Assignment

1. What is the difference between static and dynamic variables in Python?

Answer:

Static variables, also known as class variables, have a fixed memory location throughout the execution of a program. They are declared within a class or a function and retain their values between function calls.

While Dynamic variables, also referred to as instance variables, are allocated memory during runtime. Unlike static variables, dynamic variables have a memory location that changes as the program executes.

2. Explain the purpose of "pop","popitem","clear()" in a dictionary with suitable examples?

Answer: pop() – It returns and removes the element with the given key.

Popitem() - It returns and removes the key-value pair from the dictonary.

Clear()- It removes all items from the dictionary.

Ex- dict={‘name’: ‘Raj’, ’Age’: ’18’}

```
 dict.pop(‘Age’)

 dict.popitem()

dict.clear()

print(dict)
```

3. What do you mean by FrozenSet? Explain it with suitable examples?

Answer: Frozenset is a python method which creates an immutable set object from an iterable. It is a build-in function. There is no duplicate values are present in frozen set.

Ex-

```
animals = frozenset(["cat", "dog", "lion"])
```

```
print("cat" in animals)
print("elephant" in animals)
```

4. Differentiate between mutable and immutable data types in Python and give examples of mutable and immutable data types ?

Mutable Data Types:

Answer- Mutable data types are those whose values can be changed after creation. The memory location of a mutable object remains unchanged when it is modified in-place. As a result, all references to that object will change.

- list: Lists are ordered collections that can be modified by adding, removing, or changing elements.
- dict: Dictionaries are collections of key-value pairs, and we can add, remove, or modify items using their keys.
- set: Sets are unordered collections of unique elements, and we can add or remove elements from them.

Eg. :-

```
list_nums = [10, 20, 30, 40]

list_nums.append(40)

print(list_nums)
```

Immutable data types are those whose values cannot be changed after creation. When you modify an immutable object, you create a new object with the modified value, and the original object remains unchanged. As a result, any variables referencing the original object won't be updated.

- int: Integer data type represents whole numbers, and once created, their value cannot be changed.
- float: Floating-point data type represents real numbers and is immutable.
- str: String data type represents a sequence of characters, and you cannot change its individual characters.
- tuple: A tuple is an ordered collection, similar to a list, but their elements cannot be modified once they have been created.

Eg:

```
str1 = "Python"

new_str1 = str1.lower()

 print(str1)

 print(new_str1)
```

5. **What is __init__?Explain with an example?**

   Ans. __init__ is a contructor method in Python and is automatically called to allocate memory when a new object/instance is created. All classes have a __init__ method associated with them. It helps in distinguishing methods and attributes of a class from local variables.

Eg.
```
class Student:
    def __init__( self, fname, lname, age, section):
      self.firstname = fname
      self.lastname = lname
      self.age = age
      self.section = section
stu1 = Student("Sara", "Ansh", 22, "A2")
```

6. **What is docstring in Python?Explain with an example?**
   - Documentation string or docstring is a multiline string used to document a specific code segment.
   - The docstring should describe what the function or method does**.**

Eg.
```
def multiply_numbers(a, b):
  """
  Multiplies two numbers and returns the result.

  Args:
    a (int): The first number.
    b (int): The second number.

  Returns:
    int: The product of a and b.
  """
  return a * b
print(multiply_numbers(3,5))
```

7. **What are unit tests in Python?**
   - Unit test is a unit testing framework of Python.

- Unit testing means testing different components of software separately. Can you think about why unit testing is important? Imagine a scenario, you are building software that
  uses three components namely A, B, and C. Now, suppose your software breaks at a point time. How will you find which component was responsible for breaking the software? Maybe it was component A that failed, which in turn failed component B, and this actually failed the software. There can be many such combinations.
- This is why it is necessary to test each and every component properly so that we know which component might be highly responsible for the failure of the software.

8. **What is break, continue and pass in Python?**
   Ans: **Break:-**The break statement terminates the loop immediately and the control flows to the statement after the body of the loop.
   - **Continue**:-The continue statement terminates the current iteration of the statement, skips the rest of the code in the current iteration and the control flows to the next iteration of the loop.
   - **Pass:-**As explained above, the pass keyword in Python is generally used to fill up empty blocks and is similar to an empty statement represented by a semi-colon in languages such as Java, C++, Javascript, etc.

```
pat = [1, 3, 2, 1, 2, 3, 1, 0, 1, 3]
for p in pat:
  pass
  if (p == 0):
    current = p
    break
  elif (p % 2 == 0):
    continue
  print(p)    # output => 1 3 1 3 1
print(current)    # output => 0
```

9. **What is the use of self in Python?**

Ans: Self is used to represent the instance of the class. With this keyword, you can access the attributes and methods of the class in python. It binds the attributes with the given arguments. Self is used in different places and often thought to be a keyword. But unlike in C++, self is not a keyword in Python.

10. **What are global, protected and private attributes in Python?**
    - **Global** variables are public variables that are defined in the global scope. To use the variable in the global scope inside a function, we use the global keyword.
    - **Protected** attributes are attributes defined with an underscore prefixed to their identifier eg. _sara. They can still be accessed and modified from outside the class they are defined in but a responsible developer should refrain from doing so.
    - **Private** attributes are attributes with double underscore prefixed to their identifier eg. __ansh. They cannot be accessed or modified from the outside directly and will result in an AttributeError if such an attempt is made.
11. **What are modules and packages in Python?**

Ans : Python packages and Python modules are two mechanisms that allow for modular programming in Python. Modularizing has several advantages -

Simplicity: Working on a single module helps you focus on a relatively small portion of the problem at hand. This makes development easier and less error-prone.
Maintainability: Modules are designed to enforce logical boundaries between different problem domains. If they are written in a manner that reduces interdependency, it is less likely that modifications in a module might impact other parts of the program.
Reusability: Functions defined in a module can be easily reused by other parts of the application.
Scoping: Modules typically define a separate namespace, which helps avoid confusion between identifiers from other parts of the program.
Modules, in general, are simply Python files with a .py extension and can have a set of functions, classes, or variables defined and implemented. They can be imported and initialized once using the import statement. If partial functionality is needed, import the requisite classes or functions using from foo import bar.

Packages allow for hierarchical structuring of the module namespace using dot notation. As, modules help avoid clashes between global variable names, in a similar manner, packages help avoid clashes between module names.Creating a package is easy since it makes use of the system's inherent file structure. So just stuff the modules into a folder and there you have it, the folder name as the package name. Importing a module or its contents from this package requires the package name prefix to the module name

joined by a dot.
12. What are lists and tuples? What is the key difference between the two?

Ans: Lists and Tuples are both sequence data types that can store a collection of objects in Python. The objects stored in both sequences can have different data types. Lists are represented with square brackets ['sara', 6, 0.19], while tuples are represented with parantheses ('ansh', 5, 0.97).

But what is the real difference between the two? The key difference between the two is that while lists are mutable, tuples on the other hand are immutable objects. This means that lists can be modified, appended or sliced on the go but tuples remain constant and cannot be modified in any manner. You can run the following example on Python IDLE to confirm the difference:

Eg.
```
my_tuple = ('sara', 6, 5, 0.97)
my_list = ['sara', 6, 5, 0.97]
print(my_tuple[0])    # output => 'sara'
print(my_list[0])    # output => 'sara'
my_tuple[0] = 'ansh'    # modifying tuple => throws an error
my_list[0] = 'ansh'    # modifying list => list modified
print(my_tuple[0])    # output => 'sara'
print(my_list[0])    # output => 'ansh'
```

13. What is an Interpreted language & dynamically typed language?Write 5 differences between them?

An Interpreted language executes its statements line by line. Languages such as Python, Javascript, R, PHP, and Ruby are prime examples of Interpreted languages. Programs written in an interpreted language runs directly from the source code, with no intermediary compilation step.

