Section A

1. Define Machine learning with its objectives

The primary purpose of machine learning is to discover patterns in the user data and then make predictions based on these and intricate patterns for answering business questions and solving business problems.

The goal of machine learning is often — though not always — to train a model on historical, labelled data (i.e., data for which the outcome is known) in order to predict the value of some quantity on the basis of a new data item for which the target value or classification is unknown.

2. Define goal of the support vector machine (SVM).

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points. To separate the two classes of data points, there are many possible hyperplanes that could be chosen.

3. List out different algorithms can be classified under Association Rule Learning Algorithms?

Association rule learning is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable. It tries to find some interesting relations or associations among the variables of dataset. It is based on different rules to discover the interesting relations between variables in the database.

The association rule learning is one of the very important concepts of machine learning, and it is employed in **Market Basket analysis, Web usage mining, continuous production, etc.**

Association rule learning can be divided into three types of algorithms: 1. Apriori 2. Eclat 3. F-P Growth Algorithm

4. Define any algorithm you know in to solve a problem in Reinforcement Learning.

There are two important learning models in reinforcement learning:

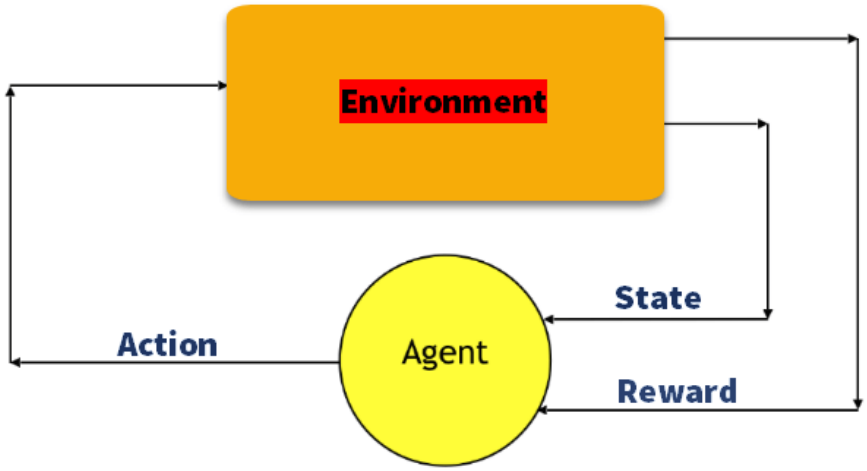
- Markov Decision Process
- Q learning

Markov Decision Process

The following parameters are used to get a solution:

- Set of actions- A
- Set of states -S
- Reward- R
- Policy- n
- Value- V

The mathematical approach for mapping a solution in reinforcement Learning is recon as a Markov Decision Process or (MDP).

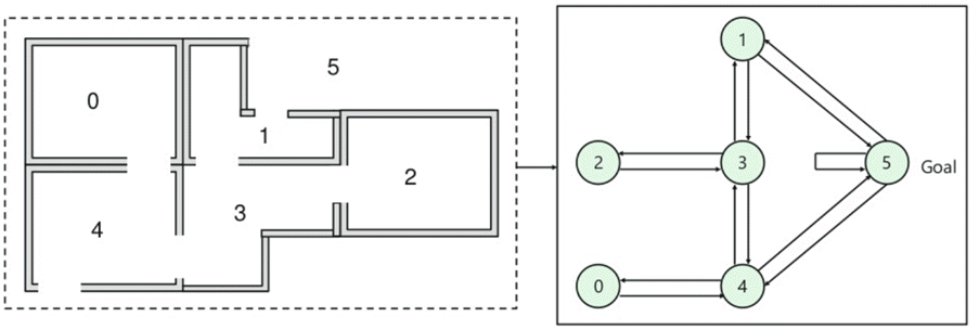


Q-Learning

Q learning is a value-based method of supplying information to inform which action an agent should take.

Let's understand this method by the following example:

- There are five rooms in a building which are connected by doors.
- Each room is numbered 0 to 4
- The outside of the building can be one big outside area (5)
- Doors number 1 and 4 lead into the building from room 5



Next, you need to associate a reward value to each door:

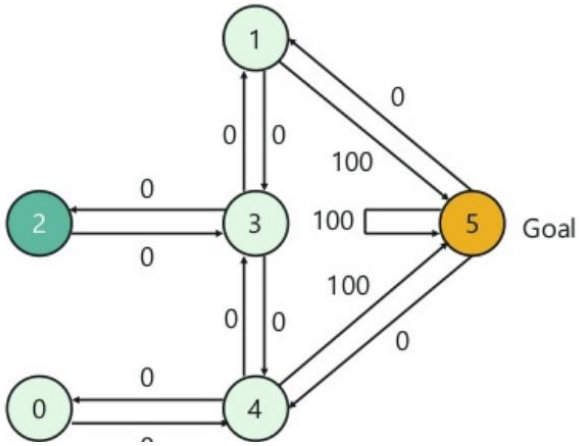
- Doors which lead directly to the goal have a reward of 100
- Doors which is not directly connected to the target room gives zero reward
- As doors are two-way, and two arrows are assigned for each room
- Every arrow in the above image contains an instant reward value

Explanation:

In this image, you can view that room represents a state

Agent's movement from one room to another represents an action

In the below-given image, a state is described as a node, while the arrows show the action.



For example, an agent traverse from room number 2 to 5

- Initial state = state 2
- State 2-> state 3
- State 3 -> state (2,1,4)
- State 4-> state (0,5,3)
- State 1-> state (5,3)
- State 0-> state 4

5. Define Agglomerative Clustering and divisive clustering.

   Agglomerative Clustering: Also known as bottom-up approach or hierarchical agglomerative clustering (HAC). A structure that is more informative than the unstructured set of clusters returned by flat clustering. This clustering algorithm does not require us to prespecify the number of clusters. Bottom-up algorithms treat each data as a singleton cluster at the outset and then successively agglomerates pairs of clusters until all clusters have been merged into a single cluster that contains all data.

Divisive clustering: Also known as a top-down approach. This algorithm also does not require to prespecify the number of clusters. Top-down clustering requires a method for splitting a cluster that contains the whole data and proceeds by splitting clusters recursively until individual data have been split into singleton clusters.

Section B

6. Differentiate between Supervised, Unsupervised and Reinforcement Learning

Criteria	Supervised ML	Unsupervised ML	Reinforcement ML
Definition	Learns by using labelled data	Trained using unlabelled data without any guidance.	Works on interacting with the environment
Type of data	Labelled data	Unlabelled data	No – predefined data
Type of problems	Regression and classification	Association and Clustering	Exploitation or Exploration
Supervision	Extra supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, SVM, KNN etc.	K – Means, C – Means, Apriori	Q – Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self Driving Cars, Gaming, Healthcare

7. How would you detect overfitting in Linear Models?

Overfitting is a modeling error that occurs when a function or model is too closely fit the training set and getting a drastic difference of fitting in test set. Overfitting the model generally takes the form of making an overly complex model to explain Model behavior in the data under study.

