**Problem Statement**

The project addresses the challenge of developing an efficient movie recommendation system using PySpark and the ALS collaborative filtering algorithm.

The primary problem is predicting user ratings for movies based on historical data, providing users with personalised recommendations for an enhanced viewing experience.

**Motivation for the Problem**

The motivation stems from the increasing demand for personalised content recommendations in the digital entertainment landscape.

A well-designed recommendation system not only improves user satisfaction but also contributes to user engagement and loyalty.

The project aims to explore how big data technologies can be harnessed to address this demand and deliver tailored movie suggestions.

**Scope and Limitations**

Scope

* The project focuses on implementing a movie recommendation system using PySpark and ALS.
* The system aims to handle moderately sized datasets, providing scalable solutions for recommendation tasks.
* Evaluation metrics, specifically Mean Squared Error (MSE), are used to assess the accuracy of the recommendation model.

Limitations

* The project may face scalability challenges with extremely large datasets due to resource constraints.
* The recommendation system relies on historical user ratings and does not consider real-time user behaviour.
* The scope does not include advanced features such as user demographics or real-time feedback incorporation.

**Dataset Description**

The dataset comprises two main files, "train.dat" and "test.dat."

The training dataset contains historical user-movie ratings, while the testing dataset includes user-movie pairs for which ratings need to be predicted.

Each record in the dataset consists of a user ID, a movie ID, and a numerical rating.

**Design of the Solution**

Data Preprocessing

Raw data is processed to extract user ID, movie ID, and rating information.

The dataset is transformed into PySpark DataFrames for efficient processing.

Model Training

An ALS collaborative filtering model is configured and trained using the training dataset.

The model's hyperparameters, such as regularisation and iterations, are tunable for optimization.

Prediction and Evaluation

The trained model is used to predict ratings for the test dataset.

The Mean Squared Error (MSE) is calculated to assess the model's predictive accuracy.

**Modules Split-up**

Data Loading and Preprocessing

Loading raw data into RDDs.

Applying data preprocessing functions for transformation.

Data Transformation to DataFrames

Creating PySpark DataFrames from processed data.

Splitting data into training and validation sets.

Model Training

Implementing the ALS model training function.

Fine-tuning hyperparameters for optimal performance.

Prediction and Evaluation

Utilising the trained model for predictions.

Evaluating model performance using MSE.

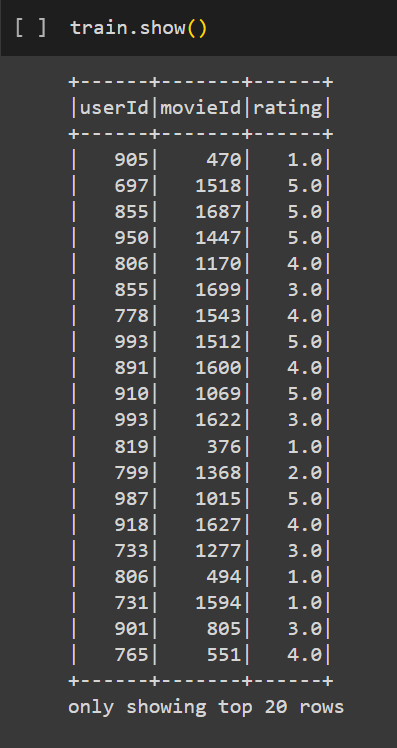
**Implementation**

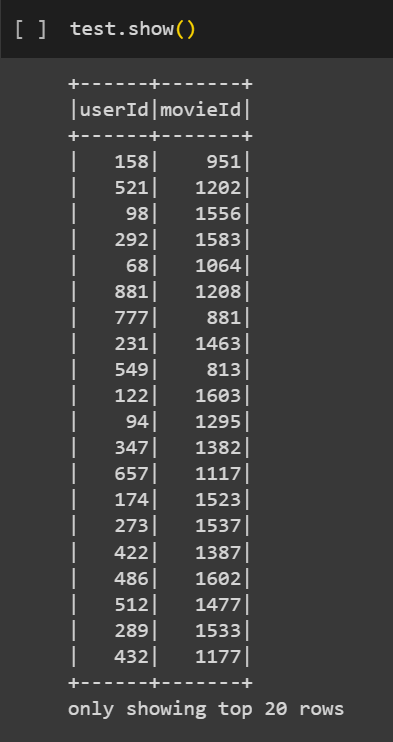
The project implementation involves coding each module, ensuring proper integration, and validating the recommendation system's functionality.

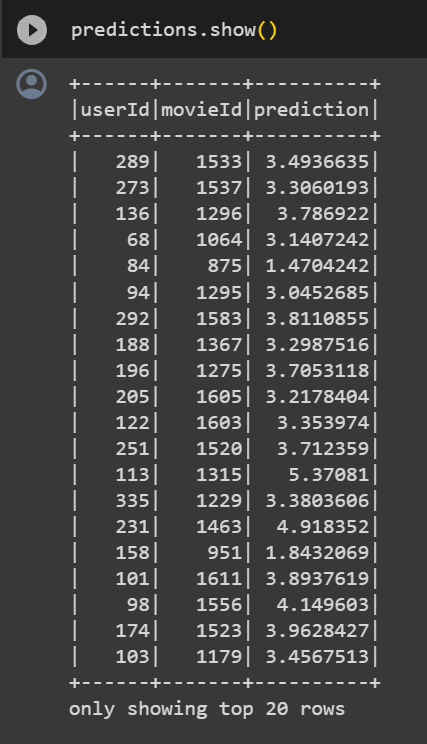
It enhances proficiency in PySpark, a distributed data processing framework for big data applications.

