REPORT ON

Data Analysis and Machine Learning Report on Anime Dataset using Pyspark.

Course: 2024F-T3 BDM 3603 - Big Data Framework 01 (DSMM Group 1)

Professor's Name: Ishant Gupta

Due Date: 7th November 2024

Submitted By:

Anmol Singh Paman

c0908043

Big Data Analytics

Introduction

The anime industry has grown substantially in recent years, with a diverse audience and a vast amount of content. Understanding the factors that contribute to anime ratings and popularity can be useful for both enthusiasts and industry professionals. This report explores an anime dataset and aims to identify patterns within the data, as well as build a predictive model to classify anime ratings based on various features. The analysis leverages PySpark for scalable data processing and machine learning.

1. Dataset Description

The dataset includes information on various anime titles, with features such as genre, rating, number of episodes, and release year. Each record represents a specific anime title with attributes that contribute to its classification. The main objective of this dataset is to support predictive analysis for classification, particularly in predicting the rating category of an anime title based on its other attributes.

Key Columns:

- **Genre**: The category or categories an anime falls under, such as action, drama, or comedy.
- **Rating**: The average user rating, serving as the target variable for classification.
- **Number of Episodes**: The total episodes in the series, which might correlate with popularity or genre.
- **Release Year**: The year the anime was first aired, used for time-based trends.

The animedataset.csv file was loaded into PySpark for distributed data processing, partitioned into four parts for improved load balancing and performance. The read.csv () function was used allowing automatic detection of column names and data types for a seamless data-loading process.

Displaying the Dataset Schema and Sample Rows: The dataset's schema was displayed to understand its structure, showing columns like username, anime_id, and genre. A preview of the first 10 rows provided insights into user ratings and anime details.

Counting Total Rows in the Dataset: The total number of rows in the dataset was counted using the count() method, revealing 35,305,695 entries. This large dataset highlights the need for efficient processing with PySpark.

2. Data Cleaning and Preprocessing

To ensure data quality and optimal performance in the machine learning model, several cleaning and preprocessing steps were undertaken using PySpark, including:

- **Handling Missing Values**: To resolve missing values in the "rank" column, the na.fill() method was used to replace any null values with 0. This guarantees that there are no missing entries in the "rank" column and that the dataset is full. The modifications are then shown by displaying the cleaned DataFrame.

- **De-duplication**: The dropDuplicates() function was used to eliminate duplicate rows from the dataset. This removes any duplication in the data and guarantees that every entry is distinct. After that, the duplicate-free DataFrame is shown to validate the modifications.
- **Data Type Conversion**: The cast("int") method was used to change the data type of the "rank" column from its original kind to an integer. This guarantees that the data in the column is formatted correctly for subsequent analysis. After that, the revised schema is shown to validate the modification.
- **Filtering Invalid Rows**: A condition was applied to the "my_score" column to filter out invalid rows. Only rows with an inclusive "my_score" value between 1 and 10 were kept. This guarantees that the dataset has accurate ratings, and the result is shown by displaying the filtered data.
- **Data Normalization**: This procedure involved first converting the "score" column to a float data type. Then, for demonstrative reasons, 0 was used to fill in the missing values in the "rank" column. Min-max scaling was used to normalise the "score" column, converting the scores into a range of 0 to 1. The modified data was shown after the resulting normalised scores were appended as a new column called normalized_score.

PySpark Transformations:

Common transformations applied included:

- filter(): Filtered rows based on specific conditions.
- drop(): Dropped unnecessary columns.
- fillna(): Filled missing values.
- withColumn() and cast(): Created or converted columns as needed.

3. Data Analysis Using Spark SQL

The cleaned DataFrame was registered as a temporary view in Spark SQL, enabling SQL-based exploration. Several queries provided insight into patterns within the dataset.

Key Analyses:

1. **Aggregation**: Summary statistics from the anime_data view were computed using SQL aggregation techniques. Both the "score" and "my_score" columns' average (AVG) and standard deviation (STDDEV) were calculated. After that, the summary data were shown for examination.

2. **Grouping by Genre**: SQL was used to group the data according to the "gender" column. The average "my_score" (AVG (my_score)) and the total number of users (COUNT(*)) were computed for each gender group. After that, the results were shown, including information about how the scores were distributed by gender.

3. **Joins**: A fictitious user_info DataFrame with columns for user_id and age was made. Next, an inner join was used to join user_info on the user_id column with the anime_data DataFrame. The resultant DataFrame, which combined user age and anime data, was shown for additional examination.

4. **Time-based analysis**: Since the dataset does not contain any time-related columns, performing time-based analysis, such as trends over time or time-series forecasting, is not possible in this case.

Insights from Analysis:

1. **Insight on Distribution by Age Group**: All age groups have a comparatively high average normalised score, with users between the ages of 27 and 30 having the highest average (0.79). The group of people who are 25 years old has the most users (801 users), while the group that is 27 years old has the fewest users (199 users).

