SINHGAD TECHNICAL EDUCATION SOCIETY'S

SINHGAD INSTITUTE OF TECHNOLOGY

Kusgaon (Bk), Lonavala 410401

DEPARTMENT OF COMPUTER ENGINEERING

LABORATORY MANUAL

Laboratory Practice - III [410246]

B.E. Computer (SEM – I)

AY 2022-23

Prepared By

Asst. Prof. R. S. Shishupal

TEACHING SCHEME

Practical: 4 Hrs/Week

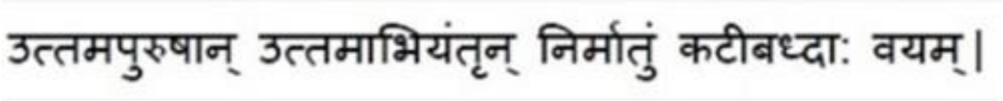
EXAMINATION SCHEME

Term Work: 50 Marks

Practical : 50 Marks

Vision and Mission of Institute

VISION



उत्तमपुरुषान् उत्तमाभियंतृन् निर्मातुं कटीबध्दा: वयम् |

We are committed to produce not only good engineers but good human beings, also.

MISSION

- We believe in and work for the holistic development of students and teachers.
- We strive to achieve this by imbibing a unique value system, transparent work culture, excellent academic and physical environment conducive to learning, creativity and technology transfer.

Vision and Mission of the Department

VISION

Computer engineering department in association with user industry will harness knowledge and potential for application based product development in future, through world-class education to empower the society around.

MISSION

Computer engineering department will be the widely recognized center of excellence for promoting value added engineering education. We will contribute by evolving innovative technology solutions to solve a wide range of complex scientific, technological and social problems.

Short Term Goals

- To continuously upgrade existing resources through qualified and experienced faculty and state-of-the-art laboratories.
- To initiate relevant value addition programs and certifications for improving employability.
- To strengthen the Institute-Industry relationship for mutual benefit.

Long Term Goals

- To establish a center of innovations in Agriculture, Tele-health and ICT sector in collaboration with Industry.
- To create center of excellence in network, security and computer vision.
- To establish a world class R&D institute for patent based research creating opportunities for faculty to be resource persons.

Program Educational Objectives (PEO’s)

1) To prepare globally competent graduates having strong fundamentals and domain knowledge to provide effective solutions for engineering problems.
2) To prepare the graduates to work as a committed professionals with strong professional ethics and values, sense of responsibilities, understanding of legal, safety, health, societal, cultural and environmental issues.
3) To prepare committed and motivated graduates with research attitude, lifelong learning, investigative approach, and multidisciplinary thinking.
4) To prepare the graduates with strong managerial and communication skills to work effectively as individual as well as in teams.

Program Outcomes: POs

Students are expected to know and be able –

1) To apply knowledge of mathematics, science, engineering fundamentals, problem solving skills, algorithmic analysis and mathematical modeling to the solution of complex engineering problems.
2) To analyze the problem by finding its domain and applying domain specific skills
3) To understand the design issues of the product/software and develop effective solutions with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
4) To find solutions of complex problems by conducting investigations applying suitable techniques.
5) To adapt the usage of modern tools and recent software.
6) To contribute towards the society by understanding the impact of Engineering on global aspect.
7) To understand environment issues and design a sustainable system.
8) To understand and follow professional ethics.
9) To function effectively as an individual and as member or leader in diverse teams and interdisciplinary settings.
10) To demonstrate effective communication at various levels.

11) To apply the knowledge of Computer Engineering for development of projects, and its finance and management.

12) To keep in touch with current technologies and inculcate the practice of lifelong learning

Program Specific Outcomes: PSOs

A graduate of the Computer Engineering Program will demonstrate-

PSO1: Professional Skills-The ability to understand, analyze and develop computer programs in the areas related to algorithms, system software, multimedia, web design, big data analytics, and networking for efficient design of computer-based systems of varying.

PSO2: Problem-Solving Skills- The ability to apply standard practices and strategies in software project development using open-ended programming environments to deliver a quality product for business success.

PSO3: Successful Career and Entrepreneurship- The ability to employ modern computer languages, environments, and platforms in creating innovative career paths to be an entrepreneur, and a zest for higher studies.

Course Objectives:

Learn effect of data preprocessing on the performance of machine learning algorithms

● Develop in depth understanding for implementation of the regression models.

● Implement and evaluate supervised and unsupervised machine learning algorithms.

● Analyze performance of an algorithm.

● Learn how to implement algorithms that follow algorithm design strategies namely divide and conquer, greedy, dynamic programming, backtracking, branch and bound.

● Understand and explore the working of Blockchain technology and its applications.

Course Outcomes:

After completion of the course, students will be able to

CO1: Apply preprocessing techniques on datasets.

CO2: Implement and evaluate linear regression and random forest regression models.

CO3: Apply and evaluate classification and clustering techniques.

CO4: Analyze performance of an algorithm.

CO5: Implement an algorithm that follows one of the following algorithm design strategies: divide and conquer, greedy, dynamic programming, backtracking, branch and bound.

CO6: Interpret the basic concepts in Blockchain technology and its applications

CERTIFICATE

This is to certify that Mr. /Ms ____ of class BE COMPUTER Div _____ Roll No.__________ Examination Seat No./PRN No.______________________________ has completed all the practical work in the **Laboratory Practice III [410246]** satisfactorily, as prescribed by Savitribai Phule Pune University , Pune in the academic year 2022 - 23 (Semester I).

Course In-charge	**Head of Department**	**Principal**
Ms. R. S. Shishupal	Dr. S. D. Babar	Dr. M. S. Gaikwad

Date:

INDEX

Sr. No	Title of experiment	Date of Performance	Marks Obtained (10)	Signature of Faculty
Group A: Design and Analysis of Algorithms				
1	Fibonacci Series	18/10/2022		
2	Huffman Encoding	31/10/2022		
3	Fractional Knapsack Problem	01/11/2022		
4	Quick Sort	07/11/2022		
5	N-Queen Problem	08/11/2022		
Group B: Machine Learning				
6	Uber Ride Fare Prediction	25/08/2022		
7	Email Spam Classification	01/09/2022		
8	Bank Customer Churn Modeling	08/09/2022		
9	K-Means Clustering	12/09/2022		
10	Gradient Descent algorithm	19/09/2022		
11	K-Nearest Neighbors	20/09/2022		
Group C: Blockchain Technology				
12	Installation of MetaMask	26/09/2022		
13	Creation of wallet using Metamask for Crypto transactions	26/09/2022		
14	Smart contract for Bank Account	29/10/2022		
15	Smart contract for Student data	17/10/2022		
16	Survey report on types of Blockchains	17/10/2022		

Name & Signature of Course In-charge

Sinhgad Institutes

Name of the Student: ________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

Design and Analysis of Algorithms

Assignment No. 01

FIBONACCI SERIES

Marks:

Date of Performance: / /2022 **Sign with Date:**

Title: Calculate Fibonacci series using non-recursive and recursive function.

Objectives:

- To understand the concept of recursive function.
- To calculate Fibonacci series using non-recursive and recursive function.

Outcomes:

- Implement non-recursive and recursive function to calculate Fibonacci series.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 5 PSOs: 1 COs: 1

Problem Statement:

Write a program non-recursive and recursive program to calculate Fibonacci numbers and analyze their time and space complexity.

Theory:

Fibonacci series

Fibonacci series is a series of numbers formed by the addition of the preceding two numbers in the series. The first two terms are zero and one respectively. The terms after this are generated by simply adding the previous two terms.

There are two ways to write the fibonacci series program:

- Fibonacci Series without using recursion
- Fibonacci Series using recursion

How do you calculate Fibonacci?

The Fibonacci Sequence is the series of numbers: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, …

The next number is found by adding up the two numbers before it:

- 2 is calculated by adding the two numbers preceding it (1+1),
- 3 is calculated by adding the two numbers preceding it (1+2),
- 5 is (2+3), and so on..

Here is an example of Fibonacci series: 0,1,1,2,3,5,8,13….etc.

In the above example, 0 and 1 are the first two terms of the series. These two terms are printed directly. The third term is made by adding the first two terms. In this case 0 and 1. So, we get 0+1=1. Hence 1 is printed as the third term. The next term is generated by using the second and third term and not using the first term. It is done until the number of terms you want or requested by the user. In the above example, we have used eight terms.

Recursive algorithm to get Fibonacci sequence:

```
1. START
2. Input the non-negative integer ‘n’
3. If (n==o || n==1)
      return n;
```

```
  else
     return fib(n-1)+fib(n-2);
4. Print, n^th Fibonacci number
5. END
```

Conclusion:

Fibonacci series is calculated using non-recursive and recursive function

A. Write short answer of following questions:

1. How are Fibonacci numbers generated?
2. Can we print Fibonacci series using recursion?
3. What is the time complexity of Fibonacci series using recursion?
4. What are the rules used for recursive function?

Sinhgad Institutes

Name of the Student: ____________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

Design and Analysis of Algorithms

Assignment No. 02

HUFFMAN ENCODING

Marks:

Date of Performance: / /2022

Sign with Date:

Title: Implement Huffman Encoding using a greedy strategy.

Objectives:

- To understand the greedy strategy.
- To implement Huffman Encoding using a greedy strategy.

Outcomes:

- Implement Huffman Encoding using a greedy strategy.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 5 PSOs: 1 COs: 1

Problem Statement:

Write a program to implement Huffman Encoding using a greedy strategy.

Theory:

What is a Greedy Method?

● A greedy algorithm is an approach for solving a problem by selecting the best option available at the moment. It doesn't worry whether the current best result will bring the overall optimal result.

● The algorithm never reverses the earlier decision even if the choice is wrong. It works in a top-down approach.

● This algorithm may not produce the best result for all the problems. It's because it always goes for the local best choice to produce the global best result.

Advantages of Greedy Approach

● The algorithm is **easier to describe**.

● This algorithm can **perform better** than other algorithms (but, not in all cases).

Drawback of Greedy Approach

● As mentioned earlier, the greedy algorithm doesn't always produce the optimal solution. This is the major disadvantage of the algorithm

● For example, suppose we want to find the longest path in the graph below from root to leaf.

Greedy Algorithm

1. To begin with, the solution set (containing answers) is empty.
2. At each step, an item is added to the solution set until a solution is reached.
3. If the solution set is feasible, the current item is kept.
4. Else, the item is rejected and never considered again.

Huffman Encoding

Huffman encoding implements the following steps.

- It assigns a variable-length code to all the given characters.
- The code length of a character depends on how frequently it occurs in the given text or string.
- A character gets the smallest code if it frequently occurs.
- A character gets the largest code if it least occurs.

Huffman coding follows a **prefix rule** that prevents ambiguity while decoding. The rule also ensures that the code assigned to the character is not treated as a prefix of the code assigned to any other character.

There are the following two major steps involved in Huffman coding:

- First, construct a **Huffman tree** from the given input string or characters or text.
- Assign, a Huffman code to each character by traversing over the tree.

Let's brief the above two steps.

Huffman Tree

Step 1: For each character of the node, create a leaf node. The leaf node of a character contains the frequency of that character.

Step 2: Set all the nodes in sorted order according to their frequency.

Step 3: There may exist a condition in which two nodes may have the same frequency. In such a case, do the following:

1. Create a new internal node.
2. The frequency of the node will be the sum of the frequency of those two nodes that have the same frequency.
3. Mark the first node as the left child and another node as the right child of the newly created internal node.

Step 4: Repeat step 2 and 3 until all the node forms a single tree. Thus, we get a Huffman tree.

Huffman Encoding Example

Suppose, we have to encode string **abracadabra.** Determine the following:

i. Huffman code for All the characters
ii. Average code length for the given String
iii. Length of the encoded string

(i) Huffman Code for All the Characters

In order to determine the code for each character, first, we construct a **Huffman tree.**

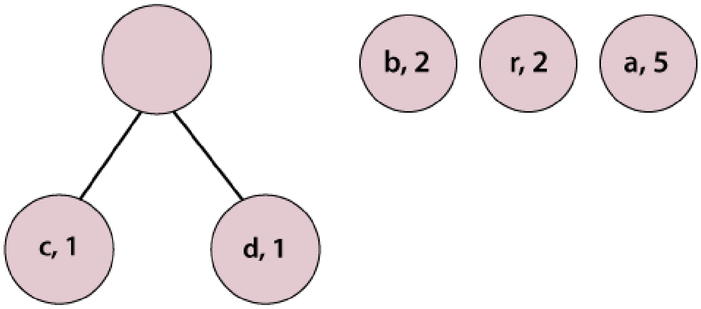
Step 1: Make pairs of characters and their frequencies.

(a, 5), (b, 2), (c, 1), (d, 1), (r, 2)

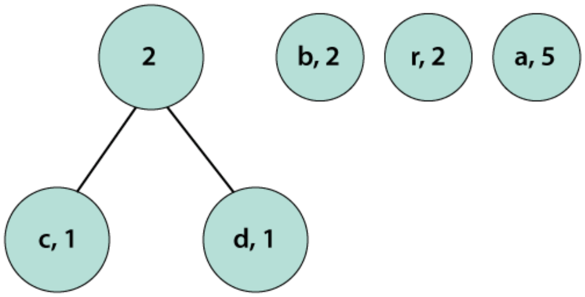
Step 2: Sort pairs with respect to frequency, we get:

(c, 1), (d, 1), (b, 2) (r, 2), (a, 5)

Step 3: Pick the first two characters and join them under a parent node.

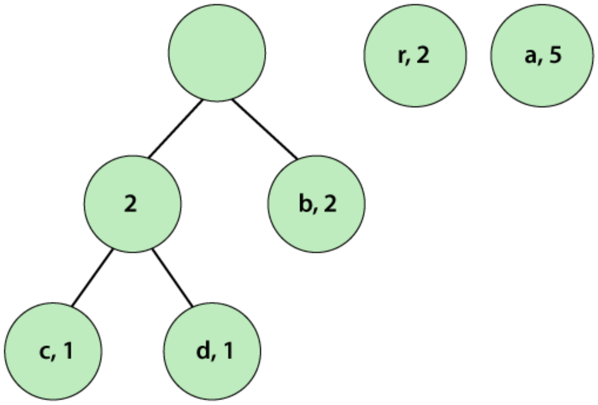


We observe that a parent node does not have a frequency so, we must assign a frequency to it. The parent node frequency will be the sum of its child nodes (left and right) i.e. 1+1=**2.**

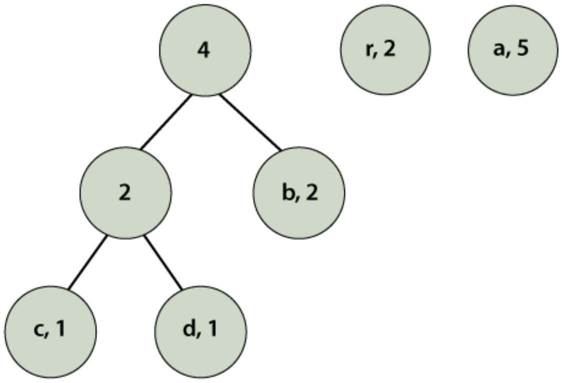


Step 4: Repeat Steps 2 and 3 until, we get a single tree.

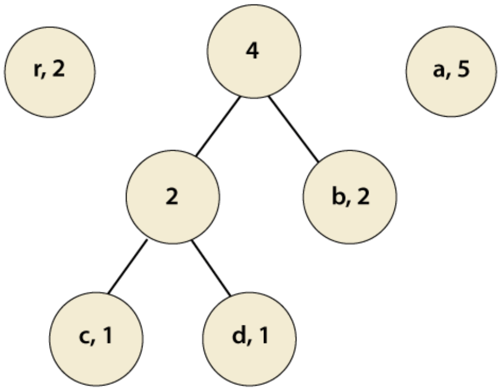
We observe that the pairs are already in a sorted (by step 2) manner. Again, pick the first two pairs and join them.



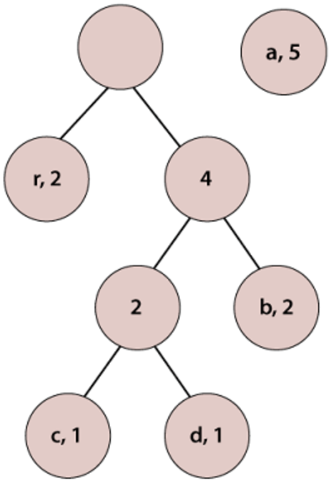
We observe that a parent node does not has a frequency so, we must assign a frequency to it. The parent node frequency will be the sum of its child nodes (left and right) i.e. 2+2=**4.**



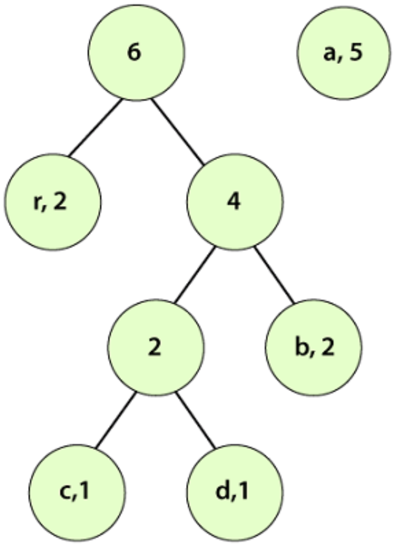
Again, we check if the pairs are in a sorted manner or not. At this step, we need to sort the pairs.



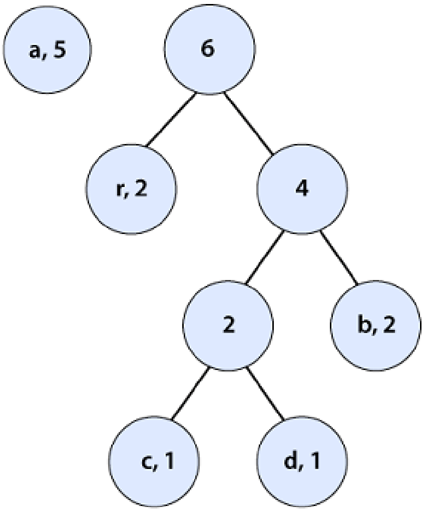
According to step 3, pick the first two pairs and join them, we get:



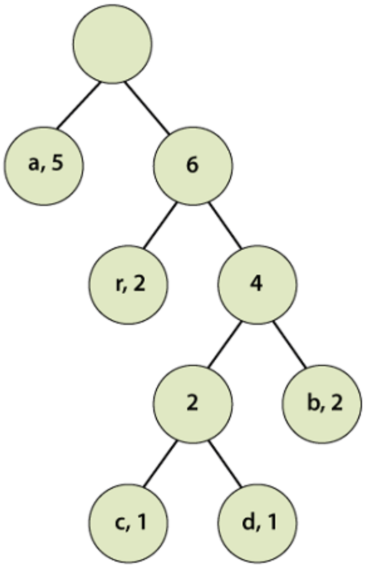
We observe that a parent node does not have a frequency so, we must assign a frequency to it. The parent node frequency will be the sum of its child nodes (left and right) i.e. 2+4=**6.**



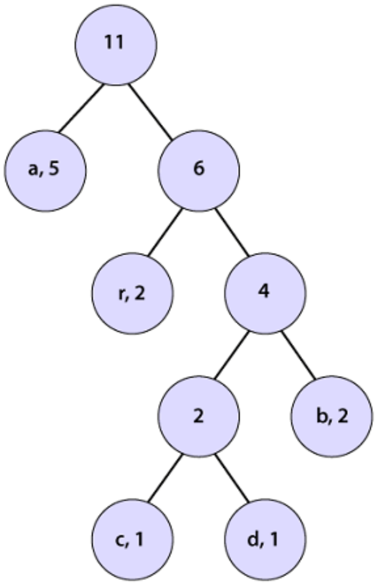
Again, we check if the pairs are in a sorted manner or not. At this step, we need to sort the pairs. After sorting the tree looks like the following:



According to step 3, pick the first two pairs and join them, we get:

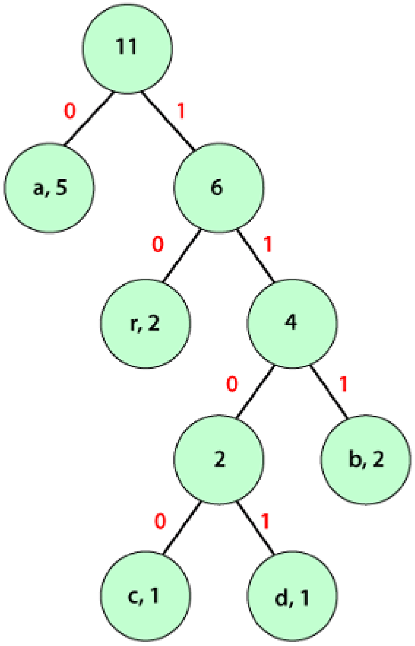


We observe that a parent node does not have a frequency so, we must assign a frequency to it. The parent node frequency will be the sum of its child nodes (left and right) i.e. 5+6=**11.**



Therefore, we get a single tree.

