1. Explain logistic regression.

Logistic Regression in Machine Learning

* Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
* Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
* Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
* In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
* The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
* Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
* Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



**Note:** Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; Therefore, it falls under the classification algorithm.

Logistic Function (Sigmoid Function):

* The sigmoid function is a mathematical function used to map the predicted values to probabilities.
* It maps any real value into another value within a range of 0 and 1.
* The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
* In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

* The dependent variable must be categorical in nature.
* The independent variable should not have multi-collinearity.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

* We know the equation of the straight line can be written as:

Logistic Regression in Machine Learning

* In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

Logistic Regression in Machine Learning

* But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

Logistic Regression in Machine Learning

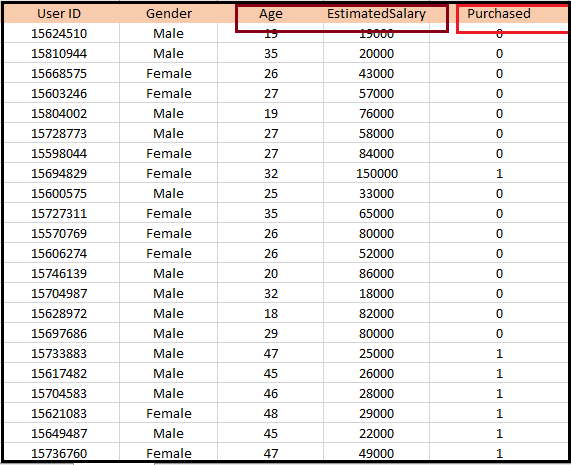
The above equation is the final equation for Logistic Regression.

Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

* **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
* **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
* **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

**Example:** There is a dataset given which contains the information of various users obtained from the social networking sites. There is a car making company that has recently launched a new SUV car. So the company wanted to check how many users from the dataset, wants to purchase the car.

For this problem, we will build a Machine Learning model using the Logistic regression algorithm. The dataset is shown in the below image. In this problem, we will predict the **purchased variable (Dependent Variable)** by using **age and salary (Independent variables)**.

2.What is Hypothesis representation in logistic regression?

# Introduction

Let’s start talking about logistic regression. In this tutorial, I’d like to show you the hypothesis representation, that is, what is the function we’re going to use to represent our hypothesis where we have a classification problem.

# Sections

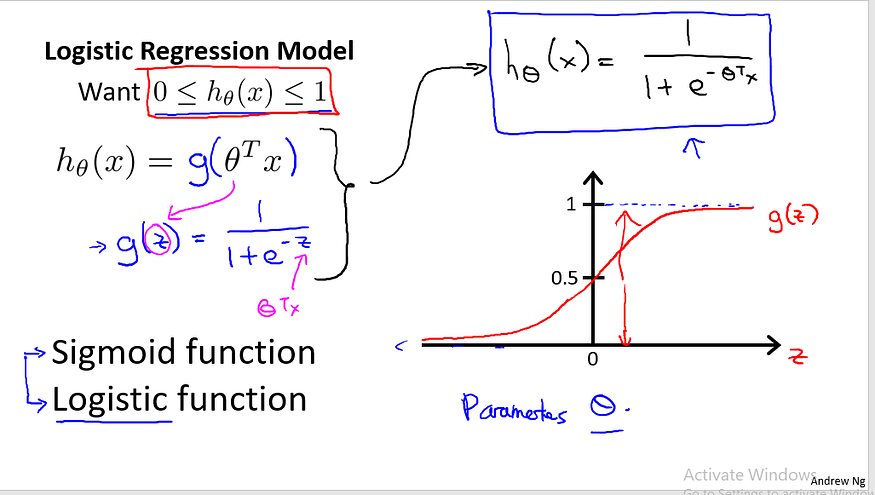
*Logistic Regression model  
Interpretation of Logistic regression model*

# Section 1-Logistic Regression model

Earlier, we said that we would like our classifier to output values that are between zero and one. So, we like to come up with a hypothesis that satisfies this property, that these predictions are maybe between zero and one. When we were using linear regression, this was the form of a hypothesis, where H of X is theta transpose X. For logistic regression,

I’m going to modify this a little bit, and make the hypothesis G of theta transpose X, where I’m going to define the function G as follows: G of Z if Z is a real number is equal to one over one plus E to the negative Z. **This called the sigmoid function or the logistic function. And the term logistic function, that’s what give rise to the name logistic progression**. **And, by the way, the terms sigmoid function and logistic function are basically synonyms and mean the same thing. So the two terms are basically interchangeable and either term can be used to refer to this function G.**

And if we take these two equations, and put them together, then here’s just an alternative way of writing out the form of my hypothesis. I’m saying that H of x is one over one plus E to the negative theta transpose X, and all I’ve done is I’ve taken the variable Z, Z here’s a real number and plugged in theta transpose X, so I end up with, you know, theta transpose X, in place of Z there.

