Report: Project Data Exploration and Analysis
-
Charles Meyer

In this project, we leverage the skills acquired in class to conduct data analysis using diverse technologies. Our objective is to explore a dataset of our choosing, define a compelling business objective, and employ a series of analytical techniques to achieve meaningful insights. This report outlines our approach, challenges encountered, and potential enhancements for future iterations.

This project aims to develop a dashboard for runners, providing various metrics, graphs, and statistics to help them manage their training and track performance over time. By leveraging big data technologies, we process running-related datasets to extract insights and present them in an interactive and user-friendly format. The project consists of two main components: a Scala-based data processing pipeline using Apache Spark for efficient data handling and a Python-based dashboard built with Dash for visualization and interaction.

The initial stages of the project involved preliminary data exploration and basic data cleaning using a Jupyter notebook. During this phase, we identified missing values in the country column, opting not to remove these rows due to completeness in other columns.

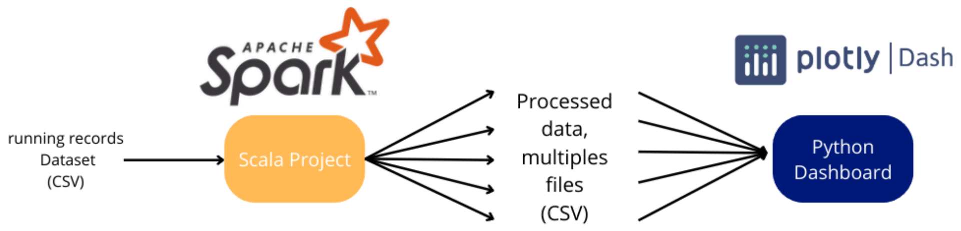
running records Dataset (CSV)
Scala Project
Processed data, multiples files (CSV)
Python Dashboard

Figure 1 : Architecture overview

The Scala component of our project is structured around several key classes: **SparkSessionBuilder** for initiating Spark sessions, **DataReader** for CSV file ingestion, **DataWriter** for output management, **DataProcessor** for core data manipulation functions, and **Main** for orchestrating these processes. We chose to forego Hadoop Distributed File System (HDFS) as our dataset sizes did not necessitate its complexity.

For data visualization and interaction, we employed the **Dash** library in Python, leveraging its familiarity and efficiency for creating our dashboard. The project was organized into multiple files including **run.py** for application startup, **graph.py** for graph creation before integrating them into the dashboard, **app/callback.py** for callback functions, **app/app.py** for page architecture, **app/dataHandler.py** for data management, and **app/assets/** for static resources.

While no specific results are presented due to the project's nature, certain dataset limitations were identified. These include the absence of detailed run-specific metrics, such as kilometer-by-kilometer pace analysis or comprehensive heart rate data, which hindered deeper analyses. Potential improvements identified include transitioning to a relational database like PostgreSQL or SQLite to reduce dashboard latency and enable backup systems. Furthermore, automating data processing upon new file reception is recommended to streamline operations.

Looking forward, integrating a machine learning model to forecast daily injury risks for runners stands as a promising future enhancement.