ABSA analysis of customer reviews

•••

With web scraping + LLM in Python

About ABSA * Aspect Based Sentiment Analysis

- A technique that identifies the sentiment associated with specific features or aspects of a product, service, or entity mentioned in a text.
- Unlike traditional sentiment analysis, which assigns an overall sentiment to a text, ABSA provides a more granular understanding

“The battery life of this phone is fantastic but the camera quality is quite disappointing in low light conditions”

->

ASPECT		SENTIMENT
battery life	->	positive
camera quality	->	negative

The Process * Web scraping

- Lightyear (lightyear.com) is a European investment platform that provides a low-fee way to invest in global stocks, ETFs, and money market funds
- The company has 1727 reviews on Trustpilot, on 73 pages
- From https://www.trustpilot.com/review/lightyear.com to https://www.trustpilot.com/review/lightyear.com?page=73
- Extracted 1441 reviews
- Dataframe, 1441 rows 5 columns:

country	date_rev	rate	review_title	review
HU	2022-02-11	5	Elegant UI	Easy to use and everything and everything…
ES	2023-01-30	4	Low fees	Sometimes when I get notifications with …

The Process * Data preparation

- review_id column added
- date_rev stored as date instead of string
- there are 206 empty reviews but review_title is always available
- !!! often the review_title repeats in the review but does not match by character
- review title was merged into review but only kept the longer text if repetition found
- review_title was eliminated
- Dataframe after preparation, 1441 rows, 5 columns:

country	date_rev	rate	review_id	review
HU	2022-02-11	5	124	Easy to use and everything and everything…
ES	2023-01-30	4	236	Sometimes when I get notifications with …

The Process * Open AI API: the prompt was perfected with Chat GPT

You are an expert in Aspect-Based Sentiment Analysis (ABSA) for customer reviews.
The company is **Lightyear**, a fintech investment app where users can invest in stocks and ETFs. Your task is to analyze each review and extract mentioned aspects, their sentiment, and relevant details, using the locked schema below:

Code	Category	Aspect_normalized
FTR_acc	Features Account	Account setup & funding/withdrawals
FTR_acc	Features Account	Local tax-wrapper availability (e.g. TBSZ)
FTR_por	Features Portfolio	Portfolio overview & detail view
FTR_por	Features Portfolio	Portfolio analytics & performance tracking
FTR_notif	Notifications & Alerts	Notifications, alerts, reminders
INT_app	Interface/Usability	Mobile app usability
INT_web	Interface/Usability	Web interface usability
INT_srch	Interface/Usability	Search & filtering
INT_nav	Interface/Usability	Navigation, menus, layout
INT_des	Interface/Usability	Design & aesthetics
TRD_frac	Trading Options	Fractional shares/ETFs
TRD_ord	Trading Options	Order types (market, limit, stop, etc.)
TRD_exec	Trading Options	Execution speed & quality
AST_var	Assets	Asset variety (stocks, ETFs, bonds, etc.)
AST_reg	Assets	Asset regional/market coverage
LNG	Language Support	App language support
LOC_reg	Regional Availability	Regional availability & features
PRC_fee	Pricing	Fees, spreads, commissions
PRC_fx	Pricing	FX/conversion costs
SUP_resp	Support	Customer support responsiveness
SUP_help	Support	Help resources, guides
GEN	Overall Satisfaction	Overall satisfaction
GEN_trust	Overall Satisfaction	Trust & brand reliability
OTH	Other	Catch-all for unclassified aspects

Instructions
1. Extract all aspects mentioned in the review. A review may contain multiple aspects.
2. For each aspect, assign: **aspect_normalized, category, code** (must match schema).
3. Add the **aspect_raw**: the reviewer's exact wording.
4. Classify sentiment: **Positive**, **Negative**, or **Neutral**.
   - Interpret sentiment correctly not literally, even if irony or sarcasm is present.
   - If the same aspect is mentioned with both positive and negative sentiment, output **separate entries** (not mixed).
5. If no aspects match, return `"aspects": []`.
6. Output must be valid JSON.

Each review must be analyzed separately and returned as:

```
{
  "review_id": <ID>,
  "aspects": [
    {"aspect_raw": "...", "aspect_normalized": "...", "category": "...", "code": "...", "sentiment": "..."},
    {"aspect_raw": "...", "aspect_normalized": "...", "category": "...", "code": "...", "sentiment": "..."}
    ...
  ]
}
```

And 5 example reviews with expected results are shared.

The Process * Dataframe after ABSA completion

- Dataframe was merged and flattened -> each aspect/sentiment is one row
- Dataframe, ready to analyze, 3985 rows, 10 columns*:

country	date_rev	rate	review_id	review
HU	2022-02-11	4	124	“The app is good but crashes often.”
HU	2022-02-11	4	124	“The app is good but crashes often.”

aspect_raw	aspect_normalized	category	code	sentiment
“app is good“	Mobile app usability	Interface/Usability	INT_app	Positive
“crashes often”	Mobile app usability	Interface/Usability	INT_app	Negative

*reviews in this sample table are examples only