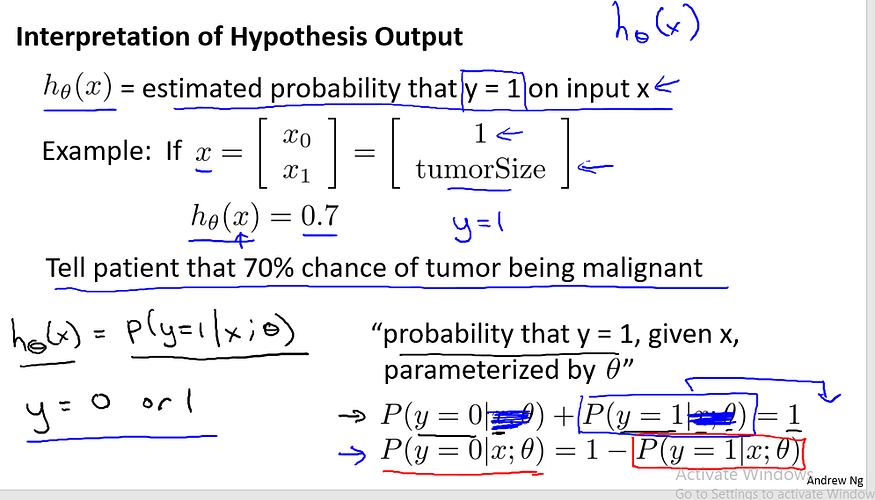


Lastly, let me show you where the sigmoid function looks like. We’re going to plot it on this figure here. The sigmoid function, G of Z, also called the logistic function, looks like this. It starts off near zero and then rises until it processes 0.5 at the origin and then it flattens out again like so. So that’s what the sigmoid function looks like. And you notice that the sigmoid function, well, it asymptotes at one, and asymptotes at zero as Z against the horizontal axis is Z. As Z goes to minus infinity, G of Z approaches zero and as G of Z approaches infinity, G of Z approaches 1, and so because G of Z offers values that are between 0 and 1 we also have that H of X must be between 0 and 1.

Finally, given this hypothesis representation, what we need to do, as before, is fit the parameters theta to our data. So given a training set, we need to pick a value for the parameters theta and this hypothesis will then let us make predictions. We’ll talk about a learning algorithm later for fitting the parameters theta.

# Section 2- Interpretation of Logistic regression model

But first let’s talk a bit about the interpretation of this model. Here’s how I’m going to interpret the output of my hypothesis H of X. When my hypothesis outputs some number, I am going to treat that number as the estimated probability that Y is equal to one on a new input example X.



Here is what I mean. Here is an example. Let’s say we’re using the tumor classification example. So we may have a feature vector X, which is this x01 as always and then our one feature is the size of the tumor. Suppose I have a patient come in and, you know they have some tumor size and I feed their feature vector X into my hypothesis and suppose my hypothesis outputs the number 0.7. I’m going to interpret my hypothesis as follows. I’m going to say that this hypothesis is telling me that for a patient with features X, the probability that Y equals one is 0 .7. In other words, I’m going to tell my patient that the tumor, sadly, has a 70% chance or a 0.7 chance of being malignant.

To write this out slightly more formally or to write this out in math, I’m going to interpret my hypothesis output as P of y equals 1, given X parametrized by theta. So, for those of you that are familiar with probability, this equation might make sense, if you’re a little less familiar with probability, you know, here’s how I read this expression, this is the probability that y is equals to one, given x instead of given that my patient has, you know, features X. Given my patient has a particular tumor size represented by my features X, and this probability is parametrized by theta. So I’m basically going to count on my hypothesis to give me estimates of the probability that Y is equal to 1

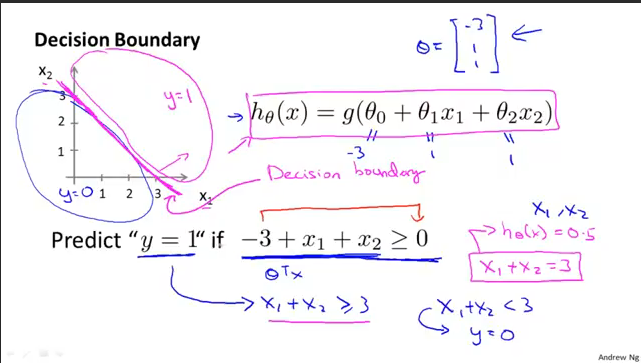
. Now since this is a classification task, we know that Y must be either zero or one, right? Those are the only two values that Y could possibly take on, either in the training set or for new patients that may walk into my office or into the doctor’s office in the future. So given H of X, we can therefore compute the probability that Y is equal to zero as well. Concretely, because Y must be either zero or one, we know that the probability of Y equals zero, plus the probability of Y equals one, must add up to one. This first equation looks a little bit more complicated but it’s basically saying that probability of Y equals zero for a particular patient with features x, and you know, given our parameter’s data, plus the probability of Y equals one for that same patient which features x and you parameters theta must add up to one, if this equation looks a little bit complicated feel free to mentally imagine it without that X and theta. And this is just saying that the probability of Y equals zero plus the probability of Y equals one must be equal to one. And we know this to be true because Y has to be either zero or one. And so the chance of Y being zero plus the chance that Y is one, you know, those two must add up to one. And so if you just take this term and move it to the right-hand side, then you end up with this equation that says probability Y equals zero is one minus probability y equals and thus if our hypothesis if H of X gives us that term you can therefore quite simply compute the probability, or compute the estimated probability that Y is equal to zero as well. So you now know what the hypothesis representation is for logistic regression and we’re seeing what the mathematical formula is defining the hypothesis for logistic regression. In the next tutrial, I’d like to try to give you better intuition about what the hypothesis function looks like. And I want to tell you something called the decision boundary and we’ll look at some visualizations together to try to get a better sense of what this hypothesis function of logistic regression really looks like.