Before we understand a dynamically typed language, we should learn about what typing is. Typing refers to type-checking in programming languages. In a strongly-typed language, such as Python, "1" + 2 will result in a type error since these languages don't allow for "type-coercion" (implicit conversion of data types). On the other hand, a weakly-typed language, such as Javascript, will simply output "12" as result.

Type-checking can be done at two stages -

Static - Data Types are checked before execution.
Dynamic - Data Types are checked during execution.
Python is an interpreted language, executes each statement line by line and thus type-checking is done on the fly, during execution. Hence, Python is a Dynamically Typed Language.

14. What are Dict and List comprehensions?

Ans:
Python comprehensions, like decorators, are syntactic sugar constructs that help build altered and filtered lists, dictionaries, or sets from a given list, dictionary, or set. Using comprehensions saves a lot of time and code that might be considerably more verbose (containing more lines of code). Let's check out some examples, where comprehensions can be truly beneficial:

Performing mathematical operations on the entire list
```
my_list = [2, 3, 5, 7, 11]
squared_list = [x**2 for x in my_list]    # list comprehension
# output => [4 , 9 , 25 , 49 , 121]
squared_dict = {x:x**2 for x in my_list}    # dict comprehension
# output => {11: 121, 2: 4 , 3: 9 , 5: 25 , 7: 49}
```
Performing conditional filtering operations on the entire list
```
my_list = [2, 3, 5, 7, 11]
squared_list = [x**2 for x in my_list if x%2 != 0]    # list comprehension
# output => [9 , 25 , 49 , 121]
squared_dict = {x:x**2 for x in my_list if x%2 != 0}    # dict comprehension
# output => {11: 121, 3: 9 , 5: 25 , 7: 49}
```
Combining multiple lists into one
Comprehensions allow for multiple iterators and hence, can be used to combine multiple lists into one.

```
a = [1, 2, 3]
b = [7, 8, 9]
[(x + y) for (x,y) in zip(a,b)]  # parallel iterators
```

```
# output => [8, 10, 12]
[(x,y) for x in a for y in b]    # nested iterators
# output => [(1, 7), (1, 8), (1, 9), (2, 7), (2, 8), (2, 9), (3, 7), (3, 8), (3, 9)]
```

Flattening a multi-dimensional list

A similar approach of nested iterators (as above) can be applied to flatten a multi-dimensional list or work upon its inner elements.

```
my_list = [[10,20,30],[40,50,60],[70,80,90]]
flattened = [x for temp in my_list for x in temp]
# output => [10, 20, 30, 40, 50, 60, 70, 80, 90]
```

15. What are decorators in Python?

Decorators in Python are essentially functions that add functionality to an existing function in Python without changing the structure of the function itself. They are represented the @decorator_name in Python and are called in a bottom-up fashion. For example:

```
# decorator function to convert to lowercase
def lowercase_decorator(function):
  def wrapper():
     func = function()
     string_lowercase = func.lower()
     return string_lowercase
  return wrapper
# decorator function to split words
def splitter_decorator(function):
  def wrapper():
     func = function()
     string_split = func.split()
     return string_split
  return wrapper
@splitter_decorator # this is executed next
@lowercase_decorator # this is executed first
def hello():
  return 'Hello World'
hello()   # output => [ 'hello' , 'world' ]
```

The beauty of the decorators lies in the fact that besides adding functionality to the output of the method, they can even accept arguments for functions and can further modify those arguments before passing it to the function itself. The inner nested function, i.e. 'wrapper' function, plays a significant role here. It is implemented to enforce encapsulation and thus, keep itself hidden from the global scope.

```
# decorator function to capitalize names
def names_decorator(function):
  def wrapper(arg1, arg2):
     arg1 = arg1.capitalize()
     arg2 = arg2.capitalize()
     string_hello = function(arg1, arg2)
```

```
      return string_hello
   return wrapper
@names_decorator
def say_hello(name1, name2):
   return 'Hello ' + name1 + '! Hello ' + name2 + '!'
say_hello('sara', 'ansh')   # output => 'Hello Sara! Hello Ansh!'
```

16. How is memory managed in Python?

Ans:

Memory management in Python is handled by the Python Memory Manager. The memory allocated by the manager is in form of a private heap space dedicated to Python. All Python objects are stored in this heap and being private, it is inaccessible to the programmer. Though, python does provide some core API functions to work upon the private heap space.

Additionally, Python has an in-built garbage collection to recycle the unused memory for the private heap space.

17. What is lambda in Python? Why is it used?

Lambda is an anonymous function in Python, that can accept any number of arguments, but can only have a single expression. It is generally used in situations requiring an anonymous function for a short time period. Lambda functions can be used in either of the two ways:

Assigning lambda functions to a variable:

```
mul = lambda a, b : a * b
print(mul(2, 5))    # output => 10
```

Wrapping lambda functions inside another function:

```
def myWrapper(n):
 return lambda a : a * n
mulFive = myWrapper(5)
print(mulFive(2))    # output => 10
```

18. Explain split() and join() functions in Python?

- You can use split() function to split a string based on a delimiter to a list of strings.
- You can use join() function to join a list of strings based on a delimiter to give a single string.

  Eg. :

  ```
   string = "This is a string."
  string_list = string.split(' ') #delimiter is ‘space’ character or ‘ ‘
  print(string_list) #output: ['This', 'is', 'a', 'string.']
  print(' '.join(string_list)) #output: This is a string.
  ```

19. What are iterators , iterable & generators in Python?

In Python, iterators are used to iterate a group of elements, containers like a list. Iterators are collections of items, and they can be a list, tuples, or a dictionary. Python iterator implements __itr__ and the next() method to iterate the stored elements. We generally use loops to iterate over the collections (list, tuple) in Python.

Iterable is an object, that one can iterate over. It generates an Iterator when passed to iter() method. An iterator is an object, which is used to iterate over an iterable object using the __next__() method. Iterators have the __next__() method, which returns the next item of the object.

In Python, the generator is a way that specifies how to implement iterators. It is a normal function except that it yields expression in the function. It does not implement __itr__ and next() method and reduces other overheads as well.

If a function contains at least a yield statement, it becomes a generator. The yield keyword pauses the current execution by saving its states and then resumes from the same when required.

20. What is the difference between xrange and range in Python?
Ans.

xrange() and range() are quite similar in terms of functionality. They both generate a sequence of integers, with the only difference that range() returns a Python list, whereas, xrange() returns an xrange object.

So how does that make a difference? It sure does, because unlike range(), xrange() doesn't generate a static list, it creates the value on the go. This technique is commonly used with an object-type generator and has been termed as "yielding".

Yielding is crucial in applications where memory is a constraint. Creating a static list as in range() can lead to a Memory Error in such conditions, while, xrange() can handle it optimally by using just enough memory for the generator (significantly less in comparison).

```
for i in xrange(10):    # numbers from o to 9
  print i       # output => 0 1 2 3 4 5 6 7 8 9
for i in xrange(1,10):    # numbers from 1 to 9
  print i       # output => 1 2 3 4 5 6 7 8 9
for i in xrange(1, 10, 2):    # skip by two for next
  print i       # output => 1 3 5 7 9
```

21. Pillars of Oops?
Like other Object-Oriented languages, when creating objects using classes, there are four(4) basic principles for writing clean and concise code. These principles are called the four pillars of object-oriented programming (OOP). These four pillars are Inheritance, Polymorphism, Encapsulation and Abstraction.

