Customer-Rating Prediction on Trustpilot Reviews

Final Report

Team TemuStars · Frank | Sebastian | Mohamed · DataScientest Cohort May 2025

1 Context & Objectives

Online marketplaces live and die by customer reviews. Temu (≈ 14 k English-language Trustpilot reviews) shows an unusually polarised 1- and 5-star pattern; predicting the star rating directly from free text therefore offers an excellent laboratory for Natural-Language-Processing pipelines, class imbalance strategies and real-time deployment.

Although we initially scraped AliExpress (~57k) and Wish (~99k), those volumes were impractical for local scraping and training time. On our mentor's advice that ~10k reviews are sufficient, we pivoted to Temu (~14k), keeping the polarised star distribution while fitting compute limits.

Goals

1. Scrape, clean and explore the Temu review corpus.
2. Predict 1–5-star ratings from text (multi-class classification).
3. Build an interactive Rating demo that returns an instant rating (and sentiment group) for any new review text.
4. Document insights for a 25 Aug oral defence (20 min + Q&A).

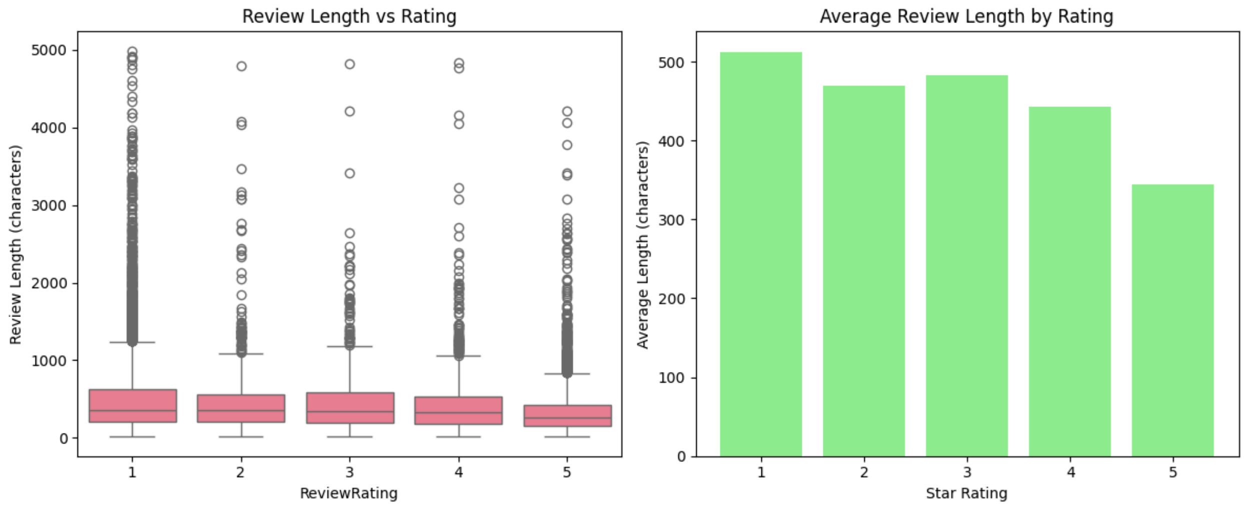
(Our original regression branch and template-based reply generator were discontinued for scope reasons but are archived under /src/discontinued_regression_way/.

2 Data Collection

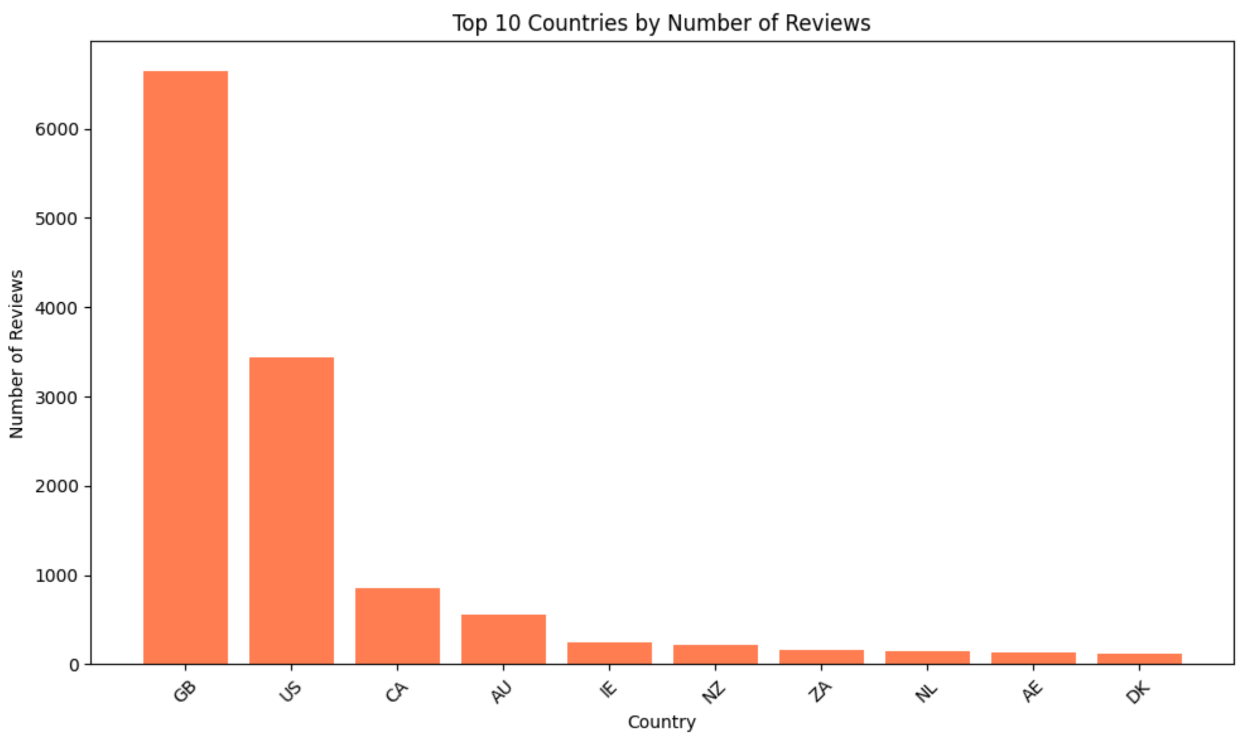
Source	Period	Raw size	Fields captured
Trustpilot API (custom **scrape_trustpilot.py**)	Sep 2022 – Jul 2025	13 855 reviews	UserId, Country, ReviewText, Rating, Date …

