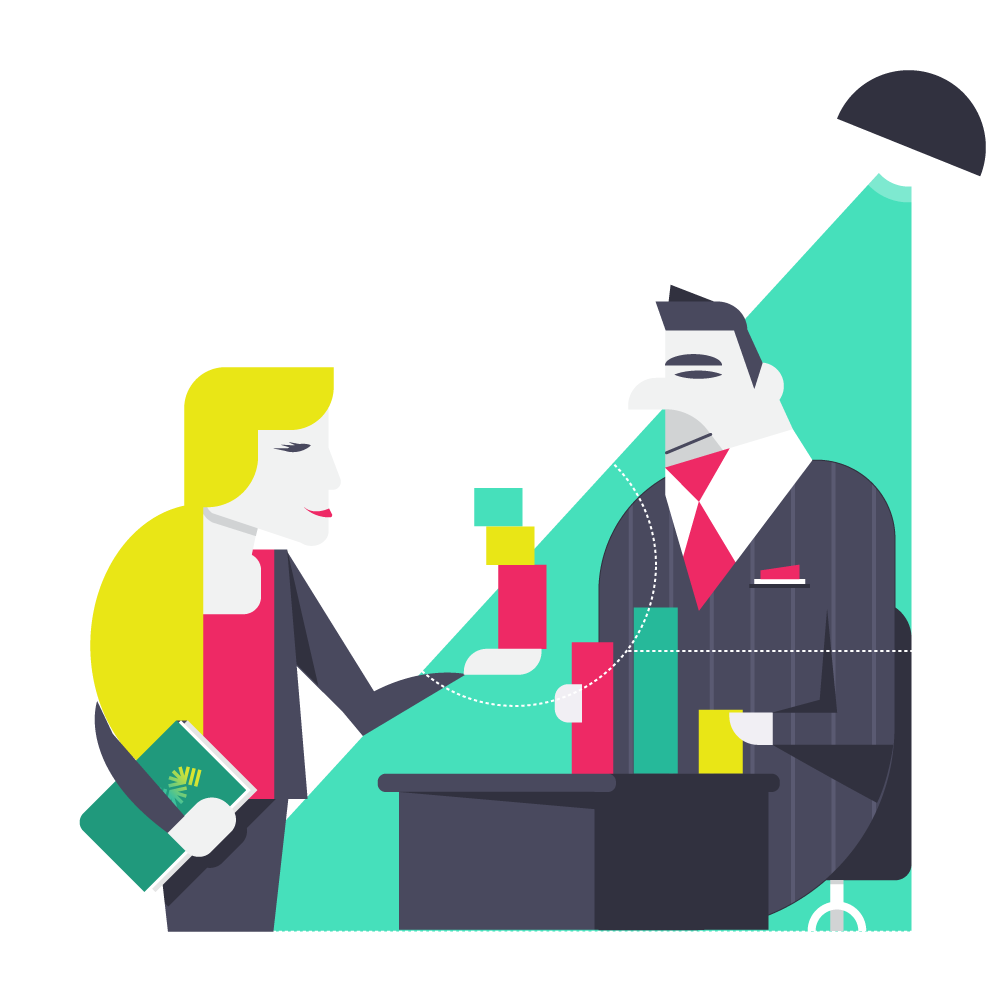
More reading: [Startup Metrics for Startups (500 Startups)](http://www.slideshare.net/dmc500hats/startup-metrics-for-pirates-long-version)

This is a tricky question. The ideal answer would demonstrate knowledge of what drives the business and how your skills could relate. For example, if you were interviewing for music-streaming startup Spotify, you could remark that your skills at developing a better recommendation model would increase user retention, which would then increase revenue in the long run.

The startup metrics Slideshare linked above will help you understand exactly what performance indicators are important for startups and tech companies as they think about revenue and growth.

Q34- What do you think of our current data process?

