Naive Bayes Classifiers

Last Updated : 10 Jul, 2024

A Naive Bayes classifiers, a family of algorithms based on Bayes' Theorem. Despite the "naive" assumption of feature independence, these classifiers are widely utilized for their simplicity and efficiency in machine learning. The article delves into theory, implementation, and applications, shedding light on their practical utility despite oversimplified assumptions.

What is Naive Bayes Classifiers?

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. To start with, let us consider a dataset.

One of the most simple and effective classification algorithms, the Naïve Bayes classifier aids in the rapid development of machine learning models with rapid prediction capabilities.

Naïve Bayes algorithm is used for classification problems. It is highly used in text classification. In text classification tasks, data contains high dimension (as each word represent one feature in the data). It is used in spam filtering, sentiment detection, rating classification etc. The advantage of using naïve Bayes is its speed. It is fast and making prediction is easy with high dimension of data.

This model predicts the probability of an instance belongs to a class with a given set of feature value. It is a probabilistic classifier. It is because it with no relation between each other. In real world, this condition satisfies rarely. It uses Bayes theorem in the algorithm for training and prediction

Why it is Called Naive Bayes?

The "Naive" part of the name indicates the simplifying assumption made by the Naïve Bayes classifier. The classifier assumes that the features used to describe an observation are conditionally independent, given the class label. The "Bayes" part of the name refers to Reverend Thomas Bayes, an 18th-century statistician and theologian who formulated Bayes' theorem.

Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as fit("Yes") or unfit("No") for playing golf.Here is a tabular representation of our dataset.

	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

	Outlook	Temperature	Humidity	Windy	Play Golf
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**.

- 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 of Naive Bayes

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

- **Feature independence:** The features of the data are conditionally independent of each other, given the class label.
- **Continuous features are normally distributed:** If a feature is continuous, then it is assumed to be normally distributed within each

each class.

- **Features are equally important:** All features are assumed to contribute equally to the prediction of the class label.
- **No missing data:** The data should not contain any missing values.

With relation to our dataset, this concept can be understood as:

- We assume that no pair of features are dependent. For example, the temperature being 'Hot' has nothing to do with the humidity or the outlook being 'Rainy' has no effect on the winds. Hence, the features are assumed to be **independent**.
- Secondly, each feature is given the same weight(or importance). For example, knowing only temperature and humidity alone can't predict the outcome accurately. None of the attributes is irrelevant and assumed to be contributing **equally** to the outcome.

> *The assumptions made by Naive Bayes are not generally correct in real-world situations. In-fact, the independence assumption is never correct but often works well in practice.Now, before moving to the formula for Naive Bayes, it is important to know about Bayes' theorem.*

Bayes' Theorem

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

where A and B are events and P(B) ≠ 0

- Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as **evidence**.
- P(A) is the **priori** of A (the prior probability, i.e. Probability of event

- P(B) is Marginal Probability: Probability of Evidence.
- P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.
- P(B|A) is Likelihood probability i.e the likelihood that a hypothesis will come true based on the evidence.

Now, with regards to our dataset, we can apply Bayes' theorem in following way:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

where, y is class variable and X is a dependent feature vector (of size *n*) where:

$$X = (x_1, x_2, x_3, ...,x_n)$$

Just to clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of dataset)

```
X = (Rainy, Hot, High, False)
y = No
```

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

With relation to our dataset, this concept can be understood as:

- We assume that no pair of features are dependent. For example, the temperature being 'Hot' has nothing to do with the humidity or the outlook being 'Rainy' has no effect on the winds. Hence, the features are assumed to be **independent**.
- Secondly, each feature is given the same weight(or importance). For example, knowing only temperature and humidity alone can't predict the outcome accurately. None of the attributes is irrelevant and assumed to be contributing **equally** to the outcome.

independent parts.

