Manuale di Installazione e Utilizzo

Pipeline Scalabile per Analisi Comportamentale su GitHub

Federico Mondelli

Anno Accademico 2024/2025

Sommario

Questo documento guida l'utente nell'installazione dell'ambiente, nella configurazione e nell'esecuzione di tutti i moduli del sistema: *Ingestor* (Acquisizione), *Validator* (Verifica) e *Analyzer* (Elaborazione e Mining).

Indice

1 Requisiti di Sistema

Prima di procedere con l'installazione, assicurarsi che la macchina soddisfi i seguenti requisiti minimi:

- **Sistema Operativo:** Windows 10/11, macOS o Linux.
- **Python:** Versione **3.10** o superiore.
- **RAM:** Minimo 8 GB (Raccomandati 16 GB per dataset estesi).
- **Spazio su Disco:** Dipende dal range temporale scaricato (stimare circa 500 MB per ogni giorno di dati grezzi).
- **Connessione Internet:** Necessaria per il modulo *Ingestor* (download da GHArchive).

2 Installazione

2.1 Passo 1: Estrazione del Progetto

Scaricare ed estrarre l'archivio del progetto in una cartella locale (es. `github_process_mining`). Aprire un terminale (Prompt dei comandi o PowerShell su Windows, Bash su Linux/macOS) e posizionarsi nella cartella radice del progetto.

2.2 Passo 2: Creazione dell'Ambiente Virtuale

Si raccomanda vivamente l'uso di un ambiente virtuale per isolare le dipendenze.

Su Windows:

```
python -m venv venv
venv\Scripts\activate
```

Su macOS / Linux:

```
python3 -m venv venv
source venv/bin/activate
```

2.3 Passo 3: Installazione delle Librerie

Il progetto richiede librerie specifiche per l'elaborazione dati (Polars), il process mining (PM4Py) e la visualizzazione. Assicurarsi che il file `requirements.txt` sia presente nella root ed eseguire:

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

Contenuto di riferimento per `requirements.txt`:

```
requests>=2.28.0
polars>=0.19.0
pm4py>=2.7.0
pandas>=2.0.0
```

```
matplotlib>=3.7.0
seaborn>=0.12.0
scikit-learn>=1.3.0
scipy>=1.10.0
python-dotenv>=1.0.0
pytest>=7.0.0
```

3 Configurazione

Creare un file chiamato `.env` nella cartella radice del progetto. Questo file definirà i percorsi di salvataggio e le variabili d'ambiente.

Esempio di contenuto `.env`:

```
# Percorso dove verranno salvati i dati grezzi (Parquet)
DATASET_PATH=data/dataset_distillato

# Percorso dove verranno salvati i risultati dell'analisi (Grafici,
    CSV)
DATA_ANALYSIS=data/dataset_analyzed

# Range temporale per l'Analisi (Formato ISO 8601)
ANALYSIS_START_DATE="2024-01-01T00:00:00Z"
ANALYSIS_END_DATE="2024-01-05T23:59:59Z"
```

Nota: Le cartelle di destinazione verranno create automaticamente dal software se non esistono.

4 Modulo 1: Dataset Ingestor (Acquisizione)

Questo modulo scarica i dati da GitHub Archive e li converte in formato Parquet.

Comando base: `python -m src.dataset_ingestor.cli`

4.1 Opzioni di Avvio

4.1.1 1. Download Range Temporale (Batch)

Scarica e processa un intervallo continuo di date. Al termine, consolida automaticamente i dati in un file Parquet giornaliero.

```
python -m src.ingestor.presentation.cli --download 2025-01-01-0
   2025-01-01-23
```

- **Formato:** `YYYY-MM-DD-H` (ore 0-23, senza zero iniziale per le ore singole).
- **Esempio:** Scarica dal 1 Gennaio h 00:00 al 1 Gennaio h 23:00.