22. How will you check if a class is a child of another class?

Following are the ways using which you can access parent class members within a child class:

By using Parent class name: You can use the name of the parent class to access the attributes as shown in the example below:

```
class Parent(object):
  # Constructor
  def __init__(self, name):
     self.name = name

class Child(Parent):
  # Constructor
  def __init__(self, name, age):
     Parent.name = name
     self.age = age

  def display(self):
     print(Parent.name, self.age)

# Driver Code
obj = Child("Interviewbit", 6)
obj.display()
```

By using super(): The parent class members can be accessed in child class using the super keyword.

```
class Parent(object):
  # Constructor
  def __init__(self, name):
     self.name = name

class Child(Parent):
  # Constructor
  def __init__(self, name, age):
     '''
     In Python 3.x, we can also use super().__init__(name)
     '''
     super(Child, self).__init__(name)
     self.age = age

  def display(self):
    # Note that Parent.name cant be used
    # here since super() is used in the constructor
    print(self.name, self.age)

# Driver Code
obj = Child("Interviewbit", 6)
obj.display()
```

23. How does inheritance work in python? Explain all types of inheritance with an example?

Inheritance gives the power to a class to access all attributes and methods of another class. It aids in code reusability and helps the developer to maintain applications without redundant code. The class inheriting from another class is a child class or also called a derived class. The class from which a child class derives the members are called parent class or superclass.

Python supports different kinds of inheritance, they are:

Single Inheritance: Child class derives members of one parent class.

```
# Parent class
class ParentClass:
    def par_func(self):
         print("I am parent class function")

# Child class
class ChildClass(ParentClass):
    def child_func(self):
         print("I am child class function")

# Driver code
obj1 = ChildClass()
obj1.par_func()
obj1.child_func()
```

Multi-level Inheritance: The members of the parent class, A, are inherited by child class which is then inherited by another child class, B. The features of the base class and the derived class are further inherited into the new derived class, C. Here, A is the grandfather class of class C.

```
# Parent class
class A:
   def __init__(self, a_name):
       self.a_name = a_name

# Intermediate class
class B(A):
   def __init__(self, b_name, a_name):
       self.b_name = b_name
       # invoke constructor of class A
       A.__init__(self, a_name)

# Child class
class C(B):
   def __init__(self,c_name, b_name, a_name):
       self.c_name = c_name
       # invoke constructor of class B
       B.__init__(self, b_name, a_name)

   def display_names(self):
      print("A name : ", self.a_name)
      print("B name : ", self.b_name)
      print("C name : ", self.c_name)

#  Driver code
obj1 = C('child', 'intermediate', 'parent')
print(obj1.a_name)
obj1.display_names()
```

Multiple Inheritance: This is achieved when one child class derives members from more than one parent class. All features of parent classes are inherited in the child class.

```
# Parent class1
class Parent1:
   def parent1_func(self):
      print("Hi I am first Parent")

# Parent class2
class Parent2:
   def parent2_func(self):
      print("Hi I am second Parent")

# Child class
class Child(Parent1, Parent2):
   def child_func(self):
      self.parent1_func()
      self.parent2_func()

# Driver's code
obj1 = Child()
obj1.child_func()
```

Hierarchical Inheritance: When a parent class is derived by more than one child class, it is called hierarchical inheritance.

```
# Base class
class A:
    def a_func(self):
       print("I am from the parent class.")

# 1st Derived class
class B(A):
    def b_func(self):
       print("I am from the first child.")

# 2nd Derived class
class C(A):
    def c_func(self):
       print("I am from the second child.")
```

```
# Driver's code
obj1 = B()
obj2 = C()
obj1.a_func()
obj1.b_func()    #child 1 method
obj2.a_func()
obj2.c_func()    #child 2 method
```

10. How do you create a class in Python?

To create a class in python, we use the keyword “class” as shown in the example below:

```
class InterviewbitEmployee:
   def __init__(self, emp_name):
       self.emp_name = emp_name
```

To instantiate or create an object from the class created above, we do the following:

```
emp_1=InterviewbitEmployee("Mr. Employee")
```

To access the name attribute, we just call the attribute using the dot operator as shown below:

```
print(emp_1.emp_name)
# Prints Mr. Employee
```

To create methods inside the class, we include the methods under the scope of the class as shown below:

```
class InterviewbitEmployee:
   def __init__(self, emp_name):
       self.emp_name = emp_name

   def introduce(self):
       print("Hello I am " + self.emp_name)
```

The self parameter in the init and introduce functions represent the reference to the current class instance which is used for accessing attributes and methods of that class. The self parameter has to be the first parameter of any method defined inside the class. The method of the class InterviewbitEmployee can be accessed as shown below:

```
emp_1.introduce()
```

The overall program would look like this:

```
class InterviewbitEmployee:
   def __init__(self, emp_name):
       self.emp_name = emp_name

   def introduce(self):
       print("Hello I am " + self.emp_name)

# create an object of InterviewbitEmployee class
emp_1 = InterviewbitEmployee("Mr Employee")
print(emp_1.emp_name)    #print employee name
```

```
emp_1.introduce()       #introduce the employee
```

24. What is encapsulation? Explain it with an example?

Encapsulation means binding the code and the data together. A Python class is an example of encapsulation.

25. What is Polymorphism in Python?

Polymorphism means the ability to take multiple forms. So, for instance, if the parent class has a method named ABC then the child class also can have a method with the same name ABC having its own parameters and variables. Python allows polymorphism.

Question 1. 2. Which of the following identifier names are invalid and why?
a) Serial_no.
b) 1st_Room
c) Hundred$
d) Total_Marks
e) total-Marks
f) Total Marks
g) True
h) _Percentag

Solution

i. Serial_no. = invalid as we can't use '.' in variable names

ii. 1st_Room = invalid as we can't begin variable names with digits

iii. Hundred$ = invalid as we can't use '$' in variable names

vi. total-Marks = invalid as '-' is not allowed in variable names

iv. Total Marks = invalid as there are two separate words in the variable name

vii. True = invalid as it is a keyword

Machine Learning

1. What is the difference beween Series & Dataframes ?

A Python one-dimensional labelled array called a Pandas Series may hold any form of data (integer, float, string, etc.). It resembles a table in a database or a column in a spreadsheet. Each component of a series has a unique identification thanks to an index. It is possible to create new Series by using lists, arrays, dictionaries, and existing Series objects. They are an essential component of the Pandas library and are commonly used for data manipulation and analysis tasks. The more complex Pandas DataFrame data structure, which resembles a two-dimensional table and is composed of multiple Series objects, also heavily relies on Series.

```
import pandas as pd
```

```
# Create a Pandas Series from a list
data = [1000, 2000, 3000, 4000, 5000]
s = pd.Series(data)

# Print the Series
print(s)
```

A form of data structure in pandas, a well-liked data analysis toolkit for Python, is a single-column DataFrame. This tabular data format has two dimensions, one column, and potentially many rows. It may be compared to a specific instance of a DataFrame in which a single column contains all of the data.
There are numerous ways to generate single-column DataFrames, including picking a single column from a bigger DataFrame or building a new DataFrame from scratch. When formatting and reshaping data in advance of analysis or visualisation, they might be helpful for executing operations on a single column of data.

```
import pandas as pd

# Create a DataFrame with a single column using a Python list
data = [1000, 2000, 3000, 4000, 5000]
df = pd.DataFrame(data, columns=['Column1'])

# Print the DataFrame
print(df)
```

2. Create a database name Travel_Planner in mysql ,and create a table name bookings in that which having attributes (user_id INT, flight_id INT,hotel_id INT, activity_id INT,booking_dae DATE) .fill with some dummy value .Now you have to read the content of his table using pandas as dataframe .Show the output.

3. Difference between loc and iloc?
   Ans: The loc() function is label based data selecting method which means that we have to pass the name of the row or column which we want to select. This method includes the last element of the range passed in it, unlike iloc(). loc() can accept the boolean data unlike iloc(). Many operations can be performed using the loc() method.
   The iloc() function is an indexed-based selecting method which means that we have to pass an integer index in the method to select a specific row/column. This method does not include the last element of the range passed in it unlike loc(). iloc() does not accept the boolean data unlike loc.