Now, if any two events A and B are independent, then,

```
P(A,B) = P(A)P(B)
```

Hence, we reach to the result:

$P(y|x_1,\ldots,x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)P(y)}{P(x_1)P(x_2)\ldots P(x_n)}$

which can be expressed as:

$P(y|x_1,\ldots,x_n) = \frac{P(y)\prod_{i=1}^{n}P(x_i|y)}{P(x_1)P(x_2)\ldots P(x_n)}$

Now, as the denominator remains constant for a given input, we can remove that term:

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

Now, we need to create a classifier model. For this, we find the probability of given set of inputs for all possible values of the class variable *y* and pick up the output with maximum probability. This can be expressed mathematically as:

$y = argmax_y P(y)\prod_{i=1}^{n}P(x_i|y)$

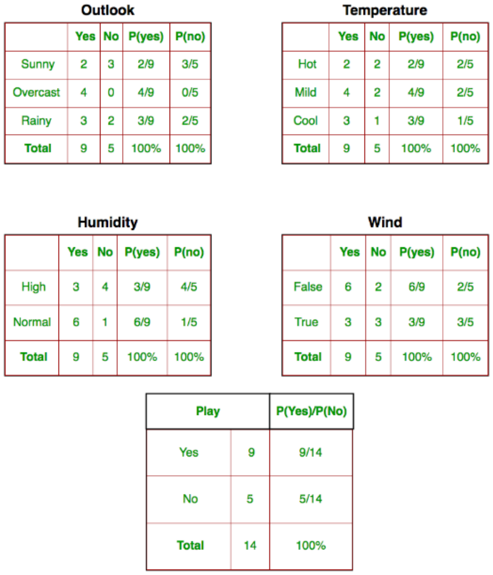
So, finally, we are left with the task of calculating $P(y)$and $P(x_i|y)$.

Please note that $P(y)$ is also called class probability and $P(x_i|y)$ is called conditional probability.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i|y)$.

Let us try to apply the above formula manually on our weather dataset. For this, we need to do some precomputations on our dataset.

We need to find$P(x_i|y_j)$for each x_i in X andy_j in y. All these calculations have been demonstrated in the tables below:



Outlook

	Yes	No	P(yes)	P(no)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rainy	3	2	3/9	2/5
Total	9	5	100%	100%

Temperature

	Yes	No	P(yes)	P(no)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
Total	9	5	100%	100%

Humidity

	Yes	No	P(yes)	P(no)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
Total	9	5	100%	100%

Wind

	Yes	No	P(yes)	P(no)
False	6	2	6/9	2/5
True	3	3	3/9	3/5
Total	9	5	100%	100%

Play		P(Yes)/P(No)
Yes	9	9/14
No	5	5/14
Total	14	100%

Naive Bayes Classifiers

So, in the figure above, we have calculated $P(x_i \mid y_j)$ for each x_i in X and y_j in y manually in the tables 1-4. For example, probability of playing golf given that the temperature is cool, i.e P(temp. = cool | play golf = Yes) = 3/9.

Also, we need to find class probabilities $P(y)$ which has been calculated in the table 5. For example, P(play golf = Yes) = 9/14.

So now, we are done with our pre-computations and the classifier is ready!

Let us test it on a new set of features (let us call it today):

```
today = (Sunny, Hot, Normal, False)
```

$$P(Yes|today) = \frac{P(SunnyOutlook|Yes)P(HotTemperature|Yes)P(NormalHumidity|Yes)P(NoWind|Yes)P(Yes)}{P(today)}$$

and probability to not play golf is given by:

$$P(No|today) = \frac{P(SunnyOutlook|No)P(HotTemperature|No)P(NormalHumidity|No)P(NoWind|No)P(No)}{P(today)}$$

$P(Yes|today) \propto \frac{3}{9}.\frac{2}{9}.\frac{6}{9}.\frac{6}{9}.\frac{9}{14} \approx 0.02116$

and

$P(No|today) \propto \frac{3}{5}.\frac{2}{5}.\frac{1}{5}.\frac{2}{5}.\frac{5}{14} \approx 0.0068$

Now, since

$P(Yes|today) + P(No|today) = 1$

These numbers can be converted into a probability by making the sum equal to 1 (normalization):

$P(Yes|today) = \frac{0.02116}{0.02116+0.0068} \approx 0.0237$

and

$P(No|today) = \frac{0.0068}{0.0141+0.0068} \approx 0.33$

Since

$P(Yes|today) > P(No|today)$

So, prediction that golf would be played is ‘Yes’.