3.Explain decision boundary logistic regression.

# Section 3- Linear Decision Boundary

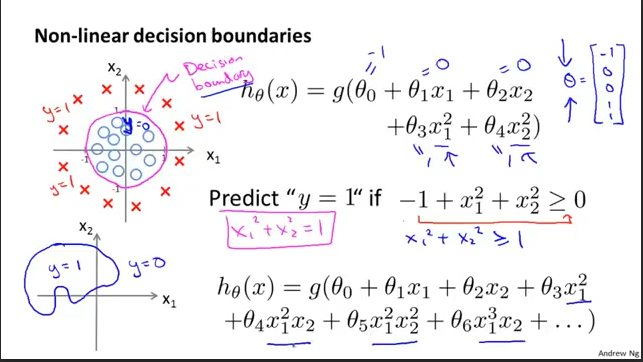
The decision boundary in logistic regression is a boundary that separates the instances of one class from the instances of the other class. It is a line (in two dimensions) or a hyperplane (in higher dimensions) that represents the region where the predicted probability of the positive class is equal to the predicted probability of the negative class.

In logistic regression, the decision boundary is determined by the weights assigned to the input features. The weights control the slope and position of the decision boundary. If the weights are such that the positive class instances have a higher probability, the decision boundary will be shifted towards the positive class. Conversely, if the weights favor the negative class, the decision boundary will be shifted towards the negative class.



Let’s use this to better understand how the hypothesis of logistic regression makes those predictions. Now, let’s suppose we have a training set like that shown on the slide, and suppose our hypothesis is H of X equals G of theta zero, plus theta one X1 plus theta two X2. We haven’t talked yet about how to fit the parameters of this model. We’ll talk about that in the next video. But suppose that variable procedure to be specified, we end up choosing the following values for the parameters. Let’s say we choose theta zero equals three, theta one equals one, theta two equals one. So this means that my parameter vector is going to be theta equals minus 311. So, we’re given this choice of my hypothesis parameters, let’s try to figure out where a hypothesis will end up predicting y equals 1 and where it will end up predicting y equals 0. Using the formulas that we worked on the previous slide, we know that Y equals 1 is more likely, that is the probability that Y equals 1 is greater than 0.5 or greater than or equal to 0.5. Whenever theta transpose x is greater than zero. And this formula that I just underlined minus three plus X1 plus X2 is, of course, theta transpose X when theta is equal to this value of the parameters that we just chose. So, for any example, for any example with features X1 and X2 that satisfy this equation that minus 3 plus X1 plus X2 is greater than or equal to 0. Our hypothesis will think that Y equals 1 is more likely, or will predict that Y is equal to one. We can also take minus three and bring this to the right and rewrite this as X1 plus X2 is greater than or equal to three. And so, equivalently, we found that this hypothesis will predict Y equals one whenever X1 plus X2 is greater than or equal to three. Let’s see what that means on the figure. If I write down the equation, X1 plus X2 equals three, this defines the equation of a straight line. And if I draw what that straight line looks like, it gives me the following line which passes through 3 and 3 on the X1 and the X2 axis. So the part of the input space, the part of the X1, X2 plane that corresponds to when X1 plus X2 is greater than or equal to three. That’s going to be this very top plane. That is everything to the up, and everything to the upper right portion of this magenta line that I just drew. And so, the region where our hypothesis will predict Y equals 1 is this region, you know, is really this huge region, this half-space over to the upper right. And let me just write that down. I’m gonna call this the Y equals one region, and in contrast the region where X1 plus X2 is less than three, that’s when we will predict that Y, Y is equal to zero, and that corresponds to this region. You know, itt’s really a half-plane, but that region on the left is the region where our hypothesis predict Y equals 0. I want to give this line, this magenta line that I drew a name. This line there is called the decision boundary. And concretely, this straight line X1 plus X equals 3. That corresponds to the set of points. So that corresponds to the region where H of X is equal to 0.5 exactly and the decision boundary, that is this straight line, that’s the line that separates the region where the hypothesis predicts Y equals one from the region where the hypothesis predicts that Y is equal to 0. And just to be clear. The decision boundary is a property of the hypothesis including the parameters theta 0, theta 1, theta 2. And in the figure I drew a training set. I drew a data set in order to help the visualization. But even if we take away the data set, you know decision boundary and a region where we predict Y equals 1 versus Y equals zero. That’s a property of the hypothesis and of the parameters of the hypothesis, and not a property of the data set. Later on, of course, we’ll talk about how to fit the parameters and there we’ll end up using the training set, or using our data, to determine the value of the parameters. But once we have particular values for the parameters: theta 0, theta 1, theta 2. Then that completely defines the decision boundary and we don’t actually need to plot a training set in order to plot the decision boundary.