So the first step to finding the Overfitting is to split the data into the Training and Testing set. If our model does much better on the training set than on the test set, then we're likely overfitting. The performance can be measured using the percentage of accuracy observed in both data sets to conclude on the presence of overfitting. If the model performs better on the training set than on the test set, it means that the model is likely overfitting.

In linear regression overfitting occurs when the model is "too complex". This usually happens when there are a large number of parameters compared to the number of observations.

8. Elaborate relationship between k-Means Clustering and PCA?

Principal Component Analysis (PCA) is a tool for dimension reduction. This technique is to transform the larger dataset into a smaller dataset by identifying the correlations and patterns with preserving most of the valuable information.

This is need for feature selection of a model. PCA aims to capture valuable information explaining high variance which results in providing the best accuracy.

K-Means Clustering:

It is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups(clusters) where each data point belongs to only one group.

K-means clustering uses “centroids”, K different randomly-initiated points in the data, and assigns every data point to the nearest centroid. After every point has been assigned, the centroid is moved to the average of all of the points assigned to it.

Relationship between PCA and K Means Clustering:

Both uses are in dimensionality reduction for visualizing patterns in data from parameters(variables).PCA in conjunction with K-means is a powerful method for visualizing high dimensional data.

k-means tries to find the least-squares partition of the data. PCA finds the least-squares cluster membership vector. The first Eigenvector has the largest variance, therefore splitting on this vector (which resembles cluster membership, not input data coordinates!) means maximizing between cluster variance.

However, when we employ PCA prior to using K-means we can visually separate almost the entire data set. That was one of the biggest goals of PCA - to reduce the number of variables by combining them into bigger, more meaningful features. Not only that, but they are 'orthogonal' to each other.

9. Compare Reinforcement Learning and Supervised Learning.

Criteria	Supervised ML	Unsupervised ML	Reinforcement ML
Definition	Learns by using labelled data	Trained using unlabelled data without any guidance.	Works on interacting with the environment
Type of data	Labelled data	Unlabelled data	No – predefined data
Type of problems	Regression and classification	Association and Clustering	Exploitation or Exploration
Supervision	Extra supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, SVM, KNN etc.	K – Means, C – Means, Apriori	Q – Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self Driving Cars, Gaming, Healthcare

10.

10	The values of independent variable x and dependent value y are given below: X: 1, 3, 5, 7, 8 Y: 3, 4, 2, 5, 7 Find the regression line y=ax+b. Estimate the value of y when x is 11.

Concept:

The normal equation for Fitting a straight line by the **least square method** is:

$\Sigma y = na + b\,\Sigma x$

$\Sigma xy = a\,\Sigma x + b\,\Sigma x^2$

Where

n = Total number of observations, a and b are the coefficients.

By solving the above two equations coefficients a and b can be obtained.

Given Data and Calculation:

x	y	x^2	xy
5	16	25	80
2	10	4	20
4	13	16	52
3	12	9	36
Σx = 14	Σy = 51	Σx^2 = 54	Σxy = 188

n = 4 So

51 = 4a + 14b

188 = 14a + 54b

Solving the above two equations **a = 6.1 and b = 1.9**

11. Compare overfitting and underfitting with an intuitive explanation of the Bias-Variance Tradeoff.

Overfitting, underfitting, and the bias-variance tradeoff are foundational concepts in machine learning. A model is **overfit** if performance on the training data, used to fit the model, is substantially better than performance on a test set, held out from the model training process. For example, the prediction error of the training data may be noticeably smaller than that of the testing data. Comparing model performance metrics between these two data sets is one of the main reasons that data are split for training and testing. This way, the model's capability for predictions with new, unseen data can be assessed.

When a model overfits the training data, it is said to have **high variance**. One way to think about this is that whatever variability exists in the training data, the model has "learned" this very well. In fact, too well. A model with high variance is likely to have learned the noise in the training set. Noise consists of the random fluctuations, or offsets from true values, in the features (independent variables) and response (dependent variable) of the data. Noise can obscure the true relationship between features and the response variable. Virtually all real-world data are noisy.

If there is random noise in the training set, then there is probably also random noise in the testing set. However, the specific values of the random fluctuations will be different than those of the training set, because after all, the noise is random. The model cannot anticipate the fluctuations in the new, unseen data of the testing set. This why testing performance of an overfit model is lower than training performance.

Overfitting is more likely in the following circumstances:

- There are a large number of features available, relative to the number of samples (observations). The more features there are, the greater the chance of discovering a spurious relationship between the features and the response.
- A complex model is used, such as deep decision trees, or neural networks. Models like these effectively engineer their own features, and have an opportunity develop more complex hypotheses about the relationship between features and the response, making overfitting more likely.

At the opposite end of the spectrum, if a model is not fitting the training data very well, this is known as **underfitting**, and the model is said to have **high bias**. In this case, the model may not be complex enough, in terms of the features or the type of model being used.

12. Scrutinize that Principal Component Analysis (PCA) is used for Dimensionality Reduction with an example.

Principal Component Analysis(PCA) is one of the most popular linear dimension reduction algorithms. It is a projection based method that transforms the data by projecting it onto a set of orthogonal(perpendicular) axes.

Reducing the number of input variables for a predictive model is referred to as dimensionality reduction.

Fewer input variables can result in a simpler predictive model that may have better performance when making predictions on new data.

Perhaps the most popular technique for dimensionality reduction in machine learning is Principal Component Analysis, or PCA for short. This is a technique that comes from the field of linear algebra and can be used as a data preparation technique to create a projection of a dataset prior to fitting a model.

With all the effectiveness PCA provides, but if the number of variables is large, it becomes hard to interpret the principal components. PCA is most suitable when variables have a linear relationship among them. Also, PCA is susceptible to big outliers.

There are many methods for Dimensionality Reduction like PCA, ICA, t-SNE, etc., we shall see PCA (Principal Component Analysis).