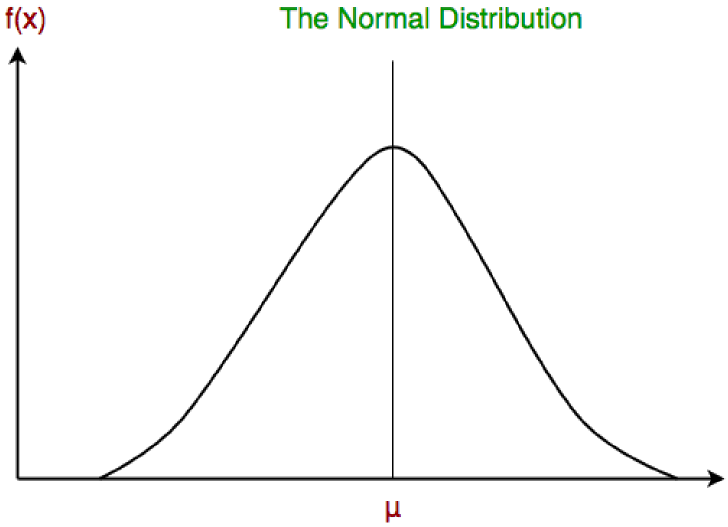
The method that we discussed above is applicable for discrete data. In case of continuous data, we need to make some assumptions regarding the distribution of values of each feature. The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i|y)$.

Types of Naive Bayes Model

There are three types of Naive Bayes Model:

Gaussian Naive Bayes classifier

In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution When plotted, it gives a bell shaped curve which is symmetric about the mean of the feature values as shown below:



Updated table of prior probabilities for outlook feature is as following:

The likelihood of the features is assumed to be Gaussian, hence, conditional probability is given by:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Now, we look at an implementation of Gaussian Naive Bayes classifier using scikit-learn.

	Yes	No	P(Yes)	P(No)
Sunny	3	2	3/9	2/5
Rainy	4	0	4/9	0/5
Overcast	2	3	2/9	3/5
Total	9	5	100%	100%

```
# load the iris dataset
from sklearn.datasets import load_iris
iris = load_iris()

# store the feature matrix (X) and response vector (y)
X = iris.data
y = iris.target

# splitting X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,

# training the model on training set
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# making predictions on the testing set
y_pred = gnb.predict(X_test)

# comparing actual response values (y_test) with predicted
from sklearn import metrics
print("Gaussian Naive Bayes model accuracy(in %):", metric
y_pred)*100)
```

Output:

```
Gaussian Naive Bayes model accuracy(in %): 95.0
```

Multinomial Naive Bayes

Feature vectors represent the frequencies with which certain events have been generated by a multinomial distribution. This is the event model typically used for document classification.

In the multivariate Bernoulli event model, features are independent booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks, where binary term occurrence(i.e. a word occurs in a document or not) features are used rather than term frequencies(i.e. frequency of a word in the document).

Advantages of Naive Bayes Classifier

- Easy to implement and computationally efficient.
- Effective in cases with a large number of features.
- Performs well even with limited training data.
- It performs well in the presence of categorical features.
- For numerical features data is assumed to come from normal distributions

Disadvantages of Naive Bayes Classifier

- Assumes that features are independent, which may not always hold in real-world data.
- Can be influenced by irrelevant attributes.
- May assign zero probability to unseen events, leading to poor generalization.