At last, we will find the code for each character with the help of the above tree. Assign a weight to each edge. Note that each **left edge-weighted is 0** and the **right edge-weighted is 1.**



We observe that input characters are only presented in the leave nodes and the internal nodes have null values. In order to find the Huffman code for each character, traverse over the Huffman tree from the root node to the leaf node of that particular character for which we want to find code. The table describes the code and code length for each character.

Character	Frequency	Code	Code Length
A	5	0	1
B	2	111	3
C	1	1100	4
D	1	1101	4
R	2	10	2

We observe that the most frequent character gets the shortest code length and the less frequent character gets the largest code length.

Now we can encode the string **(abracadabra)** that we have taken above.

1. 0 111 10 0 1100 0 1101 0 111 10 0

(ii) Average Code Length for the String

The average code length of the Huffman tree can be determined by using the formula given below:

1. Average Code Length = ∑ (frequency × code length) / ∑ (frequency)

= { (5 x 1) + (2 x 3) + (1 x 4) + (1 x 4) + (2 x 2) } / (5+2+1+1+2)

= 2.09090909

(iii) Length of the Encoded String

The length of the encoded message can be determined by using the following formula:

1. length= Total number of characters in the text x Average code length per character

= 11 x 2.09090909

= 23 bits

Huffman Encoding Algorithm

```
Huffman (C)
n=|C|
Q=C
for i=1 to n-1
   do
      z=allocate_Node()
      x=left[z]=Extract_Min(Q)
      y=right[z]=Extract_Min(Q)
      f[z]=f[x]+f[y]
Insert(Q,z)
return Extract_Min(Q)
```

The Huffman algorithm is a greedy algorithm. Since at every stage the algorithm looks for the best available options.

The time complexity of the Huffman encoding is **O(nlogn).** Where n is the number of characters in the given text.

Huffman Decoding

Huffman decoding is a technique that converts the encoded data into initial data. As we have seen in encoding, the Huffman tree is made for an input string and the characters are decoded based on their position in the tree. The decoding process is as follows:

- Start traversing over the tree from the **root** node and search for the character.
- If we move left in the binary tree, add **0** to the code.
- If we move right in the binary tree, add **1** to the code.

The child node holds the input character. It is assigned the code formed by subsequent 0s and 1s. The time complexity of decoding a string is **O(n),** where n is the length of the string.

Conclusion:
In this way we have explored Concept of Huffman Encoding using greedy method

A. Write short answer of following questions:

1. What is Huffman Encoding?
2. How many bits may be required for encoding the message ‘mississippi’?
3. Which tree is used in Huffman encoding? Give one Example
4. Why Huffman coding is lossless compression?

5. What are the advantages and disadvantages of Huffman encoding technique?

Sinhgad Institutes

Name of the Student: ___________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

Design and Analysis of Algorithms

Assignment No. 03

FRACTIONAL KNAPSACK PROBLEM

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title: Write a program to solve a fractional Knapsack problem using a greedy method.

Objectives:

- To understand greedy method.
- To solve fractional Knapsack problem using greedy method.

Outcomes:

- Implement greedy method to solve fractional Knapsack problem.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 5 PSOs: 1 COs: 1

Problem Statement:

Write a program to solve a fractional Knapsack problem using a greedy method.

Theory:

Greedy method

Among all the algorithmic approaches, the simplest and straightforward approach is the Greedy method. In this approach, the decision is taken on the basis of current available information without worrying about the effect of the current decision in future.

Greedy algorithms build a solution part by part, choosing the next part in such a way, that it gives an immediate benefit. This approach never reconsiders the choices taken previously. This approach is mainly used to solve optimization problems. Greedy method is easy to implement and quite efficient in most of the cases. Hence, we can say that Greedy algorithm is an algorithmic paradigm based on heuristic that follows local optimal choice at each step with the hope of finding global optimal solution.

In many problems, it does not produce an optimal solution though it gives an approximate (near optimal) solution in a reasonable time.

Fractional Knapsack Problem

The fractional knapsack problem is also one of the techniques which are used to solve the knapsack problem. In fractional knapsack, the items are broken in order to maximize the profit. The problem in which we break the item is known as a Fractional knapsack problem.

This problem can be solved with the help of using two techniques:

- Brute-force approach: The brute-force approach tries all the possible solutions with all the different fractions but it is a time-consuming approach.
- Greedy approach: In Greedy approach, we calculate the ratio of profit/weight, and accordingly, we will select the item. The item with the highest ratio would be selected first.

There are basically three approaches to solve the problem:

- The first approach is to select the item based on the maximum profit.
- The second approach is to select the item based on the minimum weight.
- The third approach is to calculate the ratio of profit/weight.

Consider the below example:

Objects: 1 2 3 4 5 6 7

Profit (P): 10 15 7 8 9 4

Weight(w): 1 3 5 4 1 3 2

W (Weight of the knapsack): 15

n (no of items): 7

First approach:

First approach:

Object	Profit	Weight	Remaining weight

3	15	5	15 - 5 = 10
2	10	3	10 - 3 = 7
6	9	3	7 - 3 = 4
5	8	1	4 - 1 = 3
7	7 * ¾ = 5.25	3	3 - 3 = 0

The total profit would be equal to (15 + 10 + 9 + 8 + 5.25) = 47.25

Second approach:

The second approach is to select the item based on the minimum weight.

Object	Profit	Weight	Remaining weight
1	5	1	15 - 1 = 14
5	7	1	14 - 1 = 13
7	4	2	13 - 2 = 11
2	10	3	11 - 3 = 8
6	9	3	8 - 3 = 5
4	7	4	5 - 4 = 1
3	15 * 1/5 = 3	1	1 - 1 = 0

In this case, the total profit would be equal to (5 + 7 + 4 + 10 + 9 + 7 + 3) = 46

Third approach:

In the third approach, we will calculate the ratio of profit/weight.

Objects: 1 2 3 4 5 6 7

Profit (P): 5 10 15 7 8 9 4

Weight(w): 1 3 5 4 1 3 2

In this case, we first calculate the profit/weight ratio.

Object 1: 5/1 = 5

Object 2: 10/3 = 3. 33

Object 3: 15/5 = 3

Object 4: 7/4 = 1.7

Object 5: 8/1 = 8

Object 6: 9/3 = 3

Object 7: 4/2 = 2

P:w: 5 3.3 3 1.7 8 3 2

In this approach, we will select the objects based on the maximum profit/weight ratio. Since the P/W of object 5 is maximum so we select object 5.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 8 = 7

After object 5, object 1 has the maximum profit/weight ratio, i.e., 5. So, we select object 1 shown in the below table:

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13

After object 1, object 2 has the maximum profit/weight ratio, i.e., 3.3. So, we select object 2 having profit/weight ratio as 3.3.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10

After object 2, object 3 has the maximum profit/weight ratio, i.e., 3. So, we select object 3 having profit/weight ratio as 3.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10

3	15	5	10 - 5 = 5

After object 3, object 6 has the maximum profit/weight ratio, i.e., 3. So we select object 6 having profit/weight ratio as 3.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10
3	15	5	10 - 5 = 5
6	9	3	5 - 3 = 2

After object 6, object 7 has the maximum profit/weight ratio, i.e., 2. So we select object 7 having profit/weight ratio as 2.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10
3	15	5	10 - 5 = 5
6	9	3	5 - 3 = 2

7	4	2	2 - 2 = 0

As we can observe in the above table that the remaining weight is zero which means that the knapsack is full. We cannot add more objects in the knapsack. Therefore, the total profit would be equal to (8 + 5 + 10 + 15 + 9 + 4), i.e., 51.

In the first approach, the maximum profit is 47.25. The maximum profit in the second approach is 46. The maximum profit in the third approach is 51. Therefore, we can say that the third approach, i.e., maximum profit/weight ratio is the best approach among all the approaches.

Conclusion:

Fractional Knapsack problem is solved using greedy method.

A. Write short answer of following questions:

1. Why do we use fractional knapsack?
2. Which approach is best in fractional knapsack problem?
3. What is the time complexity of fractional knapsack?
4. What is the difference between 0 1 knapsack and fractional knapsack?

Sinhgad Institutes

Name of the Student: ________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

Design and Analysis of Algorithms

Assignment No. 04

QUICK SORT

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Write a program for analysis of quick sort by using deterministic and randomized variant.

Objectives:

- To implement Deterministic and Randomized Quick sort.

Outcomes:

- To implement Deterministic and Randomized Quick sort.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 4, 5 PSOs: 1, 2 COs: 1

Problem Statement:

Write a program for analysis of quick sort by using deterministic and randomized variant.

Theory:

Quick Sort Algorithm

Sorting is a way of arranging items in a systematic manner. Quicksort is the widely used sorting algorithm that makes **n log n** comparisons in average case for sorting an array of n elements. It is a faster and highly efficient sorting algorithm. This algorithm follows the divide and conquer approach. Divide and conquer is a technique of breaking down the algorithms into subproblems, then solving the subproblems, and combining the results back together to solve the original problem.

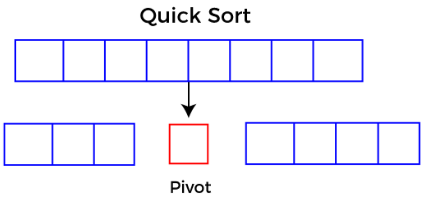
Divide: In Divide, first pick a pivot element. After that, partition or rearrange the array into two sub-arrays such that each element in the left sub-array is less than or equal to the pivot element and each element in the right sub-array is larger than the pivot element.

Conquer: Recursively, sort two subarrays with Quicksort.

Combine: Combine the already sorted array.

Quicksort picks an element as pivot, and then it partitions the given array around the picked pivot element. In quick sort, a large array is divided into two arrays in which one holds values that are smaller than the specified value (Pivot), and another array holds the values that are greater than the pivot.

After that, left and right sub-arrays are also partitioned using the same approach. It will continue until the single element remains in the sub-array.



Choosing the pivot

Picking a good pivot is necessary for the fast implementation of quicksort. However, it is typical to determine a good pivot. Some of the ways of choosing a pivot are as follows -

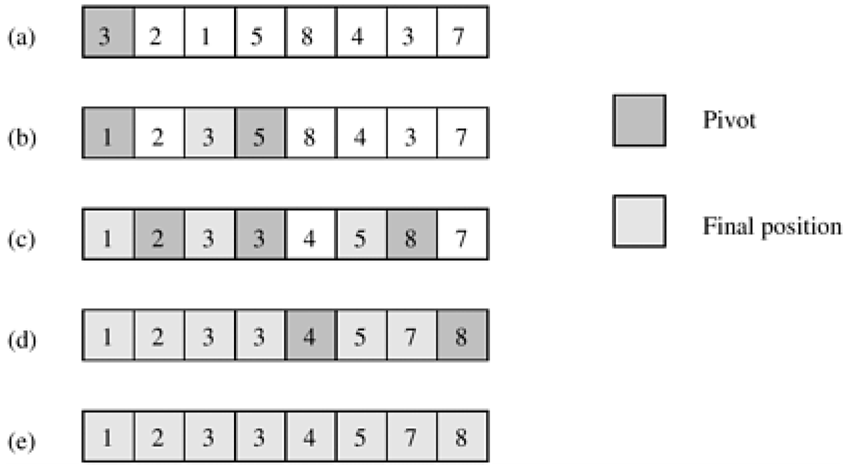
- Pivot can be random, i.e. select the random pivot from the given array.
- Pivot can either be the rightmost element of the leftmost element of the given array.
- Select median as the pivot element.

Algorithm

Quick Sort Function Algorithm

```
//start –> Starting index,  end --> Ending index
Quicksort(array, start, end)
{
if (start < end)
{
pIndex = Partition(A, start, end)
Quicksort(A,start,pIndex-1)
Quicksort(A,pIndex+1, end)
}
}
```

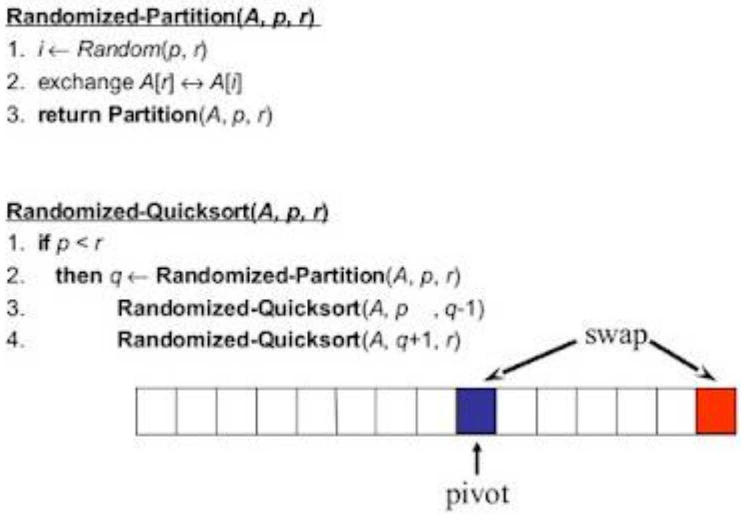
Example of the quicksort algorithm
sorting a sequence of size n = 8.



Randomized Algorithms

An algorithm that uses random numbers to decide what to do next anywhere in its logic is called Randomized Algorithm. For example, in Randomized Quick Sort, we use a random number to pick the next pivot (or we randomly shuffle the array).

Randomized Quick Sort



```
Randomized-Partition(A, p, r)
i ← Random(p, r)
exchange A[r] ↔ A[i]
return Partition(A, p, r)

Randomized-Quicksort(A, p, r)
if p < r
  then q ← Randomized-Partition(A, p, r)
       Randomized-Quicksort(A, p , q-1)
       Randomized-Quicksort(A, q+1, r)
```

In the randomized version of Quick sort we impose a distribution on input by picking the pivot element randomly. Randomized Quick Sort works well even when the array is sorted/reversely sorted and the complexity is more towards O(n log n).

Conclusion:

Thus we implemented Quick sort using Deterministic and randomized variant.

A. Write short answer of following questions :

1. What do you mean by randomized algorithm?
2. What is the purpose of using randomized quick sort?
3. How randomized quick sort algorithm works?
4. What is the time complexity of deterministic quick sort?
5. What is the time complexity of randomized quick sort?
6. Why randomized quick sort is more preferable to normal quicksort?

Sinhgad Institutes

Name of the Student: ______________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

Design and Analysis of Algorithms

Assignment No. 05

N-QUEEN PROBLEM

		Marks:	/10
Date of Performance:	/ /2022	**Sign with Date:**	

Title: Implement n-queen problem using Backtracking

Objectives:

- To understand backtracking approach.
- To solve N Queen problem using backtracking.

Outcomes:

- Implement Backtracking approach to solve N-Queen problem.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 5 PSOs: 1 COs: 1

Problem Statement:

Design n-Queens matrix having first Queen placed. Use backtracking to place remaining Queens to generate the final n-queen matrix.

Theory:

N-Queens Problem

N - Queens problem is to place n - queens in such a manner on an n x n chessboard that no queens attack each other by being in the same row, column or diagonal.

It can be seen that for n =1, the problem has a trivial solution, and no solution exists for n =2 and n =3. So first we will consider the 4 queens problem and then generate it to n - queens problem.
Given a 4 x 4 chessboard and number the rows and column of the chessboard 1 through 4.

Since, we have to place 4 queens such as q_1 q_2 q_3 and q_4 on the chessboard, such that no two queens attack each other. In such a conditional each queen must be placed on a different row, i.e., we put queen "i" on row "i."

Now, we place queen q_1 in the very first acceptable position (1, 1). Next, we put queen q_2 so that both these queens do not attack each other. We find that if we place q_2 in column 1 and 2, then the dead end is encountered. Thus the first acceptable position for q_2 in column 3, i.e. (2, 3) but then no position is left for placing queen 'q_3' safely. So we backtrack one step and place the queen 'q_2' in (2, 4), the next best possible solution. Then we obtain the position for placing 'q_3' which is (3, 2). But later this position also leads to a dead end, and no place is found where 'q_4' can be placed safely. Then we have to backtrack till 'q_1' and place it to (1, 2) and then all other queens are placed safely by moving q_2 to (2, 4), q_3 to (3, 1) and q_4 to (4, 3). That is, we get the solution (2, 4, 1, 3). This is one possible solution for the 4-queens problem. For another possible solution, the whole method is repeated for all partial solutions. The other solutions for 4 - queens problems is (3, 1, 4, 2) i.e.

	1	2	3	4
1			q_1	
2	q_2			
3				q_3
4		q_4		

Backtracking

Backtracking is **an algorithmic technique that considers searching in every possible combination for solving a computational problem**. It is known for solving problems recursively one step at a time and removing those solutions that that do not satisfy the problem constraints at any point of time.

Backtracking Algorithm

The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this row and column as part of the solution. If we do not find such a row due to clashes then we backtrack and return false.

```
1) Start in the leftmost column
2) If all queens are placed
    return true
3) Try all rows in the current column.
   Do following for every tried row.
    a) If the queen can be placed safely in this row
       then mark this [row, column] as part of the
       solution and recursively check if placing
       queen here leads to a solution.
    b) If placing the queen in [row, column] leads to
       a solution then return true.
    c) If placing queen doesn't lead to a solution then
       unmark this [row, column] (Backtrack) and go to
       step (a) to try other rows.
3) If all rows have been tried and nothing worked,
   return false to trigger backtracking.
```

Conclusion:

N Queen problem is solved using backtracking approach.

B. Write short answer of following questions:

5. What is backtracking approach?
6. How many solutions does the 4 queen problem have?
7. Which type of algorithm is used to solve the N queen problem?
8. How many directions do queens attack each other?

Sinhgad Institutes

Name of the Student: ________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

Machine Learning

Assignment No. 01

UBER RIDE FARE PREDICTION

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Uber ride fare prediction using regression algorithms

Objectives:

- To analyse Uber ride dataset to predict the fare of a ride.
- To compare performance of different regressors.

Outcomes:

- Predict the sales of a store.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 4, 5 PSOs: 1, 2 COs: 1

Problem Statement:

Predict the price of the Uber ride from a given pickup point to the agreed drop-off location.

Perform following tasks:

1. Pre-process the dataset.
2. Identify outliers.
3. Check the correlation.
4. Implement linear regression and random forest regression models.
5. Evaluate the models and compare their respective scores like R2, RMSE, etc.

Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset

Theory:

Linear Regression

In statistics, **linear regression** is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called **multiple linear regression**.

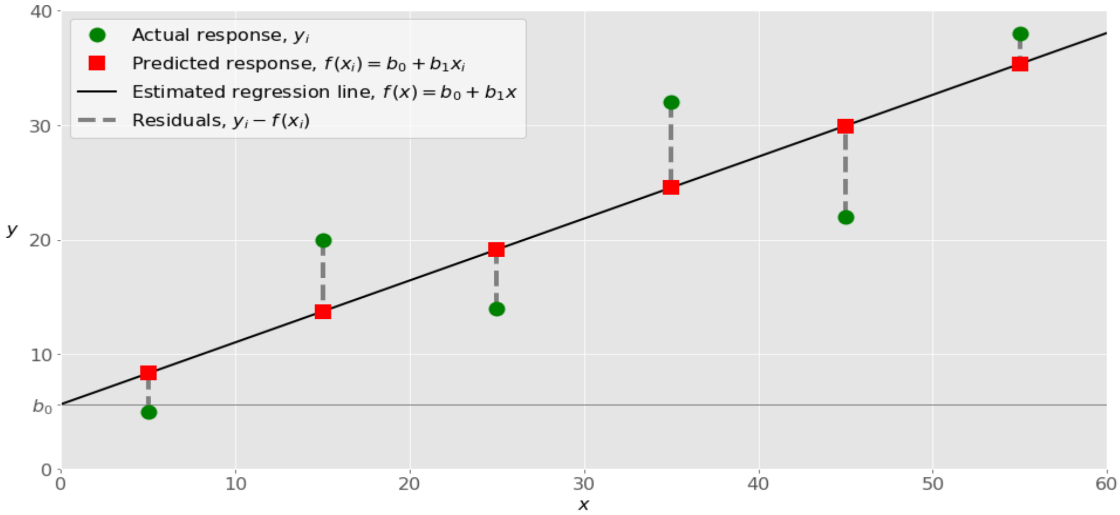
Linear Regression Equation:

Linear regression is a way to model the relationship between two variables. You might also recognize the equation as the **slope formula**. The equation has the form Y= a + bX, where Y is the dependent variable (that's the variable that goes on the Y axis), X is the independent variable (i.e. it is plotted on the X axis), b is the slope of the line and a is the y-intercept.