# Section 4- Non-Linear Decision Boundary



Let’s now look at a more complex example where, as usual, I have crosses to denote my positive examples and O’s to denote my negative examples. Given a training set like this, how can I get logistic regression to fit this sort of data? Earlier, when we were talking about polynomial regression or when we’re linear regression, we talked about how we can add extra higher order polynomial terms to the features. And we can do the same for logistic regression. Concretely, let’s say my hypothesis looks like this. Where I’ve added two extra features, X1 squared and X2 squared, to my features. So that I now have 5 parameters, theta 0 through theta 4. As before, we’ll defer to the next tutorial our discussion on how to automatically choose values for the parameters theta 0 through theta 4. But let’s say that very procedure to be specified, I end up choosing theta 0 equals minus 1, theta 1 equals 0, theta 2 equals 0, theta 3 equals 1, and theta 4 equals 1. What this means is that with this particular choice of parameters, my parameter vector theta looks like minus 1, 0, 0, 1, 1. Following our earlier discussion, this means that my hypothesis will predict that Y is equal to 1 whenever minus 1 plus X1 squared plus X2 squared is greater than or equal to 0. This is whenever theta transpose times my theta transpose my features is greater than or equal to 0. And if I take minus 1 and just bring this to the right, I’m saying that my hypothesis will predict that Y is equal to 1 whenever X1 squared plus X2 squared is greater than or equal to 1. So, what does decision boundary look like? Well, if you were to plot the curve for X1 squared plus X2 squared equals 1. Some of you will that is the equation for a circle of radius 1 centered around the origin. So, that is my decision boundary. And everything outside the circle I’m going to predict as Y equals 1. So out here is, you know, my Y equals 1 region. I’m going to predict Y equals 1 out here. And inside the circle is where I’ll predict Y is equal to 0. So, by adding these more complex or these polynomial terms to my features as well. I can get more complex decision boundaries that don’t just try to separate the positive and negative examples of a straight line. I can get in this example a decision boundary that’s a circle. Once again the decision boundary is a property not of the training set, but of the hypothesis and of the parameters. So long as we’ve given my parameter vector theta, that defines the decision boundary which is the circle. But the training set is not what we use to define decision boundary. The training set may be used to fit the parameters theta. We’ll talk about how to do that later. But once you have the parameters theta, that is what defines the decision boundary. Let me put back the training set just for visualization.

And finally, let’s look at a more complex example. So can we come up with even more complex decision boundaries and this? If I have even higher order polynomial terms, so things like X1 squared, X1 squared X2, X1 squared X2 squared, and so on. If I have much higher order polynomials, then it’s possible to show that you can get even more complex decision boundaries and logistic regression can be used to find the zero boundaries that may, for example, be an ellipse like that, or maybe with a different setting of the parameters, maybe you can get instead a different decision boundary that may even look like, you know, some funny shape like that. Or for even more complex examples you can also get decision boundaries that can look like, you know, more complex shapes like that. Where everything in here you predict Y equals 1, and everything outside you predict Y equals 0. So these higher order polynomial features you can get very complex decision boundaries. So with these visualizations, I hope that gives you a what’s the range of hypothesis functions we can represent using the representation that we have for logistic regression.

4.What is cost function for logistic regression

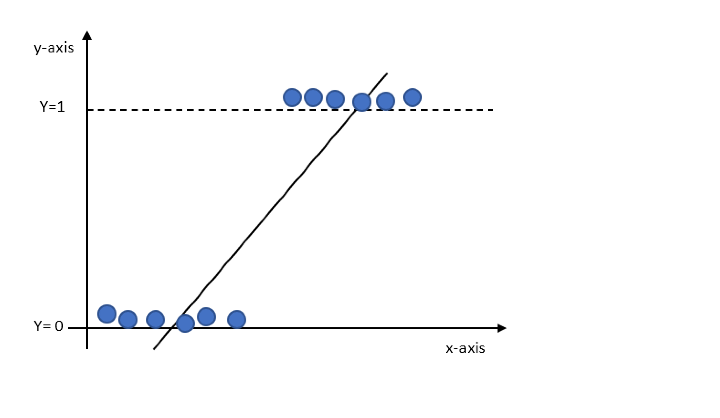
## Introduction

Logistic Regression is the simplest of all classification algorithms in Machine Learning. Logistic Regression uses log loss or cross−entropy loss instead of mean squared error for loss function. Since we already have linear regression why do we need Logistic Regression for classification and why can't use Linear Regression for classification?

Let us understand this fact through this article and explore the cost function used in Logistic Regression in detail.

## Why do we need Logistic Regression and can't use Linear Regression?

In Linear Regression, we predict a continuous value. If we fit Linear Regression to the classification task, the line of best fit will look something like the diagram below.



According to the above graph, we would have values greater than 1 and less than 0 but that does not make sense for classification as we are only interested in binary output 0 or 1.

Thus we need values to be present between the lines Y=0 and Y=1. The above line needs to be transformed such that the values lie within 0 and 1. One such transformation is applying the sigmoid function as shown below.

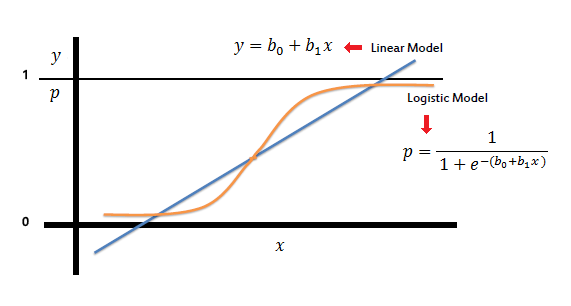
K=MX+cK=MX+c

Y=F(K)Y=F(K)

F(K)=11+e−ZF(K)=11+e−Z

Y=11+e−ZY=11+e−Z

The graph will now like as shown below



The sigmoid function gives continuous values between 0 and 1 which are probability values.