Applications of Naive Bayes Classifier

- **Spam Email Filtering**: Classifies emails as spam or non-spam based on features.
- **Text Classification**: Used in sentiment analysis, document categorization, and topic classification.
- **Medical Diagnosis:** Helps in predicting the likelihood of a disease based on symptoms.
- **Credit Scoring:** Evaluates creditworthiness of individuals for loan approval.
- **Weather Prediction**: Classifies weather conditions based on various

As we reach to the end of this article, here are some important points to ponder upon:

- In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters.
- Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution. This in turn helps to alleviate problems stemming from the curse of dimensionality.

Conclusion

In conclusion, Naive Bayes classifiers, despite their simplified assumptions, prove effective in various applications, showcasing notable performance in document classification and spam filtering. Their efficiency, speed, and ability to work with limited data make them valuable in real-world scenarios, compensating for their naive independence assumption.

Frequently Asked Questions on Naive Bayes Classifiers

What is Naive Bayes real example?

> *Naive Bayes is a simple probabilistic classifier based on Bayes' theorem. It assumes that the features of a given data point are independent of each other, which is often not the case in reality. However, despite this simplifying assumption, Naive Bayes has been shown to be surprisingly effective in a wide range of applications.*

Why is it called Naive Bayes?

not true in reality, but it does make the algorithm much simpler to compute.

What is an example of a Bayes classifier?

A Bayes classifier is a type of classifier that uses Bayes' theorem to compute the probability of a given class for a given data point. Naive Bayes is one of the most common types of Bayes classifiers.

What is better than Naive Bayes?

There are several classifiers that are better than Naive Bayes in some situations. For example, logistic regression is often more accurate than Naive Bayes, especially when the features of a data point are correlated with each other.

Can Naive Bayes probability be greater than 1?

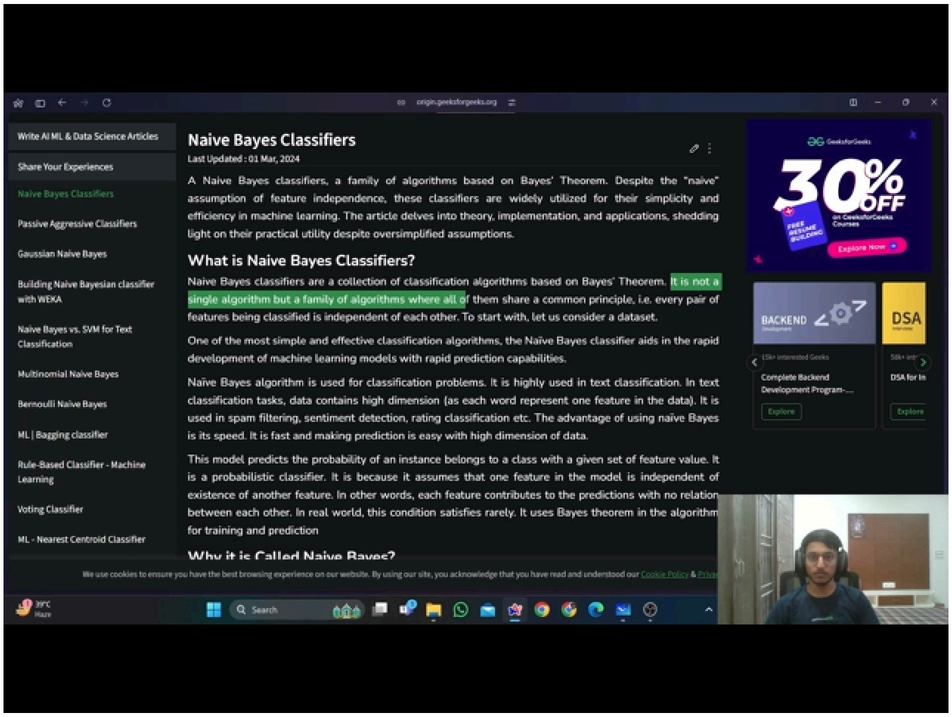
No, the probability of an event cannot be greater than 1. The probability of an event is a number between 0 and 1, where 0 indicates that the event is impossible and 1 indicates that the event is certain.