3 Exploratory Insights

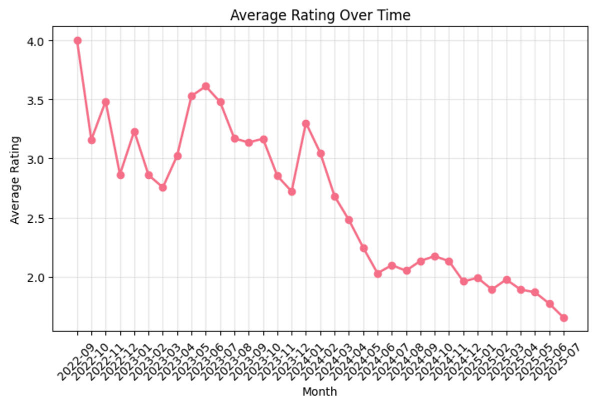
1-star texts are ~2 × longer and more emotional than 5-star.

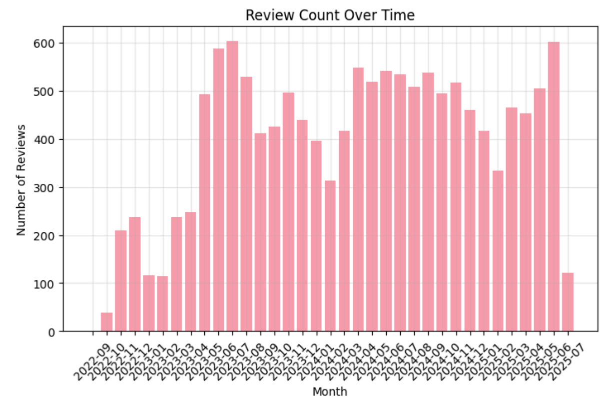


49 % GB, 25 % US → cultural bias to watch.

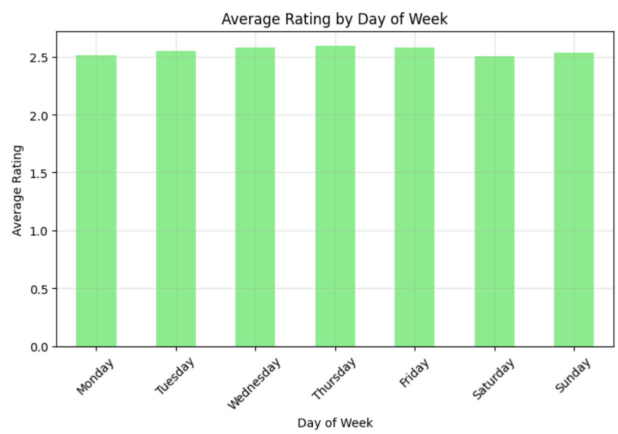


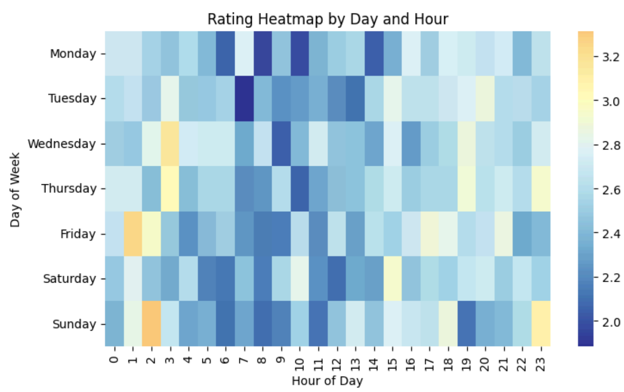
Down-trend from 4.0 → 1.7 ★ since late 2023 (possible logistic issues).