Simple Linear Regression

Simple or single-variate linear regression is the simplest case of linear regression with a single independent variable, **x** = x.

The following figure illustrates simple linear regression:



Simple Linear Regression With scikit-learn

1. Import the packages and classes you need.
2. Provide data to work with and eventually do appropriate transformations.
3. Create a regression model and fit it with existing data.
4. Check the results of model fitting to know whether the model is satisfactory.
5. Apply the model for predictions.

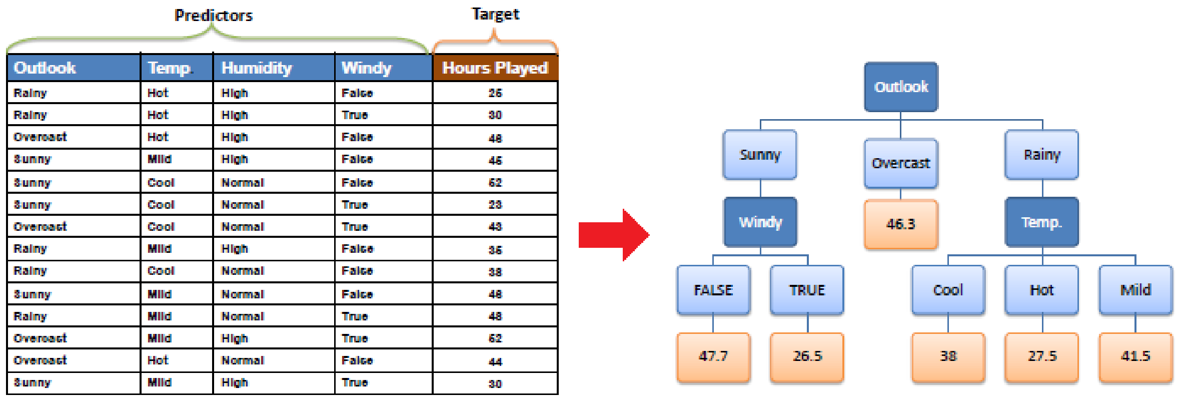
These steps are more or less general for most of the regression approaches and implementations.

Linear Regression – Implementation using scikit learn

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Cannot use Rank 1 matrix in scikit learn
X = X.reshape((m, 1))
# Creating Model
reg = LinearRegression()
# Fitting training data
reg = reg.fit(X, Y)
# Y Prediction
Y_pred = reg.predict(X)

# Calculating R2 Score
r2_score = reg.score(X, Y)
print(r2_score)
```

Decision Tree

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.



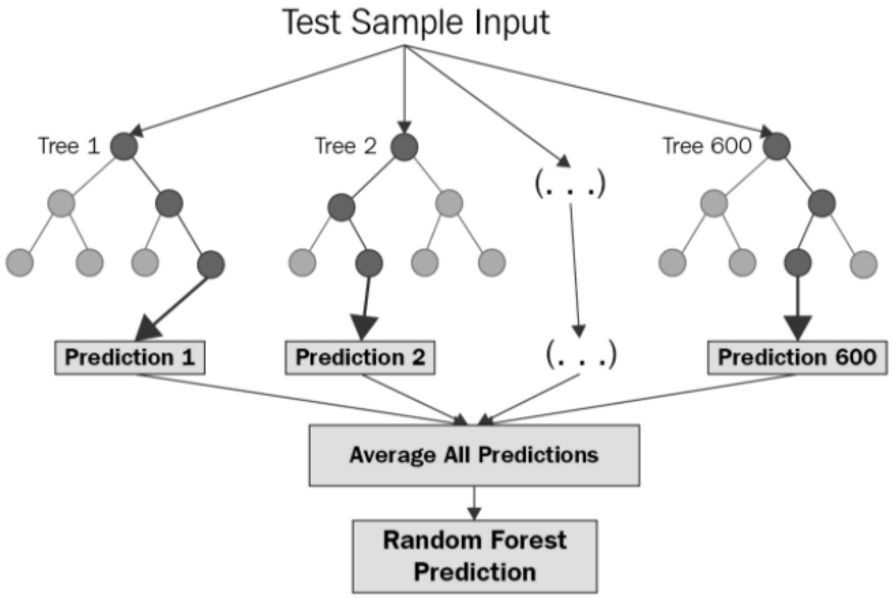
Predictors				Target
Outlook	Temp.	Humidity	Windy	Hours Played
Rainy	Hot	High	False	25
Rainy	Hot	High	True	30
Overcast	Hot	High	False	46
Sunny	Mild	High	False	45
Sunny	Cool	Normal	False	52
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	35
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	46
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30

Decision Tree Regression – Implementation using scikit learn

```
# import the regressor

from sklearn.tree import DecisionTreeRegressor

# create a regressor object
regressor = DecisionTreeRegressor(random_state = 0)

# fit the regressor with X and Y data
regressor.fit(X, y)


# predicting a new value

# test the output by changing values, like 3750
y_pred = regressor.predict(3750)

# print the predicted price
print("Predicted price: % d\n"% y_pred)
```

Random Forest Regression

Random Forest Regression is a supervised learning algorithm that uses **ensemble learning** method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.



The diagram above shows the structure of a Random Forest. You can notice that the trees run in parallel with no interaction amongst them. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees.

Random Forest Regression – Implementation using scikit learn

```
# import the regressor
from sklearn.ensemble import RandomForestRegressor

# create regressor object
regressor = RandomForestRegressor(n_estimators = 100,
random_state = 0)

# fit the regressor with x and y data
regressor.fit(x, y)


# predicting a new value
Y_pred = regressor.predict(np.array([6.5]).reshape(1, 1))
# test the output by changing values

# print the predicted price
Y_pred
```

Conclusion:

Thus we implemented and compared different regressors using PYTHON scikit-learn library.

A. Write short answer of following questions :

1. What is linear regression?
2. What is pruning in Decision Tree?
3. What is Ensemble Learning?
4. What is Entropy and Information gain in Decision tree algorithm?
5. What is Random Forest? How does it work?

Sinhgad Institutes

Name of the Student: ____________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

Machine Learning

Assignment No. 02

EMAIL SPAM CLASSIFICATION

Marks: /10

Date of Performance: / /2022

Sign with Date:

Title : Classify the email using the binary classification method

Objectives:

- To classify email using binary classification method.
- To analyse performance of KNN and SVM classifiers.

Outcomes:

- Predict the class of user.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 4, 5 PSOs: 1, 2 COs: 1

Problem Statement:

Classify the email using the binary classification method. Email Spam detection has two states: a) Normal State – Not Spam, b) Abnormal State – Spam. Use K-Nearest Neighbors and Support Vector Machine for classification. Analyze their performance.

Dataset link: The emails.csv dataset on the Kaggle https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv

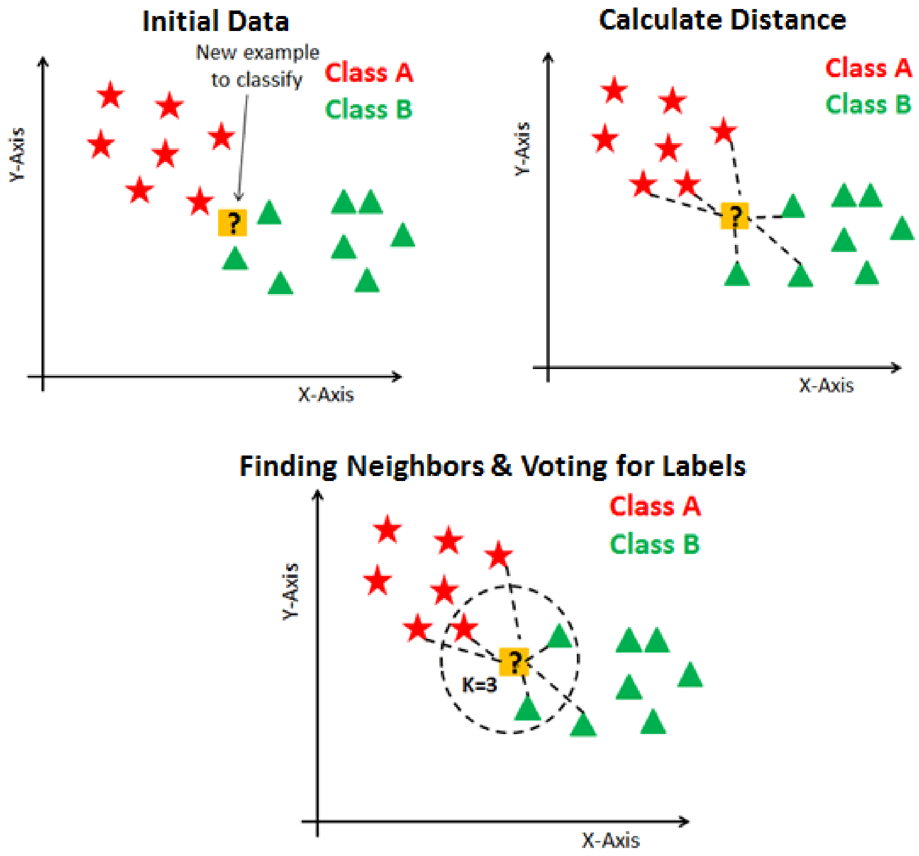
Theory:

K-Nearest Neighbors

KNN is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for underlying data distribution. In other words, the model structure determined from the dataset. This will be very helpful in practice where most of the real world datasets do not follow mathematical theoretical assumptions. Lazy algorithm means it does not need any training data points for model generation. All training data used in the testing phase. This makes training faster and testing phase slower and costlier. Costly testing phase means time and memory. In the worst case, KNN needs more time to scan all data points and scanning all data points will require more memory for storing training data.

How does the KNN algorithm work?

In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor. K is generally an odd number if the number of classes is 2. When K=1, then the algorithm is known as the nearest neighbor algorithm. This is the simplest case. Suppose P1 is the point, for which label needs to predict. First, you find the one closest point to P1 and then the label of the nearest point assigned to P1.



Suppose P1 is the point, for which label needs to predict. First, you find the k closest point to P1 and then classify points by majority vote of its k neighbors. Each object votes for their class and the class with the most votes is taken as the prediction. For finding closest similar points, you find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance.

KNN Classifier Building in Scikit-learn

Generating Model

First, import the KNeighborsClassifier module and create KNN classifier object by passing argument number of neighbors in KNeighborsClassifier() function.

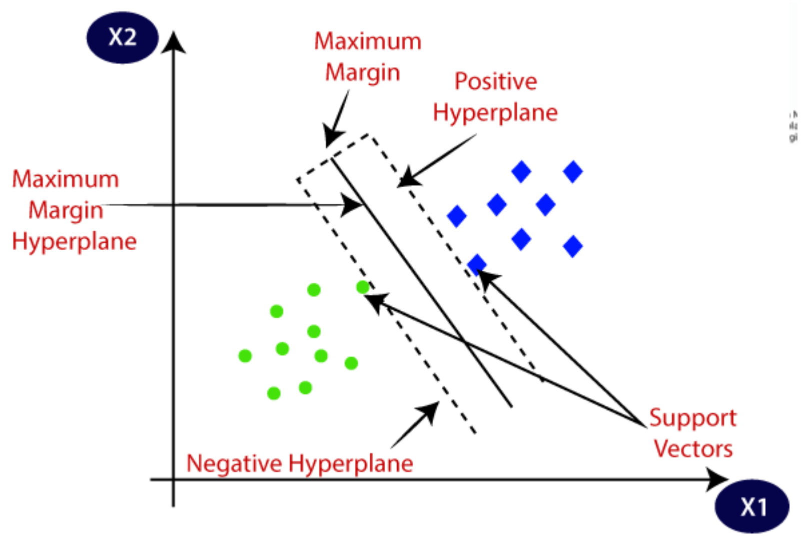
Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

```
From sklearn.neighbors import KNeighborsClassifier

Model= KNeighborsClassifier(n_neighbors=3)

# Train the model using the training sets
Mpdel.fit(features,label)

#Predict Output
predicted=model.predict([[0,2]])# 0:Overcast, 2:Mild
print(predicted)
```

Support Vector Machine Algorithm

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

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.

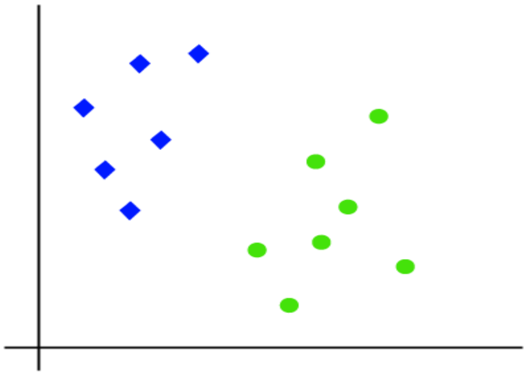


SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

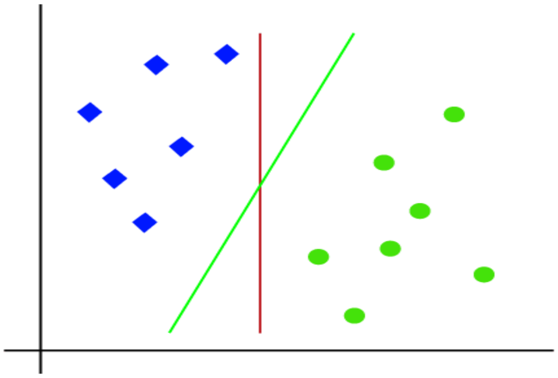
How does SVM works?

Linear SVM:

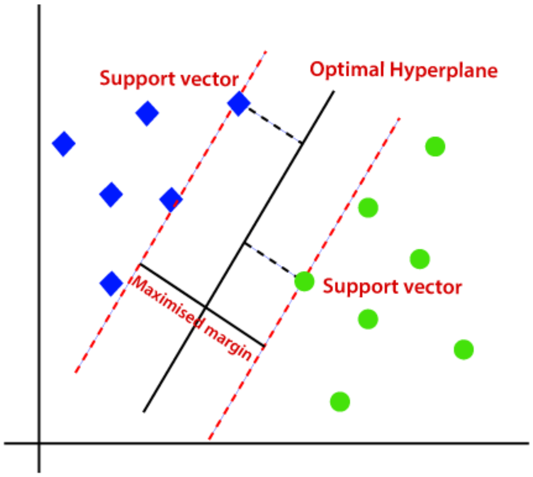
The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:



Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as **margin**. And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.



Support Vector Machine Classifier Building in Scikit-learn

```
from sklearn.svm import SVC # "Support vector classifier"
classifier = SVC(kernel='linear', random_state=0)
classifier.fit(x_train, y_train)
#Predicting the test set result
y_pred= classifier.predict(x_test)
#Creating the Confusion matrix
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_test, y_pred)
```

Evaluating Model

Accuracy can be computed by comparing actual test set values and predicted values.

```
# Model Accuracy, how often is the classifier correct?