4.1.2 2. Download Ore Singole (Atomico)

Scarica specifiche ore. Utile per debug o per recuperare porzioni mancanti a causa di errori di rete.

```
python -m src.dataset_ingestor.cli --hours 2025-01-01-10
   2025-01-01-15
```

4.1.3 3. Visualizza Info Dataset

Mostra statistiche sui dati presenti in locale (dimensione, copertura).

```
python -m src.dataset_ingestor.cli --info
```

4.1.4 4. Reset Dataset

Cancella l'intero dataset scaricato e gli indici locali, permettendo di ripartire da zero. Da usare con cautela.

```
python -m src.dataset_ingestor.cli --reset
```

- **Azione:** Elimina ricorsivamente la cartella dei dati definita nel file `.env`.
- **Interazione:** Richiede una conferma manuale (s/n) da terminale prima di procedere all'eliminazione.

5 Modulo 2: Dataset Validator (Verifica)

Strumento di raccordo per verificare la coerenza temporale prima dell'analisi. Scansiona gli indici generati dall'Ingestor.

```
python src/tools/validate_dataset.py --path data/dataset_distillato
```

Attenzione: Assicurarsi che il path corrisponda a quello definito nel `.env`.

Output: Suggerisce il periodo contiguo più lungo (senza buchi) da copiare nelle variabili `ANALYSIS_START_DATE` e `ANALYSIS_END_DATE` del file `.env`.

6 Modulo 3: Dataset Analysis (Elaborazione)

Questo è il cuore del sistema. Esegue stratificazione, mining e confronto. I comandi vanno eseguiti **rigorosamente in questo ordine**.

Comando base: `python -m src.dataset_analysis.cli`

6.1 1. Preparazione Completa e Stratificazione

Carica i dati Parquet, calcola le metriche, definisce i quantili dinamici e assegna gli Archetipi.

```
python -m src.analyzer.infrastructure.cli full
```

Output: Genera il file master `repositories_stratified.parquet` e il report di distribuzione.

6.2 2. Process Discovery (Mining)

Estrae gli eventi per ogni Archetipo, applica i filtri semantici (es. rimozione bot) ed esegue l'algoritmo **Heuristics Miner** in modalità ibrida (aggregazione Polars + mining PM4Py).

```
python -m src.analyzer.infrastructure.cli process_discovery
```

Output: Salva i modelli di processo (`.pkl`), i log `.xes` e le immagini dei grafi (`.png`).

6.3 3. Confronto Strutturale

Carica i modelli generati e calcola le metriche di distanza (Jaccard, Frobenius, TVD) e complessità.

```
python -m src.analyzer.infrastructure.cli structural_comparison
```

Output: Genera Heatmap differenziali, Radar Chart, MDS Plot e tabelle CSV nella cartella `final_analysis`.

7 Esecuzione dei Test

Per verificare che tutti i componenti funzionino correttamente (Unit Test e Integration Test), eseguire:

```
pytest
```

Se l'installazione è corretta, il terminale mostrerà una serie di punti verdi indicanti il superamento dei test.

8 Workflow Riassuntivo

Per riprodurre l'esperimento completo da zero:

1. **Ingestor:** Scaricare un giorno di dati.
   `python -m src.ingestor.presentation.cli -download 2024-01-01-0 2024-01-01-23`
2. **Validator:** Controllare che il giorno sia completo.
   `python src/utils/validate_dataset.py -path data/dataset_distillato`
3. **Configurazione:** Aggiornare il file `.env` con le date corrette.
4. **Analyzer (Step 1):** Preparare e stratificare.
   `python -m src.analyzer.infrastructure.cli full`
5. **Analyzer (Step 2):** Generare i modelli di processo.
   `python -m src.analyzer.infrastructure.cli process_discovery`
6. **Analyzer (Step 3):** Eseguire il confronto strutturale.
   `python -m src.analyzer.infrastructure.cli structural_comparison`

I risultati finali (immagini PNG e file CSV) si troveranno nella cartella definita in `DATA_ANALYSIS`.