4. What is the difference between supervised and unsupervised learning?

Ans: Supervised learning assumes the availability of a teacher or supervisor who classifies the training examples, whereas unsupervised learning must identify the pattern-class information as a part of the learning process.

Supervised learning algorithms utilize the information on the class membership of each training instance. This information allows supervised learning algorithms to detect pattern misclassifications as feedback to themselves. In unsupervised learning algorithms, unlabeled instances are used. They blindly or heuristically process them. Unsupervised learning algorithms often have less computational complexity and less accuracy than supervised learning algorithms.

5. Explain the bias-variance tradeoff ?

Ans: The bias-variance decomposition essentially decomposes the learning error from any algorithm by adding the bias, variance, and a bit of irreducible error due to noise in the underlying dataset.
Necessarily, if you make the model more complex and add more variables, you'll lose bias but gain variance. To get the optimally-reduced amount of error, you'll have to trade off bias and variance. Neither high bias nor high variance is desired.

- High bias and low variance algorithms train models that are consistent, but inaccurate on average.
- High variance and low bias algorithms train models that are accurate but inconsistent.

6. What are precision and recall? How are they different from accuracy?

Ans.: Precision
Precision is the ratio of several events you can correctly recall to the total number of events you recall (mix of correct and wrong recalls).

Precision = (True Positive) / (True Positive + False Positive)

Recall
A recall is the ratio of the number of events you can recall the number of total events.

Recall = (True Positive) / (True Positive + False Negative)

Precision and recall are two evaluation metrics used to measure the performance of a classifier in binary and multiclass classification problems.

Precision measures the accuracy of positive predictions, while recall measures the completeness of positive predictions.

7. What is overfitting and how can it be prevented?

Ans.:
Overfitting happens when the model learns patterns as well as the noises present in the data this leads to high performance on the training data but very low performance for data that the model has not seen earlier. To avoid overfitting there are multiple methods that we can use:

- Early stopping of the model's training in case of validation training stops increasing but the training keeps going on.

- Using regularization methods like L1 or L2 regularization which is used to penalize the model's weights to avoid overfitting.

8. Explain the concept of cross-validation?
Ans.: Cross-Validation in Machine Learning is a statistical resampling technique that uses different parts of the dataset to train and test a machine learning algorithm on different iterations. The aim of cross-validation is to test the model's ability to predict a new set of data that was not used to train the model. Cross-validation avoids the overfitting of data.

K-Fold Cross Validation is the most popular resampling technique that divides the whole dataset into K sets of equal sizes.

9. What is the difference between a classification and a regression problem?
Ans.: Classification is the process of finding or discovering a model or function that helps in separating the data into multiple categorical classes i.e. discrete values. In classification, data is categorized under different labels according to some parameters given in the input and then the labels are predicted for the data.

In a classification task, we are supposed to predict discrete target variables(class labels) using independent features.
In the classification task, we are supposed to find a decision boundary that can separate the different classes in the target variable.

Regression Algorithms
Regression is the process of finding a model or function for distinguishing the data into continuous real values instead of using classes or discrete values. It can also identify the distribution movement depending on the historical data. Because a regression predictive model predicts a quantity, therefore, the skill of the model must be reported as an error in those predictions.

In a regression task, we are supposed to predict a continuous target variable using independent features.
In the regression tasks, we are faced with generally two types of problems linear and non-linear regression.
10. Explain the concept of ensemble learning?
Ans: Ensemble learning is a combination of the results obtained from multiple machine learning models to increase the accuracy for improved decision-making.

Example: A Random Forest with 100 trees can provide much better results than using just one decision tree.
Methods for Independently Constructing Ensembles –

- Majority Vote
- Bagging and Random Forest
- Randomness Injection
- Feature-Selection Ensembles
- Error-Correcting Output Coding

Methods for Coordinated Construction of Ensembles –

- Boosting
- Stacking

11. What is gradient descent and how does it work?

Ans.: Gradient Descent is an optimization algorithm for finding a local minimum of a differentiable function. Gradient descent in machine learning is simply used to find the values of a function's parameters (coefficients) that minimize a cost function as far as possible.

A gradient simply measures the change in all weights with regard to the change in error. You can also think of a gradient as the slope of a function. The higher the gradient, the steeper the slope and the faster a model can learn. But if the slope is zero, the model stops learning. In mathematical terms, a gradient is a partial derivative with respect to its inputs.

Imagine you have a machine learning problem and want to train your algorithm with gradient descent to minimize your cost-function J(w, b) and reach its local minimum by tweaking its parameters (w and b). The image below shows the horizontal axes representing the parameters (w and b), while the cost function J(w, b) is represented on the vertical axes. Gradient descent is a convex function

We know we want to find the values of w and b that correspond to the minimum of the cost function (marked with the red arrow). To start finding the right values we initialize w and b with some random numbers. Gradient descent then starts at that point (somewhere around the top of our illustration), and it takes one step after another in the steepest downside direction (i.e., from the top to the bottom of the illustration) until it reaches the point where the cost function is as small as possible.

12. Describe the difference between both gradient descent and stochastic Gradient descent?

Ans.: Gradient Descent is an optimization algorithm for finding a local minimum of a differentiable function. Gradient descent in machine learning is simply used to find the values of a function's parameters (coefficients) that minimize a cost function as far as possible.

A gradient simply measures the change in all weights with regard to the change in error. You can also think of a gradient as the slope of a function. The higher the gradient, the steeper the slope and the faster a model can learn. But if the slope is zero, the model stops learning. In mathematical terms, a gradient is a partial derivative with respect to its inputs.

Stochastic Gradient Descent

By contrast, stochastic gradient descent (SGD) does this for each training example within the dataset, meaning it updates the parameters for each training example one by one. Depending on the problem, this can make SGD faster than batch gradient descent.

One advantage is the frequent updates allow us to have a pretty detailed rate of improvement.

The frequent updates, however, are more computationally expensive than the batch gradient descent approach. Additionally, the frequency of those updates can result in noisy gradients, which may cause the error rate to jump around instead of slowly decreasing.

13. What is the curse of dimensionality in machine learning?
Ans.: The Curse of Dimensionality refers to the phenomenon where the efficiency and effectiveness of algorithms deteriorate as the dimensionality of the data increases exponentially.
In high-dimensional spaces, data points become sparse, making it challenging to discern meaningful patterns or relationships due to the vast amount of data required to adequately sample the space.
The Curse of Dimensionality significantly impacts machine learning algorithms in various ways. It leads to increased computational complexity, longer training times, and higher resource requirements. Moreover, it escalates the risk of overfitting and spurious correlations, hindering the algorithms' ability to generalize well to unseen data.

14. Explain the difference between L1 and L2 regularization.
Ans.: Regularization is a technique used to avoid overfitting by trying to make the model simpler. One way to apply regularization is by adding the weights to the loss function. This is done to consider minimizing unimportant weights. In L1 regularization, we add the sum of the absolute of the weights to the loss function. In L2 regularization, we add the sum of the squares of the weights to the loss function.

So, both L1 and L2 regularization are ways to reduce overfitting, but to understand the difference it's better to know how they are calculated:
Loss (L2): Cost function + L * weights ^{2}
Loss (L1): Cost function + L * |weights|
Where L is the regularization parameter.

L2 regularization penalizes huge parameters preventing any of the single parameters to get too large. But weights never become zeros. It adds parameters square to the loss. Averting the model from overfitting any single feature.

L1 regularization penalizes weights by adding a term to the loss function which is the absolute value of the loss. This leads to removing small values of the parameters until the parameter hits zero and stays there for the rest of the epochs. Removing completely this specific variable from our calculation. So, it helps for simplifying our model and for feature selection as it shrinks the coefficient to zero which is not significant in the model.

