

BT4222 Project Source Code

Sentiment Analysis was built and tested with Python 3.10/3.11 on macOS due to the long processing time, which exceeds the maximum inactivity window on Google Colab.

Table of Contents

- Dataset Access
- Dataset Purpose and Content
- Repository Layout
- Running the Workflow
- Purpose and Content of Scripts
- Reproducing Results

Dataset Access

Raw CSVs are hosted outside the repository because of size limits. Download them from the shared drive and place them under `src/datasets` :

- [BT4222 Group 9 Datasets \(Google Drive\)](#)
- Keep filenames unchanged; the notebooks expect the naming convention shown in the repository layout.

Dataset Purpose and Content

- `games.csv` : Master catalog of Steam titles keyed by `gameid`, including title, developers, publishers, genres, supported languages, and release date.
- `prices.csv` : Regional price snapshots for each `gameid` across `usd`, `eur`, `gbp`, `jpy`, and `rub`, tracked by `date_acquired`.
- `purchased_games.csv` : Ownership table mapping each `playerid` to a serialized list of owned `gameid`s.
- `players.csv` : Player metadata with `playerid`, `country`, and account creation timestamp.
- `reviews.csv` : Raw review corpus with text plus engagement metadata (`helpful`, `funny`, `awards`) and `posted` date.
- `reviews_lang_detect.csv` : Reviews with detected `language` appended for multilingual filtering (same schema as `reviews.csv` + `language`).
- `english_reviews.csv` : Filtered subset of `reviews_lang_detect.csv` limited to reviews with `language == "en"`.
- `english_reviews_1k.csv` : 1,000-row sample from `english_reviews.csv` for faster experimentation.
- `sentiment_1k.csv` : The 1k English sample with an added `sentiment_score` column for quick validation.
- `sentiment_reviews_18oct.csv` : Full English review corpus enriched with `sentiment_score` outputs from the sentiment pipeline (snapshot dated 18 Oct).
- `games_encoded.csv` : Game metadata with cleaned genres, one-hot genre flags, and `release_age_days` for downstream modeling.

- `2_games_prices_merged.csv` : `games_encoded.csv` joined with aggregated price metrics (`base_price`, `price_volatility`, `avg_discount`).
- `2_price_features.csv` : Per-game pricing features derived from `prices.csv` (one row per `gameid`).
- `3_purchase_features.csv` : Player-level purchase features including library text fields, parsed `library_list`, `library_size`, `avg_purchase_price`, and `price_coverage`.
- `game_features_and_clusters.csv` : Scaled game features (age, price metrics, sentiment, genre flags) with assigned cluster labels (`cluster`).
- `recommendations_for_all_players.csv` : Final recommender output with one row per suggested `gameid` and `title` for each `playerid`, including `similarity_score` and source `cluster`.

Repository Layout

```

.
├── README.md
└── src
    ├── datasets/           # Raw + intermediate CSVs (download separately)
    └── eda/
        └── Dataset_Statistics.ipynb
    ├── feature-engineering/
        ├── FeatureEngineering.ipynb
        ├── ReviewSampling.ipynb
        └── sentiment/
            ├── run_sentiment.py
            ├── sentiment_analyser.py
            └── Sample1kTesting.ipynb
    └── model/
        ├── CollaborativeBasedFiltering.ipynb
        ├── ContentBasedFiltering_BeyondAccuracyMetrics.ipynb
        └── ContentBasedFiltering_Final.ipynb

```

Running the Workflow

- **Exploratory analysis:** Start with `src/eda/Dataset_Statistics.ipynb` to understand the raw data distributions and missingness.
- **Feature engineering:**
 - Use `src/feature-engineering/ReviewSampling.ipynb` to downsample reviews for manageable experiments.
 - Run `src/feature-engineering/FeatureEngineering.ipynb` to build player-level and game-level features, including price, engagement, and textual indicators.

- **Sentiment enrichment:**
 - Execute the sentiment script for large-scale processing:

```
python src/feature-engineering/sentiment/run_sentiment.py \
--input english_reviews.csv \
--output sentiment_reviews_full.csv \
--workers 4
```

Adjust `--workers` to match your CPU (3–4 workers ≈ 3 hours on ~30k reviews). For quick tests, use `english_reviews_1k.csv`.

- **Modelling:**
 - `src/model/ContentBasedFiltering_Final.ipynb` builds similarity-based recommenders using engineered features.
 - `src/model/CollaborativeBasedFiltering.ipynb` experiments with matrix factorisation and neighbourhood-based approaches.

Purpose and Content of Scripts

- **Dataset_Statistics.ipynb** – descriptive stats, missing value checks, and sanity checks for the raw tables.
- **FeatureEngineering.ipynb** – merges purchases, prices, and reviews into analytical tables, exporting `2_price_features.csv`, `3_purchase_features.csv`, etc.
- **ReviewSampling.ipynb** – sampling workflows for balanced sentiment analysis experiments, which includes language detection and removal of non-english reviews.
- **run_sentiment.py** – multi-processing of sentiment analysis; wraps `sentiment_analyser.py` (mandatory to run locally).
- **Sample1kTesting.ipynb** – tests for the sentiment pipeline on the 1k review subset.
- **ContentBasedFiltering_Final.ipynb / CollaborativeBasedFiltering.ipynb** – evaluate recommenders, calibrate hyperparameters, and export recommendation lists (e.g., `recommendations_for_all_players.csv`).

Reproducing Results

1. Download the full dataset bundle and place it in `src/datasets`.
2. Run the sentiment script (or use the precomputed `sentiment_reviews_18oct.csv` if available).
3. Execute feature-engineering notebooks to regenerate intermediate CSVs.
4. Open modelling notebooks to rebuild final recommendation outputs. Results are cached to `src/datasets`.