print("Accuracy:",metrics.accuracy_score(y_test,
y_pred))
```

Conclusion:

Thus we implemented SVM and KNN classifiers using PYTHON scikit-learn library.

A. Write short answer of following questions :

1. Explain the Confusion Matrix with Respect to Machine Learning Algorithms.
2. Explain the K Nearest Neighbor Algorithm.
3. Explain Support Vector Machine Algorithm.
4. Explain following Classification evaluation metrics.
    a. Accuracy
    b. Precision
    c. Recall
    d. AUC-ROC
    e. F-beta score

Sinhgad Institutes

Name of the Student: ________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

Machine Learning

Assignment No. 03

BANK CUSTOMER CHURN MODELING

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Bank customer churn modeling using the neural network based classifier.

Objectives:

- To build a neural network-based classifier.
- To determine whether the bank customer will leave or not in the next 6 months.

Outcomes:

- Predict the class of user.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Given a bank customer, build a neural network-based classifier that can determine whether they will leave or not in the next 6 months.

Dataset Description: The case study is from an open-source dataset from Kaggle. The dataset contains 10,000 sample points with 14 distinct features such as CustomerId, CreditScore, Geography, Gender, Age, Tenure, Balance, etc.

Link to the Kaggle project:

https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling

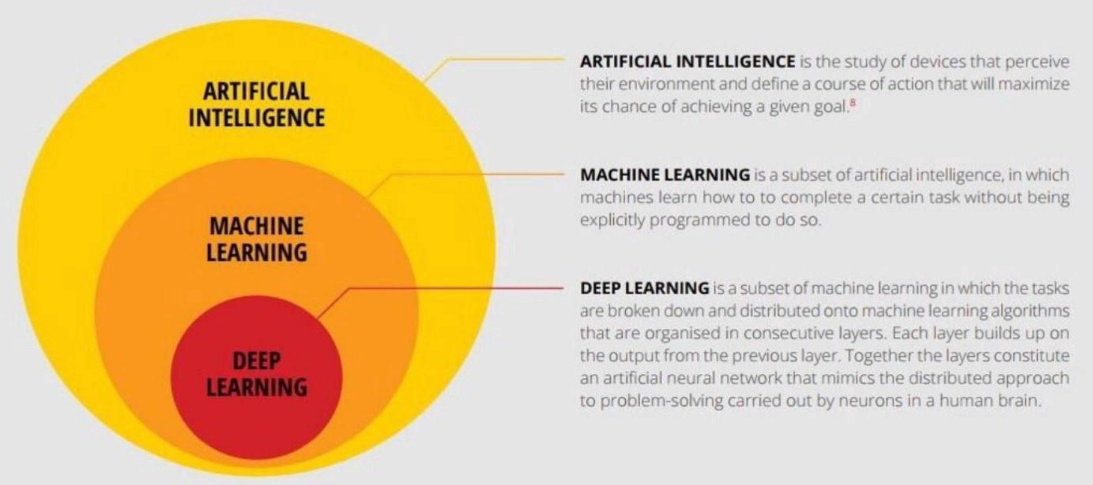
Perform following steps:

1. Read the dataset.
2. Distinguish the feature and target set and divide the data set into training and test sets.
3. Normalize the train and test data.
4. Initialize and build the model. Identify the points of improvement and implement the same.
5. Print the accuracy score and confusion matrix

Theory:

What is Deep Learning?

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



ARTIFICIAL INTELLIGENCE is the study of devices that perceive their environment and define a course of action that will maximize its chance of achieving a given goal.[8]

MACHINE LEARNING is a subset of artificial intelligence, in which machines learn how to to complete a certain task without being explicitly programmed to do so.

DEEP LEARNING is a subset of machine learning in which the tasks are broken down and distributed onto machine learning algorithms that are organised in consecutive layers. Each layer builds up on the output from the previous layer. Together the layers constitute an artificial neural network that mimics the distributed approach to problem-solving carried out by neurons in a human brain.

Applications of deep learning: -

Applications of deep learning are vast and many of great technologies now use deep learning to improve the task. Some of the examples are:-

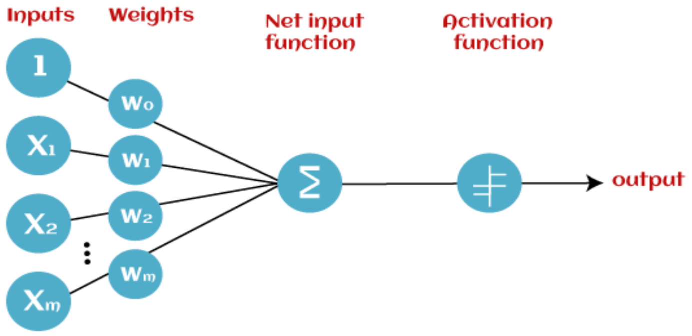
1. Self-driving cars
2. Voice search and virtual assistants
3. Machine translation
4. Image caption generation
5. Colorization of Black and White Images
6. Game playing ai(Open Ai dota bot, google brain alpha go).
7. Real-time object recognition in the image (Google lens).

Basic building block of deep learning:

Perceptron

Perceptron is the building block of neural networks. Perceptron is Machine Learning algorithm for supervised learning of various binary classification tasks. Further, ***Perceptron is also understood as an Artificial Neuron or neural network unit that helps to detect certain input data computations in business intelligence***.

Perceptron model is also treated as one of the best and simplest types of Artificial Neural networks. However, it is a supervised learning algorithm of binary classifiers. Hence, we can consider it as a single-layer neural network with four main parameters, i.e., **input values, weights and Bias, net sum, and an activation function.**



Basic Components of Perceptron

- **Input Nodes or Input Layer:**

This is the primary component of Perceptron which accepts the initial data into the system for further processing. Each input node contains a real numerical value.

- **Wight and Bias:**

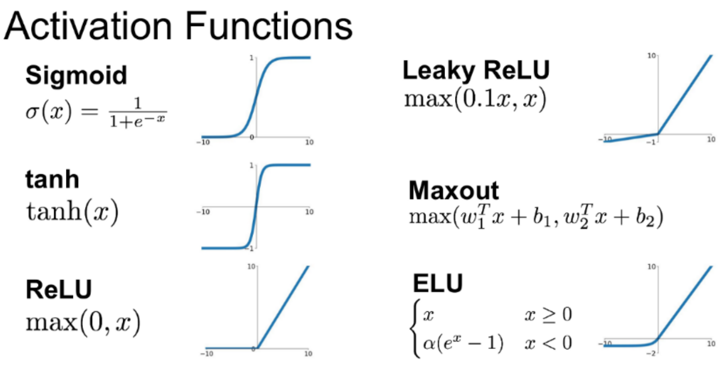
Weight parameter represents the strength of the connection between units. This is another most important parameter of Perceptron components. Weight is directly proportional to the strength of the associated input neuron in deciding the output. Further, Bias can be considered as the line of intercept in a linear equation.

- **Activation Function:**

These are the final and important components that help to determine whether the neuron will fire or not. Activation Function can be considered primarily as a step function.

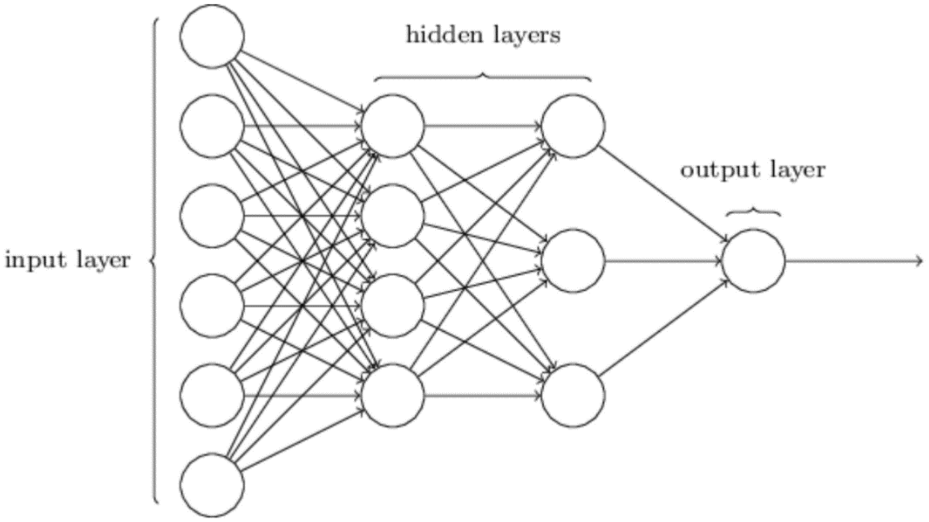
Some of the popular activation functions that are used while building the deep learning models are as follows:

- Sigmoid function
- Hyperbolic tangent function
- Rectified linear unit (RELU) function
- Leaky RELU function
- Maxout function
- Exponential Linear unit (ELU) function



What is Neural Network?

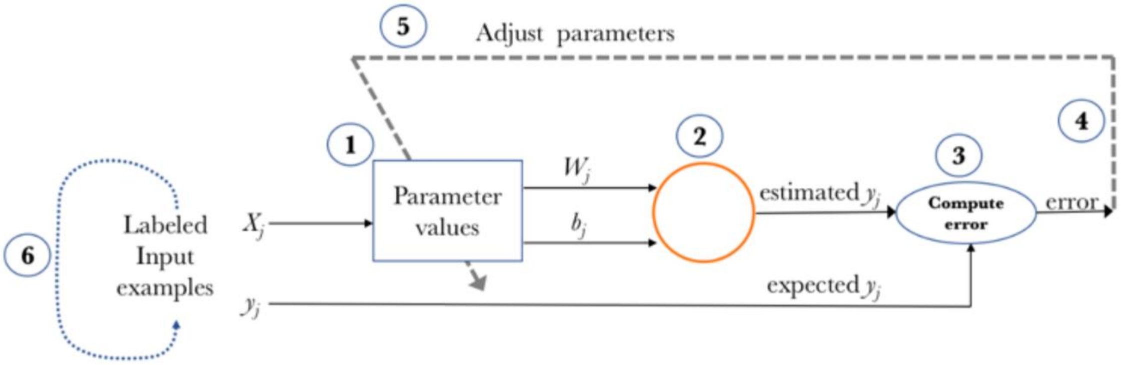
A neural network also known as artificial neural network(ANN) is the basic building block of deep learning. It consists of layers of sigmoid neuron stacked together to form a bigger architecture.



Each circle in the above image is a sigmoid neuron. It consists of 3 types of layers, an input layer, an output layer, and hidden layers. All the previous layers are fully connected with the next layer as can be seen in the image so it is sometimes also referred to as the fully connected neural network. Each neuron has its own weight values. The first layer(input) takes the independent variable of data as input. Output layer predicts the class. The number of hidden layers and number of neurons in hidden layers is not fixed and you can choose any number and tinker to get the best results.

How Neural Networks learn:-

The complete learning process of neural network is given as,



Neural networks are just the weighted sum of the inputs. So the learning of neural networks is based on updating these weights. We need a method to update the weights. It is based on how good the neural network is performing. Performance of the neural network means how good are the predictions based on the actual labels that are to be predicted. The value at the output layer is calculated by crossing through the neural network and finding the value of each neuron. This process of crossing through the neural network is called **forward propagation**.

For measuring the performance of a neural network we introduce **loss function** which calculates how bad the neural network is. Loss function depends on the task we are trying to solve, there are plenty of loss functions but we will discuss two most used ones.

1. **Cross-entropy loss:-** It is used for the classification problem and calculates some value using a function based on the true input label and the predicted output label. The function is given by,

cross entropy loss = −(y*log(p)+(1−y)*log(1−p))

Here y is true label and p is predicted label.

2. **Mean Squared Error(MSE):-** It is used in the regression problem and calculates the distance between the true value and predicted value. It is given as,

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y_i})^2$$

$*\, n$ is the number of data points
$*\, Y_i$ represents observed values
$*\, \hat{Y_i}$ represents predicted values

Once the loss is calculated we need a method to change the weights of the neural network with respect to the calculated loss. We backpropagate through the neural network and update the weights. based on the relative contribution that each neuron has contributed to the original output. This process is repeated, layer by layer, until all the neurons in the network have received a loss signal that describes their relative contribution to the total loss. The change to be made in weights is calculated using **gradient descent**.

Building neural network in Keras:

Keras is a fast, open-source, and easy-to-use Neural Network Library written in Python that runs at top of Theano or Tensorflow. Tensorflow provides low-level as well as high-level API, indeed Keras only provide High-level API.

As a beginner, it is recommended to work with Keras first and then move to TensorFlow. The reason is using Tensorflow functions as a beginner is a little bit complex to understand and interpret but Keras functionality is simple.

1. **Preprocess and load data-** As we have already discussed data is the key for the working of neural network and we need to process it before feeding to the neural network. In this step, we will also visualize data which will help us to gain insight into the data.

2. **Define model-** Now we need a neural network model. This means we need to specify the number of hidden layers in the neural network and their size, the input and output size.

```
# instantiate the model
model = tk.Sequential()
# Adding the input layer
model.add(tk.layers.Input(shape=(11,)))
# Adding the first hidden layer
model.add(tk.layers.Dense(units=6,activation='relu',
kernel_initializer='he_uniform'))
# Adding the second hidden layer
model.add(tk.layers.Dense(units=6,activation='relu',
kernel_initializer='he_uniform'))
# Adding the output layer
model.add(tk.layers.Dense(units=1,activation='sigmoid',kernel_initializer='glorot_uniform'))
```

3. **Loss and optimizer-** Now we need to define the loss function according to our task. We also need to specify the optimizer to use with learning rate and other hyperparameters of the optimizer.

```
# Compiling the model
model.compile(optimizer='Adam',loss='binary_crossentropy',metrics=['Precision','accuracy'])
```

4. **Fit model-** This is the training step of the neural network. Here we need to define the number of epochs for which we need to train the neural network.