2. **Insight on User Scores by Gender**: There are significantly more male users (25,386,353) than female users (9,678,908). On average, male users have a higher score (4.73) compared to female users (4.24).

3. **Official Scores vs. User Ratings:** The official scores are generally higher than user ratings, with an average of 7.53 compared to 4.59. Additionally, user ratings exhibit more variability, as indicated by a standard deviation of 3.91, compared to 0.73 for the official scores.

4. Machine Learning Model

Based on the dataset structure and the insights, a classification model was chosen to predict the rating category of an anime. The goal was to predict whether an anime would fall into high or low rating categories based on its features.

Model Development Process:

Data Splitting: The data was split into training and testing sets, with 80% of the data used for training and 20% for testing.

Pipeline Construction:

- **Feature Engineering**: The dataset underwent a number of modifications in this step: Users with a "my_score" of seven or more received a value of one in a new column called label, while others received a value of zero.

  Numerical features such as "score," "scored_by," and "popularity" were chosen for additional examination.

  The category columns "gender" and "type" were transformed into numerical indices (gender_index and type_index) using the StringIndexer. With the handleInvalid="keep" option making sure that any incorrect values are handled correctly, this transformation aids in getting categorical data ready for machine learning models.

- **VectorAssembler**: Several input columns, including gender_index, type_index, score, scored_by, and popularity, were combined into a single vector column called "features" using the VectorAssembler. In machine learning workflows, this transformation is frequently used to combine several feature columns into a single one that may be fed into machine learning models. Any rows with invalid values in the chosen columns are guaranteed to be skipped during the transformation if the handleInvalid="skip" option is used.

- **Model Selection**: An 80-20 ratio was used to divide the dataset into training and test sets. Using the chosen features, a RandomForestClassifier was initialised to predict the label. The classifier was placed in a pipeline that included preprocessing stages (indexing and feature building). To ensure model compatibility, the score column was set to double. Both training and testing were conducted using a 10% sample from each set. Predictions were then made on the test set after the model was trained on the sampled training data.

- **Evaluation Metrics**: The code first displays a sample of predictions, comparing the actual user scores (my_score) with the model's predictions. It then evaluates the model's performance using two metrics: **accuracy** (60%), indicating the proportion of correct predictions, and **F1-score** (0.60), which balances precision and recall.

5. Model Tuning and Evaluation

Hyperparameter tuning was conducted using cross-validation to optimize model performance

Preprocessing Categorical Data: To enable its use in machine learning models, the categorical gender column is transformed into a numeric index (genderIndex) using StringIndexer.

Feature Engineering: Most machine learning models in PySpark require a single vector, which is created by the VectorAssembler by combining the chosen characteristics (anime_id, user_id, genderIndex, and score).

Model Initialization: The combined features vector is used to forecast the target variable my_score in a Linear Regression model that has been initialised. In order to provide predictions, the model will learn how the user score and the input features relate to one another.

The process further involves setting up a range of options to fine-tune the model, specifically focusing on adjusting regularization parameters to avoid overfitting and control the complexity of the model.

Cross-Validation: CrossValidator is configured to run the linear regression model (lr) through three rounds of cross-validation. It adjusts the hyperparameters specified in the paramGrid and uses Root Mean Squared Error (RMSE) to assess the model's performance. For quicker execution, parallelism is set to two.

Final Model Evaluation: To assess the model's performance, the **Root Mean Squared Error (RMSE)** metric is used, which measures the accuracy of predictions by calculating the difference between predicted and actual scores. In this case, the RMSE value is **5.65**, indicating the model's average error in predicting the scores.

Model Performance

Based on a predetermined threshold, the projected and actual scores are converted into binary classifications. The F1 score, which strikes a compromise between precision and recall, and accuracy, which gauges the percentage of accurate predictions, are then employed as two important evaluation metrics. These metrics aid in evaluating the model's overall effectiveness and accuracy in data classification. The findings shed light on how well the model differentiates between various outcomes**.**

- **Accuracy**: Approximately 54.47% of the model's predictions were correct, indicating that it successfully predicted the correct class for just over half of the instances.
- **F1-Score**: The F1 score is a harmonic mean of precision and recall. A score of 0.55 suggests that the model's ability to balance precision (correct positive predictions) and recall (correctly identifying all actual positives) is moderate, but there's still room for improvement in both aspects.

Conclusion

The analysis provided valuable insights into the characteristics of high-rated anime and the factors that may contribute to better user ratings. The machine learning model developed successfully predicts anime rating categories based on features such as genre, release year, and episode count, providing a useful tool for stakeholders in the anime industry. Future work could explore additional feature engineering or use ensemble methods for potentially improved performance. This analysis illustrates how data-driven approaches can uncover trends and provide actionable predictions in niche entertainment domains.