### Log loss and Cost function for Logistic Regression

One of the popular metrics to evaluate models for classification by using probabilities is log loss.

F=−∑i=1Myilog(pθ(xi))+(1−yi)log(1−pθ(xi))F=−∑i=1Myilog⁡(p𝜃(xi))+(1−yi)log⁡(1−p𝜃(xi))

The cost function can be written as

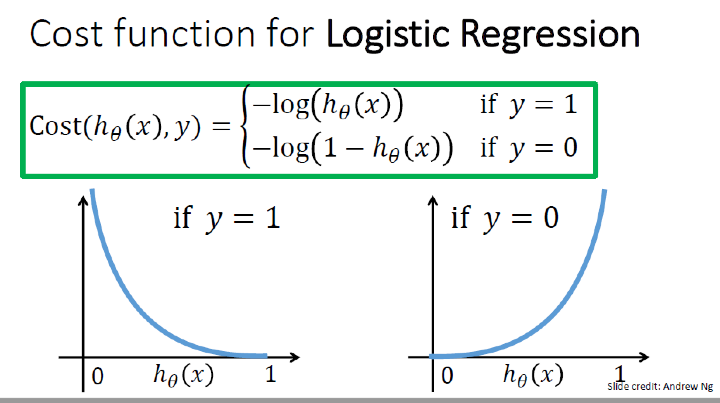
F(θ)=1n∑i=1n12[pθ(xi)−Yi]2F(𝜃)=1n∑i=1n12[p𝜃(xi)−Yi]2

For Logistic Regression,

pθ(x)=g(θTx)p𝜃(x)=g(𝜃Tx)

The above equation leads to a non−convex function that acts as the cost function. The cost function logistic regression is log loss and is summarized below.

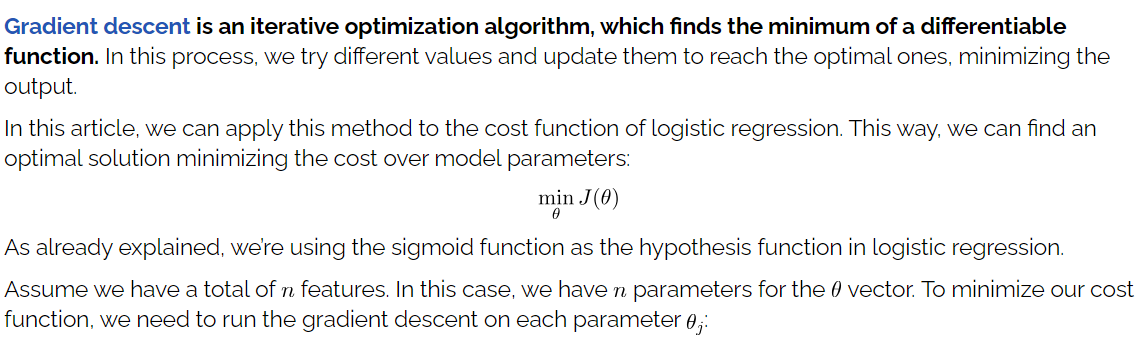
cost(pθ(x),(y))=(−log(pθ(x))ify=1 −log(1−pθ(x))ify=0)cost(p𝜃(x),(y))=(−log⁡(𝑝𝜃(𝑥))𝑖𝑓𝑦=1 −log⁡(1−𝑝𝜃(𝑥))𝑖𝑓𝑦=0)

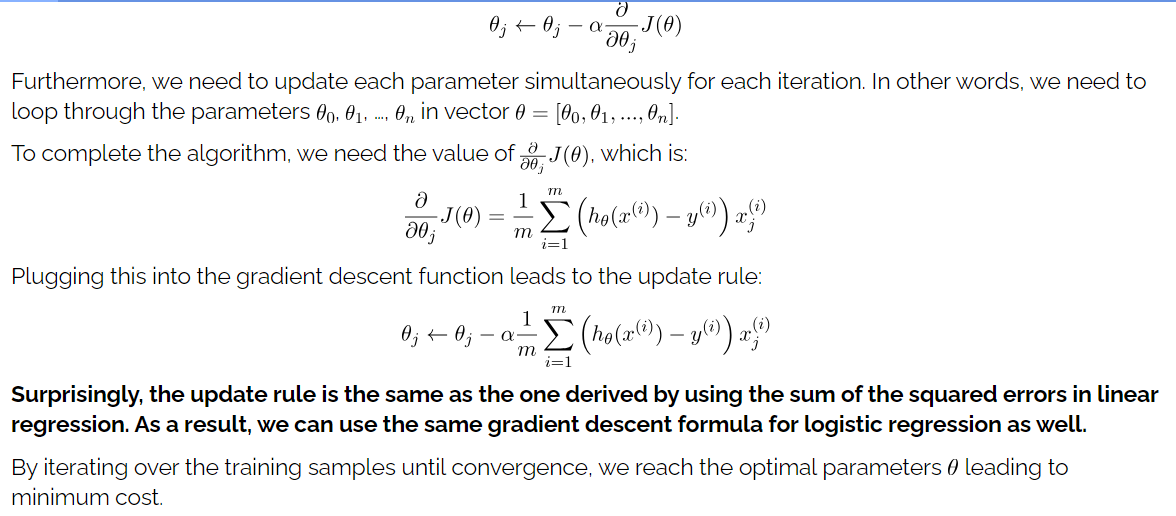


The gradient descent update equation becomes,

θk:=θk−α∑i=1n[pθ(xi)−yi]xij

5.Explain Gradient Descent for Logistic Regression.