```
model.fit(x=x_train1_sc,
          y=y_train,
          epochs=100,
          batch_size=32,
          validation_data=(x_test1_sc,y_test))
```

Epoch, iteration, and batch are different types that are used for processing the datasets and algorithms for gradient descent. All these three methods, i.e., epoch, iteration, and batch size are basically ways of working on the gradient descent depending on the size of the data set.

Epoch: It represents one iteration over the entire training dataset (everything put into the training model).

Batch: This refers to when we are not able to pass the entire dataset into the neural network at once due to the problem of high computations, so we divide the dataset into several batches.

Iteration: Let’s have 10,000 images as our training dataset and we choose a batch size of 200. then an epoch should run (10000/200) iterations i.e, 50 iterations.

Conclusion:

Thus we implemented ANN classifiers using PYTHON Keras library.

A. Write short answer of following questions:

1. What do you mean by Perceptron?
2. What is the use of the Loss functions?
3. What is the role of the Activation functions in Neural Networks?
4. What is the difference between Forward propagation and Backward Propagation in Neural Networks?
5. Explain Gradient Descent and its types.

Sinhgad Institutes

Name of the Student: ________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

Machine Learning

Assignment No. 04

K-MEANS CLUSTERING

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csv dataset. Determine the number of clusters using the elbow method.

Objectives:

- To implement K-Means clustering.
- To determine the number of clusters using the elbow method.

Outcomes:

- Determine the number of clusters using the elbow method.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csv dataset. Determine the number of clusters using the elbow method.

Dataset link : https://www.kaggle.com/datasets/kyanyoga/sample-sales-data

Theory:

K-Means Clustering Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.

What is K-Means Algorithm?

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 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.

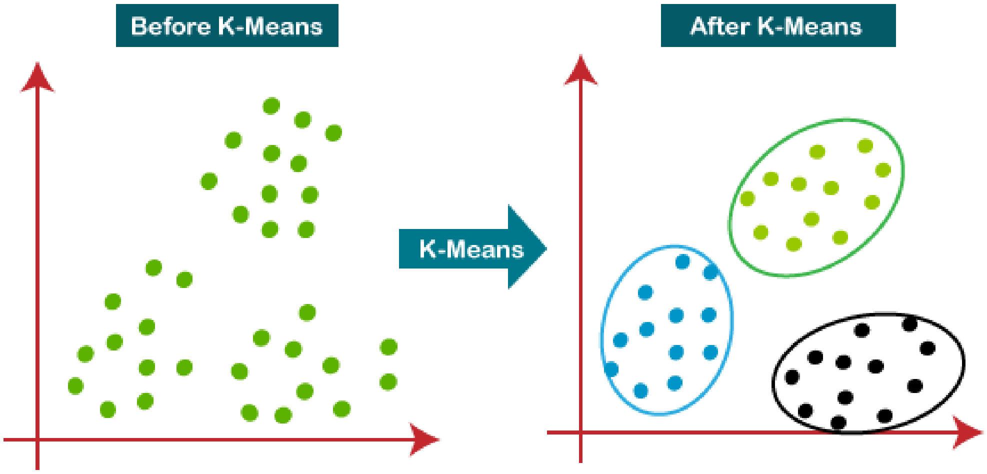
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

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.

The below diagram explains the working of the K-means Clustering Algorithm:



How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

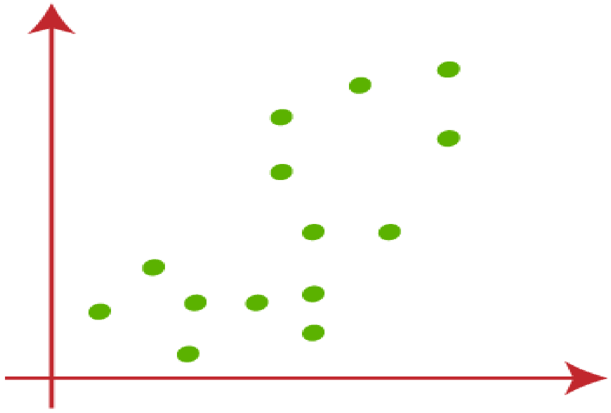
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

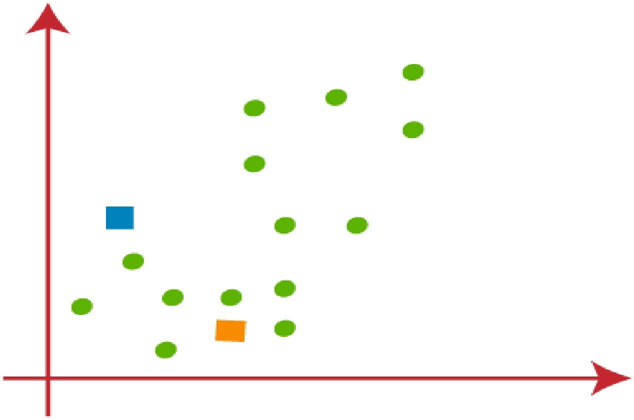
Let's understand the above steps by considering the visual plots:

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

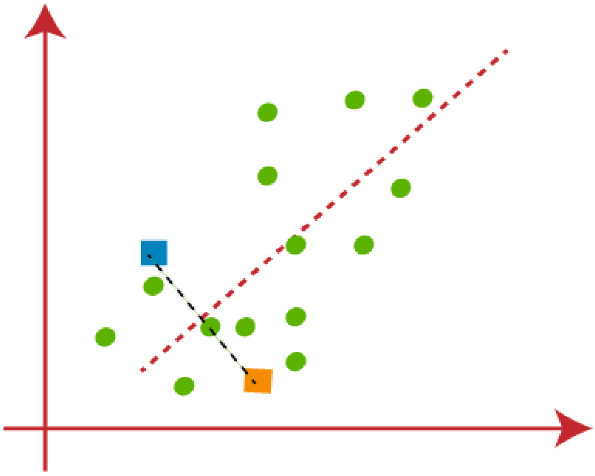


- ○ Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
- ○ We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points

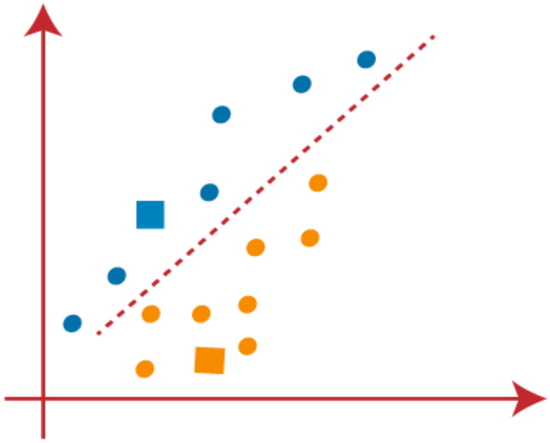
as k points, which are not the part of our dataset. Consider the below image:



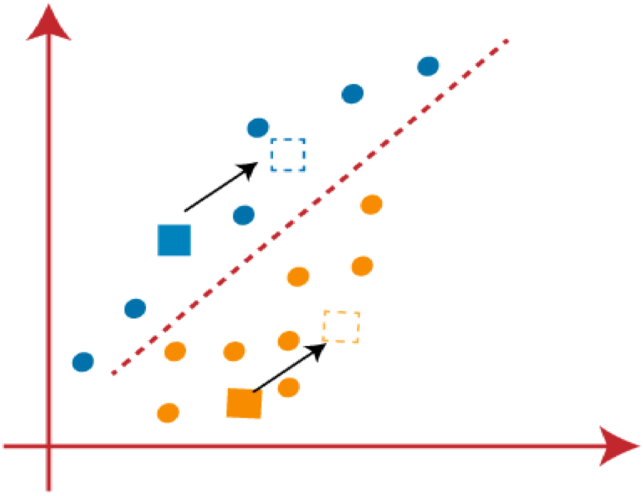
- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:



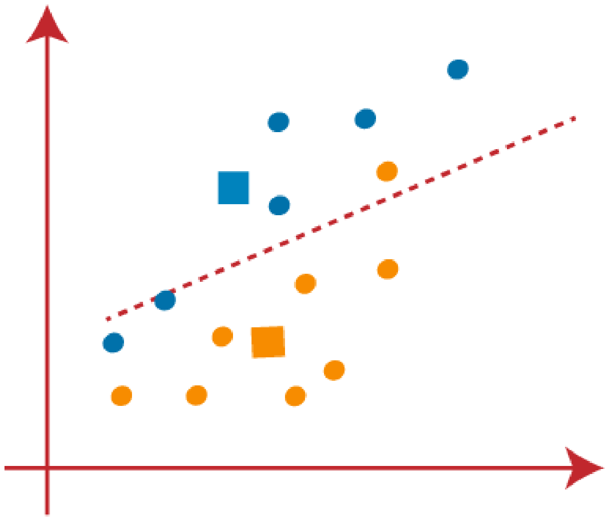
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



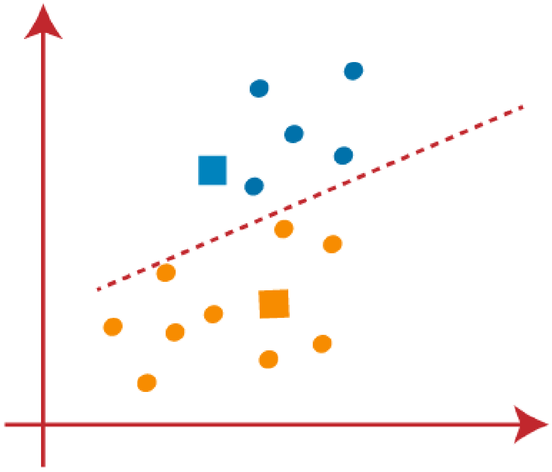
- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:



- Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:

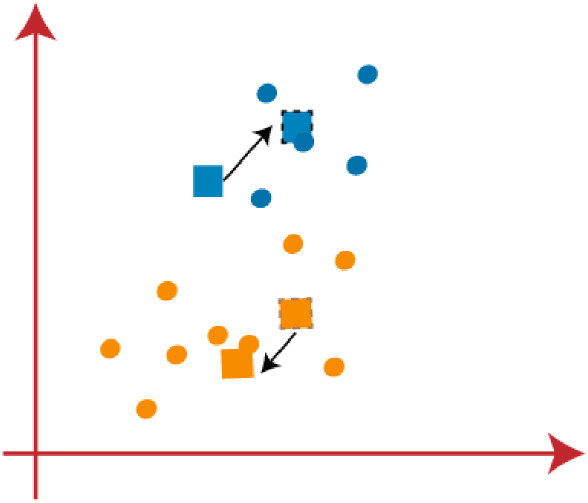


From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

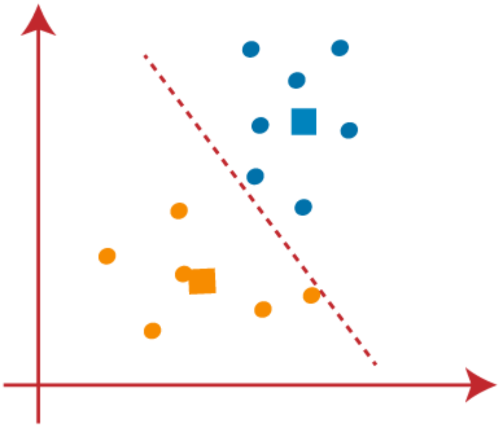


As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

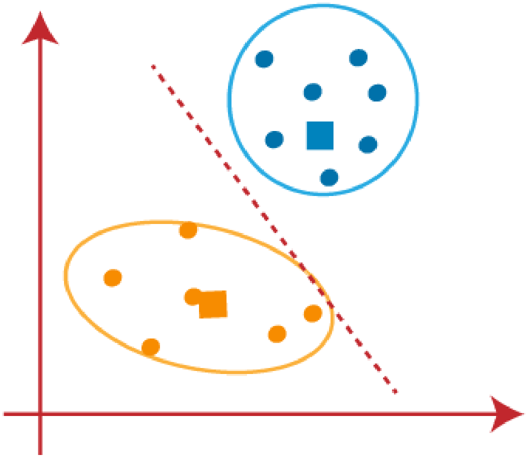
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



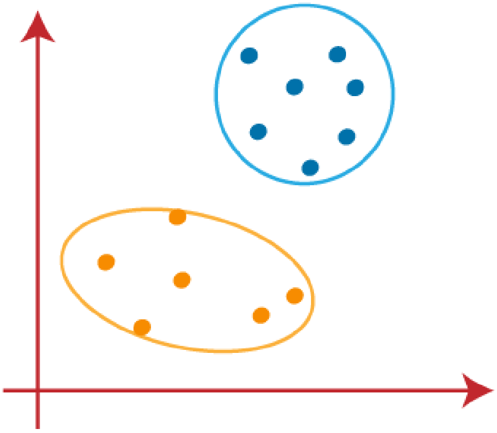
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

WCSS= $\sum_{\text{Pi in Cluster1}}$ distance(P_i C_1)^2 +$\sum_{\text{Pi in Cluster2}}$distance($P_i$ C_2)^2+$\sum_{\text{Pi in CLuster3}}$ distance(P_i C_3)^2

In the above formula of WCSS,

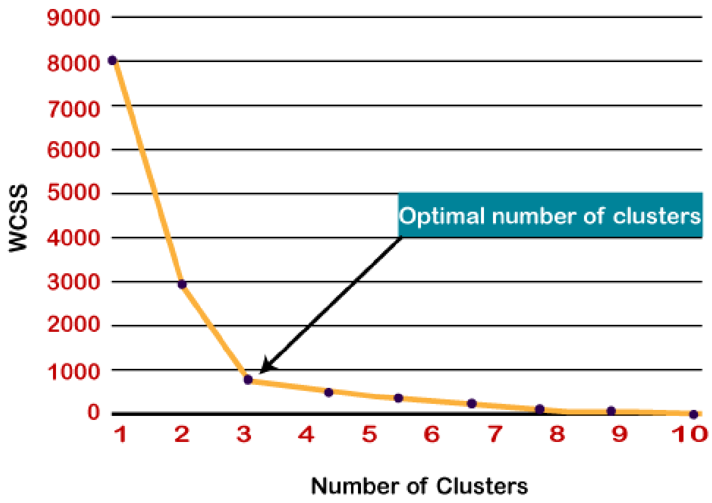
$\sum_{\text{Pi in Cluster1}}$ distance(P_i C_1)^2: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



Conclusion:

Thus, we implemented K-Means clustering using PYTHON scikit-learn library.

A. Write short answer of following questions:

1. How K means clustering algorithm works?
2. What is elbow method in K-means algorithm?
3. Can you use k-means with categorical data?
4. Is clustering sensitive to outliers?
5. What are the two main problems of k-means clustering algorithm?

Sinhgad Institutes

Name of the Student: ________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

Machine Learning

Assignment No. 05

GRADIENT DESCENT ALGORITHM

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Implement Gradient Descent Algorithm to find the local minima of a function.

Objectives:

- To implement Gradient Descent Algorithm to find the local minima of a function.

Outcomes:

- Find the local minima of a function.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Implement Gradient Descent Algorithm to find the local minima of a function.
For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.

Theory:

Gradient Descent in Machine Learning

Gradient Descent is known as one of the most commonly used optimization algorithms to train machine learning models by means of minimizing errors between actual and expected results. Further, gradient descent is also used to train Neural Networks.

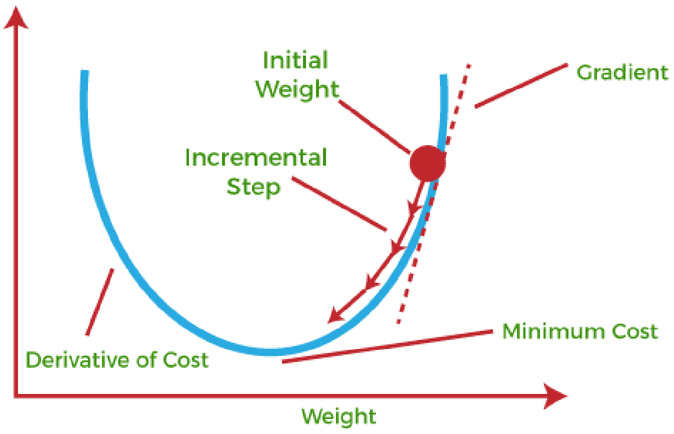
In mathematical terminology, Optimization algorithm refers to the task of minimizing/maximizing an objective function f(x) parameterized by x. Similarly, in machine learning, optimization is the task of minimizing the cost function parameterized by the model's parameters. The main objective of gradient descent is to minimize the convex function using iteration of parameter updates. Once these machine learning models are optimized, these models can be used as powerful tools for Artificial Intelligence and various computer science applications.

What is Gradient Descent or Steepest Descent?

Gradient descent was initially discovered by **"Augustin-Louis Cauchy"** in mid of 18th century. ***Gradient Descent is defined as one of the most commonly used iterative optimization algorithms of machine learning to train the machine learning and deep learning models. It helps in finding the local minimum of a function.***

The best way to define the local minimum or local maximum of a function using gradient descent is as follows:

- If we move towards a negative gradient or away from the gradient of the function at the current point, it will give the **local minimum** of that function.
- Whenever we move towards a positive gradient or towards the gradient of the function at the current point, we will get the **local maximum** of that function.



This entire procedure is known as Gradient Ascent, which is also known as steepest descent. ***The main objective of using a gradient descent algorithm is to minimize the cost function using iteration.*** To achieve this goal, it performs two steps iteratively:

- Calculates the first-order derivative of the function to compute the gradient or slope of that function.
- Move away from the direction of the gradient, which means slope increased from the current point by alpha times, where Alpha is defined as Learning Rate. It is a tuning parameter in the optimization process which helps to decide the length of the steps.

What is Cost-function?

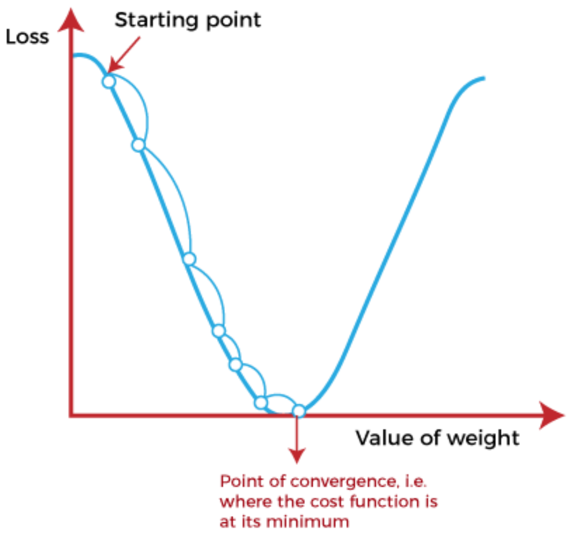
The cost function is defined as the measurement of difference or error between actual values and expected values at the current position and present in the form of a single real number. It helps to increase and improve machine learning efficiency by providing feedback to this model so that it can minimize error and find the local or global minimum.

How does Gradient Descent work?

Before starting the working principle of gradient descent, we should know some basic concepts to find out the slope of a line from linear regression. The equation for simple linear regression is given as:

1. Y=mX+c

Where 'm' represents the slope of the line, and 'c' represents the intercepts on the y-axis.



The starting point(shown in above fig.) is used to evaluate the performance as it is considered just as an arbitrary point. At this starting point, we will derive the first derivative or slope and then use a tangent line to calculate the steepness of this slope. Further, this slope will inform the updates to the parameters (weights and bias).

The slope becomes steeper at the starting point or arbitrary point, but whenever new parameters are generated, then steepness gradually reduces, and at the lowest point, it approaches the lowest point, which is called **a point of convergence.**

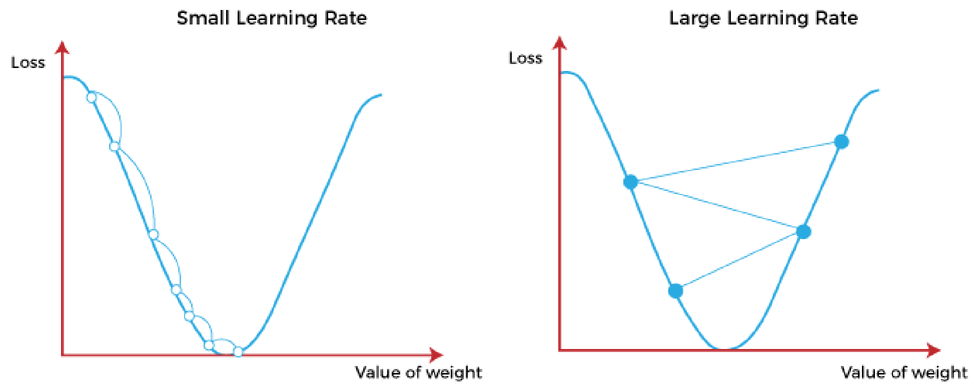
The main objective of gradient descent is to minimize the cost function or the error between expected and actual. To minimize the cost function, two data points are required:

- **Direction & Learning Rate**

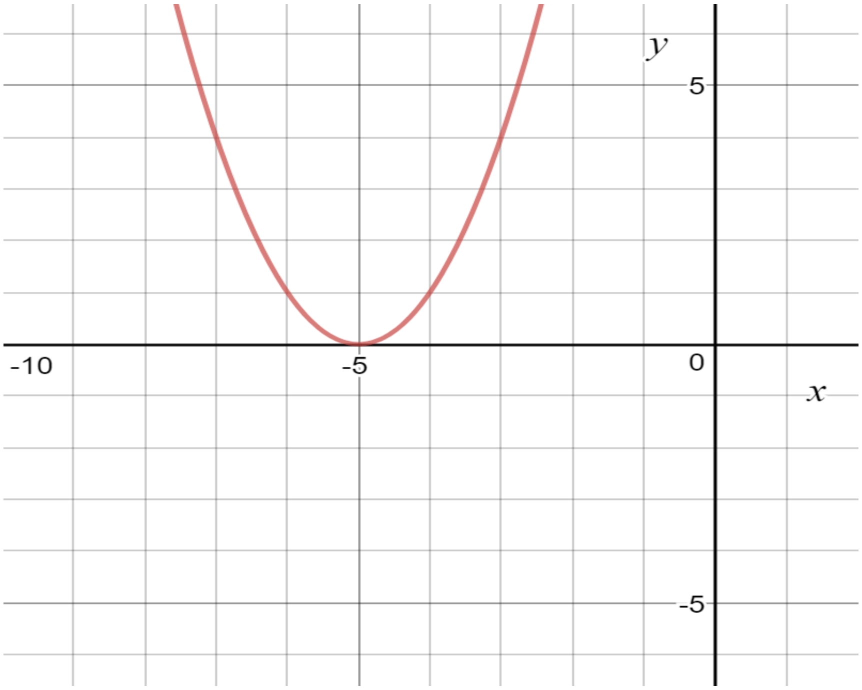
These two factors are used to determine the partial derivative calculation of future iteration and allow it to the point of convergence or local minimum or global minimum. Let's discuss learning rate factors in brief;

Learning Rate:

It is defined as the step size taken to reach the minimum or lowest point. This is typically a small value that is evaluated and updated based on the behavior of the cost function. If the learning rate is high, it results in larger steps but also leads to risks of overshooting the minimum. At the same time, a low learning rate shows the small step sizes, which compromises overall efficiency but gives the advantage of more precision.



Example :
Question : Find the local minima of the function $y=(x+5)^2$ starting from the point x=3



Solution : We know the answer just by looking at the graph. y = (x+5)² reaches it's minimum value when x = -5 (i.e when x=-5, y=0). Hence x=-5 is the local and global minima of the function.

Now, let's see how to obtain the same numerically using gradient descent.

Step 1 : Initialize x =3. Then, find the gradient of the function, dy/dx = 2*(x+5).

Step 2 : Move in the direction of the negative of the gradient. But wait, how much to move? For that, we require a learning rate. Let us assume the **learning rate → 0.01**

Step 3 : Let's perform 2 iterations of gradient descent

Initialize Parameters :

$$X_0 = 3$$

$$Learning\ rate = 0.01$$

$$\frac{dy}{dx} = \frac{d}{dx}(x+5)^2 = 2 * (x+5)$$

Iteration 1 :

$$X_1 = X_0 - (learning\ rate) * (\frac{dy}{dx})$$

$$X_1 = 3 - (0.01) * \big(2 * (3+5)\big) = 2.84$$

Iteration 2 :

$$X_2 = X_1 - (learning\ rate) * (\frac{dy}{dx})$$

$$X_2 = 2.84 - (0.01) * \big(2 * (2.84+5)\big) = 2.6832$$

Step 4 : We can observe that the X value is slowly decreasing and should converge to -5 (the local minima). However, how many iterations should we perform?
Let us set a precision variable in our algorithm which calculates the difference between two consecutive "x" values . If the difference between x values from 2 consecutive iterations is lesser than the precision we set, stop the algorithm !

Conclusion:

Thus, we implemented Gradient Descent algorithm in Python.

A. Write short answer of following questions:

1. What is gradient descent explain with example?
2. Which ML algorithms use gradient descent?
3. How is the gradient descent useful in machine learning implementation?
4. What happens when learning rate is too high gradient descent?
5. How many types of gradient descent are there? Which gradient descent is best?

Sinhgad Institutes

Name of the Student: ____________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

Machine Learning

Assignment No. 06

K-NEAREST NEIGHBORS

		Marks:	/10
Date of Performance:	/ /2022	**Sign with Date:**	

Title : Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset.

Objectives:

- To implement k-Nearest Neighbors Algorithm.
- To compute confusion matrix, accuracy, error rate, precision and recall on the given dataset.

Outcomes:

- Classify the record using K-Nearest Neighbors algorithm.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset.

Dataset link : https://www.kaggle.com/datasets/abdallamahgoub/diabetes

Theory:

K-Nearest Neighbors Algorithm for Machine Learning

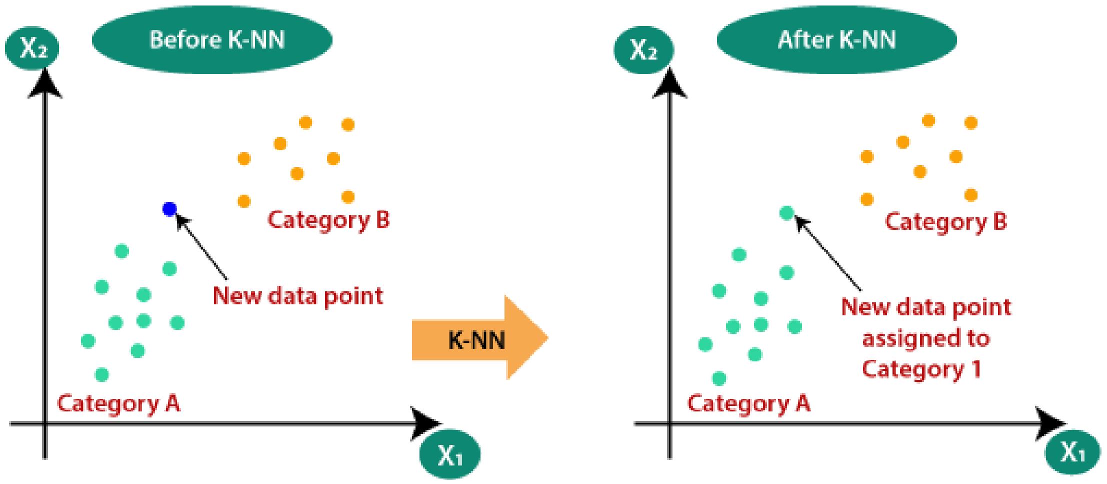
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

- 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.

KNN Classifier
?
Input value
Predicted Output

<u>Why do we need a K-NN Algorithm?</u>

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:

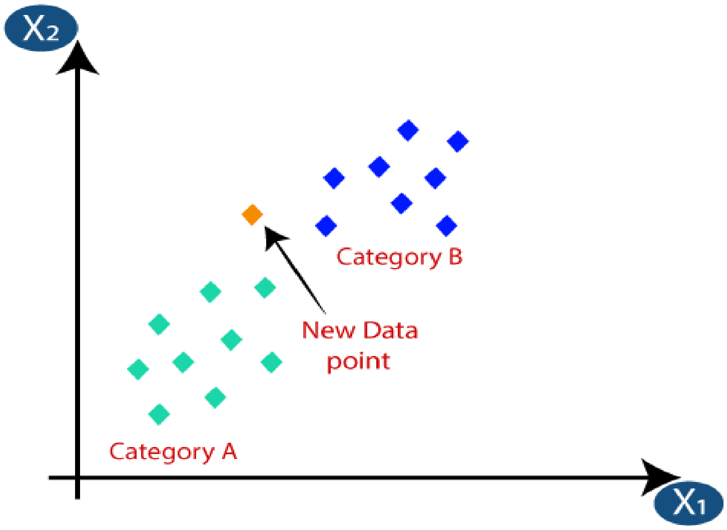


How does K-NN work?

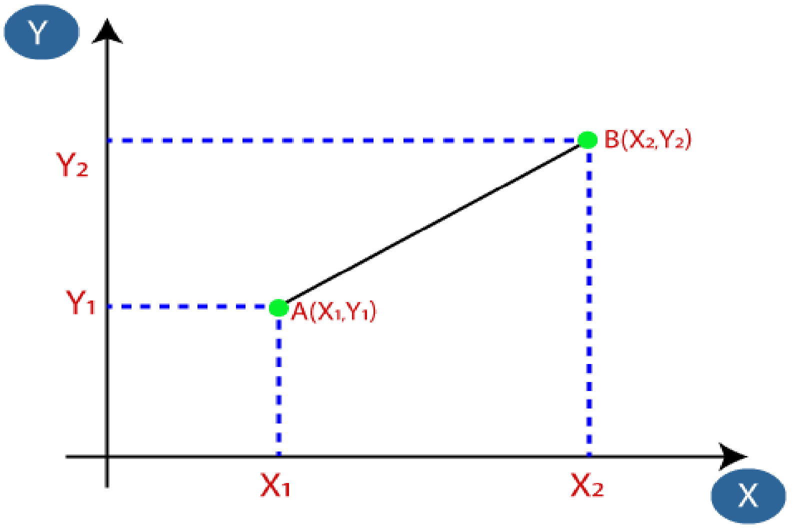
The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance.
- **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:

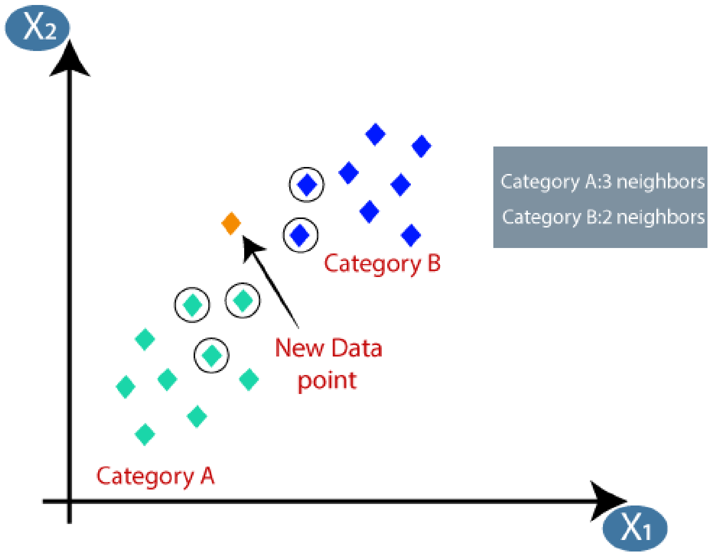


- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



Euclidean Distance between A_1 and B_2 = $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

- 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.

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.

KNN Classifier Building in Scikit-learn

Generating Model

First, import the KNeighborsClassifier module and create KNN classifier object by passing argument number of neighbors in KNeighborsClassifier() function.

Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