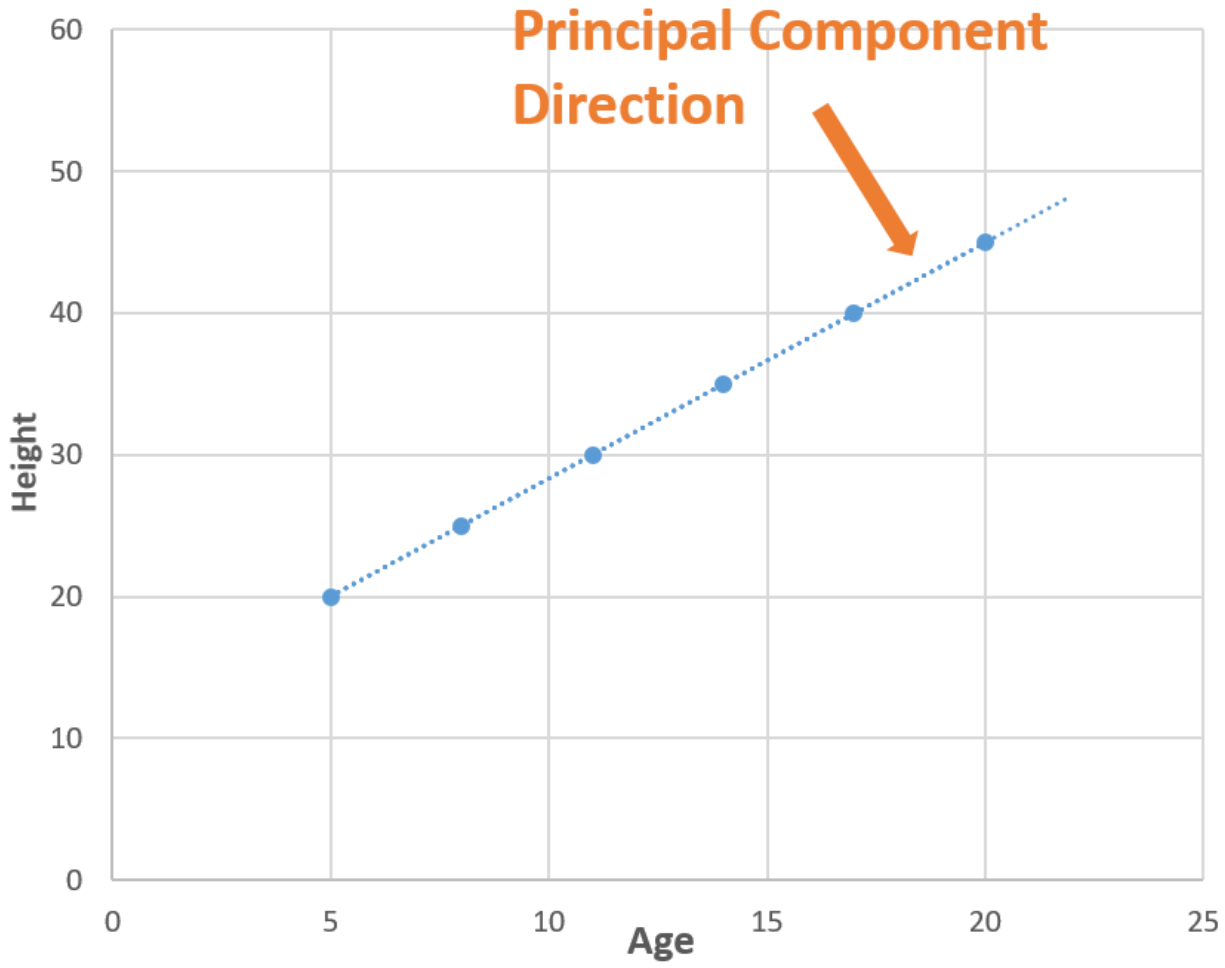
Let's first understand what is *information* in data. Consider the following imaginary data, which has Age, Weight, and Height of people.

Age	Weight	Height
5	20	4
8	25	4
11	30	4
14	35	4
17	40	4
20	45	4

Fig.1

The information lies in the variance!
As the 'Height' of all the people is the same i.e. the variance is 0 thus it's not adding any information, so we can remove the 'Height' column without losing any information.

Now we know that information is variance, let's understand the working of PCA. In PCA, we find new dimensions (features) that capture maximum variance (i.e. information). To understand this we shall use the previous example. After removing 'Height', we are left with 'Age' and 'Weight'. These two features are caring all the information. In other words, we can say that we require 2 features (Age and Height) to hold the information, and if we can find a new feature that alone can hold all the information, we can replace origination 2 features with a new single feature, achieving dimensionality reduction!



Now consider a line (blue dotted line) that is passing through all the points. The blue dotted line is capturing all the information so, we can replace 'Age' and 'Weight' (2 Dimensions) with the blue dotted line (1 Dimension) without losing any information, and in this way, we have done dimensionality reduction (2 dimensions to 1 dimension). The blue dotted line is called **Principal Component.**

SET 2

1. Define hypothesis testing and Normal Distribution.

Hypothesis testing is a form of statistical inference that uses data from a sample to draw conclusions about a population parameter or a population probability distribution. First, a tentative assumption is made about the parameter or distribution. This assumption is called the null hypothesis and is denoted by H0. An alternative hypothesis (denoted Ha), which is the opposite of what is stated in the null hypothesis, is then defined. The hypothesis-testing procedure involves using sample data to determine whether or not H0 can be rejected. If H0 is rejected, the statistical conclusion is that the alternative hypothesis Ha is true.

2. Define possibility of conversion for Regression into Classification and vice versa.

Some regression models are already classification models - e.g. logistic regression. One could set the cut point at any particular level to get a classification. Usually one would choose a 50-50 split, but there may be reasons not to (the cost of the two types of classification error might be different).

Regression trees turn into classification trees if the dependent variable changes. In general, it is not a good idea to turn a continuous dependent variable (as for regression trees) into a categorical one - it loses information. But there might be times when it is necessary (e.g. to make certain kinds of decisions).

Similarly, if you categorize the dependent variable, a linear regression is inappropriate and a logistic regression model is better.

Regression models can be very sensitive to outliers. Also, a practical challenge is, a predicted value might be far off from the real value in extreme ranges, however it still may fall in the correct side of the data distribution with respect to the mean. For e.g. you have a device that measures heart rate from your finger tips images (just cooking up an example here), it’ll be lot easier from data science standpoint to first try to predict whether the pulse rate is normal or high or below normal.

The definitions of what is “normal” is very clear from medical standpoint. However, you may wish to redefine your response variable segments, first using a clustering technique.

Regression models can be very sensitive to outliers. Also, a practical challenge is, a predicted value might be far off from the real value in extreme ranges, however it still may fall in the correct side of the data distribution with respect to the mean. For e.g. you have a device that measures heart rate from your finger tips images (just cooking up an example here), it’ll be lot easier from data science standpoint to first try to predict whether the pulse rate is normal or high or below normal.

The definitions of what is “normal” is very clear from medical standpoint. However, you may wish to redefine your response variable segments, first using a clustering technique.

Before we do this, it is important to clarify the distinction between regression and classification models. Regression models predict a continuous variable, such as rainfall amount or sunlight intensity. They can also predict probabilities, such as the probability that an image contains a cat. A probability-predicting regression model can be used as part of a classifier by imposing a decision rule - for example, if the probability is 50% or more, decide it's a cat.

Logistic regression predicts probabilities, and is therefore a regression algorithm. However, it is commonly described as a classification method in the machine learning literature, because it can be (and is often) used to make classifiers. There are also "true" classification algorithms, such as SVM, which only predict an outcome and do not provide a probability. We won't discuss this kind of algorithm here.

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

Above, malignant tumors get 11 and non-malignant ones get 00, and the green line is our hypothesis h(x)h(x). To make predictions we may say that for any given tumor size xx, if h(x)h(x) gets bigger than 0.50.5 we predict malignant tumor, otherwise we predict benign.

Looks like this way we could correctly predict every single training set sample, but now let's change the task a bit.

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

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

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

Hope this explains why linear regression is not the best fit for classification problems! Also, you might want to watch VI. Logistic Regression. Classification video on ml-class.org which explains the idea in more detail.

