# PIPELINE ETL AUTOMATIZZATA

Team 01 - classe data

DEVELHOPE

CHRISTIAN BUONO

MANUELA RIZZUTO

GIANLUIGI PEDICINI

# MISSION E OBIETTIVI:

Realizzare una pipeline ETL per l’analisi di un dataset contenente informazioni sulle app del Playstore di Google
Comprendere le preferenze degli utenti e capire:

- Quali sono le app più scaricate
- Quali app generano più profitto
- cosa influisce positivamente o negativamente sulle recensioni

in che modo le aziende possono migliorare i loro prodotti o servizi in base a queste informazioni e rispondere alle esigenze dei clienti?

# STRUTTURA PROGETTO SU GITHUB

GitHub

- data
  - clean_data
  - raw_data
- graphs
- src
- dag.py
- main.py
- datasets_analysis.ipynb
- requirements.txt

# Processo ETL

## extract / trasform / load

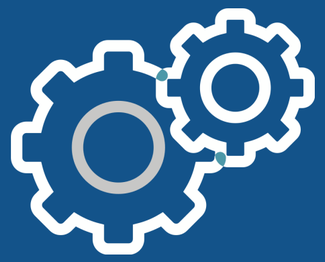
**estrazione/trasformazione/caricamento**, è il processo di raccolta dei dati da una sorgente e della loro successiva trasformazione, organizzazione e centralizzazione in una repository di destinazione(warehouse).

I dati sono di importanza fondamentale in tutti i processi aziendali; per poter disporre di dati di qualità, è necessario sottoporli a un processo ETL con l' obiettivo di ottenere dati puliti e accessibili da utilizzare per le attività di analisi aziendali.

# Tecnologie utilizzate

- Python
  - Linguaggio di programmazione
- Airflow
  - Strumento di orchestrazione
- Tableau
  - Dashboard
- GitHub
  - Versionamento e pubblicazione del progetto

# PIPELINE



Nel processo ETL delle app contenute nel playstore di Google intervengono
le classi  per svolgere attività come la pulizia dei dati, l'aggregazione, la normalizzazione, la gestione di errori e la creazione di nuove colonne o calcoli.

1 - DATA INGESTOR

2 - DATA CLEANER

3 - DATA EXPLORATOR

4 - DATA ANALYZER

5 - DATA VISUALIZER

# DATA INGESTOR

Gestisce l'acquisizione, la ricezione e l'elaborazione di dati da diverse fonti o formati. Il suo ruolo principale è quello di semplificare il processo di importazione, estrazione e manipolazione di dati

# DATA CLEANER

Esegue operazioni di pulizia e trasformazione dei dati. Il suo obiettivo principale è garantire che i dati in ingresso siano coerenti, privi di errori e pronti per l'analisi o l'elaborazione successiva(rimozione di dati duplicati; trattamento di valori mancanti sostituendoli con valori predefiniti; portare i dati a una forma coerente, ad esempio, normalizzando formati di data e ora; rimozione di caratteri speciali o spazi vuoti superflui).

# DATA EXPLORATOR

**Si occupa di esplorare ed analizzare i dati per ottenere informazioni significative da essi. Esegue filtraggio e selezione dei dati: seleziona porzioni specifiche dei dati o applica filtri per esaminare solo parti rilevanti del dataset.**

- **Informazioni numeriche**
- **Grafici di distribuzione**
- **Studio della correlazione**

Insights

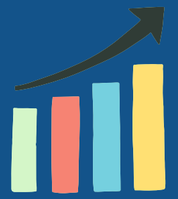
# DATA ANALYZER

Effettua la **Sentiment Analisys**, un' analisi automatizzate di un testo possibile grazie a degli algoritmi che assegnano un valore alle parole, questo può essere positivo (+1) se la parola è positiva a sua volta, o negativo (-1), se la parola è negativa.

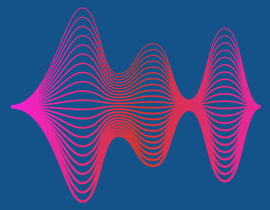
Utile per monitorare il comportamento dei clienti su un particolare prodotto.
Automatizza l'attività di report sulle preferenze del cliente.

AFINN

# DATA VISUALIZER



Questa classe è  utilizzata per rendere i dati più comprensibili e accessibili agli utenti, consentendo loro di analizzare e interpretare le informazioni in modo più efficace attraverso la creazione di grafici a barre, grafici a torta, grafici a dispersione, grafici a linee, istogramma, heatmap e altri tipi di grafici.



# DAG

## Directed Acyclic Graph

concetto centrale di Airflow, Questo modello è un grafo che non presenta cicli, ma percorsi sequenziali provenienti dallo stesso batch

raccoglie le attività organizzate, con dipendenze e relazioni, stabilisce l’ordine in cui devono essere eseguite e la frequenza con cui eseguirle

Un DAG è definita in un file Python, è costituita da operatori che descrivono il lavoro da svolgere tramite tasks e le relazioni che indicano l’ordine di esecuzione delle tasks

# PERCHE’ ABBIAMO UTILIZZATO L’ORGANIZZAZIONE MODULARE DEL CODICE?

1. Riusabilità
2. Separazione delle responsabilità
3. Testabilità
4. Collaborazione
5. Manutenibilità

1,813,303

0.09476

# GRAZIE!

CHRISTIAN BUONO

GIANLUIGI PEDICINI

MANUELA RIZZUTO