```
From sklearn.neighbors import KNeighborsClassifier

Model= KNeighborsClassifier(n_neighbors=3)

# Train the model using the training sets
Model.fit(features,label)

#Predict Output
predicted=model.predict([[0,2]])# 0:Overcast, 2:Mild
print(predicted)
```

Conclusion:

Thus, we implemented K-Nearest Neighbors classifiers using PYTHON scikit-learn library.

A. Write short answer of following questions:

1. Why KNN is called lazy learner?
2. Is KNN clustering or classification?
3. Can KNN be used for regression?
4. What does K represent in KNN algorithm?
5. How KNN is used in machine learning?
6. What distance metrics can be used in KNN?

Name of the Student: ______________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

BLOCKCHAIN TECHNOLOGY

Assignment No. 01

Installation of MetaMask

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Installation of MetaMask and study spending Ether per transaction.

Objectives:

- To learn new technology such as MetaMask.
- To build application using MetaMask.

Outcomes:

- Installation of MetaMask and study spending Ether per transaction.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Installation of MetaMask and study spending Ether per transaction.

Theory:

Introduction to Blockchain

- Blockchain can be described as a data structure that holds transactional records and while ensuring security, transparency, and decentralization. You can also think of it as a chain or records stored in the forms of blocks which are controlled by no single authority.
- A blockchain is a distributed ledger that is completely open to any and everyone on the network. Once an information is stored on a blockchain, it is extremely difficult to change or alter it.
- Each transaction on a blockchain is secured with a digital signature that proves its authenticity. Due to the use of encryption and digital signatures, the data stored on the blockchain is tamper-proof and cannot be changed.
- Blockchain technology allows all the network participants to reach an agreement, commonly known as consensus. All the data stored on a blockchain is recorded digitally and has a common history which is available for all the network participants. This way, the chances of any fraudulent activity or duplication of transactions is eliminated without the need of a third-party.

Blockchain Features

The following features make the revolutionary technology of blockchain stand out:

- **Decentralized**

  Blockchains are decentralized in nature meaning that no single person or group holds the authority of the overall network. While everybody in the network has the copy of the distributed ledger with them, no one can modify it on his or her own. This unique feature of blockchain allows transparency and security while giving power to the users.

- **Peer-to-Peer Network**

  With the use of Blockchain, the interaction between two parties through a peer-to-peermodel is easily accomplished without the requirement of any third party. Blockchain uses P2P protocol which allows all the network participants to hold an identical copy of transactions, enabling approval through a machine consensus. For example, if you wish to make any transaction from one part of the world to another, you can do that with blockchain all by yourself within a few seconds. Moreover, any interruptions or extra charges will not be deducted in the transfer.

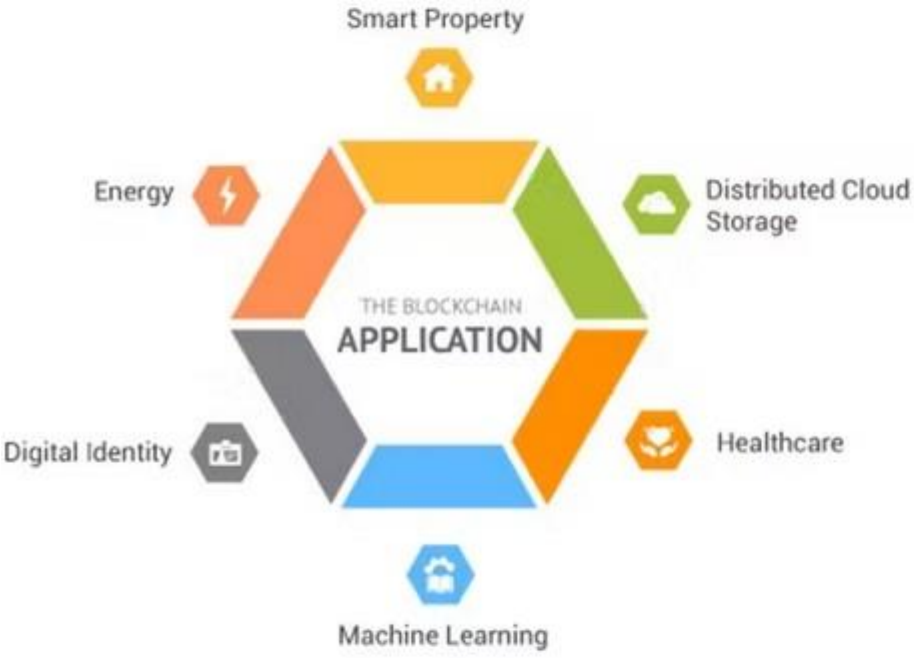
- **Immutable**

  The immutability property of a blockchain refers to the fact that any data once written on the blockchain cannot be changed. To understand immutability, consider sending email as an example. Once you send an email to a bunch of people, you cannot take it back. In order to find a way around, you'll have to ask all the recipients to delete your email which is pretty tedious. This is how immutability works.

- **Tamper-Proof**

  With the property of immutability embedded in blockchains, it becomes easier to detect tampering of any data. Blockchains are considered tamper-proof as any change in even one single block can be detected and addressed smoothly. There are two key ways of detecting tampering namely, hashes and blocks.

Popular Applications of Blockchain Technology



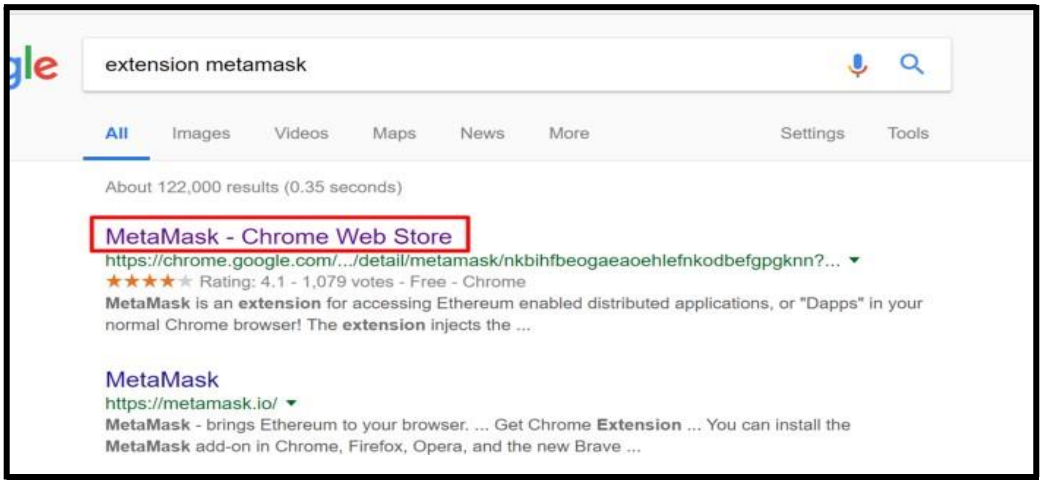
Benefits of Blockchain Technology

- **Time-saving:** No central Authority verification needed for settlements making theprocess faster and cheaper.
- **Cost-saving:** A Blockchain network reduces expenses in several ways. No need for third-party verification. Participants can share assets directly. Intermediaries are reduced. Transaction efforts are minimized as every participant has a copy of shared ledger.
- **Tighter security:** No one can temper with Blockchain Data as it is shared among millions of participants. The system is safe against cybercrimes and Fraud.
- In finance market trading, Fibonacci retracement levels are widely used in technicalanalysis.

How to use MetaMask: A step by step guide

MetaMask is one of the most popular browser extensions that serves as a way of storing your Ethereum and other ERC-20 Tokens. The extension is free and secure, allowing web applications to read and interact with Ethereum's blockchain.

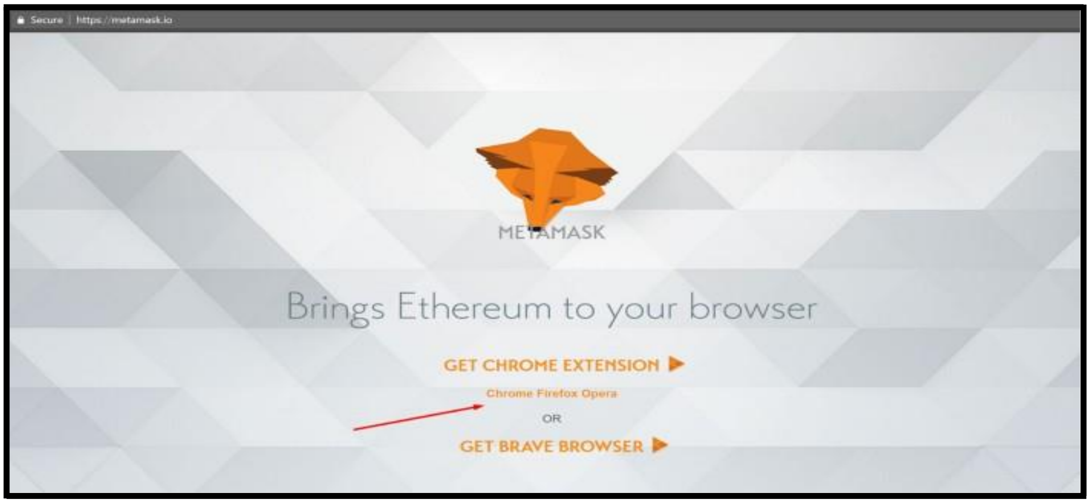
Step 1. Install MetaMask on your browser.
To create a new wallet, you have to install the extension first. Depending on your browser, there are different marketplaces to find it. Most browsers have MetaMask on their stores, so it’s not that hard to see it, but either way, here they are Chrome, Firefox, and Opera.



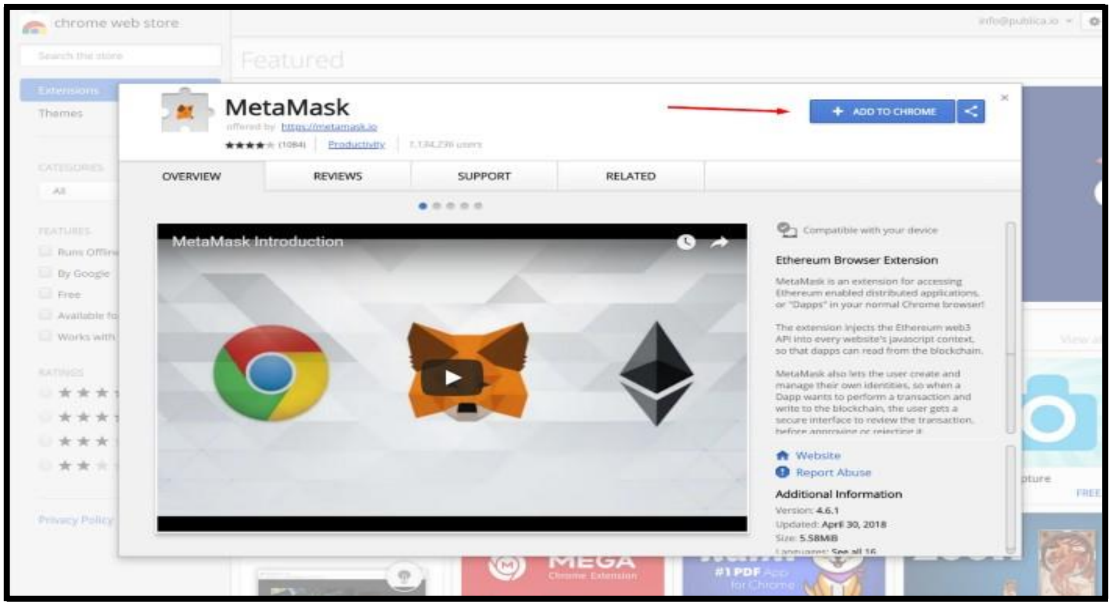
About 122,000 results (0.35 seconds)

MetaMask - Chrome Web Store
https://chrome.google.com/.../detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn?...
★★★★★ Rating: 4.1 - 1,079 votes - Free - Chrome
MetaMask is an **extension** for accessing Ethereum enabled distributed applications, or "Dapps" in your normal Chrome browser! The **extension** injects the ...

MetaMask
https://metamask.io/
MetaMask - brings Ethereum to your browser. ... Get Chrome **Extension** ... You can install the **MetaMask** add-on in Chrome, Firefox, Opera, and the new Brave ...



- Click on **Install MetaMask** as a Google Chrome extension.
- Click **Add to Chrome**.
- Click **Add Extension**.



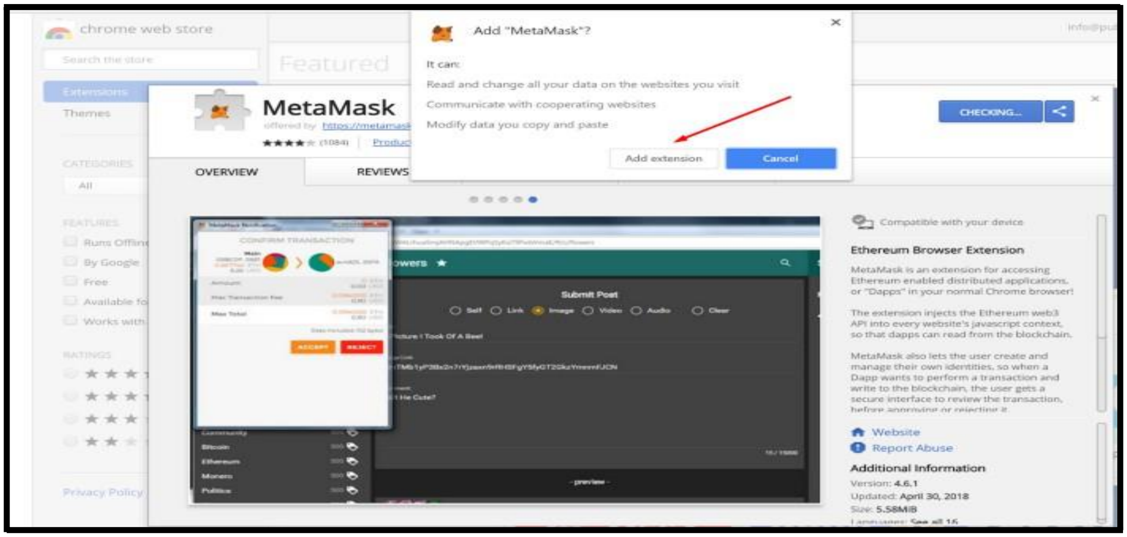
Ethereum Browser Extension

MetaMask is an extension for accessing Ethereum enabled distributed applications, or "Dapps" in your normal Chrome browser!

The extension injects the Ethereum web3 API into every website's javascript context, so that dapps can read from the blockchain.

MetaMask also lets the user create and manage their own identities, so when a Dapp wants to perform a transaction and write to the blockchain, the user gets a secure interface to review the transaction, before approving or rejecting it.

Additional Information
Version: 4.6.1
Updated: April 30, 2018
Size: 5.58MiB
Languages: See all 16



Add "MetaMask"?

It can:

Read and change all your data on the websites you visit

Communicate with cooperating websites

Modify data you copy and paste

Add extension Cancel

Ethereum Browser Extension

MetaMask is an extension for accessing Ethereum enabled distributed applications, or "Dapps" in your normal Chrome browser!

The extension injects the Ethereum web3 API into every website's javascript context, so that dapps can read from the blockchain.

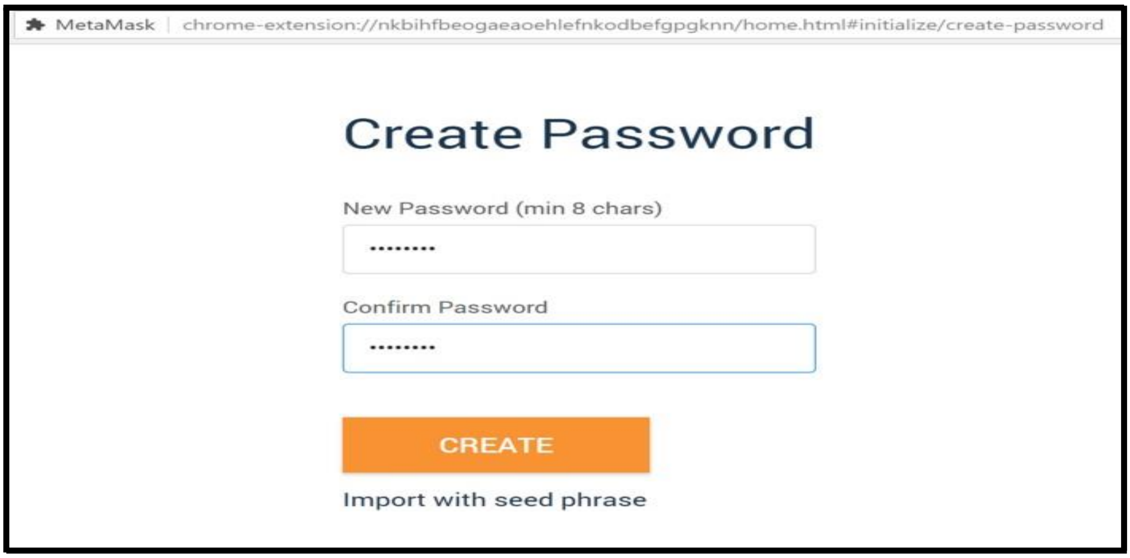
MetaMask also lets the user create and manage their own identities, so when a Dapp wants to perform a transaction and write to the blockchain, the user gets a secure interface to review the transaction, before approving or rejecting it.

Additional Information
Version: 4.6.1
Updated: April 30, 2018
Size: 5.58MiB
Languages: See all 16

It's as easy as that to install the extension on your browser, continue reading the next stepto figure out how to create an account.

Step 2. Create an account.

- Click on the extension icon in the upper right corner to open MetaMask.
- To install the latest version and be up to date, **click Try it now**.
- **Click Continue.**
- You will be prompted to create a new password. **Click Create**.



MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#initialize/create-password

Create Password

New Password (min 8 chars)

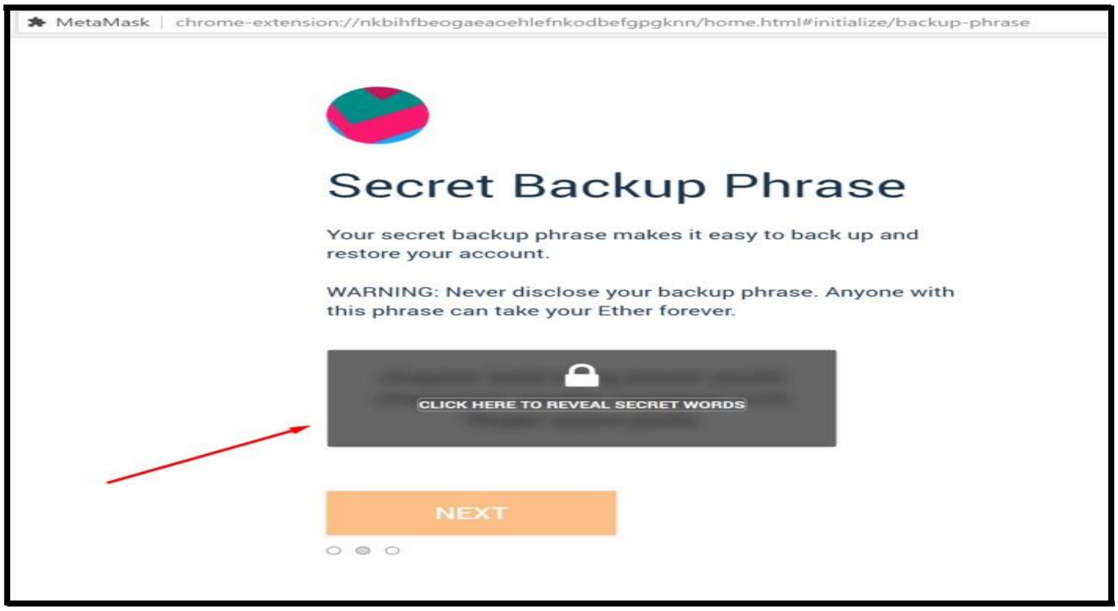
••••••••

Confirm Password

••••••••

- Proceed by **clicking Next** and accept the Terms of Use.

Click Reveal Secret Words. There you will see a 12 words seed phrase. This is reallyimportant and usually not a good idea to store digitally, so take your time and write it down

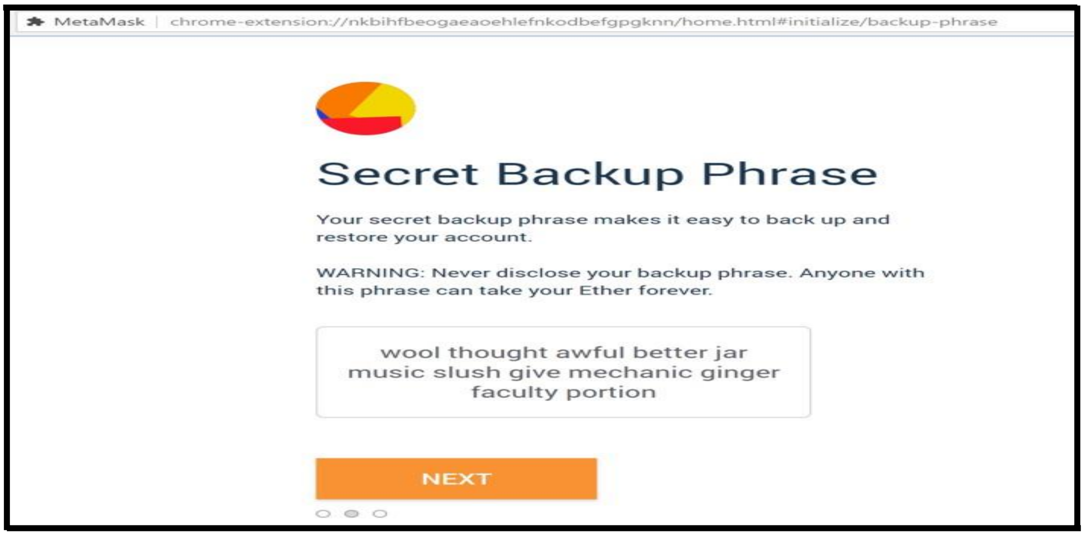


MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#initialize/backup-phrase

Secret Backup Phrase

Your secret backup phrase makes it easy to back up and restore your account.

WARNING: Never disclose your backup phrase. Anyone with this phrase can take your Ether forever.



MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#initialize/backup-phrase

Secret Backup Phrase

Your secret backup phrase makes it easy to back up and restore your account.

WARNING: Never disclose your backup phrase. Anyone with this phrase can take your Ether forever.

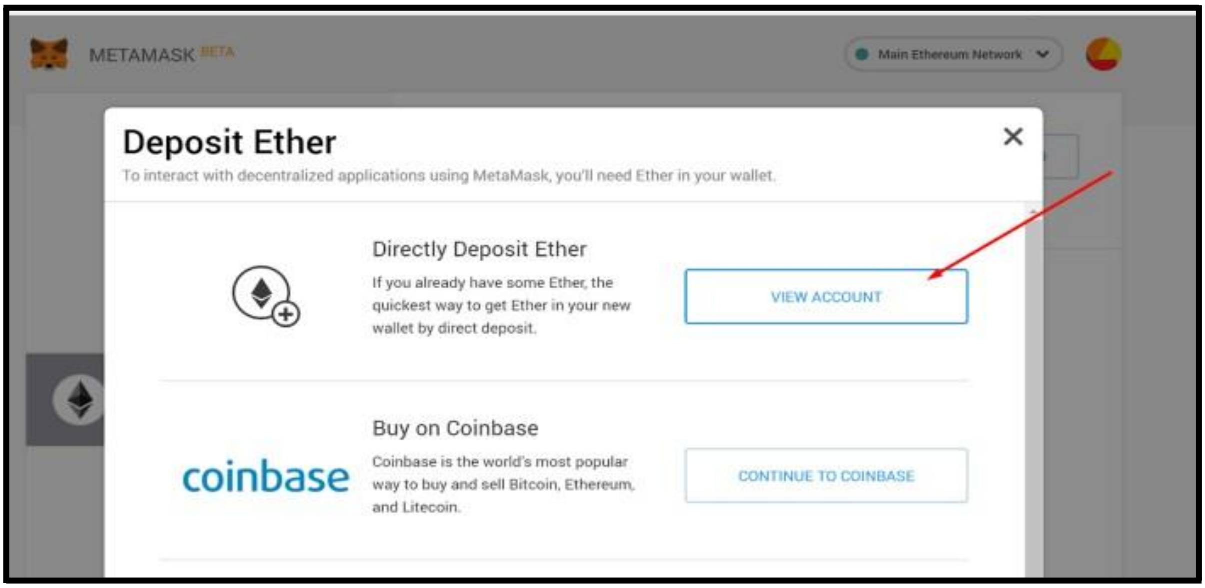
wool thought awful better jar music slush give mechanic ginger faculty portion

- Verify your secret phrase by selecting the previously generated phrase in order. **Click Confirm**.

And that’s it; now you have created your MetaMask account successfully. A new Ethereum wallet address has just been created for you. It’s waiting for you to deposit funds, and if you want to learnhow to do that, look at the next step below.

Step 3. Depositing funds.

- Click on **View Account**.



Deposit Ether

To interact with decentralized applications using MetaMask, you'll need Ether in your wallet.

Directly Deposit Ether

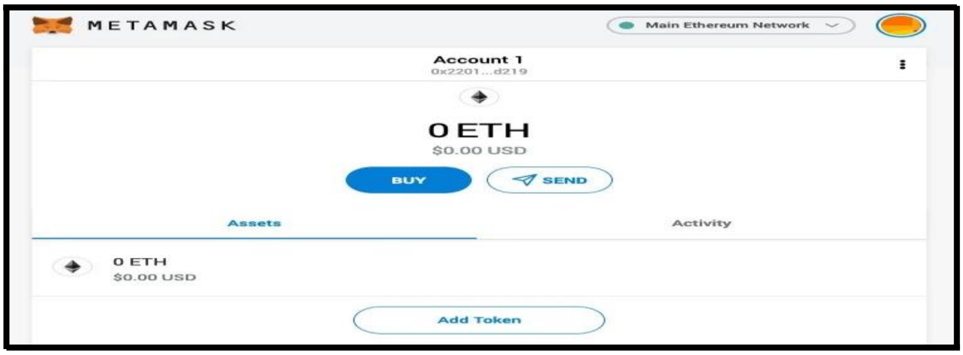
If you already have some Ether, the quickest way to get Ether in your new wallet by direct deposit.

Buy on Coinbase

Coinbase is the world's most popular way to buy and sell Bitcoin, Ethereum, and Litecoin.

You can now see your public address and share it with other people. There are some methods to buy coins offered by MetaMask, but you can do it differently as well; you just need your address.

If you ever get logged out, you’ll be able to log back in again by clicking the MetaMask icon, which will have been added to your web browser (usually found next to the URL bar).



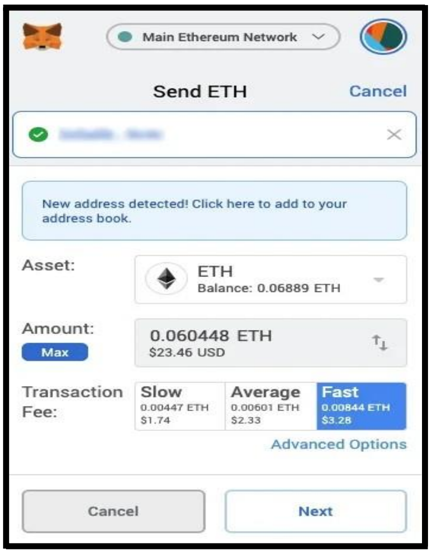
Account 1
0x2201...d219

0 ETH
$0.00 USD

0 ETH
$0.00 USD

You can now access your list of assets in the ‘Assets’ tab and view your transaction history in the‘Activity’ tab.

- Sending crypto is as simple as clicking the ‘Send’ button, entering the recipient address and amount to send, and selecting a transaction fee. You can also manually adjust the transaction fee using the ‘Advanced Options’ button, using information from ETH Gas Station or similar platforms to choose a more acceptable gas price.
- After clicking ‘Next’, you will then be able to either confirm or reject the transaction on the subsequent page.



[redacted]

New address detected! Click here to add to your address book.

Asset: ETH
Balance: 0.06889 ETH

Amount: 0.060448 ETH
$23.46 USD

Transaction Fee:	Slow 0.00447 ETH $1.74	Average 0.00601 ETH $2.33	Fast 0.00844 ETH $3.28

- To use MetaMask to interact with a dapp or smart contract, you’ll usually need to find a ‘Connect to Wallet’ button or similar element on the platform you are trying to use. Afterclicking this, you should then see a prompt asking whether you want to let the dapp
connect to your wallet.

What advantages does MetaMask have?

- **Popular** - It is commonly used, so users only need one plugin to access a wide range ofdapps.

- **Simple** - Instead of managing private keys, users just need to remember a list of words, andtransactions are signed on their behalf.
- **Saves space** - Users don't have to download the Ethereum blockchain, as MetaMask sendsrequests to nodes outside of the user's computer.
- **Integrated** - Dapps are designed to work with MetaMask, so it becomes much easier to sendEther in and out.

Conclusion:

Thus, we have explored Blockchain and MetaMask wallet for transaction of digital currency.

A. Write short answer of following questions:

1. What Are the Different Types of Blockchain Technology?
2. What Are the Key Features/Properties of Blockchain?
3. What Type of Records You Can Keep in A Blockchain?
4. What is the difference between Ethereum and Bitcoin?
5. What are Merkle Trees? Explain their concept.
6. What is Double Spending in transaction operation
7. Give real-life use cases of blockchain.

Sinhgad Institutes

Name of the Student: ________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

BLOCKCHAIN TECHNOLOGY

Assignment No. 02

Creation of wallet using Metamask for crypto transactions

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Create your own wallet using Metamask for crypto transactions.

Objectives:

- To learn about cryptocurrencies.
- To learn howtransaction done by using different digital currency.

Outcomes:

- Create your own wallet using Metamask for crypto transactions.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Create your own wallet using MetaMask for crypto transactions.

Theory:

Introduction to Cryptocurrency

- Cryptocurrency is a digital payment system that doesn't rely on banks to verify transactions. It's a peer-to-peer system that can enable anyone anywhere to send and receive payments. Instead of being physical money carried around and exchanged in the real world, cryptocurrency payments exist purely as digital entries to an online database describing specific transactions. When you transfer cryptocurrency funds, the transactions are recorded in a public ledger. Cryptocurrency is stored in digital wallets.
- Cryptocurrency received its name because it uses encryption to verify transactions. This means advanced coding is involved in storing and transmitting cryptocurrency data between wallets and to public ledgers. The aim of encryption is to provide security and safety.
- The first cryptocurrency was Bitcoin, which was founded in 2009 and remains the best known today. Much of the interest in cryptocurrencies is to trade for profit, with speculatorsat times driving prices skyward.

How does cryptocurrency work?

- Cryptocurrencies run on a distributed public ledger called blockchain, a record of all transactions updated and held by currency holders.
- Units of cryptocurrency are created through a process called mining, which involves using computer power to solve complicated mathematical problems that generate coins. Users can also buy the currencies from brokers, then store and spend them using cryptographic wallets.
- If you own cryptocurrency, you don't own anything tangible. What you own is a key that allows you to move a record or a unit of measure from one person to another without atrusted third party.
- Although Bitcoin has been around since 2009, cryptocurrencies and applications of blockchain technology are still emerging in financial terms, and more uses are expected in the future. Transactions including bonds, stocks, and other financial assets could eventually be traded using the technology.

Cryptocurrency examples

There are thousands of cryptocurrencies. Some of the best known include:

- **Bitcoin:**

Founded in 2009, Bitcoin was the first cryptocurrency and is still the most commonly traded. The currency was developed by Satoshi Nakamoto – widely believed to be a pseudonym for an individual or group of people whose precise identity remains unknown.

- **Ethereum:**

Developed in 2015, Ethereum is a blockchain platform with its own cryptocurrency, called Ether (ETH) or Ethereum. It is the most popular cryptocurrency after Bitcoin.

- **Litecoin:**

This currency is most similar to bitcoin but has moved more quickly to develop new innovations, including faster payments and processes to allow more transactions.

- **Ripple:**

Ripple is a distributed ledger system that was founded in 2012. Ripple can be used to track different kinds of transactions, not just cryptocurrency. The company behind it has worked with various banks and financial institutions.

- Non-Bitcoin cryptocurrencies are collectively known as "altcoins" to distinguish them from the original.

How to store cryptocurrency

- Once you have purchased cryptocurrency, you need to store it safely to protect it from hacks or theft. Usually, cryptocurrency is stored in crypto wallets, which are physical devices or online software used to store the private keys to your cryptocurrencies securely. Someexchanges provide wallet services, making it easy for you to store directly through the platform. However, not all exchanges or brokers automatically provide wallet services for you.
- There are different wallet providers to choose from. The terms “hot wallet” and “cold wallet”are used:
- **Hot wallet storage:** "hot wallets" refer to crypto storage that uses online software to protectthe private keys to your assets.
- **Cold wallet storage:** Unlike hot wallets, cold wallets (also known as hardware wallets) relyon offline electronic devices to securely store your private keys.

Conclusion:

Thus, we have explored Concept Cryptocurrency and learn howtransactions are done using digital currency.

A. Write short answer of following questions:

1. What is Bitcoin?
2. What Are the biggest Four common cryptocurrency scams
3. Explain How safe are money e-transfers?
4. What is cryptojacking and how does it work?

Name of the Student: ________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

BLOCKCHAIN TECHNOLOGY

Assignment No. 03

SMART CONTRACT FOR BANK ACCOUNT

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Write a smart contract on a test network, for Bank account of a customer.

Objectives:

- To write smart contract using Solidity language.

Outcomes:

- Write smart contract for different applications using Solidity language

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Write a smart contract on a test network, for Bank account of a customer for following operations:

- Deposit money
- Withdraw Money
- Show balance

Theory:

What is Ethereum?

Ethereum is a decentralized ie. blockchain platform that runs smart contracts i.e. applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third-party interference.

The Ethereum Virtual Machine (EVM)

The Ethereum Virtual Machine, also known as EVM, is the runtime environment for smart contracts in Ethereum. The Ethereum Virtual Machine focuses on providing security and executing untrusted code by computers all over the world.

The EVM specialised in preventing Denial-of-service attacks and ensures that programs do not have access to each other's state, ensuring communication can be established without any potential interference.

The Ethereum Virtual Machine has been designed to serve as a runtime environment for smart contracts based on Ethereum.

What is Smart Contract?

A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract. Smart contracts allow the performance of credible transactions without third parties. These transactions are trackable and irreversible.

The concept of smart contracts was first proposed by Nick Szabo in 1994. Szabo is a legal scholar and cryptographer known for laying the groundwork for digital currency.

Remix IDE

Remix IDE is an open source web and desktop application. It fosters a fast development cycle and has a rich set of plugins with intuitive GUIs. Remix is used for the entire journey of contract development as well as act as a playground for learning and teaching Ethereum. Remix IDE is part of the Remix Project which is a platform for development tools that use a plugin architecture. It encompasses sub-projects including Remix Plugin Engine, Remix Libs, and of course Remix-IDE.

Remix IDE is a powerful open source tool that helps you write Solidity contracts straight from the browser. It is written in JavaScript and supports both usage in the browser, but run locally and in a desktop version. Remix IDE has modules for testing, debugging and deploying of smart contracts and much more.

Solidity

Solidity is a contract-oriented, high-level programming language for implementing smart contracts. Solidity is highly influenced by C++, Python and JavaScript and has been designed to target the Ethereum Virtual Machine (EVM).

Solidity is statically typed, supports inheritance, libraries and complex user-defined types programming language.

You can use Solidity to create contracts for uses such as voting, crowdfunding, blind auctions, and multi-signature wallets.

First Solidity Smart Contract

In the `contracts` folder, create a file called `AdditionContract.sol`. All Solidity files have the `.sol` extension. In all `.sol` files, we specify the SPDX License Identifier on the first line. Here we can specify MIT or whatever license is applicable to your code.

Pragma

The next line is always the `pragma`, which specifies which Solidity version the source code is written for. There are a few options here, for example `pragma solidity 0.8.0;` specifies that it's written for version 0.8.0, `pragma solidity >=0.6.0 <0.8.0;` means that we're good with any version between 0.6.0 and 0.7.9. Below is the line that we have to add to our contract. The `^` means that it will not compile on anything less than 0.8.0 and it will also not work with a 0.5.0 compiler.