EDIT

probabilityislogic asked what a good classifier would do. In this particular example you would probably use logistic regression which might learn a hypothesis like this (I'm just making this up):

Note that both linear regression and logistic regression give you a straight line (or a higher order polynomial) but those lines have different meaning:

- h(x)h(x) for linear regression interpolates, or extrapolates, the output and predicts the value for xx we haven't seen. It's simply like plugging a new xx and getting a raw number, and is more suitable for tasks like predicting, say car price based on {car size, car age} etc.
- h(x)h(x) for logistic regression tells you the probability that xx belongs to the "positive" class. This is why it is called a regression algorithm - it estimates a continuous quantity, the probability. However, if you set a threshold on the probability, such as h(x)>0.5h(x)>0.5, you obtain a classifier, and in many cases this is what is done with the output from a logistic regression model. This is equivalent to putting a line on the plot: all points sitting above the classifier line belong to one class while the points below belong to the other class.

So, the bottom line is that in classification scenario we use a completely different reasoning and a completely different algorithm than in regression scenario.

3. Differentiate KNN and K-means Clustering.

4. Define dendrogram in Hierarchical Clustering Algorithm.

A dendrogram is a diagram that shows the hierarchical relationship between objects. It is most commonly created as an output from hierarchical clustering. The main use of a dendrogram is to work out the best way to allocate objects to clusters.

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

5. List out different algorithms to solve a problem in Reinforcement Learning.

There are two important learning models in reinforcement learning:

- Markov Decision Process
- Q learning

Markov Decision Process

The following parameters are used to get a solution:

- Set of actions- A
- Set of states -S
- Reward- R
- Policy- n
- Value- V

The mathematical approach for mapping a solution in reinforcement Learning is recon as a Markov Decision Process or (MDP).

Q-Learning

Q learning is a value-based method of supplying information to inform which action an agent should take.

Let's understand this method by the following example:

- There are five rooms in a building which are connected by doors.
- Each room is numbered 0 to 4
- The outside of the building can be one big outside area (5)
- Doors number 1 and 4 lead into the building from room 5

Next, you need to associate a reward value to each door:

- Doors which lead directly to the goal have a reward of 100
- Doors which is not directly connected to the target room gives zero reward
- As doors are two-way, and two arrows are assigned for each room
- Every arrow in the above image contains an instant reward value

Explanation:

In this image, you can view that room represents a state

Agent's movement from one room to another represents an action

In the below-given image, a state is described as a node, while the arrows show the action.

For example, an agent traverse from room number 2 to 5

- Initial state = state 2
- State 2-> state 3
- State 3 -> state (2,1,4)
- State 4-> state (0,5,3)
- State 1-> state (5,3)
- State 0-> state 4

6. Differentiate Root Mean Squared Error (RMSE) with Mean Squared Error (MSE) for Linear Regression?

The Mean Squared Error (MSE) is a measure of how close a fitted line is to data points. For every data point, you take the distance vertically from the point to the corresponding y value on the curve fit (the error), and square the value. Then you add up all those values for all data points, and, in the case of a fit with two parameters such as a linear fit, divide by the number of points minus two. The squaring is done so negative values do not cancel positive values. The smaller the Mean Squared Error, the closer the fit is to the data. The MSE has the units squared of whatever is plotted on the vertical axis.

RMSE stands for root mean square error and MSE stands for mean square error. It is just the square root of the mean square error. That is probably the most easily interpreted statistic, since it has the same units as the quantity plotted on the vertical axis.

They are the most common measures of accuracy for a linear regression model. The formulas are below.

RMSE is the square root of MSE. MSE is measured in units that are the square of the target variable, while RMSE is measured in the same units as the target variable. Due to its formulation, MSE, just like the squared loss function that it derives from, effectively penalizes larger errors more severely.

7. Describe over fitting with comparison to underfitting? Give any one method to avoid over fitting.

1. Train with more data.
2. Data augmentation.

3. Addition of noise to the input data.
4. Feature selection.
5. Cross-validation.
6. Simplify data.
7. Regularization.
8. Ensembling

8. Elaborate Apriori algorithm using confidence, support, and lift with an appropriate example.

The Apriori algorithm uses frequent itemsets to generate association rules, and it is designed to work on the databases that contain transactions. With the help of these association rule, it determines how strongly or how weakly two objects are connected. This algorithm uses a breadth-first search and Hash Tree to calculate the itemset associations efficiently. It is the iterative process for finding the frequent itemsets from the large dataset.

Consider the following dataset and we will find frequent itemsets and generate association rules for them.

minimum support count is 2

minimum confidence is 60%

Step-1: K=1

(I) Create a table containing support count of each item present in dataset – Called C1(candidate set)

(II) compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.

Step-2: K=2

- Generate candidate set C2 using L1 (this is called join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common.
- Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent.Check for each itemset)
- Now find support count of these itemsets by searching in dataset.

(II) compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

Step-3:

- Generate candidate set C3 using L2 (join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common. So here, for L2, first element should match.

So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}

- Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset.(Here subset of {I1, I2, I3} are {I1, I2},{I2, I3},{I1, I3} which are frequent. For {I2, I3, I4}, subset {I3, I4} is not frequent so remove it. Similarly check for every itemset)
- find support count of these remaining itemset by searching in dataset.

(II) Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.

Step-4:

- Generate candidate set C4 using L3 (join step). Condition of joining Lk-1 and Lk-1 (K=4) is that, they should have (K-2) elements in common. So here, for L3, first 2 elements (items) should match.
- Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contains {I1, I3, I5}, which is not frequent). So no itemset in C4
- We stop here because no frequent itemsets are found further

---

Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence –

A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.

Confidence(A->B)=Support_count(A∪B)/Support_count(A)

So here, by taking an example of any frequent itemset, we will show the rule generation.

Itemset {I1, I2, I3} //from L3

SO rules can be

[I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%

[I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%

[I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%

[I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6*100=33%

[I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7*100=28%

[I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6*100=33%

So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

9. Compare Naive Bayes with Logistic Regression to solve classification problems.

Naïve Bayes is a classification method based on Bayes' theorem that derives the probability of the given feature vector being associated with a label. Naïve Bayes has a naive assumption of conditional independence for every feature, which means that the algorithm expects the features to be independent which not always is the case.

Logistic regression is a linear classification method that learns the probability of a sample belonging to a certain class. Logistic regression tries to find the optimal decision boundary that best separates the classes.

1. Both algorithms are used for classification problems

The first similarity is the classification use case, where both Naive Bayes and Logistic regression are used to determine if a sample belongs to a certain class, for example, if an e-mail is spam or ham.

2. Algorithm's Learning mechanism

The learning mechanism is a bit different between the two models, where Naive Bayes is a generative model and Logistic regression is a discriminative model. What does this mean?

Generative model: Naive Bayes models the joint distribution of the feature X and target Y, and then predicts the posterior probability given as P(y|x)

Discriminative model: Logistic regression directly models the posterior probability of P(y|x) by learning the input to output mapping by minimising the error.

