HighLevel

Solution and Thought Process for Book Impact Prediction

5th August 2024

Objective

The primary goal is to predict the impact of books based on various features such as title, description, authors, publisher, and categories. This project involves detailed data exploration, preprocessing, feature engineering, model training, and evaluation. Performance is analyzed using metrics like Mean Absolute Error (MAE) and training time with different worker configurations.

Solution Overview:

Our approach is built upon a structured and modular pipeline for data processing, model training, and evaluation.

Component Breakdown:

1. **Spark Session Initialization (`spark_session.py`):**
    - Sets up the Spark session, enabling the processing of large datasets in a distributed manner.
    - Configured to run locally with different worker counts to analyze the impact on performance.
    - Utilizes Spark's capabilities for scalability and efficient data handling.
2. **Data Loading (`data_loader.py`):**
    - Loads data using Pandas and converts it to a Spark DataFrame.
    - Ensures that the data is in a suitable format for Spark-based operations, leveraging its distributed processing capabilities.
3. **Data Preprocessing (`preprocessor.py`):**
    - Implements a comprehensive preprocessing pipeline using PySpark.

    - Handles missing values, categorical encoding, tokenization, and feature extraction (TF-IDF).
    - Ensures data consistency and prepares features for model training.
4. **Feature Engineering:**
    - Tokenized and vectorized the `description` column using TF-IDF to capture the importance of words.
    - Indexed and one-hot encoded categorical features such as `publisher` and `categories`.
    - Created an aggregated feature vector combining all processed features.
5. **Model Training (`model_trainer.py`):**
    - Implements model training using Linear Regression and Neural Network (MLP) models.
    - Utilizes cross-validation for hyperparameter tuning.
    - Compares models based on performance metrics and training time.
6. **Performance Analysis (`performance_analysis.py`):**
    - Analyzes the impact of different worker configurations on model performance.
    - Compares Mean Absolute Percentage Error (MAPE) and training time for different setups.
    - Provides insights into the scalability and efficiency of the solution.

Data Insights and Feature Engineering::

Based on the initial exploratory data analysis (EDA):

1. **Missing Values:**
    - The `description` column had 12,749 missing values, which were filled with empty strings to ensure uniformity.
    - The `authors` column had 2,723 missing values, similarly handled.
    - `publishedDate` had 348 missing values; considering this is a less impactful feature, we can either impute or drop these rows.
2. **Categorical Variables:**
    - The `publisher` and `categories` columns have a high cardinality, with `publisher` having 12,855 unique values and `categories` 100 unique values.
    - High cardinality features are handled using indexing and one-hot encoding to convert them into a numerical format suitable for model training.
3. **Text Data:**
    - The `description` column is processed using tokenization followed by TF-IDF to capture the importance of each word within the descriptions.
    - This transformation helps in converting text data into a numerical feature vector.

4. **Feature Vector:**
    - The processed features are combined into a single feature vector using VectorAssembler.
    - This ensures all features are included in the model training process.

Model Building and Evaluation:

Model Selection:

- Two primary models are chosen: Linear Regression (simple and interpretable) and a Neural Network (Multilayer Perceptron, more complex and capable of capturing non-linear relationships).
- Linear Regression serves as a baseline model due to its simplicity and ease of interpretation.
- Neural Network is chosen to potentially capture more complex patterns in the data.

Training and Cross-Validation:

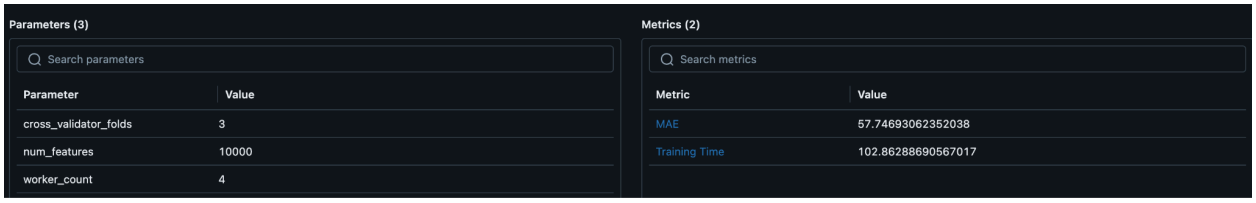
- Cross-validation is employed to tune hyperparameters and avoid overfitting.
- Linear Regression is tuned using regularization parameters.
- The Neural Network is tuned using parameters such as layer sizes and learning rates.