15. What is a confusion matrix and how is it used?
Ans.: A confusion matrix (or error matrix) is a specific table that is used to measure the performance of an algorithm. It is mostly used in supervised learning; in unsupervised learning, it's called the matching matrix. The confusion matrix has two parameters:

- Actual
- Predicted
-

It also has identical sets of features in both of these dimensions.

Consider a confusion matrix (binary matrix) shown below:

Confusion Matrix

Here,

For actual values:

Total Yes = 12+1 = 13

Total No = 3+9 = 12

Similarly, for predicted values:

Total Yes = 12+3 = 15

Total No = 1+9 = 10

For a model to be accurate, the values across the diagonals should be high. The total sum of all the values in the matrix equals the total observations in the test data set.

For the above matrix, total observations = 12+3+1+9 = 25

Now, accuracy = sum of the values across the diagonal/total dataset

= (12+9) / 25

= 21 / 25

= 84%
16. Define AUC-ROC curve?

Ans.: AUC ROC stands for "Area Under the Curve" of the "Receiver Operating Characteristic" curve. The AUC ROC curve is basically a way of measuring the performance of an ML model. AUC measures the ability of a binary classifier to distinguish between classes and is used as a summary of the ROC curve.
It is commonly used in machine learning to assess the ability of a model to distinguish between two classes, typically the positive class (e.g., presence of a disease) and the negative class (e.g., absence of a disease).

17. Explain the k-nearest neighbours algorithm?
Ans.: K nearest neighbor algorithm is a classification algorithm that works in a way that a new data point is assigned to a neighboring group to which it is most similar.

In K nearest neighbors, K can be an integer greater than 1. So, for every new data point, we want to classify, we compute to which neighboring group it is closest.

Let us classify an object using the following example. Consider there are three clusters:

Football
Basketball
Tennis ball
Kluster 1

Let the new data point to be classified is a black ball. We use KNN to classify it. Assume K = 5 (initially).

Next, we find the K (five) nearest data points, as shown.

Kluster 2

Observe that all five selected points do not belong to the same cluster. There are three tennis balls and one each of basketball and football.
When multiple classes are involved, we prefer the majority. Here the majority is with the tennis ball, so the new data point is assigned to this cluster.

18. Explain the basis concept of a Support Vector Machine (SVM)?
Ans: Support Vectors are data points that are nearest to the hyper plane. It influences the position and orientation of the hyper plane. Removing the support vectors will alter the position of the hyper plane. The support vectors help us build our support vector machine model.

19. How does the kernel trick work in SVM?
Ans: The kernel trick relies on the inner products of vectors. For SVMs, the decision function is based on the dot products of vectors within the input space. Kernel functions replace these dot products with a non-linear function that computes a dot product in a higher-dimensional spaceWhat are the different types of kernels used in SVM and when would you use each.
The kernel trick is typically expressed as:

$K(x,y)=\phi(x)\cdot\phi(y)$
where,

x and y are two vectors in the original input space
\phi
is the mapping function to the higher-dimensional space.

20. What is the hyperplane in SVM and how is it determined?
Ans.: A hyperplane is a decision boundary that differentiates the two classes in SVM. A data point falling on either side of the hyperplane can be attributed to different classes. The dimension of the hyperplane depends on the number of input features in the

dataset. If we have 2 input features the hyper-plane will be a line. likewise, if the number of features is 3, it will become a two-dimensional plane.

21. What are the pros and cons of using a Support Vector Machine (SVM)?
Ans.: **Advantages**

- Performs well at classifying non-linear data
- Optimizing margins can help reduce the overfitting of data and allow for capacity control
- Learning without a local minima
- There are many kernels (transformations) that could be used to fit the data unlike any other algorithm
- Often provides sparse solutions
- Performs well on data sets that have many attributes, even if there are relatively very few cases on which to train the model

Limitations

- Choice of the right Kernel
    - The number of possible kernels is infinite and can make it hard to choose the right one
    - Most software uses a few kernels that generalize to many situations, but no kernel generalizes to every situation
- Can be computationally intensive
    - Algorithms can be complex
- Can overfit the model

22. Explain the difference between a hard margin and a soft margin in SVM?
Ans.: When the data is linearly separable, and we don't want to have any misclassifications, we use SVM with a hard margin. However, when a linear boundary is not feasible, or we want to allow some misclassifications in the hope of achieving better generality, we can opt for a soft margin for our classifier.

23. Describe the process of constructing a decision tree?
Ans.: A decision tree typically starts with a single node, which branches into possible outcomes. Each of those outcomes leads to additional nodes, which branch off into other possibilities. This gives it a tree-like shape.

24. Describe the working principle of a decision tree?
Ans.: The process of creating a decision tree involves:

Selecting the Best Attribute: Using a metric like Gini impurity, entropy, or information gain, the best attribute to split the data is selected.
Splitting the Dataset: The dataset is split into subsets based on the selected attribute.
Repeating the Process: The process is repeated recursively for each subset, creating a new internal node or leaf node until a stopping criterion is met (e.g., all instances in a node belong to the same class or a predefined depth is reached).

25. What is information gain and how is it used in decision trees?
Ans.: Information Gain: Measures the reduction in entropy or Gini impurity after a dataset is split on an attribute.
InformationGain=Entropyparent−∑i=1(|Di||D|∗Entropy(Di)), where Di is the subset of D after splitting by an attribute.

26. Explain Gini impurity and its role in decision trees?
Gini Impurity: Measures the likelihood of an incorrect classification of a new instance if it was randomly classified according to the distribution of classes in the dataset.
Gini=1−∑i=1(pi)2Gini=1−∑i=1n(pi)2, where pi is the probability of an instance being classified into a particular class.

27. What are the advantages and disadvantages of decision trees?
Ans.: Adavantges

- Simplicity and Interpretability: Decision trees are easy to understand and interpret. The visual representation closely mirrors human decision-making processes.
- Versatility: Can be used for both classification and regression tasks.
- No Need for Feature Scaling: Decision trees do not require normalization or scaling of the data.
- Handles Non-linear Relationships: Capable of capturing non-linear relationships between features and target variables.

Disadvantages

- Overfitting: Decision trees can easily overfit the training data, especially if they are deep with many nodes.
- Instability: Small variations in the data can result in a completely different tree being generated.
- Bias towards Features with More Levels: Features with more levels can dominate the tree structure.

28. How do random forests improve upon decision trees?
Ans.: The problem with decision trees is that they can easily become too complex, leading to overfitting. Overfitting occurs when a model becomes too specific to the training data, and as a result, performs poorly on new data. Decision trees are particularly prone to overfitting when the data is noisy or contains a lot of irrelevant features.

29. How does a random forest algorithm work?
Ans.: Random forests are an extension of decision trees that work by constructing multiple decision trees and combining their results. Instead of relying on a single decision tree, random forests generate a collection of decision trees, each trained on a random subset of the input data.

The idea behind this approach is that by combining multiple models, the overall prediction accuracy will be improved. Random forests also help to reduce overfitting by using a technique called “bootstrap aggregating” or “bagging”.

30. What is bootstrapping in the context of random forests?
Ans.: Bootstrapping is a statistical resampling technique that involves random sampling of a dataset with replacement. It is often used as a means of quantifying the uncertainty associated with a machine learning model.

31. Explain the concept of feature importance in random forests?
Ans.: The final feature importance, at the Random Forest level, is it's average over all the trees. The sum of the feature's importance value on each trees is calculated and divided by the total number of trees: RFfi sub(i)= the importance of feature i calculated from all trees in the Random Forest model.