You might wonder what posterior probability is, let me give you a hint. Posterior probability can be defined as the probability of event A happening given that event B has occurred, in more layman terms this means that the previous belief can be updated when we have new information. For example, let's say we think the stock market will go up by 50% next year, this prediction can be updated when we get new information such as updated GPD numbers, interest rates etc.

3. Model assumptions

Naïve Bayes assumes all the features to be conditionally independent. So, if some of the features are in fact dependent on each other (in case of a large feature space), the prediction might be poor.

Logistic regression splits feature space linearly, and typically works reasonably well even when some of the variables are correlated.

4. Approach to be followed to improve model results

Naïve Bayes: When the training data size is small relative to the number of features, the information/data on prior probabilities help in improving the results

Logistic regression: When the training data size is small relative to the number of features, including regularisation such as Lasso and Ridge regression can help reduce overfitting and result in a more generalised model.

10. Differentiate Random Forest with Decision Tree and Explain how is it possible to perform Unsupervised Learning with Random Forest?

As stated above, many unsupervised learning methods require the inclusion of an input dissimilarity measure among the observations. Hence, if a dissimilarity matrix can be produced using Random Forest, we can successfully implement unsupervised learning. The patterns found in the process will be used to make clusters.

An artificial class label is created that distinguishes the ‘observed’ data from suitably generated ‘synthetic’ data. The observed data is the original unlabeled data, while the synthetic data is drawn from a reference distribution. Supervised learning methods, which distinguish observed data from synthetic data, yield a dissimilarity measure that can be used as input in subsequent unsupervised learning methods.

11. Can PCA be used for regression-based problem statements? If yes, then explain the scenario where we can use it.

Yes, we can use Principal Components for regression problem statements.

PCA would perform well in cases when the first few Principal Components are sufficient to capture most of the variation in the independent variables as well as the relationship with the dependent variable.

The only problem with this approach is that the new reduced set of features would be modeled by ignoring the dependent variable Y when applying a PCA and while these features may do a good overall job of explaining the variation in X, the model will perform poorly if these variables don’t explain the variation in Y.

Yes, we can use Principal Components for regression problem statements.

PCA would perform well in cases when the first few Principal Components are sufficient to capture most of the variation in the independent variables as well as the relationship with the dependent variable.

The only problem with this approach is that the new reduced set of features would be modeled by ignoring the dependent variable Y when applying a PCA and while these features may do a good overall job of explaining the variation in X, the model will perform poorly if these variables don’t explain the variation in Y.

12. Compare Feature Extraction and Feature Selection techniques. Explain how dimensionality can be reduced using subset selection procedure.

The main difference:- Feature Extraction transforms an arbitrary data, such as text or images, into numerical features that is understood by machine learning algorithms. Feature Selection on the other hand is a machine learning technique applied on these (numerical) features.

Feature selection is the process of choosing precise features, from a features pool. This helps in simplification, regularization and shortening training time. This can be done with various techniques: e.g. Linear Regression, Decision Trees.

Feature extraction is the process of converting the raw data into some other data type, with which the algorithm works is called Feature Extraction. Feature extraction creates a new, smaller set of features that captures most of the useful information in the data.

The main difference between them is Feature selection keeps a subset of the original features while feature extraction creates new ones.

Feature Selection:- This module is used for feature selection/dimensionality reduction on given datasets. This is done either to improve estimators' accuracy scores or to boost their performance on very high-dimensional datasets.

Feature Extraction:- This module is used to extract features in a format supported by machine learning algorithms from the given datasets consisting of formats such as text and image.

The main difference:- Feature Extraction transforms an arbitrary data, such as text or images, into numerical features that is understood by machine learning algorithms. Feature Selection on the other hand is a machine learning technique applied on these (numerical) features.

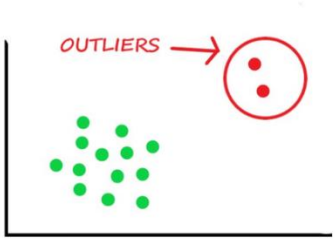
SET 3

1. Differentiate a missing value with an outlier

Outlier is the value far from the main group. Missing value is the value of blank. We often meet them when we analyze large size data. Outlier and missing value are also called "abnormal value", "noise", "trash", "bad data" and "incomplete data".

Row no	State	Salary	Yrs of Experience
1	NY	57400	Mid
2	TX		Entry
3	NJ	90000	High
4	VT	36900	Entry
5	TX		Mid
6	CA	76600	High
7	NY	85000	High
8	CA		Entry
9	CT	45000	Entry

Missing values



2. Define Ensemble Learning.

Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem.

Ensemble learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of an unfortunate selection of a poor one. Other applications of ensemble learning include assigning a confidence to the decision made by the model, selecting optimal (or near optimal) features, data fusion, incremental learning, nonstationary learning and error-correcting.

Ensemble learning helps improve machine learning results by combining several models. This approach allows the production of better predictive performance compared to a single model. Basic idea is to learn a set of classifiers (experts) and to allow them to vote.

Advantage: Improvement in predictive accuracy.
Disadvantage: It is difficult to understand an ensemble of classifiers.

3. List out various advantages of using Decision Trees

- Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.

- A decision tree does not require normalization of data.

- A decision tree does not require scaling of data as well.
- Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
- A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.

4. Differentiate Unsupervised Learning and Reinforcement Learning

Criteria	Supervised ML	Unsupervised ML	Reinforcement ML
Definition	Learns by using labelled data	Trained using unlabelled data without any guidance.	Works on interacting with the environment
Type of data	Labelled data	Unlabelled data	No – predefined data
Type of problems	Regression and classification	Association and Clustering	Exploitation or Exploration
Supervision	Extra supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, SVM, KNN etc.	K – Means, C – Means, Apriori	Q – Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self Driving Cars, Gaming, Healthcare

5. Define any problems using Naive Bayes for Classification

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

The dataset is divided into two parts, namely, **feature matrix** and the **response vector**.

- Feature matrix contains all the vectors(rows) of dataset in which each vector consists of the value of **dependent features**. In above dataset, features are ‘Outlook’, ‘Temperature’, ‘Humidity’ and ‘Windy’.
- Response vector contains the value of **class variable**(prediction or output) for each row of feature matrix. In above dataset, the class variable name is ‘Play golf’.

Assumption:

The fundamental Naive Bayes assumption is that each feature makes an:

- independent
- equal

contribution to the outcome.

	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

The dataset is divided into two parts, namely, **feature matrix** and the **response vector**.

6. Compare SVM and logistic regression in handling outliers.

SVM supports both linear and non-linear solutions using kernel trick. SVM handles outliers better than LR. Both perform well when the training data is less, and there are large number of features.

Aspects	Logistic Regression	Support Vector Machines
Multicolinearity check	Important	Not important
Outliers Handling	Cannot handle well, will skew the probability functions for labels	Can handle, outliers may not intervene with the maximum margin distance
Scaling	Important to make sure no dominance which affect coefficients	Important to ensure no dominance to affect margin distance
Optimization Function	Uses Maximum likelihood to maximize the probability of reaching to a certain label decision.	Uses Maximum Margin Distance to separate positive and negative plane by using kernels (shapes)

7. Is Feature Scaling required for the KNN Algorithm? Explain with proper justification.

Yes, feature scaling is required to get the better performance of the KNN algorithm.
For Example, Imagine a dataset having n number of instances and N number of features. There is one feature having values ranging between 0 and 1. Meanwhile, there is also a feature that varies from -999 to 999. When these values are substituted in the formula of Euclidean Distance, this will affect the performance by giving higher weightage to variables having a higher magnitude.

