MODULARISATION OF CODE

WITH TECHNICAL ARCHITECT'S PERSPECTIVE

INTRODUCTION

PURPOSE: THIS PRESENTATION AIMS TO SHOWCASE THE PROCESS OF MODULARIZING CODE FOR MACHINE LEARNING MODELS AND THE PROGRESS WE'VE MADE IN ACHIEVING THIS GOAL.

MODULARIZATION IS A CRUCIAL STEP IN SOFTWARE DEVELOPMENT THAT IMPROVES CODE READABILITY, MAINTAINABILITY, AND REUSABILITY.

THROUGHOUT THIS PRESENTATION, WE WILL DISCUSS THE APPROACH TAKEN, THE BENEFITS OF MODULARIZATION, AND THE RESULTS ACHIEVED. LET'S GET STARTED!

MODULARISATION

01

Improved Readability:

- Modularization organizes code into logical components, making it easier to understand and maintain.

02

Enhanced Maintainability:

- Modular code is easier to debug, update, and extend, leading to reduced maintenance efforts.

03

Increased Reusability:

Modular components can be reused across different projects or within the same project, promoting code efficiency.

MODULARIZATION IS THE PROCESS OF BREAKING DOWN A COMPLEX SYSTEM INTO SMALLER, MANAGEABLE MODULES OR COMPONENTS.

FOR MACHINE LEARNING MODELS, MODULARIZATION IS PARTICULARLY BENEFICIAL AS IT ALLOWS FOR BETTER ORGANIZATION OF DATA PREPROCESSING, MODEL BUILDING, TRAINING, EVALUATION, AND DEPLOYMENT PROCESSES.

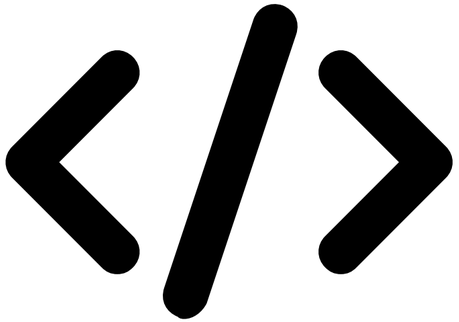
Modularization Approach

1. Identifying Components: We began by identifying distinct components or functionalities within our machine learning pipeline, such as data preprocessing, model training, evaluation, and deployment.

2. Creating Modules: Each identified component was encapsulated into separate modules, allowing for clear separation of concerns and improved code organization.

3. Defining Interfaces: We defined clear interfaces between modules, specifying how they interact and communicate with each other. This promotes modularity and facilitates integration with other parts of the system.

4. Implementing Abstraction: We employed abstraction techniques to hide implementation details and expose only essential functionalities through well-defined interfaces. This enhances code flexibility and reduces dependencies.



5. Testing and Validation: Comprehensive testing and validation were conducted at each stage of modularization to ensure that individual modules function correctly and integrate seamlessly within the overall system.

Modularized Code

- This slide showcases snippets of the modularized code for each component/module, demonstrating how the code is organized and structured.

Functionality of Each Module

01

Data Preprocessing Module:

- Describes how this module handles data cleaning, transformation, and feature engineering tasks to prepare the data for model training.

02

Model Training Module:

Explains the process of model selection, training, and optimization within this module, highlighting the algorithms and techniques used.

03

Evaluation Module:

Discusses how this module evaluates the performance of trained models using various metrics such as accuracy, precision, recall.

04

Deployment Module:

If a deployment module exists, mention its functionality in deploying trained models to production environments or integrating them into applications.

CONCLUSION

- We discussed the importance of modularization in improving code readability, maintainability, and reusability, which are crucial aspects of machine learning development.
- By modularizing our code, we have organized our machine learning pipeline into distinct components/modules, facilitating easier understanding, maintenance, and collaboration.
- Modularization enables us to effectively manage the complexity of machine learning projects, streamline development workflows, and adapt to changing requirements.
- Moving forward, we will continue to refine and enhance our modularized codebase, leveraging the benefits it brings to our development process.

THANK YOU