32. What are the key hyperparameters of a random forest and how do they affect the model?
Ans.: The key hyperparameters of a random forest are:

- max_depth : The max_depth of a tree in Random Forest is defined as the longest path between the root node and the leaf node.
- min_sample_split: min_sample_split – a parameter that tells the decision tree in a random forest the minimum required number of observations in any given node in order to split it.
- max_leaf_nodes: This hyperparameter sets a condition on the splitting of the nodes in the tree and hence restricts the growth of the tree. If after splitting we have more terminal nodes than the specified number of terminal nodes, it will stop the splitting and the tree will not grow further.
- min_samples_leaf: Time to shift our focus to min_sample_leaf. This Random Forest hyperparameter specifies the minimum number of samples that should be present in the leaf node after splitting a node.
- n_estimators : This means that choosing a large number of estimators in a random forest model is not the best idea. Although it will not degrade the model, it can save you the computational complexity and prevent the use of a fire extinguisher on your CPU!
- max_sample (bootstrap sample): The max_samples hyperparameter determines what fraction of the original dataset is given to any individual tree. You might be thinking that more data is always better. Let’s try to see if that makes sense.
- max_features: Finally, we will observe the effect of the max_features hyperparameter. This resembles the number of maximum features provided to each tree in a random forest.

33. Describe the logistic regression model and its assumptions?
Ans: Logistic regression is a statistical model used to predict the probability of a binary outcome based on one or more independent variables. Its primary purpose in machine learning is to classify data into different categories and understand the relationship between the independent and outcome variables.
Critical Assumptions of Logistic Regression
In logistic regression analysis, the assumptions of linearity and independence are important because they ensure that the relationships between the independent and dependent variables are consistent. This lets you make accurate predictions. Violating these assumptions can compromise the validity of the analysis and its usefulness in

making informed pricing decisions, thus highlighting the importance of these assumptions.

34. How does logistic regression handle binary classification problems?
Ans.: Logistic regression is a binary classification technique and it is an approach for prediction tasks. The core of this method is the logistic function that is a sigmoid function (an S-shaped curve). Through this function the logistic regression maps a weighted linear combination of features into real values between 0 and 1. These real values can be interpreted as probabilities: rather than predicting a class, predicting a probability of belonging to a given class of data. Logistic regression has been used in several medical diagnostic applications.

35. What is the sigmoid function and how is it used in logistic regression?
Ans.: Logistic regression is used for binary classification, where the goal is to predict one of two possible outcomes, typically represented as 0 and 1. The sigmoid function is at the core of logistic regression, serving as the link function that maps the linear combination of input features to a probability.

36. Explain the concept of the cost function in logistic regression?
Ans.: The logistic regression cost function, also known as the log loss or cross-entropy loss, is a measure of the error between the predicted probabilities and true class labels.

37. How can logistic regression be extended to handle multilclass classification?
Ans.: It can be extended to handle multiclass problems by using techniques such as softmax regression or multinomial logistic regression. These extensions allow logistic regression to classify instances into multiple classes by estimating the probabilities of each class.

38. What is the difference between L1 and L2 regularization in logistic regression?
Ans.: L1 Regularization (Lasso): Encourages sparsity in the model parameters. Some coefficients can shrink to zero, effectively performing feature selection. L2 Regularization (Ridge): It shrinks the coefficients evenly but does not necessarily bring them to zero. It helps with multicollinearity and model stability.

39. What is XGBoost and how does it differ from other boosting algorithms?
Ans.: XGBoost builds a predictive model by combining the predictions of multiple individual models, often decision trees, in an iterative manner. The algorithm works by sequentially adding weak learners to the ensemble, with each new learner focusing on correcting the errors made by the existing ones.

40. Explain the concept of boosting in the context of ensemble learning?

Ans.: Boosting is an ensemble learning method that combines a set of weak learners into a strong learner to minimize training errors. Boosting algorithms can improve the predictive power of your data mining initiatives.

41. How does XGBoost handle missing values?
Ans. XGBoost supports missing values by default. In tree algorithms, branch directions for missing values are learned during training. It is important to note that the gblinear booster treats missing values as zeros. During the training time XGB decides whether the missing values should fall into the right node or left.

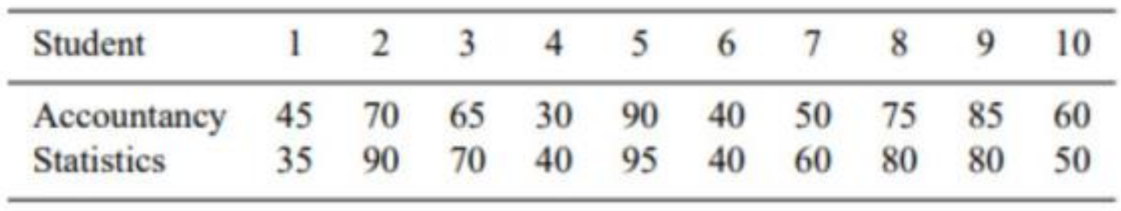
42. What are the key hyperparameters in XGBoost and how do they affect model performance?
43. Describe the process of Gradient boosting in XGBoost?
Ans Gradient boosting is a supervised learning algorithm. This means that it takes a set of labelled training instances as input and builds a model that aims to correctly predict the label of each training example based on other non-label information that we know about the example (known as features of the instance)What are the advantages and disadvanta0es of using XGBoost?
Ans. Advantages:

- XGB consists of a number of hyper-parameters that can be tuned — a primary advantage over gradient boosting machines.
- XGBoost has an in-built capability to handle missing values.
- It provides various intuitive features, such as parallelisation, distributed computing, cache optimisation, and more.

Disadvantages:

- Like any other boosting method, XGB is sensitive to outliers.
- Unlike LightGBM, in XGB, one has to manually create dummy variable/ label encoding for categorical features before feeding them into the models.

Que. 24. Calculate coefficient of correlation between the marks obtained by 10 students in Accountancy and
statistics:



Student	1	2	3	4	5	6	7	8	9	10
Accountancy	45	70	65	30	90	40	50	75	85	60
Statistics	35	90	70	40	95	40	60	80	80	50

Use Karl Pearson's Coefficient of Correlation Method to find it.

X	Y	X^2	Y^2	XY
45	35	2025	1225	1575
70	90	4900	8100	6300
65	70	4225	4900	4550

30	40	900	1600	1200
90	95	8100	9025	8500
40	40	1600	1600	1600
50	60	2500	3600	3000
75	80	5625	6400	6000
85	80	7225	6400	6800
60	50	3600	2500	3000
Total=610	Total=640	Total=40700	Total=45350	Total=42575

N=10
R=10*42575-610*640/sqrt(10*40700-(610)^2)*sqrt(10*45350-(640)^2)
R=35350/186.81*209.52
R=0.9031
Hence, Karl Pearson's coefficient is 0.9031.

27. In a partially destroyed laboratory record of an analysis of correlation data, the following results only are
legible: Variance of x = 9, Regression equations are: (i) 8x−10y = −66; (ii) 40x − 18y = 214. What are (a) the
mean values of x and y, (b) the coefficient of correlation between x and y, (c) the σ of y.

Ans. The regression equations are:
1. 8x - 10y = -66
2. 40x - 18y = 214

(a) Mean Values of x and y

The regression equations can be rewritten in the form y = mx + c .
First, let's solve for x and y when the lines intersect. This intersection point will give us the mean values of x and y.