KNN and K-Means are one of the most commonly and widely used machine learning algorithms. KNN is a supervised learning algorithm and can be used to solve both classification as well as regression problems. K-Means, on the other hand, is an unsupervised learning algorithm which is widely used to cluster data into different groups.

One thing which is common in both these algorithms is that both KNN and K-Means are distance based algorithms. KNN chooses the k closest neighbors and then based on these neighbors, assigns a class (for classification problems) or predicts a value (for regression problems) for a new observation. K-Means clusters the similar points together. The similarity here is defined by the

distance between the points. Lesser the distance between the points, more is the similarity and vice versa.

8. Explain how the Random Forests give output for Classification and Regression problems

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

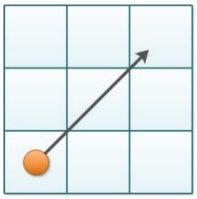
The random forest is a classification algorithm consisting of many decisions trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

A random forest regressor. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

In addition to classification, Random Forests can also be used for regression tasks. A Random Forest's nonlinear nature can give it a leg up over linear algorithms, making it a great option. However, it is important to know your data and keep in mind that a Random Forest can't extrapolate.

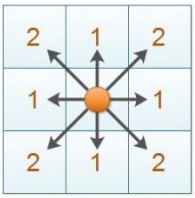
9. Differentiate Manhattan Distance and Euclidean Distance.

Euclidean Distance



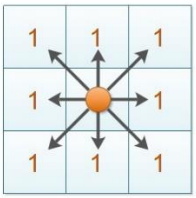
$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Manhattan Distance



$$|x_1 - x_2| + |y_1 - y_2|$$

Chebyshev Distance



$$\max(|x_1 - x_2|, |y_1 - y_2|)$$

Q10

10 Given the data in Table, reduce the dimension from 2 to 1 using the Principal Component Analysis (PCA) algorithm.

Feature	Example 1	Example 2	Example 3	Example 4
X_1	4	8	13	7
X_2	11	4	5	14

11 Find the frequent itemsets and generate the association rules using the Apriori algorithm using given dataset which has various transactions.

TID	ITEMSETS
T1	A, B
T2	B, D
T3	B, C
T4	A, B, D
T5	A, C
T6	B, C
T7	A, C
T8	A, B, C, E
T9	A, B, C

Given: Minimum Support= 2, Minimum Confidence= 50%

12 Justify with elaboration the following statement:
The k-means algorithm is based on the strong initial condition to decide the Number of clusters through the assignment of 'k' initial centroids or means.

https://www.gatevidyalay.com/principal-component-analysis-dimension-reduction/

PCA Algorithm-

The steps involved in PCA Algorithm are as follows-

Step-01: Get data.

Step-02: Compute the mean vector (μ).

Step-03: Subtract mean from the given data.

Step-04: Calculate the covariance matrix.

Step-05: Calculate the eigen vectors and eigen values of the covariance matrix.

Step-06: Choosing components and forming a feature vector.

Step-07: Deriving the new data set.

Problem-01:

Given data = { 2, 3, 4, 5, 6, 7 ; 1, 5, 3, 6, 7, 8 }.

Compute the principal component using PCA Algorithm.

OR

Consider the two dimensional patterns (2, 1), (3, 5), (4, 3), (5, 6), (6, 7), (7, 8).

Compute the principal component using PCA Algorithm.

OR

Compute the principal component of following data-

CLASS 1

X = 2 , 3 , 4

Y = 1 , 5 , 3

CLASS 2

X = 5 , 6 , 7

Y = 6 , 7 , 8

Solution-

We use the above discussed PCA Algorithm-

Step-01:

Get data.

The given feature vectors are-

- $x_1 = (2, 1)$
- $x_2 = (3, 5)$
- $x_3 = (4, 3)$
- $x_4 = (5, 6)$
- $x_5 = (6, 7)$
- $x_6 = (7, 8)$

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix} \begin{bmatrix} 5 \\ 6 \end{bmatrix} \begin{bmatrix} 6 \\ 7 \end{bmatrix} \begin{bmatrix} 7 \\ 8 \end{bmatrix}$$

Step-02:

Calculate the mean vector (μ).

Mean vector (μ)

= ((2 + 3 + 4 + 5 + 6 + 7) / 6, (1 + 5 + 3 + 6 + 7 + 8) / 6)

= (4.5, 5)

Thus,

$$\text{Mean vector } (\mu) = \begin{bmatrix} 4.5 \\ 5 \end{bmatrix}$$

Step-03:

Subtract mean vector (μ) from the given feature vectors.

- $x_1 - \mu = (2 - 4.5, 1 - 5) = (-2.5, -4)$
- $x_2 - \mu = (3 - 4.5, 5 - 5) = (-1.5, 0)$
- $x_3 - \mu = (4 - 4.5, 3 - 5) = (-0.5, -2)$
- $x_4 - \mu = (5 - 4.5, 6 - 5) = (0.5, 1)$
- $x_5 - \mu = (6 - 4.5, 7 - 5) = (1.5, 2)$
- $x_6 - \mu = (7 - 4.5, 8 - 5) = (2.5, 3)$

Feature vectors (x_i) after subtracting mean vector (μ) are-

$$\begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 2.5 \\ 3 \end{bmatrix}$$

Step-04:

Calculate the covariance matrix.

Covariance matrix is given by-

$$\text{Covariance Matrix} = \frac{\sum (x_i - \mu)(x_i - \mu)^t}{n}$$

Now,

$$m_1 = (x_1 - \mu)(x_1 - \mu)^t = \begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -2.5 & -4 \end{bmatrix} = \begin{bmatrix} 6.25 & 10 \\ 10 & 16 \end{bmatrix}$$

$$m_2 = (x_2 - \mu)(x_2 - \mu)^t = \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -1.5 & 0 \end{bmatrix} = \begin{bmatrix} 2.25 & 0 \\ 0 & 0 \end{bmatrix}$$

$$m_3 = (x_3 - \mu)(x_3 - \mu)^t = \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} -0.5 & -2 \end{bmatrix} = \begin{bmatrix} 0.25 & 1 \\ 1 & 4 \end{bmatrix}$$

$$m_4 = (x_4 - \mu)(x_4 - \mu)^t = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 0.5 & 1 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$$m_5 = (x_5 - \mu)(x_5 - \mu)^t = \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 1.5 & 2 \end{bmatrix} = \begin{bmatrix} 2.25 & 3 \\ 3 & 4 \end{bmatrix}$$

$$m_6 = (x_6 - \mu)(x_6 - \mu)^t = \begin{bmatrix} 2.5 \\ 3 \end{bmatrix} \begin{bmatrix} 2.5 & 3 \end{bmatrix} = \begin{bmatrix} 6.25 & 7.5 \\ 7.5 & 9 \end{bmatrix}$$

Now,

Covariance matrix

$= (m_1 + m_2 + m_3 + m_4 + m_5 + m_6) / 6$

On adding the above matrices and dividing by 6, we get-

$$\text{Covariance Matrix} = \frac{1}{6}\begin{bmatrix} 17.5 & 22 \\ 22 & 34 \end{bmatrix}$$

$$\text{Covariance Matrix} = \begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix}$$

Step-05:

Calculate the eigen values and eigen vectors of the covariance matrix.