More reading: [The Data Science Process Email Course – Springboard](https://www.springboard.com/resources/data-science-process/)

[](https://www.springboard.com/resources/data-science-process/)

This kind of question requires you to listen carefully and impart feedback in a manner that is constructive and insightful. Your interviewer is trying to gauge if you’d be a valuable member of their team and whether you grasp the nuances of why certain things are set the way they are in the company’s data process based on company- or industry-specific conditions. They’re trying to see if you can be an intellectual peer. Act accordingly.

### Machine Learning Interview Questions: General Machine Learning Interest

This series of machine learning interview questions attempts to gauge your passion and interest in machine learning. The right answers will serve as a testament for your commitment to being a lifelong learner in machine learning.

Q35- What are the last machine learning papers you’ve read?

More reading: [What are some of the best research papers/books for machine learning?](https://www.quora.com/What-are-some-of-the-best-research-papers-books-for-Machine-learning)

Keeping up with the latest scientific literature on machine learning is a must if you want to demonstrate interest in a machine learning position. This overview of [deep learning in Nature](http://www.cs.toronto.edu/~hinton/absps/NatureDeepReview.pdf) by the scions of deep learning themselves (from Hinton to Bengio to LeCun) can be a good reference paper and an overview of what’s happening in deep learning — and the kind of paper you might want to cite.

Q36- Do you have research experience in machine learning?

Related to the last point, most organizations hiring for machine learning positions will look for your formal experience in the field. Research papers, co-authored or supervised by leaders in the field, can make the difference between you being hired and not. Make sure you have a summary of your research experience and papers ready — and an explanation for your background and lack of formal research experience if you don’t.

Q37- What are your favorite use cases of machine learning models?

More reading: [What are the typical use cases for different machine learning algorithms? (Quora)](https://www.quora.com/What-are-the-typical-use-cases-for-different-machine-learning-algorithms)

The Quora thread above contains some examples, such as decision trees that categorize people into different tiers of intelligence based on IQ scores. Make sure that you have a few examples in mind and describe what resonated with you. It’s important that you demonstrate an interest in how machine learning is implemented.

Q38- How would you approach the “Netflix Prize” competition?

More reading: [Netflix Prize (Wikipedia)](https://en.wikipedia.org/wiki/Netflix_Prize)

The Netflix Prize was a famed competition where Netflix offered $1,000,000 for a better collaborative filtering algorithm. The team that won called BellKor had a 10% improvement and used an ensemble of different methods to win. Some familiarity with the case and its solution will help demonstrate you’ve paid attention to machine learning for a while.

Q39- Where do you usually source datasets?

More reading: [19 Free Public Data Sets For Your First Data Science Project (Springboard)](https://www.springboard.com/blog/free-public-data-sets-data-science-project/)

Machine learning interview questions like these try to get at the heart of your machine learning interest. Somebody who is truly passionate about machine learning will have gone off and done side projects on their own, and have a good idea of what great datasets are out there. If you’re missing any, check out [Quandl](https://www.quandl.com/) for economic and financial data, and [Kaggle’s Datasets](https://www.kaggle.com/datasets) collection for another great list.

Q40- How do you think Google is training data for self-driving cars?

More reading: [Waymo Tech](https://waymo.com/tech/)

Machine learning interview questions like this one really test your knowledge of different machine learning methods, and your inventiveness if you don’t know the answer. Google is currently using [recaptcha](https://www.google.com/recaptcha) to source labelled data on storefronts and traffic signs. They are also building on training data collected by Sebastian Thrun at GoogleX — some of which was obtained by his grad students driving buggies on desert dunes!

Q41- How would you simulate the approach AlphaGo took to beat Lee Sidol at Go?

More reading: [Mastering the game of Go with deep neural networks and tree search (Nature)](http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html)

AlphaGo beating Lee Sidol, the best human player at Go, in a best-of-five series was a truly seminal event in the history of machine learning and deep learning. The Nature paper above describes how this was accomplished with “Monte-Carlo tree search with deep neural networks that have been trained by supervised learning, from human expert games, and by reinforcement learning from games of self-play.”