8x - 10y = -66 -------------Equation 1
40x - 18y = 214 -------------Equation 2

To solve these simultaneously, we can multiply Equation 1 by 2 to make the coefficients of y comparable:
16x - 20y = -132 -------------Equation 3
40x - 18y = 214 ---------------Equation 2

Subtract Equation 3 from Equation 2:
(40x - 18y) - (16x - 20y) = 214 - (-132)
40x - 18y - 16x + 20y = 214 + 132
24x + 2y = 346
24x + 2y = 346
Divide by 2:
12x + y = 173 -------------Equation 4

Now solve for y from Equation 4:

y = 173 - 12x

Substitute this y into Equation 1 to solve for x:
8x - 10(173 - 12x) = -66
8x - 1730 + 120x = -66
128x - 1730 = -66
128x = 1664
x = 1664/128= 13

Now substitute x = 13 back into Equation 4 to find y:
y = 173 - 12(13) = 173 - 156 = 17

So, the mean values are:
bar{x} = 13
bar{y} = 17

(b) The Coefficient of Correlation (r)

The coefficient of correlation can be found using the slopes of the regression lines. The regression equations can be rewritten as:
y = 4/5x + 33/5
y = 40/18x – 214/18
Simplifying, the slopes are:
m1 = 4/5
m2 =20/9
The correlation coefficient r is given by:
r = sqrt(m1 * m2)
Substitute m1 and m2 :
r = sqrt(4/5 *20/9)= sqrt (80/45) = sqrt(16/9) = 4/3

Since the coefficient of correlation cannot be greater than 1, we must have made a calculation error. Let's re-evaluate the slopes correctly.

Slope of the first equation when y = dependent variable:
y = 8/10x + 66/10 = 4/5

Slope of the second equation (when y = dependent variable):
y = 40/18x – 214/18 = 20/9

Correctly:
r = sqrt (4/5*9/20) = sqrt(36/100) = 6/10 = 0.6

Thus, the correct correlation coefficient is:
r = 0.6

(c) The Standard Deviation of sigma(y)

Given that the variance of x (sigma(x)^2) is 9, the standard deviation of x is:
Sigma(x) = sqrt(9) = 3

Sigma(y) = b(y.x) *sigma(x)

To find the regression coefficient b(y.x), we can use:

b(yx) = sigma(y)/sigma(x) = sigma(x)*sigma(y) . r

Thus we have:
Sigma(y)= b(yx). Sigma(x)

Regression coefficient b(yx)can be calculated from slope:
B(yx) = 20/9
Sigma(y) = 20/9* 3 = 60/9 = 20/3

So the standard deviation of y is:
Sigma(y) = 20/3

30. Which of the following options are correct about Normal Distribution Curve. (a) Within a range 0.6745 of σ on both sides the middle 50% of the observations occur i,e. mean ±0.6745σ covers 50% area 25% on each side. (b) Mean ±1S.D. (i,e.μ ± 1σ) covers 68.268% area, 34.134 % area lies on either side of the mean. (c) Mean ±2S.D. (i,e. μ ± 2σ) covers 95.45% area, 47.725% area lies on either side of the mean. (d) Mean ±3 S.D. (i,e. μ ±3σ) covers 99.73% area, 49.856% area lies on the either side of the mean. (e) Only 0.27% area is outside the range μ ±3σ.
To determine which of the given statements about the normal distribution curve are correct, let's review some key properties of the normal distribution:

1. The normal distribution is symmetric about the mean.
2. The total area under the normal distribution curve is 1 (or 100%).
3. Specific percentages of the data fall within certain numbers of standard deviations (σ) from the mean (μ).

The standard properties of the normal distribution are as follows:

- About 68.27% of the data falls within ±1σ of the mean.
- About 95.45% of the data falls within ±2σ of the mean.
- About 99.73% of the data falls within ±3σ of the mean.

Given this, let's evaluate each statement:

(a) Within a range 0.6745 of σ on both sides the middle 50% of the observations occur, i.e., mean ±0.6745σ covers 50% area 25% on each side.

This statement is true. The value 0.6745σ corresponds to the 25th and 75th percentiles of the normal distribution, meaning 50% of the data lies between mean ±0.6745σ.

(b) Mean ±1S.D. (i.e., μ ± 1σ) covers 68.268% area, 34.134% area lies on either side of the mean.

This statement is true. Approximately 68.27% of the data falls within ±1σ of the mean, so about 34.134% lies on each side of the mean.

(c) Mean ±2S.D. (i.e., μ ± 2σ) covers 95.45% area, 47.725% area lies on either side of the mean.

This statement is true. Approximately 95.45% of the data falls within ±2σ of the mean, so about 47.725% lies on each side of the mean.

(d) Mean ±3 S.D. (i.e., μ ±3σ) covers 99.73% area, 49.856% area lies on either side of the mean.

This statement is incorrect. While it's true that approximately 99.73% of the data falls within ±3σ of the mean, the area on each side of the mean is not 49.856%. Instead, it should be half of 99.73%, which is 49.865%.

(e) Only 0.27% area is outside the range μ ±3σ.

This statement is true. Since 99.73% of the data lies within ±3σ of the mean, the remaining 0.27% of the data lies outside this range.

Therefore, the correct statements are (a), (b), (c), and (e).

31. The mean of a distribution is 60 with a standard deviation of 10. Assuming that the distribution is normal,
what percentage of items be (i) between 60 and 72, (ii) between 50 and 60, (iii) beyond 72 and (iv) between
70 and 80?

Solve:

z=x−μ/σ

(i) Percentage of items between 60 and 72

First, calculate the z-scores for 60 and 72:

z60=60−60/10=0

z72=72−60=1.2

Using the standard normal distribution table (or a calculator), the area to the left of z=0 is 0.5 (50%) and the area to the left of z=1.2 is approximately 0.8849 (88.49%).

Therefore, the percentage of items between 60 and 72 is:

0.8849−0.5=0.3849 or 38.49%0.8849 - 0.5 = 0.3849 \text{ or }
38.49\%0.8849−0.5=0.3849 or 38.49%

(ii) Percentage of items between 50 and 60

First, calculate the z-scores for 50 and 60:

$z50=50-60/10=-1$

$z60=60-60/10=0$

Using the standard normal distribution table (or a calculator), the area to the left of $z=-1$ is approximately 0.1587 (15.87%) and the area to the left of $z=0$ is 0.5 (50%).

Therefore, the percentage of items between 50 and 60 is:

$0.5-0.1587=0.3413$ or 34.13%

(iii) Percentage of items beyond 72

First, calculate the z-score for 72:

$z72=72-60/10=1.2$

Using the standard normal distribution table (or a calculator), the area to the left of $z=1.2$ is approximately 0.8849 (88.49%).

Therefore, the percentage of items beyond 72 is:

$1-0.8849=0.1151$ or 11.51%

(iv) Percentage of items between 70 and 80

First, calculate the z-scores for 70 and 80:

$z70=70-60/10=1$

$z80=80-60/10=2$

Using the standard normal distribution table (or a calculator), the area to the left of $z=1$ is approximately 0.8413 (84.13%) and the area to the left of $z=2$ is approximately 0.9772 (97.72%).

Therefore, the percentage of items between 70 and 80 is:

$0.9772-0.8413=0.1359$ or 13.59%

33. If the height of 500 students are normally distributed with mean 65 inch and standard deviation 5 inch. How many students have height : a) greater than 70 inch. b) between 60 and 70 inch

$z=\sigma x-\mu$

(a) Number of students with height greater than 70 inches

First, calculate the z-score for 70 inches:

$z=70-65/5=1$

Using the standard normal distribution table (or a calculator), the area to the left of $z=1$is approximately 0.8413 (84.13%).

Therefore, the percentage of students with heights greater than 70 inches is:

$1-0.8413=0.1587$ or 15.87%

Given that there are 500 students, the number of students with height greater than 70 inches is:

$500\times0.1587=79.35$

Since the number of students must be a whole number, we round to the nearest whole number:

≈79 students

(b) Number of students with height between 60 and 70 inches

First, calculate the z-scores for 60 inches and 70 inches:

$z60=60-65/5=-1$

$z70=70-65/5=1$

Using the standard normal distribution table (or a calculator), the area to the left of $z=-1$ is approximately 0.1587 (15.87%) and the area to the left of $z=1z = 1z=1$ is approximately 0.8413 (84.13%).