6.Explain Naïve Bayes Classifier

Naïve Bayes Classifier Algorithm

* Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
* It is mainly used in *text classification* that includes a high-dimensional training dataset.
* Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
* **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object**.
* Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles**.

Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

* **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
* **Bayes**: It is called Bayes because it depends on the principle of [Bayes' Theorem](https://www.javatpoint.com/bayes-theorem-in-artifical-intelligence).

Bayes' Theorem:

* Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
* The formula for Bayes' theorem is given as:

Naïve Bayes Classifier Algorithm

**Where,**

**P(A|B) is Posterior probability**: Probability of hypothesis A on the observed event B.

**P(B|A) is Likelihood probability**: Probability of the evidence given that the probability of a hypothesis is true.

**P(A) is Prior Probability**: Probability of hypothesis before observing the evidence.

**P(B) is Marginal Probability**: Probability of Evidence.

Working of Naïve Bayes' Classifier:

Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.

**Problem**: If the weather is sunny, then the Player should play or not?

**Solution**: To solve this, first consider the below dataset:

|  |  |  |
| --- | --- | --- |
|  | **Outlook** | **Play** |
| **0** | Rainy | Yes |
| **1** | Sunny | Yes |
| **2** | Overcast | Yes |
| **3** | Overcast | Yes |
| **4** | Sunny | No |
| **5** | Rainy | Yes |
| **6** | Sunny | Yes |
| **7** | Overcast | Yes |
| **8** | Rainy | No |
| **9** | Sunny | No |
| **10** | Sunny | Yes |
| **11** | Rainy | No |
| **12** | Overcast | Yes |
| **13** | Overcast | Yes |

**Frequency table for the Weather Conditions:**

|  |  |  |
| --- | --- | --- |
| Weather | Yes | No |
| Overcast | 5 | 0 |
| Rainy | 2 | 2 |
| Sunny | 3 | 2 |
| Total | 10 | 5 |

**Likelihood table weather condition:**

|  |  |  |  |
| --- | --- | --- | --- |
| Weather | No | Yes |  |
| Overcast | 0 | 5 | 5/14= 0.35 |
| Rainy | 2 | 2 | 4/14=0.29 |
| Sunny | 2 | 3 | 5/14=0.35 |
| All | 4/14=0.29 | 10/14=0.71 |  |

**Applying Bayes'theorem:**

**P(Yes|Sunny)= P(Sunny|Yes)\*P(Yes)/P(Sunny)**

P(Sunny|Yes)= 3/10= 0.3

P(Sunny)= 0.35

P(Yes)=0.71

So P(Yes|Sunny) = 0.3\*0.71/0.35= **0.60**

**P(No|Sunny)= P(Sunny|No)\*P(No)/P(Sunny)**

P(Sunny|NO)= 2/4=0.5

P(No)= 0.29

P(Sunny)= 0.35

So P(No|Sunny)= 0.5\*0.29/0.35 = **0.41**

So as we can see from the above calculation that **P(Yes|Sunny)>P(No|Sunny)**

**Hence on a Sunny day, Player can play the game.**

Advantages of Naïve Bayes Classifier:

* Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
* It can be used for Binary as well as Multi-class Classifications.
* It performs well in Multi-class predictions as compared to the other Algorithms.
* It is the most popular choice for **text classification problems**.

Disadvantages of Naïve Bayes Classifier:

* Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

**7.What is Over fitting &Under fitting**

Overfitting and Underfitting in Machine Learning

Overfitting and Underfitting are the two main problems that occur in machine learning and degrade the performance of the machine learning models.

The main goal of each machine learning model is **to generalize well**. Here **generalization** defines the ability of an ML model to provide a suitable output by adapting the given set of unknown input. It means after providing training on the dataset, it can produce reliable and accurate output. Hence, the underfitting and overfitting are the two terms that need to be checked for the performance of the model and whether the model is generalizing well or not.

Before understanding the overfitting and underfitting, let's understand some basic term that will help to understand this topic well:

* **Signal:** It refers to the true underlying pattern of the data that helps the machine learning model to learn from the data.
* **Noise:** Noise is unnecessary and irrelevant data that reduces the performance of the model.
* **Bias:** Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.
* **Variance:** If the machine learning model performs well with the training dataset, but does not perform well with the test dataset, then variance occurs.

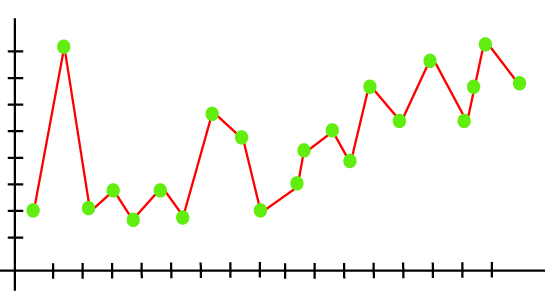
Overfitting

Overfitting occurs when our [machine learning](https://www.javatpoint.com/machine-learning) model tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model. The overfitted model has **low bias** and **high variance.**

The chances of occurrence of overfitting increase as much we provide training to our model. It means the more we train our model, the more chances of occurring the overfitted model.