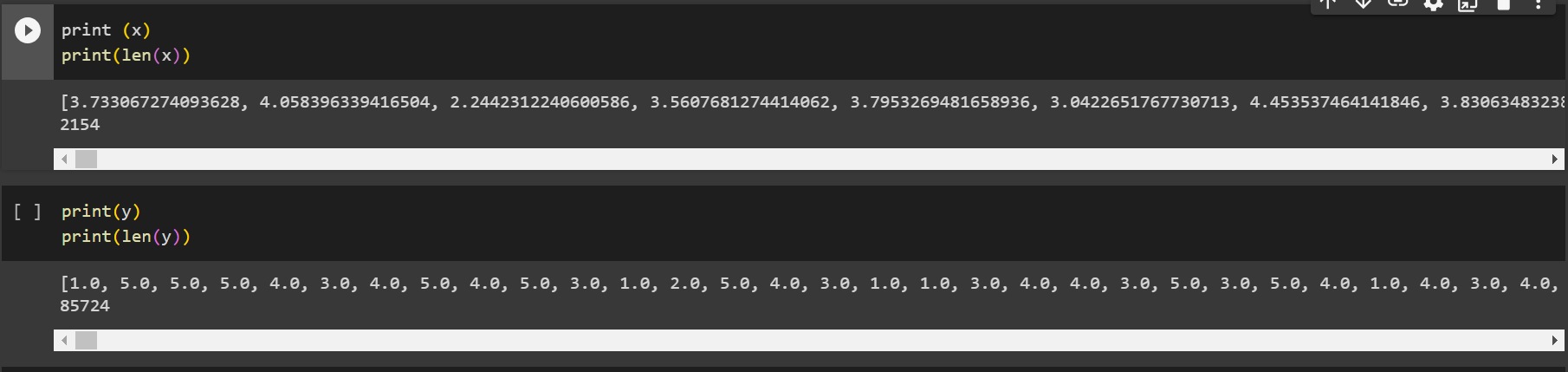
It includes setting up a Spark session, working with Resilient Distributed Datasets (RDDs), and leveraging PySpark's capabilities for efficient data processing.

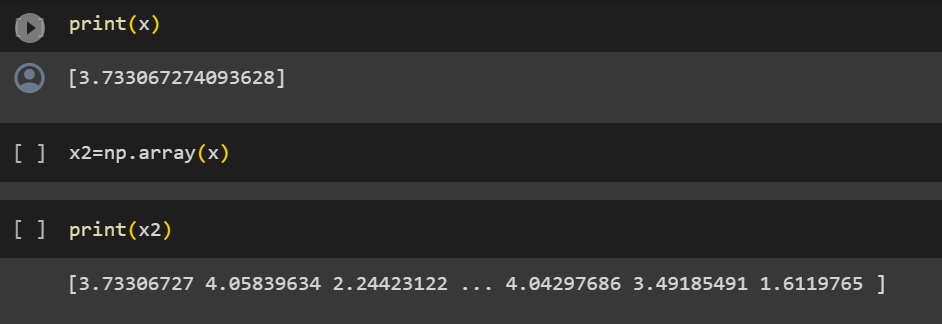
**Output Screenshots**

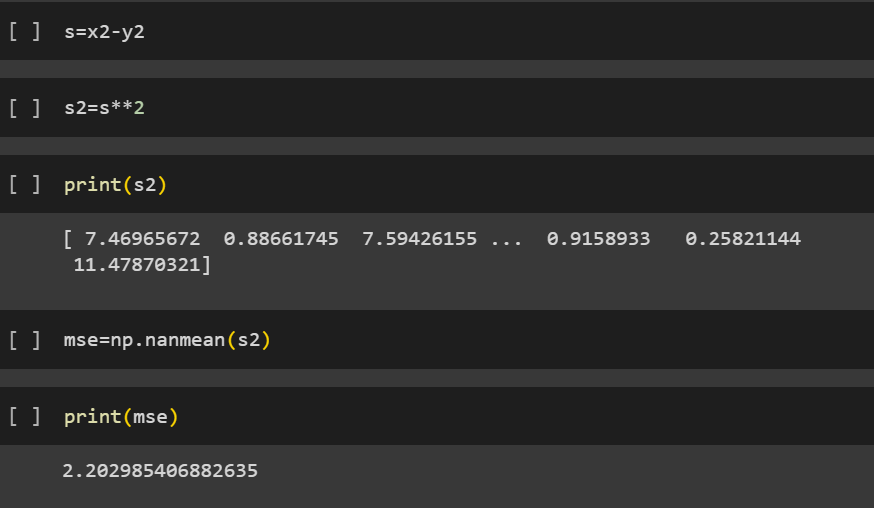












**Technologies Used:**

* PySpark: Leveraged for distributed data processing and machine learning.
* ALS (Alternating Least Squares): Collaborative filtering algorithm for recommendation tasks.
* RDDs (Resilient Distributed Datasets): The foundational data structure in PySpark for distributed data processing.
* Python: The primary programming language for implementing the recommendation system.

**Inference and Future Extension**

The successful implementation of the movie recommendation system demonstrates the potential of integrating PySpark, ALS, and RDDs to address personalised content delivery.

Future extensions may include:

* Integration of additional user features for improved recommendations.
* Exploration of real-time user behaviour data for dynamic recommendations.
* Scaling the system to handle larger datasets and increasing user bases.