Therefore, the percentage of students with heights between 60 and 70 inches is:

$0.8413-0.1587=0.6826$ or 68.26%

Given that there are 500 students, the number of students with height between 60 and 70 inches is:

$500\times0.6826=341.3$

Since the number of students must be a whole number, we round to the nearest whole number:

≈341 students

35.A random sample of size 25 from a population gives the sample standard derivation to be 9.0. Test the hypothesis that the population standard derivation is 10.5. Hint(Use chi-square distribution).
Ans. To test the hypothesis that the population standard deviation is 10.5 using a sample standard deviation of 9.0 and a sample size of 25, we use the chi-square (χ2) test for variance.

Hypotheses:

- Null Hypothesis (H0): The population standard deviation is 10.5 (σ=10.5).
- Alternative Hypothesis (Ha): The population standard deviation is not 10.5 (σ≠10.5).

The test statistic is:

χ2=(n−1)s2/σ2

where:

- n=25n = 25n=25 (sample size)
- s=9.0s = 9.0s=9.0 (sample standard deviation)
- σ=10.5\sigma = 10.5σ=10.5 (hypothesized population standard deviation)

χ2=10.52(25−1)×9^2/10.5^2
χ2=24×81/110.25
χ2=1944/110.25
χ2≈17.63

The degrees of freedom (df) is n−1=24.
The critical values at α/2=0.025 (two-tailed) are approximately
X^2 0.025,24=39.364 and χ^20.975,24 = 12.401.
Since our test statistic χ2=17.63 falls between the critical values, we fail to reject the null hypothesis.

37.100 students of a PW IOI obtained the following grades in Data Science paper :
Grade :[A, B, C, D, E]
Total Frequency :[15, 17, 30, 22, 16, 100]
Using the χ 2 test , examine the hypothesis that the distribution of grades is uniform.

Ans. Hypotheses:

- Null Hypothesis (H0): The distribution of grades is uniform.
- Alternative Hypothesis (Ha): The distribution of grades is not uniform.

The observed frequencies (Oi) are:

[15,17,30,22,16]

Since we have 5 grades and 100 students, the expected frequency (Ei) for each grade under the uniform distribution is:

$$Ei=100/5=20$$

The chi-square test statistic is:

$$\chi2=\sum (Oi-Ei)^2/ Ei$$

$\chi2=(15-20)^2/20+ (17-20)^2/20+(30-20)^2/20+(22-20)^2/20+(16-20)^2/20$
$\chi2=(-5)^2/20+(-3)^2/20+(10)^2/20+(2)^2/20+(-4)^2/20$

$\chi2=25/20+9/20+100/20+4/20+16/20$

$\chi2=1.25+0.45+5+0.2+0.8$

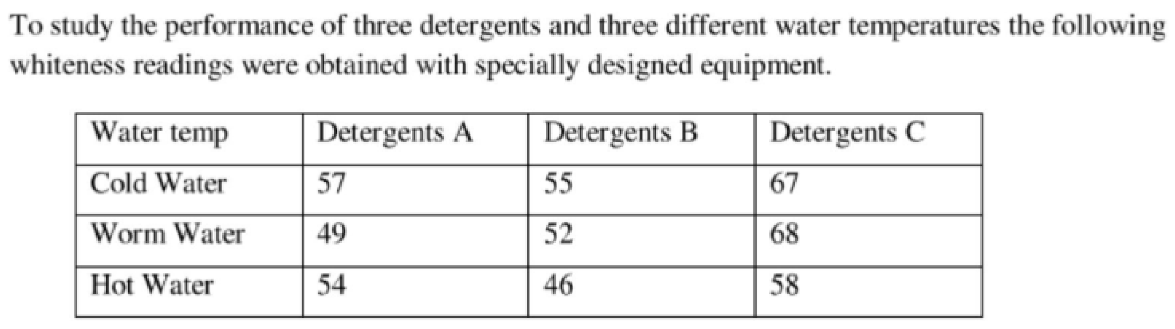
$\chi2=7.7$

df=k-1=5-1=4

df=4

The critical value at $\alpha=0.05$ is χ^20.05,4=9.488

Since our test statistic $\chi2=7.7$ is less than the critical value, we fail to reject the null hypothesis.



To study the performance of three detergents and three different water temperatures the following whiteness readings were obtained with specially designed equipment.

Water temp	Detergents A	Detergents B	Detergents C
Cold Water	57	55	67
Worm Water	49	52	68
Hot Water	54	46	58

Calculate the grand mean (Bar(Xgrand)

Bar(Xgrand) = (57+55+67+49+52+68+54+46+58)/9=506/9=56.22

Now calculate mean:

Detergent Means (X¯A,X¯B,X¯C):

Bar(A)= 57+49+54/3

=160/3=53.33

Bar(B)=(55+52+46)/3=153/3=51

Bar(C)= 67+68+58/3=193/3=64.33

Water Temperature Means

Bar(X)Cold=(57+55+67)/3=179/3 = 59.67

Bar(X)Warm=(49+52+68)/3=169/3 = 56.33

Bar(X)Hot=(54+46+58)/3=158/3=52.67

Now calculate the sum of squares:

SST=(57−56.22)^2+(55−56.22)^2+(67−56.22)^2+(49−56.22)^2+(52−56.22)^2+(68−56.22)^2+(54−56.22)^2+(46−56.22)^2+(58−56.22)^2

SST=0.6084+1.4884+115.3924+52.2884+17.7984+139.6884+5.2484+105.7284+3.1684

SST =441.408

Sum of Squares for Rows (SSR):

SSR=3((59.67−56.22)^2+(56.33−56.22)^2+(52.67−56.22)^2)
SSR=3(11.9889+0.0121+12.6129)
SSR=3×24.6139=73.842

Sum of Squares for Columns (SSC)
SSC=3((53.33−56.22)^2+(51−56.22)^2+(64.33−56.22)^2)
SSC=3(8.3378+27.2484+65.2822)
SSC=3×100.8684=302.605

Sum of Squares for Error (SSE):
SSE=SST−SSR−SSC
SSE=441.408−73.842−302.605
SSE=64.961

Calculate the degrees of freedom:

- **Degrees of Freedom for Rows (dfR):**

  $dfR=r-1=3-1=2$

- **Degrees of Freedom for Columns (dfC):**

  $dfC=c-1=3-1=2$

  - **Degrees of Freedom for Error (dfE):**

  $dfE=(r-1)(c-1)=2\times2=4$

- **Calculate the Mean Squares:**
  - **Mean Square for Rows (MSR):**

    $MSR=SSR/dfR = 73.842/2=36.921$

  - **Mean Square for Columns (MSC):**

    $MSC=SSC/dfC=302.605/2=151.302$

  - **Mean Square for Error (MSE):**

    $MSE=SSE/dfE=64.961/4=16.24025$

- **Calculate the F-statistics:**
  - **F-statistic for Rows:**

    $FR=MSR/MSE=36.921/16.24025 = 2.27$

  - **F-statistic for Columns:**

    $FC=MSC/MSE=151.302/16.24025 = 9.32$

- **Determine the critical F-value and compare:**

  For $\alpha=0.05$ and $dfR=2$, $dfE=4$:

  - Critical F-value for rows: $F(0.05,2,4)\approx6.94$

  For $\alpha=0.05$ and $dfC=2,dfE=4$:

  - Critical F-value for columns: $F(0.05,2,4)\approx6.94$

-

- For rows (water temperature): The calculated F-statistic $F_R \approx 2.27$ is less than the critical F-value 6.94, so we fail to reject the null hypothesis. There is no significant difference in whiteness readings due to water temperature.
- For columns (detergents): The calculated F-statistic $F_C \approx 9.32$ is greater than the critical F-value 6.94, so we reject the null hypothesis. There is a significant difference in whiteness readings due to the type of detergent used.