Are you passionate about data and looking to make one giant leap into your career? Our **Data Science Course** will help you change your game and, most importantly, allow students, professionals, and working adults to tide over into the data science immersion. Master state-of-the-art methodologies, powerful tools, and industry best practices, hands-on

GeeksforGeeks 184

Next Article

Passive Aggressive Classifiers


origin.geeksforgeeks.org
Write AI ML & Data Science Articles
Share Your Experiences
Naive Bayes Classifiers
Passive Aggressive Classifiers
Gaussian Naive Bayes
Building Naive Bayesian classifier with WEKA
Naive Bayes vs. SVM for Text Classification
Multinomial Naive Bayes
Bernoulli Naive Bayes
ML | Bagging classifier
Rule-Based Classifier - Machine Learning
Voting Classifier
ML - Nearest Centroid Classifier
30% OFF
on GeeksforGeeks Courses
Explore Now
BACKEND
DSA
Complete Backend Development Program-...
Explore
Search

Naive Bayes Classifiers

Last Updated : 01 Mar, 2024

A Naive Bayes classifiers, a family of algorithms based on Bayes' Theorem. Despite the "naive" assumption of feature independence, these classifiers are widely utilized for their simplicity and efficiency in machine learning. The article delves into theory, implementation, and applications, shedding light on their practical utility despite oversimplified assumptions.

What is Naive Bayes Classifiers?

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. To start with, let us consider a dataset.

One of the most simple and effective classification algorithms, the Naïve Bayes classifier aids in the rapid development of machine learning models with rapid prediction capabilities.

Naïve Bayes algorithm is used for classification problems. It is highly used in text classification. In text classification tasks, data contains high dimension (as each word represent one feature in the data). It is used in spam filtering, sentiment detection, rating classification etc. The advantage of using naïve Bayes is its speed. It is fast and making prediction is easy with high dimension of data.

This model predicts the probability of an instance belongs to a class with a given set of feature value. It is a probabilistic classifier. It is because it assumes that one feature in the model is independent of existence of another feature. In other words, each feature contributes to the predictions with no relation between each other. In real world, this condition satisfies rarely. It uses Bayes theorem in the algorithm for training and prediction

Why it is Called Naive Bayes?

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our Cookie Policy & Priva

Naive Bayes Classifiers

Similar Reads

Variable importance for support vector machine and naive Bayes...

Understanding the importance of variables in a model is crucial for interpreting and improving the model's performance. Variable importance

Complement Naive Bayes (CNB) Algorithm

Naive Bayes algorithms are a group of very popular and commonly used Machine Learning algorithms used for classification. There are many differe...

7 min read

Bernoulli Naive Bayes

Supervised learning is a subcategory of machine learning algorithms. In this way, the models are trained on labeled datasets. Under supervised learning...

5 min read

Gaussian Naive Bayes

In the vast field of machine learning, classification algorithms play a pivotal role in making sense of data. One such algorithm, Gaussian Naive Bayes,...

6 min read

Gaussian Naive Bayes using Sklearn

In the world of machine learning, Gaussian Naive Bayes is a simple yet powerful algorithm used for classification tasks. It belongs to the Naive...

8 min read

Classification of Text Documents using the approach of Naive Bayes

In natural language processing and machine learning, the Naïve Bayes approach is a potent and popular method for classifying text documents....

10 min read

Multinomial Naive Bayes

Multinomial Naive Bayes (MNB) is a very popular and efficient machine learning algorithm that is based on Bayes' theorem. It is commonly used for...

9 min read

9 min read

Applying Multinomial Naive Bayes to NLP Problems

Multinomial Naive Bayes (MNB) is a popular machine learning algorithm for text classification problems in Natural Language Processing (NLP). It is...

