Workshop 3: Machine Learning and Data Streaming - Documentation

Author: Isabella Pérez Caviedes - 2230603

Date: 21/05/2025

Project: Workshop 3: Machine Learning and Data Streaming

Repository: https://github.com/isabellaperezcav/Workshop_03-Machine-learning-and-Data-streaming

Version: 1.0

Table of Contents

---

1. Introduction

This documentation describes the solution for Workshop 3: Machine Learning and Data Streaming. The project focuses on developing a regression model to predict happiness scores of various countries using data from five CSV files. The solution includes exploratory data analysis (EDA), extract-transform-load (ETL) processes, training a regression model, data streaming using Apache Kafka, prediction, storing results in a PostgreSQL database, and model evaluation. Technologies used include Python, Jupyter Notebook, Scikit-learn, Apache Kafka, PostgreSQL, and Docker. The main goal is to build a scalable and accurate predictive system following industry-standard practices.

---

2. Problem Statement

The task is to design and implement a regression model capable of predicting happiness scores for different countries based on historical data provided in five CSV files, each representing a different year. The requirements are:

- Perform EDA and ETL to preprocess and extract features from the CSV files.
- Train a regression model using an 80-20 data split (80% for training, 20% for testing).
- Stream the transformed data using Apache Kafka.
- Use the trained model in a Kafka consumer to predict happiness scores and store both predictions and input features in a PostgreSQL database.
- Evaluate model performance using appropriate metrics on the test data.

It is important to note that datasets may vary between years, requiring careful feature comparison to ensure consistency.

---

3. Solution Approach

The solution is structured into the following key steps:

1. **EDA and ETL**: Analyze datasets, handle missing values, encode categorical variables, scale features, and select relevant features based on consistency and correlation.
2. **Model Training**: Split the preprocessed data into training and testing sets, evaluate multiple regression models, select the best-performing one, and serialize it for deployment.
3. **Data Streaming**: Use Apache Kafka to stream transformed data, with a producer sending data to a topic and a consumer receiving it for prediction.
4. **Prediction and Storage**: Load the trained model in the Kafka consumer, generate predictions, and store them along with input features in a PostgreSQL database.
5. **Model Evaluation**: Evaluate model performance using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the coefficient of determination (R^2).

---

4. Implementation Details

4.1 Exploratory Data Analysis (EDA) and ETL

The dataset consists of five CSV files with happiness data from various countries across different years. Due to potential inconsistencies in features between years, a detailed analysis was conducted.

- **Data Analysis**: Each CSV file was examined to identify missing values, data types, and distributions, in order to understand its structure and quality.
- **Feature Selection**: Features that were consistent across all years and highly correlated with happiness scores were selected to ensure robustness.
- **Preprocessing**: Missing values were imputed, categorical variables were one-hot encoded, and numerical features were scaled using standard scaling.

4.2 Model Training

- **Data Splitting**: The preprocessed data was split into training and test sets using an 80-20 ratio.
- **Model Evaluation**: Four regression models were tested: Ridge, Random Forest, XGBoost, and Lasso. Random Forest was chosen for its superior performance.
- **Model Serialization**: The trained Random Forest model was saved as `random_forest_model.pkl` using Python's `pickle` module.

4.3 Data Streaming

- **Kafka Configuration**: Apache Kafka facilitated data streaming, with a producer sending transformed data to the topic `happiness_topic`.
- **Producer**: The script `KafkaProducer.py` reads and transmits the preprocessed data.
- **Consumer**: The script `KafkaConsumer.py` subscribes to the topic and processes incoming data for prediction and stores the information in a PostgreSQL database.

4.4 Prediction and Storage

- **Model Loading**: The serialized model is loaded into the consumer using `pickle`.
- **Prediction**: Predictions for the streamed data are generated using the loaded model.
- **Database Storage**: Predictions and their corresponding input features are stored in a PostgreSQL database in the `predicciones` table.

4.5 Model Evaluation

- **Metrics**: The model's performance was evaluated using MAE, RMSE, and R^2.
- **Results**: The Random Forest model achieved a MAE of 0.008105, RMSE of 0.018131, and R^2 of 0.999730, reflecting high predictive accuracy.
- Cross-validation of the selected model was performed to assess its **generalization** capability *(to verify whether the model works well only on the current dataset or would also perform well on new, unseen data)*.

5. Results

The Random Forest model stood out as the top performer. Its evaluation metrics on the test data are presented below:

Model	MAE	RMSE	R^2
Random Forest	0.008105	0.018131	0.999730

The high R^2 value and low errors demonstrate the model's effectiveness in predicting happiness scores.

---

6. Conclusion

This project successfully integrates machine learning and data streaming to predict happiness scores. It includes comprehensive EDA, robust model training, efficient data transmission, accurate predictions, and reliable storage. The use of Apache Kafka and PostgreSQL ensures scalability and operational efficiency, fulfilling the workshop's objectives.

---

7. Future Work

Possible improvements include:

- Advanced feature engineering to uncover deeper relationships in the data.
- Hyperparameter tuning (e.g., grid search or random search) to further optimize model performance.

---

8. References

- Scikit-learn Documentation: https://scikit-learn.org/stable/documentation.html
- Apache Kafka Documentation: https://kafka.apache.org/documentation/
- PostgreSQL Documentation: https://www.postgresql.org/docs/
- Project Repository: https://github.com/isabellaperezcav/Workshop_03-Machine-learning-and-Data-streaming

---

9. Prerequisites

- Docker installed (https://docs.docker.com/get-docker/)
- Python 3 and required dependencies:

  ```
  bash
  CopiarEditar
  pip install -r requirements.txt
  ```

---

10. How to Run the Project

1. **Start Kafka and Zookeeper**:

   ```
   bash
   CopiarEditar
   docker-compose up
   ```

2. **Run the Kafka Producer**:

   ```
   bash
   CopiarEditar
   cd KafkaETL
   python KafkaProducer.py
   ```

3. **Run the Kafka Consumer**:

   ```
   bash
   CopiarEditar
   cd KafkaETL
   python KafkaConsumer.py
   ```

4. **View Results**:
    - Check results in PostgreSQL or in the terminal.
    - Explore `notebooks/eda_and_model_training.ipynb`.
    - Refer to `pdf/work03_ETL_isabellaperezcav.mp4`.

---

11. Notes

- Make sure Docker is running before launching Kafka.
- Check topic names in `producer.py` and `consumer.py` (default: `happiness_topic`).
- Use tools like pgAdmin4 to query the PostgreSQL database.