λ is an eigen value for a matrix M if it is a solution of the characteristic equation $|M – \lambda I| = 0$.

So, we have-

$$\begin{vmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{vmatrix} - \begin{vmatrix} \lambda & 0 \\ 0 & \lambda \end{vmatrix} = 0$$

$$\begin{vmatrix} 2.92 - \lambda & 3.67 \\ 3.67 & 5.67 - \lambda \end{vmatrix} = 0$$

From here,

$(2.92 – \lambda)(5.67 – \lambda) – (3.67 \times 3.67) = 0$

$16.56 – 2.92\lambda – 5.67\lambda + \lambda^2 – 13.47 = 0$

$\lambda^2 – 8.59\lambda + 3.09 = 0$

Solving this quadratic equation, we get $\lambda = 8.22, 0.38$

Thus, two eigen values are $\lambda_1 = 8.22$ and $\lambda_2 = 0.38$.

Clearly, the second eigen value is very small compared to the first eigen value.

So, the second eigen vector can be left out.

Eigen vector corresponding to the greatest eigen value is the principal component for the given data set.

So. we find the eigen vector corresponding to eigen value λ_1.

We use the following equation to find the eigen vector-

$$MX = \lambda X$$

where-

- M = Covariance Matrix
- X = Eigen vector
- λ = Eigen value

Substituting the values in the above equation, we get-

$$\begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \end{bmatrix} = 8.22 \begin{bmatrix} X1 \\ X2 \end{bmatrix}$$

Solving these, we get-

$2.92X_1 + 3.67X_2 = 8.22X_1$

$3.67X_1 + 5.67X_2 = 8.22X_2$

On simplification, we get-

$5.3X_1 = 3.67X_2$ ……(1)

$3.67X_1 = 2.55X_2$ ……(2)

From (1) and (2), **$X_1 = 0.69X_2$**

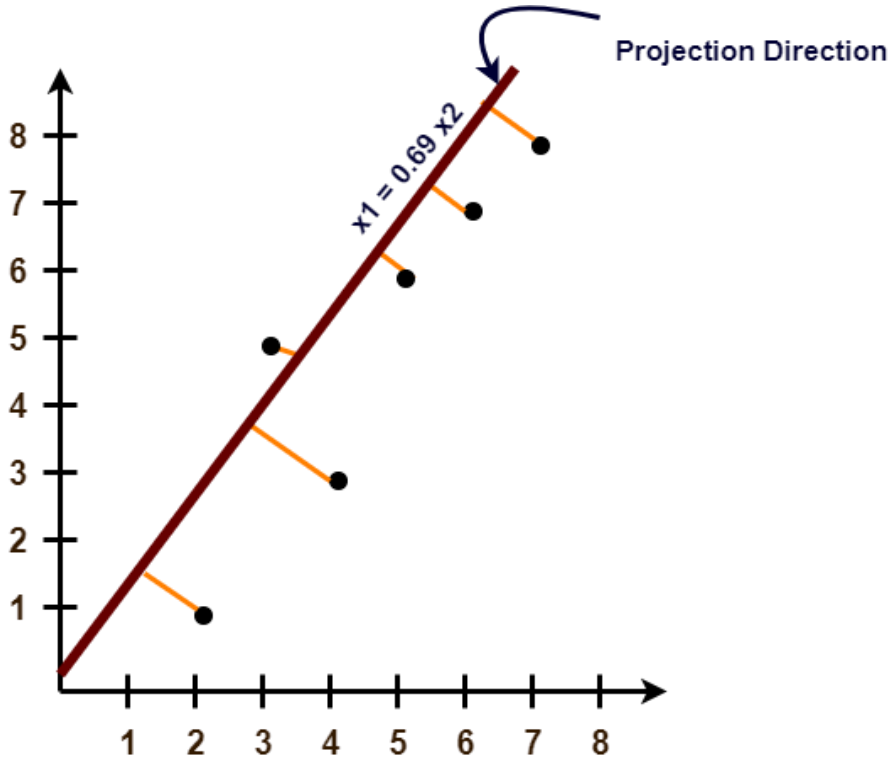
From (2), the eigen vector is-

$$\text{Eigen Vector :} \quad \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

Thus, principal component for the given data set is-

$$\text{Principal Component :} \quad \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

Lastly, we project the data points onto the new subspace as-



11.

Example: Suppose we have the following dataset that has various transactions, and from this dataset, we need to find the frequent itemsets and generate the association rules using the Apriori algorithm:

TID	ITEMSETS
T1	A, B
T2	B, D
T3	B, C
T4	A, B, D
T5	A, C
T6	B, C
T7	A, C
T8	A, B, C, E
T9	A, B, C

Given: Minimum Support= 2, Minimum Confidence= 50%

Solution:

Step-1: Calculating C1 and L1:

- In the first step, we will create a table that contains support count (The frequency of each itemset individually in the dataset) of each itemset in the given dataset. This table is called the **Candidate set or C1.**

Itemset	Support_Count
A	6
B	7
C	5
D	2
E	1

- Now, we will take out all the itemsets that have the greater support count that the Minimum Support (2). It will give us the table for the **frequent itemset L1.** Since all the itemsets have greater or equal support count than the minimum support, except the E, so E itemset will be removed.

Itemset	Support_Count
A	6
B	7
C	5
D	2

Step-2: Candidate Generation C2, and L2:

- In this step, we will generate C2 with the help of L1. In C2, we will create the pair of the itemsets of L1 in the form of subsets.
- After creating the subsets, we will again find the support count from the main transaction table of datasets, i.e., how many times these pairs have occurred together in the given

dataset. So, we will get the below table for C2:

Itemset	Support_Count
{A, B}	4
{A,C}	4
{A, D}	1
{B, C}	4
{B, D}	2
{C, D}	0

- Again, we need to compare the C2 Support count with the minimum support count, and after comparing, the itemset with less support count will be eliminated from the table C2. It will give us the below table for L2

Itemset	Support_Count
{A, B}	4
{A, C}	4
{B, C}	4
{B, D}	2

A, B, C, D

Step-3: Candidate generation C3, and L3:

- For C3, we will repeat the same two processes, but now we will form the C3 table with subsets of three itemsets together, and will calculate the support count from the dataset. It will give the below table:

Itemset	Support_Count
{A, B, C}	2
{B, C, D}	1
{A, C, D}	0
{A, B, D}	0

- Now we will create the L3 table. As we can see from the above C3 table, there is only one combination of itemset that has support count equal to the minimum support count. So, the L3 will have only one combination, i.e., **{A, B, C}.**

Step-4: Finding the association rules for the subsets:

To generate the association rules, first, we will create a new table with the possible rules from the occurred combination {A, B.C}. For all the rules, we will calculate the Confidence using formula **sup(A ^B)/A.** After calculating the confidence value for all rules, we will exclude the rules that have less confidence than the minimum threshold(50%).

