Weekly Report: MD. Ariful Islam Shakil

Week-05 (April 21 – April 25)

1. **Machine Learning specialization** (Course 03: Unsupervised Learning, Recommenders, Reinforcement Learning)
    - Studied foundational reinforcement learning concepts, including the definitions of return, state, action, and policy.
    - Gained an understanding of the Bellman equations and their role in reinforcement learning frameworks.
    - Explored the concept of the state-action value function and how it supports decision-making processes in dynamic environments.
    - Learned to work with continuous state spaces and the challenges they present.
    - Successfully built and implemented a **Deep Q-Learning Network (DQN)** as a part of practical assignments.
    - **Achievement:** Successfully completed the "Unsupervised Learning, Recommenders, Reinforcement Learning" course and received a **course certificate**.
2. **Mathematics for Machine Learning: Multivariate Calculus**
    - Week – 03:
        - Applied the multivariate chain rule to differentiate complex nested functions.
        - Studied the structure and operational mechanism of neural networks.
        - Applied multivariate calculus tools to map the relationship between network parameters and network outputs.
        - Implemented the **Backpropagation algorithm** on a small-scale neural network to better understand the learning process.
    - Week – 04:
        - Recognized and analyzed power series approximations of functions.

- Investigated the behavior of power series for ill-behaved (non-smooth) functions.
- Explored the concept and practical applications of linearization for function approximation.
- Selected appropriate methods for multivariate function approximation, depending on the function behavior and approximation needs.

3. OOP Test Case Writing

I have written unit test case for OOP documentation using pytest. This test case includes:

- **Classes and Objects**: Initialization, attributes, and method behaviors.
- **Inheritance**: Proper extension and overriding of base class functionality.
- **Polymorphism**: Dynamic method binding and function overloading behaviors.
- **Abstraction**: Validation of abstract base classes and method implementations.
- **Encapsulation**: Ensured proper data hiding and access control.
- **Decorators**: Tested functionality of decorators to modify class methods.
- **Properties**: Verified the correctness of getter and setter methods using Python's property decorators.

4. Fast API:

- Practiced implementing **CRUD operations** (Create, Read, Update, Delete) using **FastAPI** framework.
- Integrated **DynamoDB** as the backend database for storing and managing records.
- Worked on setting up endpoints for data creation, retrieval, updating, and deletion.

5. Leet Code Problem Solving

- Some problem based on Tree Data Structure.