Overfitting is the main problem that occurs in [supervised learning](https://www.javatpoint.com/supervised-machine-learning).

**Example:** The concept of the overfitting can be understood by the below graph of the linear regression output:



As we can see from the above graph, the model tries to cover all the data points present in the scatter plot. It may look efficient, but in reality, it is not so. Because the goal of the regression model to find the best fit line, but here we have not got any best fit, so, it will generate the prediction errors.

How to avoid the Overfitting in Model

Both overfitting and underfitting cause the degraded performance of the machine learning model. But the main cause is overfitting, so there are some ways by which we can reduce the occurrence of overfitting in our model.

* **Cross-Validation**
* **Training with more data**
* **Removing features**
* **Early stopping the training**
* **Regularization**
* **Ensembling**

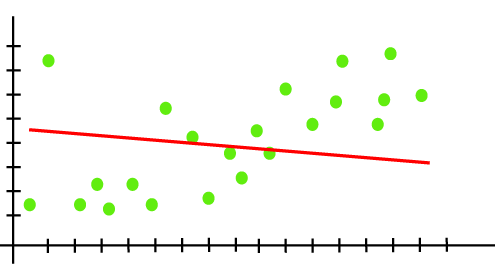
Underfitting

Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data. To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data. As a result, it may fail to find the best fit of the dominant trend in the data.

In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.

An underfitted model has high bias and low variance.

**Example:** We can understand the underfitting using below output of the linear regression model:



As we can see from the above diagram, the model is unable to capture the data points present in the plot.

How to avoid underfitting:

* By increasing the training time of the model.
* By increasing the number of features.

**8.Explain instance based classifier.**

* Instance-based learning are the systems that learn the training examples by heart and then generalizes to new instances based on some similarity measure.
* It is called instance-based because it builds the hypotheses from the training instances.
* It is also known as memory-based learning or lazy-learning (because they delay processing until a new instance must be classified).
* The time complexity of this algorithm depends upon the size of training data.
* Each time whenever a new query is encountered, its previously stores data is examined & assign to a target function value for the new instance.
* The worst-case time complexity of this algorithm is O (n), where n is the number of training instances.

For example, If we were to create a spam filter with an instance-based learning algorithm, instead of just flagging emails that are already marked as spam emails, our spam filter would be programmed to also flag emails that are very similar to them.

1. Instead of estimating for the entire instance set, local approximations can be made to the target function.
2. This algorithm can adapt to new data easily, one which is collected as we go .

* Disadvantages:

1. Classification costs are high
2. Large amount of memory required to store the data, and each query involves starting the identification of a local model from scratch.

* Some of the instance-based learning algorithms

1. K Nearest Neighbor (KNN)
2. Self-Organizing Map (SOM)
3. Learning Vector Quantization (LVQ)
4. Locally Weighted Learning (LWL)
5. Case-Based Reasoning

**9.Explain K- Nearest Neighbor Classifier**

K-Nearest Neighbor(KNN) Algorithm for Machine Learning

* K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
* K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
* K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
* K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
* It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
* KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
* **Example:** Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



Why do we need a K-NN Algorithm?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

* **Step-1:** Select the number K of the neighbors
* **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
* **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
* **Step-4:** Among these k neighbors, count the number of the data points in each category.
* **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
* **Step-6:** Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



* Firstly, we will choose the number of neighbors, so we will choose the k=5.
* Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



* By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:



* As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

How to select the value of K in the K-NN Algorithm?

Below are some points to remember while selecting the value of K in the K-NN algorithm:

* There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
* A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
* Large values for K are good, but it may find some difficulties.

Advantages of KNN Algorithm:

* It is simple to implement.
* It is robust to the noisy training data
* It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

* Always needs to determine the value of K which may be complex some time.
* The computation cost is high because of calculating the distance between the data points for all the training samples.

**10.Explain Bayesian Network**

# Bayesian Belief Network in artificial intelligence

Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:

"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."

It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.

Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

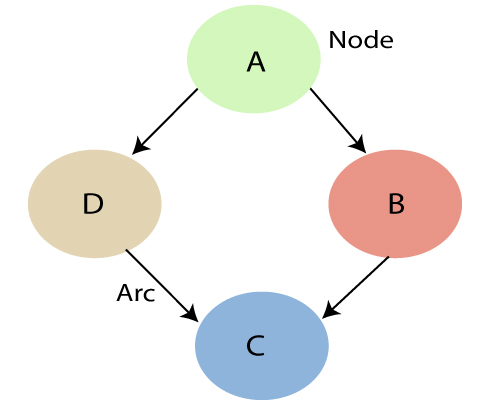
Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network. It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.

Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:

* **Directed Acyclic Graph**
* **Table of conditional probabilities.**

The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

**A Bayesian network graph is made up of nodes and Arcs (directed links), where:**



* Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.
* **Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph.  
  These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other
  + **In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
  + **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
  + **Node C is independent of node A.**

#### Note: The Bayesian network graph does not contain any cyclic graph. Hence, it is known as a directed acyclic graph or DAG.

The Bayesian network has mainly two components:

* **Causal Component**
* **Actual numbers**

Each node in the Bayesian network has condition probability distribution **P(Xi |Parent(Xi) )**, which determines the effect of the parent on that node.