Consider the below table:

Rules	**Support**	**Confidence**
A ^B → C	2	Sup{(A ^B) ^C}/sup(A ^B)= 2/4=0.5=50%
B^C → A	2	Sup{(B^C) ^A}/sup(B ^C)= 2/4=0.5=50%
A^C → B	2	Sup{(A ^C) ^B}/sup(A ^C)= 2/4=0.5=50%
C→ A ^B	2	Sup{(C^(A ^B)}/sup(C)= 2/5=0.4=40%
A→ B^C	2	Sup{(A^(B ^C)}/sup(A)= 2/6=0.33=33.33%
B→ B^C	2	Sup{(B^(B ^C)}/sup(B)= 2/7=0.28=28%

As the given threshold or minimum confidence is 50%, so the first three rules **A ^B → C, B^C → A, and A^C → B** can be considered as the strong association rules for the given problem.

12.

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

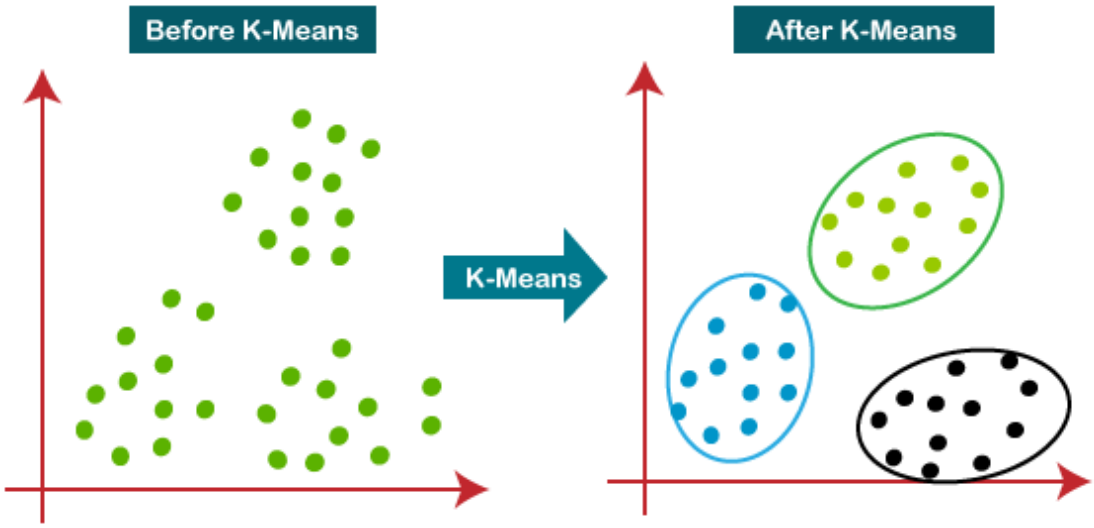
It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.



Our aim here is to minimize the distance between the points within a cluster.

There is an algorithm that tries to minimize the distance of the points in a cluster with their centroid – the k-means clustering technique.

K-means is a centroid-based algorithm, or a distance-based algorithm, where we calculate the distances to assign a point to a cluster. In K-Means, each cluster is associated with a centroid.

The main objective of the K-Means algorithm is to minimize the sum of distances between the points and their respective cluster centroid.

Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible.

Let’s now take an example to understand how K-Means actually works:

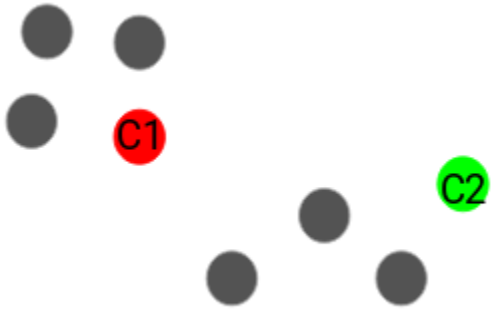
We have these 8 points and we want to apply k-means to create clusters for these points. Here’s how we can do it.

Step 1: Choose the number of clusters *k*

The first step in k-means is to pick the number of clusters, k.

Step 2: Select k random points from the data as centroids

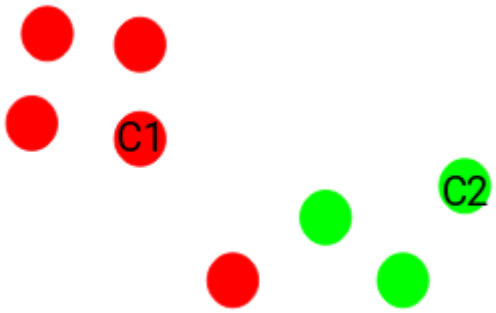
Next, we randomly select the centroid for each cluster. Let’s say we want to have 2 clusters, so k is equal to 2 here. We then randomly select the centroid:



Here, the red and green circles represent the centroid for these clusters.

Step 3: Assign all the points to the closest cluster centroid

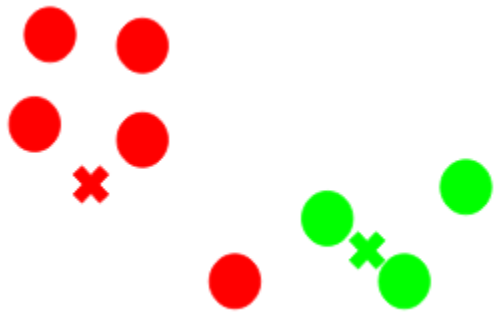
Once we have initialized the centroids, we assign each point to the closest cluster centroid:



Here you can see that the points which are closer to the red point are assigned to the red cluster whereas the points which are closer to the green point are assigned to the green cluster.

Step 4: Recompute the centroids of newly formed clusters

Now, once we have assigned all of the points to either cluster, the next step is to compute the centroids of newly formed clusters:



Here, the red and green crosses are the new centroids.

Step 5: Repeat steps 3 and 4

We then repeat steps 3 and 4:

The step of computing the centroid and assigning all the points to the cluster based on their distance from the centroid is a single iteration. But wait – when should we stop this process? It can't run till eternity, right?

Stopping Criteria for K-Means Clustering

There are essentially three stopping criteria that can be adopted to stop the K-means algorithm:

1. Centroids of newly formed clusters do not change
2. Points remain in the same cluster
3. Maximum number of iterations are reached

We can stop the algorithm if the centroids of newly formed clusters are not changing. Even after multiple iterations, if we are getting the same centroids for all the clusters, we can say that the algorithm is not learning any new pattern and it is a sign to stop the training.

Another clear sign that we should stop the training process if the points remain in the same cluster even after training the algorithm for multiple iterations.

Finally, we can stop the training if the maximum number of iterations is reached. Suppose if we have set the number of iterations as 100. The process will repeat for 100 iterations before stopping.