```
pragma solidity ^0.8.0;
```

Contracts

Contracts are created, surprisingly, with the contract keyword. After that, you enter the name of your contract and open a curly brace.

Constructor

Now we move on to the constructor, whose goal is similar to Javascript constructors or to initializers in some other languages.

Function

We declare functions by using the function keyword. After that follows the function's name, and then a couple of keywords. Here we have public, which means that this function is available to the outside world, view, which means the function will not modify the state of the contract, then the returns keyword, signaling that the function will return something, followed by the type of the return thing and its storage location.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.16;
contract AdditionContract
{
   int a;
   int b;
   int res;
   constructor()
   {
      a=10;
      b=20;
   }
   function add() view public returns(int)
   {
      return a+b;
   }
   function getData() view public returns(int,int,int)
   {
      return(a,b,res);
   }
}
```

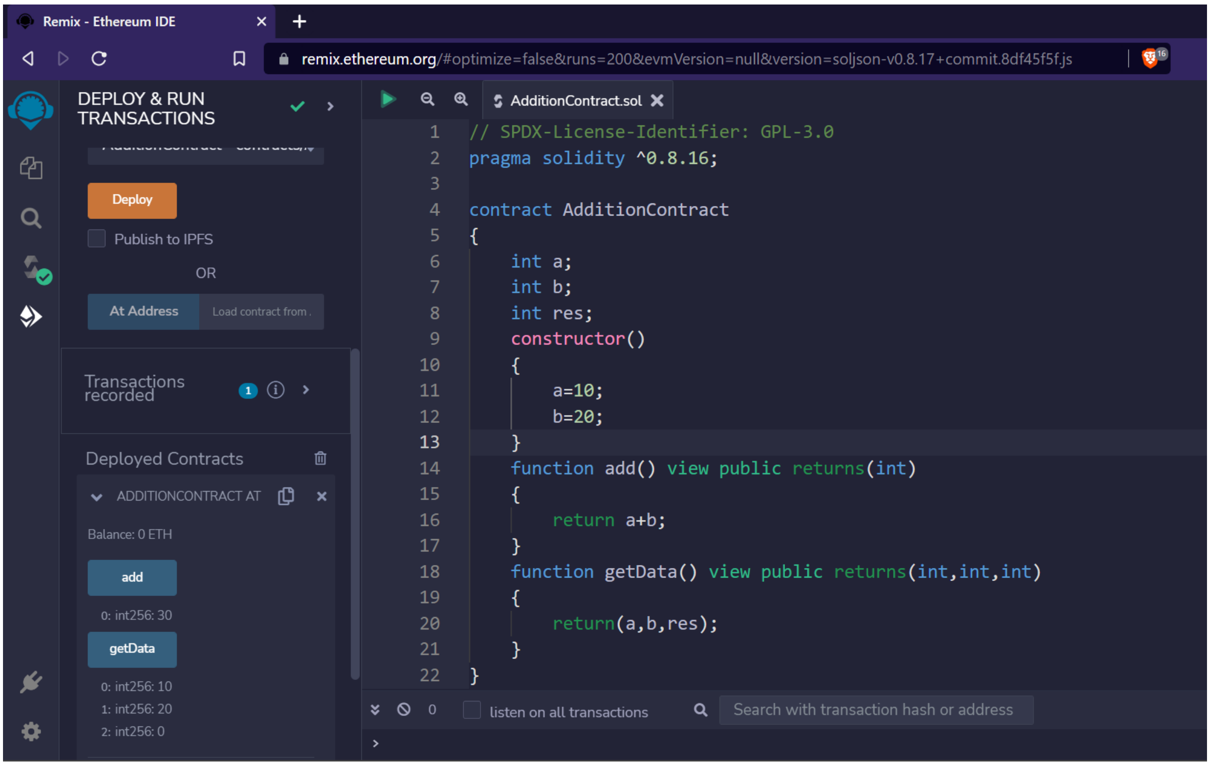
Steps to Execute Solidity Smart Contract using Remix IDE

Step 1: Open Remix IDE on any of your browsers, select on the *New File* and click on *Solidity* to choose the environment.