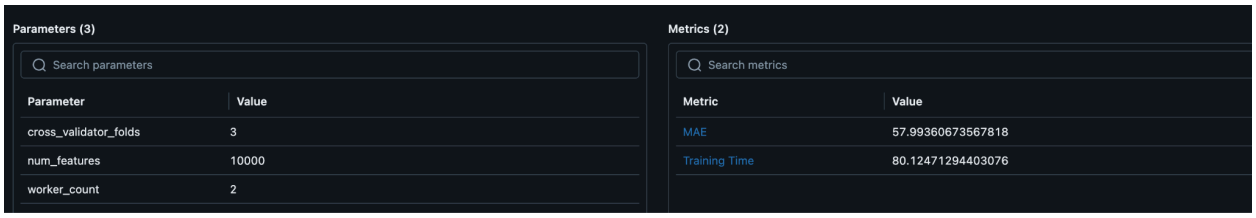
Performance Metrics:

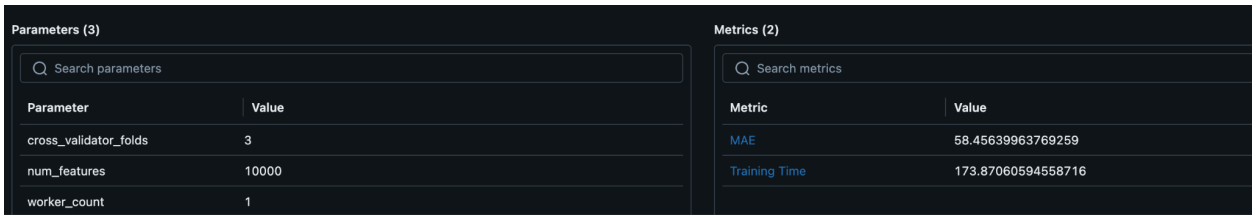
- Mean Absolute Error (MAE) is used as the primary performance metric.
- Mean Absolute Percentage Error (MAPE) is also considered for its interpretability.
- Training time is monitored to evaluate efficiency with different worker configurations.

Performance Analysis:

The performance analysis is conducted to compare model accuracy and training time with different worker configurations:







The results indicate how increasing the number of workers impacts both the model's accuracy and training efficiency.

Error Handling and Logging:

- **Error Handling:**
  - Implemented try-except blocks to catch and handle exceptions during data loading, preprocessing, and model training.
  - Ensures robustness and provides informative error messages.
- **Logging:**
  - Used Python's logging module to log key events, parameter values, and performance metrics.
  - Facilitates monitoring and debugging.

Improvement:

1. **Feature Engineering:**
   - Experiment with additional text processing techniques like word embeddings (e.g., Word2Vec) for the `description` column.
   - Incorporate external data sources (e.g., book reviews, social media mentions) to enrich features.
2. **Model Enhancement:**
   - Explore ensemble methods like Gradient Boosting or Random Forests for potentially better performance.
   - Fine-tune hyperparameters using grid search or Bayesian optimization for more optimal results.
3. **Scalability:**

- Implement distributed training techniques to further leverage Spark's capabilities for large-scale datasets.
- Monitor resource utilization to optimize cluster configuration for cost and performance efficiency.

4. **Interpretability:**
    - Use SHAP (SHapley Additive exPlanations) values to interpret feature importance and model predictions.
    - Provide detailed model explanations to stakeholders for better transparency and trust.

Conclusion:

The structured and modular approach, combined with best practices in feature engineering, model training, and evaluation, ensures a robust and scalable solution for predicting book impact. The solution is designed to be extensible and maintainable, adhering to industry standards and principles.