Report Finale: Analisi della Pipeline di Data Quality e Machine Learning

Introduzione

Questo report presenta i risultati dell'analisi del file `pipeline.py` rispetto alla documentazione contenuta nel PDF "Progetto_finale_Archi.pdf". L'obiettivo dell'analisi era verificare la coerenza e la correttezza dell'implementazione della pipeline rispetto alla documentazione.

Struttura della Pipeline

La pipeline implementata nel file `pipeline.py` è strutturata utilizzando il framework Luigi per la gestione delle dipendenze tra le diverse task. La struttura generale è coerente con quanto descritto nel PDF.

La pipeline è organizzata nelle seguenti componenti principali:

1. **Configurazione iniziale**:
2. Importazione delle librerie necessarie
3. Configurazione del logger
4. Definizione dei percorsi predefiniti tramite il file di configurazione Luigi
5. Definizione delle soglie per le misure di qualità dei dati
6. **Task di preprocessing dei dati**:
7. `DataPreprocessing` : Rimozione di valori mancanti e duplicati
8. `DataTransformation` : Encoding di variabili categoriche e rimozione di colonne non necessarie
9. `PCATask` : Riduzione della dimensionalità tramite PCA
10. `SplitDataset` : Suddivisione del dataset in set di training e test
11. **Task di modellazione Machine Learning**:
12. `SVMModel` : Addestramento di un modello Support Vector Machine
13. `DTCModel` : Addestramento di un modello Decision Tree Classifier
14. `PerformanceEval` : Valutazione delle performance dei modelli

15. **Task di verifica della qualità dei dati**:

16. `Completeness`: Verifica della completezza dei dati
17. `Consistency`: Verifica della consistenza dei dati
18. `Uniqueness`: Verifica dell'unicità dei dati

19. `Accuracy`: Verifica dell'accuratezza dei dati

20. **Task wrapper**:

21. `FullPipeline`: Task wrapper che esegue l'intera pipeline

Coerenza con la Documentazione

Nel complesso, l'implementazione della pipeline è coerente con quanto descritto nella documentazione. Le task sono implementate correttamente e seguono la struttura descritta nel PDF.

Punti di Forza

1. **Struttura modulare**: La pipeline è organizzata in task ben definite, ognuna con una responsabilità specifica. Questo rende il codice facile da mantenere e estendere.

2. **Gestione delle dipendenze**: L'uso del framework Luigi per gestire le dipendenze tra le task garantisce che le task vengano eseguite nell'ordine corretto e che le dipendenze siano soddisfatte.

3. **Logging dettagliato**: Il codice include un logging dettagliato che facilita il debug e il monitoraggio dell'esecuzione della pipeline.

4. **Verifica della qualità dei dati**: La pipeline include task specifiche per verificare la qualità dei dati secondo le quattro dimensioni principali (completezza, consistenza, unicità, accuratezza).

5. **Configurazione flessibile**: L'uso di parametri Luigi e di un file di configurazione rende la pipeline flessibile e configurabile.

Incongruenze e Problemi

1. **Riferimento a Neural Network**: Nel codice ci sono riferimenti a un modello Neural Network che non è completamente implementato. Questo potrebbe causare errori durante l'esecuzione della pipeline.

2. **Mancanza di implementazione delle funzioni di utilità**: Nel codice si fa riferimento a funzioni di utilità come `get_global_metrics` e `get_confidence_intervals` dal modulo `utils.evaluation`, ma l'implementazione di queste funzioni non è presente nel file `pipeline.py`.

3. **Mancanza di implementazione delle funzioni di data quality**: Nel codice si fa riferimento a funzioni di data quality come `completeness_test`, `consistency_test`, `uniqueness_test` e `accuracy_test` dal modulo `utils.data_quality`, ma l'implementazione di queste funzioni non è presente nel file `pipeline.py`.

Raccomandazioni

1. **Rimuovere i riferimenti al modello Neural Network**: Se il modello Neural Network non è necessario, rimuovere tutti i riferimenti ad esso dal codice. Se invece è necessario, implementare la task `NNModel` e aggiornare la task `PerformanceEval` per utilizzarla correttamente.

2. **Verificare l'implementazione delle funzioni di utilità**: Assicurarsi che i moduli `utils.evaluation` e `utils.data_quality` esistano e che le funzioni utilizzate nel codice siano implementate correttamente.

3. **Aggiungere test unitari**: Aggiungere test unitari per verificare il corretto funzionamento delle task e delle funzioni di utilità.

4. **Migliorare la documentazione del codice**: Aggiungere più commenti e documentazione al codice per facilitare la comprensione e la manutenzione.

5. **Considerare l'uso di tipi di dati più specifici**: Utilizzare tipi di dati più specifici per i parametri delle task, ad esempio utilizzando `luigi.PathParameter` invece di `luigi.Parameter` per i percorsi dei file.

Conclusione

L'implementazione della pipeline è generalmente coerente con la documentazione e ben strutturata. Le incongruenze identificate sono principalmente legate a riferimenti a un modello Neural Network che non è completamente implementato. Queste incongruenze possono essere facilmente risolte seguendo le raccomandazioni fornite.

La pipeline è un buon esempio di come utilizzare il framework Luigi per implementare un flusso di lavoro di data science, con task ben definite e dipendenze gestite in modo efficiente.