7 min read

ML | Naive Bayes Scratch Implementation using Python

A key concept in probability theory, the Bayes theorem, provides the foundation for the probabilistic classifier known as Naive Bayes. It is a simp...

8 min read

Article Tags : AI-ML-DS Machine Learning Machine Learning

Practice Tags : Machine Learning Machine Learning

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:-
A-143, 9th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305) | Registered Address:- K 061,
Tower K, Gulshan Vivante Apartment,
Sector 137, Noida, Gautam Buddh
Nagar, Uttar Pradesh, 201305

In Media

Contact Us

Advertise with us

GFG Corporate Solution

Placement Training Program

Offline Classes (Delhi/NCR)

DSA in JAVA/C++

Master System Design

Master CP

GeeksforGeeks Videos

Geeks Community

Languages

Python

Java

C++

PHP

GoLang

SQL

R Language

Android Tutorial

DSA

Data Structures

Algorithms

DSA for Beginners

Basic DSA Problems

DSA Roadmap

DSA Interview Questions

Competitive Programming

Data Science & ML

Data Science With Python

Data Science For Beginner

Machine Learning

ML Maths

Data Visualisation

Pandas

NumPy

NLP

Deep Learning

Web Technologies

HTML

CSS

JavaScript

TypeScript

ReactJS

NextJS

NodeJs

Bootstrap

Tailwind CSS

Python Tutorial

- Python Programming Examples
- Django Tutorial
- Python Projects
- Python Tkinter
- Web Scraping
- OpenCV Tutorial
- Python Interview Question

Computer Science

- GATE CS Notes
- Operating Systems
- Computer Network
- Database Management System
- Software Engineering
- Digital Logic Design
- Engineering Maths

DevOps

- Git
- AWS
- Docker
- Kubernetes
- Azure
- GCP
- DevOps Roadmap

System Design

- High Level Design
- Low Level Design
- UML Diagrams
- Interview Guide
- Design Patterns
- OOAD
- System Design Bootcamp
- Interview Questions

School Subjects

- Mathematics
- Physics
- Chemistry
- Biology
- Social Science
- English Grammar

Commerce

- Accountancy
- Business Studies
- Economics
- Management
- HR Management
- Finance
- Income Tax

Databases

SQL

MYSQL

PostgreSQL

PL/SQL

MongoDB

Competitive Exams

JEE Advanced

UGC NET

UPSC

SSC CGL

SBI PO

SBI Clerk

IBPS PO

IBPS Clerk

Free Online Tools

Typing Test

Image Editor

Code Formatters

Code Converters

Currency Converter

Random Number Generator

Random Password Generator

DSA/Placements

DSA - Self Paced Course

DSA in JavaScript - Self Paced Course

DSA in Python - Self Paced

C Programming Course Online - Learn C with Data Structures

Complete Interview Preparation

Master Competitive Programming

Core CS Subject for Interview Preparation

Mastering System Design: LLD to HLD

Tech Interview 101 - From DSA to System Design [LIVE]

DSA to Development [HYBRID]

Placement Preparation Crash Course [LIVE]

Machine Learning/Data Science

Preparation Corner

Company-Wise Recruitment Process

Resume Templates

Aptitude Preparation

Puzzles

Company-Wise Preparation

Companies

Colleges

More Tutorials

Software Development

Software Testing

Product Management

Project Management

Linux

Excel

All Cheat Sheets

Recent Articles

Write & Earn

Write an Article

Improve an Article

Pick Topics to Write

Share your Experiences

Internships

Development/Testing

JavaScript Full Course

React JS Course

React Native Course

Django Web Development Course

Complete Bootstrap Course

Full Stack Development - [LIVE]

JAVA Backend Development - [LIVE]

Complete Software Testing Course [LIVE]

Android Mastery with Kotlin [LIVE]

Programming Languages