Bayesian network is based on Joint probability distribution and conditional probability. So let's first understand the joint probability distribution:

## Joint probability distribution:

If we have variables x1, x2, x3,....., xn, then the probabilities of a different combination of x1, x2, x3.. xn, are known as Joint probability distribution.

**P[x1, x2, x3,....., xn]**, it can be written as the following way in terms of the joint probability distribution.

**= P[x1| x2, x3,....., xn]P[x2, x3,....., xn]**

**= P[x1| x2, x3,....., xn]P[x2|x3,....., xn]....P[xn-1|xn]P[xn].**

In general for each variable Xi, we can write the equation as:

P(Xi|Xi-1,........., X1) = P(Xi |Parents(Xi ))

## Explanation of Bayesian network:

Let's understand the Bayesian network through an example by creating a directed acyclic graph:

**Example:** Harry installed a new burglar alarm at his home to detect burglary. The alarm reliably responds at detecting a burglary but also responds for minor earthquakes. Harry has two neighbors David and Sophia, who have taken a responsibility to inform Harry at work when they hear the alarm. David always calls Harry when he hears the alarm, but sometimes he got confused with the phone ringing and calls at that time too. On the other hand, Sophia likes to listen to high music, so sometimes she misses to hear the alarm. Here we would like to compute the probability of Burglary Alarm.

**Problem:**

**Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.**

**Solution:**

* The Bayesian network for the above problem is given below. The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but David and Sophia's calls depend on alarm probability.
* The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.
* The conditional distributions for each node are given as conditional probabilities table or CPT.
* Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.
* In CPT, a boolean variable with k boolean parents contains 2K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values

**List of all events occurring in this network:**

* **Burglary (B)**
* **Earthquake(E)**
* **Alarm(A)**
* **David Calls(D)**
* **Sophia calls(S)**

We can write the events of problem statement in the form of probability: **P[D, S, A, B, E]**, can rewrite the above probability statement using joint probability distribution:

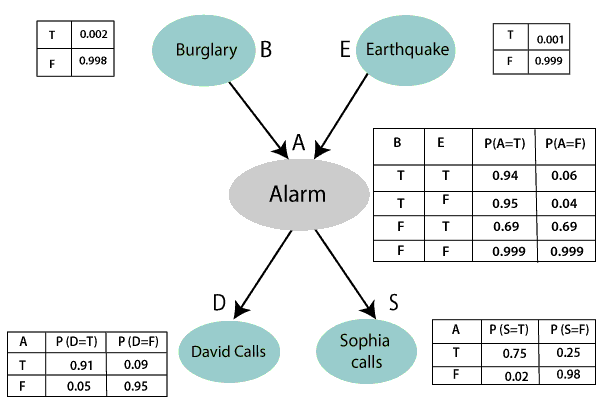
**P[D, S, A, B, E]= P[D | S, A, B, E]. P[S, A, B, E]**

**=P[D | S, A, B, E]. P[S | A, B, E]. P[A, B, E]**

**= P [D| A]. P [ S| A, B, E]. P[ A, B, E]**

**= P[D | A]. P[ S | A]. P[A| B, E]. P[B, E]**

**= P[D | A ]. P[S | A]. P[A| B, E]. P[B |E]. P[E]**



Let's take the observed probability for the Burglary and earthquake component:

P(B= True) = 0.002, which is the probability of burglary.

P(B= False)= 0.998, which is the probability of no burglary.

P(E= True)= 0.001, which is the probability of a minor earthquake

P(E= False)= 0.999, Which is the probability that an earthquake not occurred.

We can provide the conditional probabilities as per the below tables:

**Conditional probability table for Alarm A:**

The Conditional probability of Alarm A depends on Burglar and earthquake:

|  |  |  |  |
| --- | --- | --- | --- |
| **B** | **E** | **P(A= True)** | **P(A= False)** |
| True | True | 0.94 | 0.06 |
| True | False | 0.95 | 0.04 |
| False | True | 0.31 | 0.69 |
| False | False | 0.001 | 0.999 |

**Conditional probability table for David Calls:**

The Conditional probability of David that he will call depends on the probability of Alarm.

|  |  |  |
| --- | --- | --- |
| **A** | **P(D= True)** | **P(D= False)** |
| True | 0.91 | 0.09 |
| False | 0.05 | 0.95 |

**Conditional probability table for Sophia Calls:**

The Conditional probability of Sophia that she calls is depending on its Parent Node "Alarm."

|  |  |  |
| --- | --- | --- |
| **A** | **P(S= True)** | **P(S= False)** |
| True | 0.75 | 0.25 |
| False | 0.02 | 0.98 |

From the formula of joint distribution, we can write the problem statement in the form of probability distribution:

**P(S, D, A, ¬B, ¬E) = P (S|A) \*P (D|A)\*P (A|¬B ^ ¬E) \*P (¬B) \*P (¬E).**

= 0.75\* 0.91\* 0.001\* 0.998\*0.999

**= 0.00068045.**

**Hence, a Bayesian network can answer any query about the domain by using Joint distribution.**

**The semantics of Bayesian Network:**

There are two ways to understand the semantics of the Bayesian network, which is given below:

**1. To understand the network as the representation of the Joint probability distribution.**

It is helpful to understand how to construct the network.

**2. To understand the network as an encoding of a collection of conditional independence statements.**