WEEKTASK-3

Date:26.07.2025 - 27.07.2025
Name:Nancy M

1.Create a class BankAccount in Python with private attributes __accountno,__name, __balance.
Add
parameterized constructor
methods:
deposit(amount)
withdraw(amount)
set_accountno
get_accountno
set_name
get_name
get_balance()
set_balance()

```
class BankAccount:
    def __init__(self, accountno, name, balance):
        self.__accountno = accountno
        self.__name = name
        self.__balance = balance

    def deposit(self, amount):
        if amount > 0:
            self.__balance += amount

    def withdraw(self, amount):
        if 0 < amount <= self.__balance:
            self.__balance -= amount

    def set_accountno(self, accountno):
        self.__accountno = accountno

    def get_accountno(self):
        return self.__accountno

    def set_name(self, name):
        self.__name = name

    def get_name(self):
```

```
        return self.__name

    def set_balance(self, balance):
        self.__balance = balance

    def get_balance(self):
        return self.__balance
```

Output:
python bankacc.py
none

2.How will you define a static method in Python?Explore and give an example.

```
class MathUtils:
    @staticmethod
    def add(x, y):
        return x + y


result = MathUtils.add(5, 3)
print(result)
```

Output:
PS D:\Python> python static.py
8

3.Give examples for dunder methods in Python other than __str__ and __init__ .

```
class Ball:
    def __init__(self, radius):
        self.radius = radius

    def __mul__(self, other):
        return Ball(self.radius * other.radius)

    def __gt__(self, other):
        return self.radius > other.radius

    def __repr__(self):
        return f"Ball(radius={self.radius})"

b1 = Ball(3)
b2 = Ball(4)
```

```
b3 = b1 * b2

print(b3)
print(b1 > b2)
print(b2 > b1)
```

Output:
```
PS D:\Python> python dunder.py
Ball(radius=12)
False
True
```

4. Explore some supervised and unsupervised models in ML.

Machine Learning Models Overview

Machine learning models fall into two main categories: supervised and unsupervised learning, based on the data and objective.

Supervised Learning

Trained on labeled data to predict outputs for new inputs.

- Classification (predicts categories):
    - *Logistic Regression*: Binary classification.
    - *SVM*: Finds optimal class-separating hyperplane.
    - *Decision Trees / Random Forests*: Tree-based decision models.
    - *KNN*: Classifies based on nearest neighbors.
- Regression (predicts continuous values):
    - *Linear & Polynomial Regression*: Model linear or nonlinear trends.
    - *Ridge & Lasso*: Regularized models to avoid overfitting.

Unsupervised Learning

Works on unlabeled data to uncover patterns or structure.

- Clustering (groups similar data):
    - *K-Means*, *Hierarchical*, *DBSCAN*.
- Dimensionality Reduction (simplifies data):
    - *PCA*, *t-SNE*.
- Association Rule Mining (finds variable relationships):
    - *Apriori Algorithm*.

5.Implement Stack with class in Python.

```
class Stack:
    def __init__(self):
        self.items = []

    def push(self, item):
        self.items.append(item)

    def pop(self):
        if not self.is_empty():
            return self.items.pop()
        return None

    def peek(self):
        if not self.is_empty():
            return self.items[-1]
        return None

    def is_empty(self):
        return len(self.items) == 0

    def size(self):
        return len(self.items)


s = Stack()
s.push(10)
s.push(20)
```

```
print(s.pop())
print(s.peek())
```

Output:

```
PS D:\Python> python stack.py
20
10
```