Step 2: Write the Smart contract in the code section, and click the *Compile button* under the Compiler window to compile the contract.

Step 3: To execute the code, click on the *Deploy button* under Deploy and Run Transactions window.

Step 4: After deploying the code click on the method calls under the drop-down of deployed contracts to run the program, and for output, check to click on the drop-down on the console.



Remix - Ethereum IDE
remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.17+commit.8df45f5f.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from .
Transactions recorded 1
AdditionContract.sol
listen on all transactions
Search with transaction hash or address

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.16;

contract AdditionContract
{
    int a;
    int b;
    int res;
    constructor()
    {
        a=10;
        b=20;
    }
    function add() view public returns(int)
    {
        return a+b;
    }
    function getData() view public returns(int,int,int)
    {
        return(a,b,res);
    }
}
```

Deployed Contracts

ADDITIONCONTRACT AT

Balance: 0 ETH

add

```
0: int256: 30
```

getData

```
0: int256: 10
1: int256: 20
2: int256: 0
```

Conclusion:

Thus, we have written smart contract for bank account of a customer using Solidity language.

A. Write short answer of following questions:

1. How is Solidity used in blockchain?
2. What is smart contract in Solidity?
3. How do you create a smart contract in Solidity?
4. How do you deploy a contract in remix?
5. What are the function types available in Solidity?

Sinhgad Institutes

Name of the Student: ___________________________________ **Roll No:** ___

CLASS: - B. E. [COMP] **Division: A, B & C** **Course: LP-III**

BLOCKCHAIN TECHNOLOGY

Assignment No. 04

SMART CONTRACT FOR STUDENT DATA

		Marks:	/10
Date of Performance:	/ /2022	**Sign with Date:**	

Title : Write a program in Solidity to create Student data.

Objectives:

- To write smart contract using Solidity language to create student data.

Outcomes:

- Write smart contract for different applications using Solidity language.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III **POs: 1, 2, 3, 4, 5** **PSOs: 1, 2** **COs: 1**

Problem Statement:

Write a program in solidity to create Student data. Use the following constructs:

- Structures
- Arrays
- Fallback

Deploy this as smart contract on Ethereum and Observe the transaction fee and Gas values.

Theory:

What is Ethereum?

Ethereum is a decentralized ie. blockchain platform that runs smart contracts i.e. applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third-party interference.

The Ethereum Virtual Machine (EVM)

The Ethereum Virtual Machine, also known as EVM, is the runtime environment for smart contracts in Ethereum. The Ethereum Virtual Machine focuses on providing security and executing untrusted code by computers all over the world.

The EVM specialised in preventing Denial-of-service attacks and ensures that programs do not have access to each other's state, ensuring communication can be established without any potential interference.

The Ethereum Virtual Machine has been designed to serve as a runtime environment for smart contracts based on Ethereum.

What is Smart Contract?

A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract. Smart contracts allow the performance of credible transactions without third parties. These transactions are trackable and irreversible.

Arrays in Solidity

Array is a data structure, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Declaring Arrays

To declare an array of fixed size in Solidity, the programmer specifies the type of the elements and the number of elements required by an array as follows −

```
type arrayName [ arraySize ];
```

Structure in Solidity

Struct types are used to represent a record. Suppose you want to keep track of your books in a library. You might want to track the following attributes about each book −

- Title
- Author
- Subject
- Book ID

Defining a Struct

To define a Struct, you must use the **struct** keyword. The struct keyword defines a new data type, with more than one member. The format of the struct statement is as follows –

```
struct struct_name {
   type1 type_name_1;
   type2 type_name_2;
   type3 type_name_3;
}
```

Example

```
struct Book {
   string title;
   string author;
   uint book_id;
}
```

Fallback function in Solidity

In Solidity, a **fallback function** is an external function with neither a *name*, *parameters*, or *return values*. It is executed in one of the following cases:

- If a function identifier doesn't match any of the available functions in a smart contract.
- If there was no data supplied along with the function call.

Only one such unnamed function can be assigned to a contract.

Properties of the fallback function

- They are unnamed functions.
- They cannot accept arguments.
- They cannot return anything.
- There can be only one fallback function in a smart contract.
- It is compulsory to mark it external.
- It should be marked as *payable*. If not, the contract will throw an exception if it receives ether without any data.
- It is limited to 2300 gas if invoked by other functions.

Error Handling in Solidity

Solidity provides various functions for error handling. Generally when an error occurs, the state is reverted back to its original state. Other checks are to prevent unauthorized code access. Following are some of the important methods used in error handling −

- **assert(bool condition)** − In case condition is not met, this method call causes an invalid opcode and any changes done to state got reverted. This method is to be used for internal errors.
- **require(bool condition)** − In case condition is not met, this method call reverts to original state. - This method is to be used for errors in inputs or external components.
- **require(bool condition, string memory message)** − In case condition is not met, this method call reverts to original state. - This method is to be used for errors in inputs or external components. It provides an option to provide a custom message.
- **revert()** − This method aborts the execution and revert any changes done to the state.
- **revert(string memory reason)** − This method aborts the execution and revert any changes done to the state. It provides an option to provide a custom message.

Data Locations

Solidity provides four types of data locations.

- Storage
- Memory
- Calldata
- Stack

Storage

The storage location is permanent data, which means that this data can be accessed into all functions within the contract. To make it more simple, you can think of it as the hard disk data of your computer where all the data gets stored permanently. Similarly, the storage variable is stored in the state of a smart contract and is persistent between function calls. Keep in mind that storage data location is expensive compared to other data locations.

Memory

The memory location is temporary data and cheaper than the storage location. It can only be accessible within the function. Usually, Memory data is used to save temporary variables for calculation during function execution. Once the function gets executed, its contents are discarded. You can think of it as a RAM of each individual function.

Calldata

Calldata is non-modifiable and non-persistent data location where all the passing values to the function are stored. Also, Calldata is the default location of parameters (not return parameters) of external functions.

Stack

Stack is a non-persistent data maintained by EVM (Ethereum Virtual Machine). EVM uses stack data location to load the variables during execution. Stack location has the limitation up to 1024 levels.

Gas Fees on Ethereum

Gas fees represent the compensation paid to miners and stakers who help make Ethereum network transactions possible.

In order to send and receive crypto on most blockchains, you must pay a transaction fee. This transaction fee can vary widely (from less than USD 0.0001 to over USD 100) and depends on the blockchain you’re using and its current demand for block space. On Ethereum, the transaction fee required to use the network is referred to as the gas fee (or gas price). Ethereum’s native coin is ether (ETH) and transaction fees are paid using ETH.

Conclusion:

Thus, we have written smart contract for student data using Solidity language.

A. Write short answer of following questions:

1. What is gas transaction fee?
2. How do you write an array in Solidity?
3. What are the 4 memory locations of Solidity?
4. How do you set a struct in Solidity? How do you access structure members in Solidity?
5. What is the fallback function?
6. In which scenarios is a fallback function triggered?

Sinhgad Institutes

Name of the Student: ________________________________ Roll No: ___

CLASS: - B. E. [COMP] Division: A, B & C Course: LP-III

BLOCKCHAIN TECHNOLOGY

Assignment No. 05

SURVEY REPORT ON TYPES OF BLOCKCHAINS

Marks: /10

Date of Performance: / /2022 **Sign with Date:**

Title : Write a survey report on types of Blockchains and its real time use cases.

Objectives:

- To study types of Blockchains.
- To study real time use cases of Blockchains.

Outcomes:

- Write a survey report on types of Blockchains and its real time use cases.

PEOs, POs, PSOs and COs satisfied

PEOs: I, III POs: 1, 2, 3, 4, 5 PSOs: 1, 2 COs: 1

Problem Statement:

Write a survey report on types of Blockchains and its real time use cases.

Theory:

What Is a Blockchain?

A blockchain is a distributed database or ledger that is shared among the nodes of a computer network. As a database, a blockchain stores information electronically in digital format. Blockchains are best known for their crucial role in cryptocurrency systems, such as Bitcoin, for maintaining a secure and decentralized record of transactions. The innovation with a blockchain is that it guarantees the fidelity and security of a record of data and generates trust without the need for a trusted third party.

One key difference between a typical database and a blockchain is how the data is structured. A blockchain collects information together in groups, known as blocks, that hold sets of information. Blocks have certain storage capacities and, when filled, are closed and linked to the previously filled block, forming a chain of data known as the blockchain. All new information that follows that freshly added block is compiled into a newly formed block that will then also be added to the chain once filled.

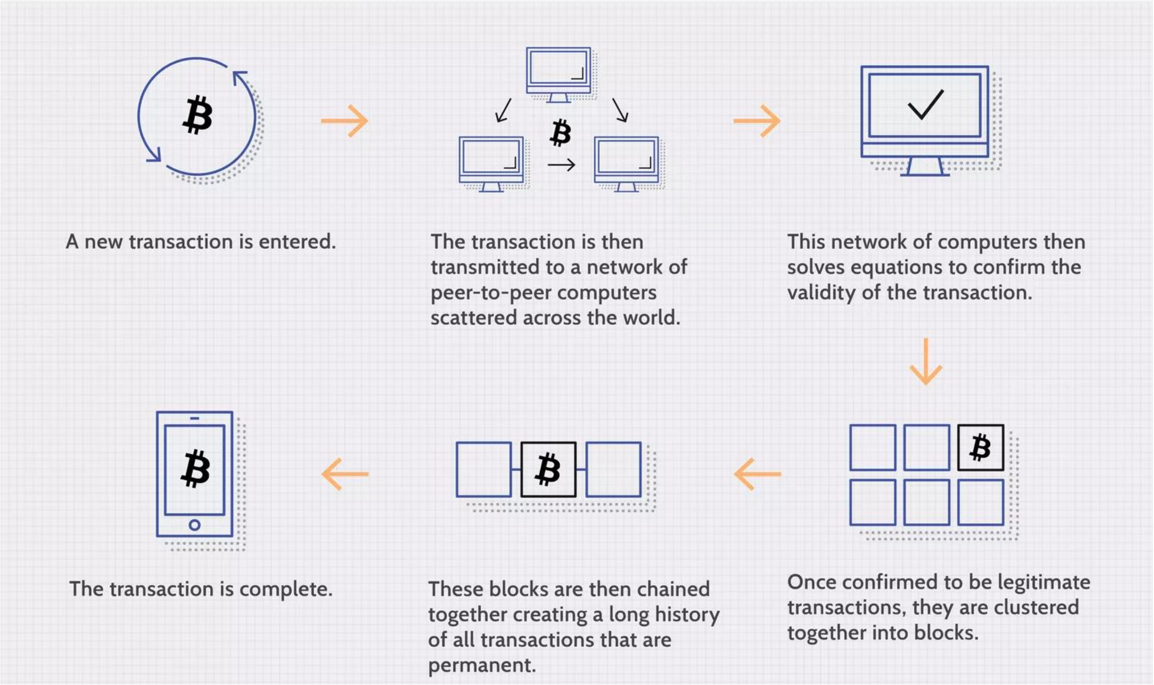
A database usually structures its data into tables, whereas a blockchain, as its name implies, structures its data into chunks (blocks) that are strung together. This data structure inherently makes an irreversible timeline of data when implemented in a decentralized nature. When a block is filled, it is set in stone and becomes a part of this timeline. Each block in the chain is given an exact timestamp when it is added to the chain.

How Does a Blockchain Work?

The goal of blockchain is to allow digital information to be recorded and distributed, but not edited. In this way, a blockchain is the foundation for immutable ledgers, or records of transactions that cannot be altered, deleted, or destroyed. This is why blockchains are also known as a distributed ledger technology (DLT).

First proposed as a research project in 1991, the blockchain concept predated its first widespread application in use: Bitcoin, in 2009. In the years since, the use of blockchains has exploded via the creation of various cryptocurrencies, decentralized finance (DeFi) applications, non-fungible tokens (NFTs), and smart contracts.

Transaction Process


A new transaction is entered.
The transaction is then transmitted to a network of peer-to-peer computers scattered across the world.
This network of computers then solves equations to confirm the validity of the transaction.
Once confirmed to be legitimate transactions, they are clustered together into blocks.
These blocks are then chained together creating a long history of all transactions that are permanent.
The transaction is complete.

Advantage using blockchain :

1. It provides greater trust among users.
2. It provides greater security among data.
3. Reduce the cost of production.
4. Improve Speed.
5. Invocation and tokenization.
6. It provides immutable records.
7. Smart contracts

Disadvantages using blockchain:

1. Data modification is not possible.
2. It requires large storage for a large database.
3. The owner cannot access the private key again if they forget or lose it.

Real life application of blockchain:

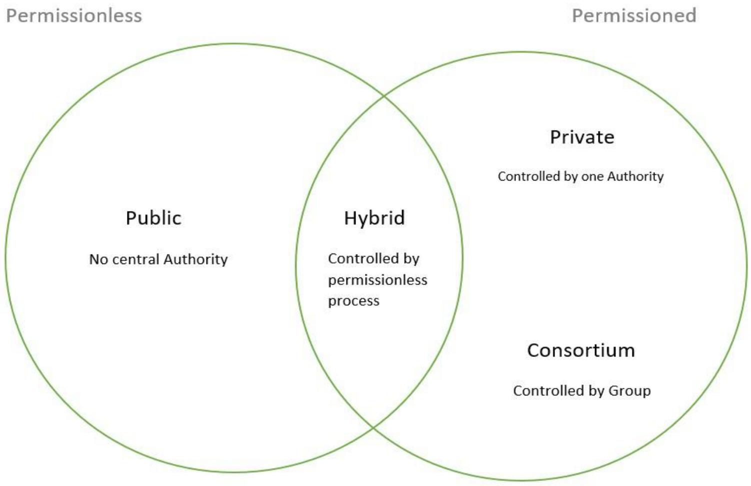
Here is a list of real-world problem where we can use blockchain:

1. In a secure and full-proof voting management system.
2. To supply chain management.
3. In healthcare management.
4. Real estate project.
5. NFT marketplace.
6. Avoid copyright and original content creation.
7. In the personal identity system
8. To make an immutable data backup.
9. Internet of Things

Types of Blockchain

There are 4 types of blockchain:

- Public Blockchain.
- Private Blockchain.
- Hybrid Blockchain.
- Consortium Blockchain.



1. Public Blockchain

These blockchains are completely open to following the idea of decentralization. They don't have any restrictions, anyone having a computer and internet can participate in the network.

- As the name is public this blockchain is open to the public, which means it is not owned by anyone.

- Anyone having internet and a computer with good hardware can participate in this public blockchain.
- All the computer in the network hold the copy of other nodes or block present in the network
- In this public blockchain, we can also perform verification of transactions or records

Advantages:

- **Trustable:** There are algorithms to detect no fraud. Participants need not worry about the other nodes in the network
- **Secure:** This blockchain is large in size as it is open to the public. In a large size, there is greater distribution of records
- **Anonymous Nature:** It is a secure platform to make your transaction properly at the same time, you are not required to reveal your name and identity in order to participate.
- **Decentralized:** There is no single platform that maintains the network, instead every user has a copy of the ledger.

Disadvantages:

- **Processing:** The rate of the transaction process is very slow, due to its large size. Verification of each node is a very time-consuming process.
- **Energy Consumption:** Proof of work is high energy-consuming. It requires good computer hardware to participate in the network
- **Acceptance:** No central authority is there so governments are facing the issue to implement the technology faster.

Use Cases: Public Blockchain is secured with proof of work or proof of stake they can be used to displace traditional financial systems. The more advanced side of this blockchain is the smart contract that enabled this blockchain to support decentralization. Examples of public blockchain are Bitcoin, Ethereum.

2. Private Blockchain

These blockchains are not as decentralized as the public blockchain only selected nodes can participate in the process, making it more secure than the others.

- These are not as open as a public blockchain.
- They are open to some authorized users only.
- These blockchains are operated in a closed network.
- In this few people are allowed to participate in a network within a company/organization.

Advantages:

- **Speed:** The rate of the transaction is high, due to its small size. Verification of each node is less time-consuming.
- **Scalability:** We can modify the scalability. The size of the network can be decided manually.

- **Privacy:** It has increased the level of privacy for confidentiality reasons as the businesses required.
- **Balanced:** It is more balanced as only some user has the access to the transaction which improves the performance of the network.

Disadvantages:

- Security- The number of nodes in this type is limited so chances of manipulation are there. These blockchains are more vulnerable.
- Centralized- Trust building is one of the main disadvantages due to its central nature. Organizations can use this for malpractices.
- Count- Since there are few nodes if nodes go offline the entire system of blockchain can be endangered.

Use Cases: With proper security and maintenance, this blockchain is a great asset to secure information without exposing it to the public eye. Therefore companies use them for internal auditing, voting, and asset management. An example of private blockchains is Hyperledger, Corda.

3. Hybrid Blockchain

It is the mixed content of the private and public blockchain, where some part is controlled by some organization and other makes are made visible as a public blockchain.

- It is a combination of both public and private blockchain.
- Permission-based and permissionless systems are used.
- User access information via smart contracts
- Even a primary entity owns a hybrid blockchain it cannot alter the transaction

Advantages:

- **Ecosystem:** Most advantageous thing about this blockchain is its hybrid nature. It cannot be hacked as 51% of users don't have access to the network
- **Cost:** Transactions are cheap as only a few nodes verify the transaction. All the nodes don't carry the verification hence less computational cost.
- **Architecture:** It is highly customizable and still maintains integrity, security, and transparency.
- **Operations:** It can choose the participants in the blockchain and decide which transaction can be made public.

Disadvantages:

- **Efficiency:** Not everyone is in the position to implement a hybrid Blockchain. The organization also faces some difficulty in terms of efficiency in maintenance.
- **Transparency:** There is a possibility that someone can hide information from the user. If someone wants to get access through a hybrid blockchain it depends on the organization whether they will give or not.

- **Ecosystem:** Due to its closed ecosystem this blockchain lacks the incentives for network participation.

Use Case: It provides a greater solution to the health care industry, government, real estate, and financial companies. It provides a remedy where data is to be accessed publicly but needs to be shielded privately. Examples of Hybrid Blockchain are Ripple network and XRP token.

4. Consortium Blockchain

It is a creative approach that solves the needs of the organization. This blockchain validates the transaction and also initiates or receives transactions.

- Also known as Federated Blockchain.
- This is an innovative method to solve the organization's needs.
- Some part is public and some part is private.
- In this type, more than one organization manages the blockchain.

Advantages:

- **Speed:** A limited number of users make verification fast. The high speed makes this more usable for organizations.
- **Authority:** Multiple organizations can take part and make it decentralized at every level. Decentralized authority, makes it more secure.
- **Privacy:** The information of the checked blocks is unknown to the public view. but any member belonging to the blockchain can access it.
- **Flexible:** There is much divergence in the flexibility of the blockchain. Since it is not a very large decision can be taken faster.

Disadvantages:

- **Approval:** All the members approve the protocol making it less flexible. Since one or more organizations are involved, there can be differences in the vision of interest.
- **Transparency:** It can be hacked if the organization becomes corrupt. Organizations may hide information from the users.
- **Vulnerability:** If few nodes are getting compromised there is a greater chance of vulnerability in this blockchain

Conclusion:

Thus, we have written smart contract for student data using Solidity language.

A. Write short answer of following questions:

1. Write a survey report on types of Blockchains and its real time use cases.
2. What is blockchain and how does it work?
3. What are the 4 different types of blockchain technology?
4. What is the main purpose of blockchain?