Night-time reviews are harsher; Fridays slightly friendlier.

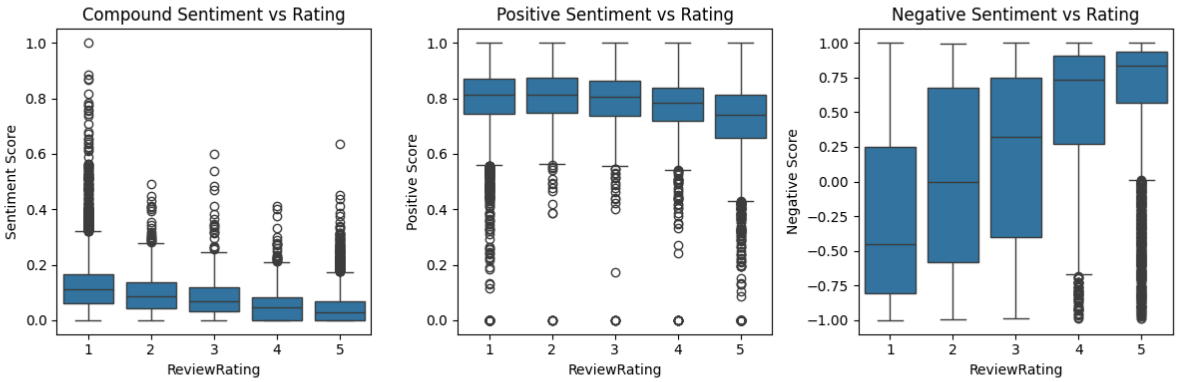




4 Text Pre-processing & Feature Engineering

Step	Toolkit	Output
HTML stripping, lower-casing, lemmatisation, stop-word removal	**spaCy**, **NLTK**	Clean corpus
Sentiment scores (compound / pos / neu / neg)	**VADER**	3 numeric cols
Surface features	length, capitals-ratio, punctuation counts	8 cols
TF-IDF 1–2-grams	max_features=5 000	sparse 10 k × 5 001
Final feature matrix	5 011 dims	

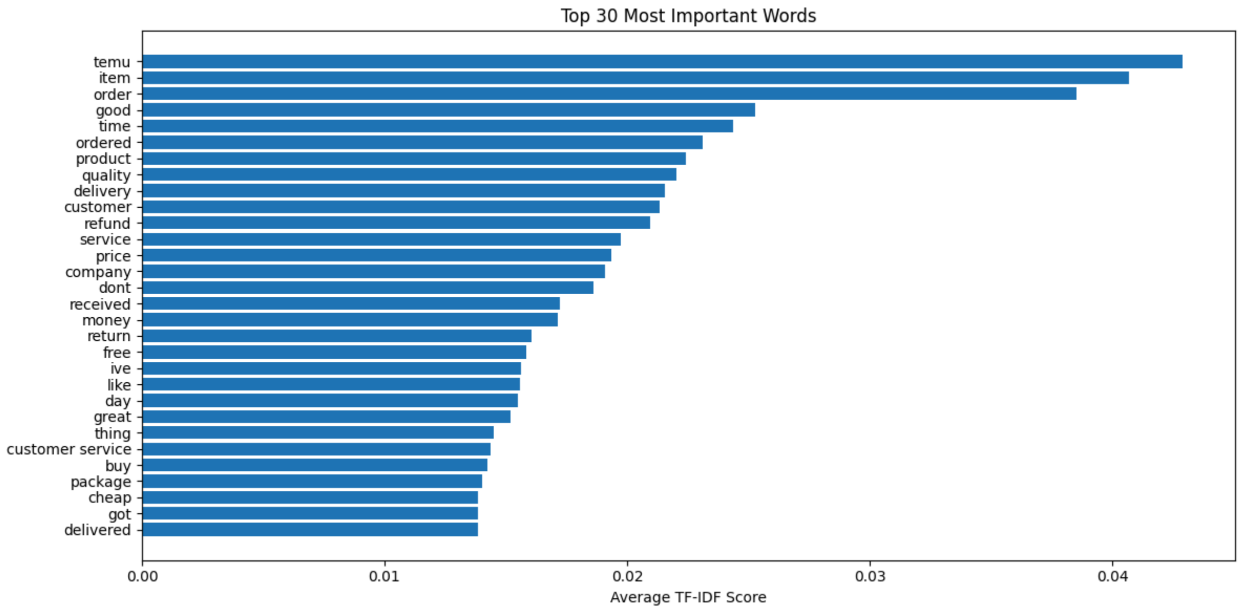
VADER tracks the stars as expected: the negative score is highest for 1★ and collapses towards 0 by 5★; the positive score does the reverse; the compound score shifts accordingly with star level. This validates using VADER features alongside TF-IDF.



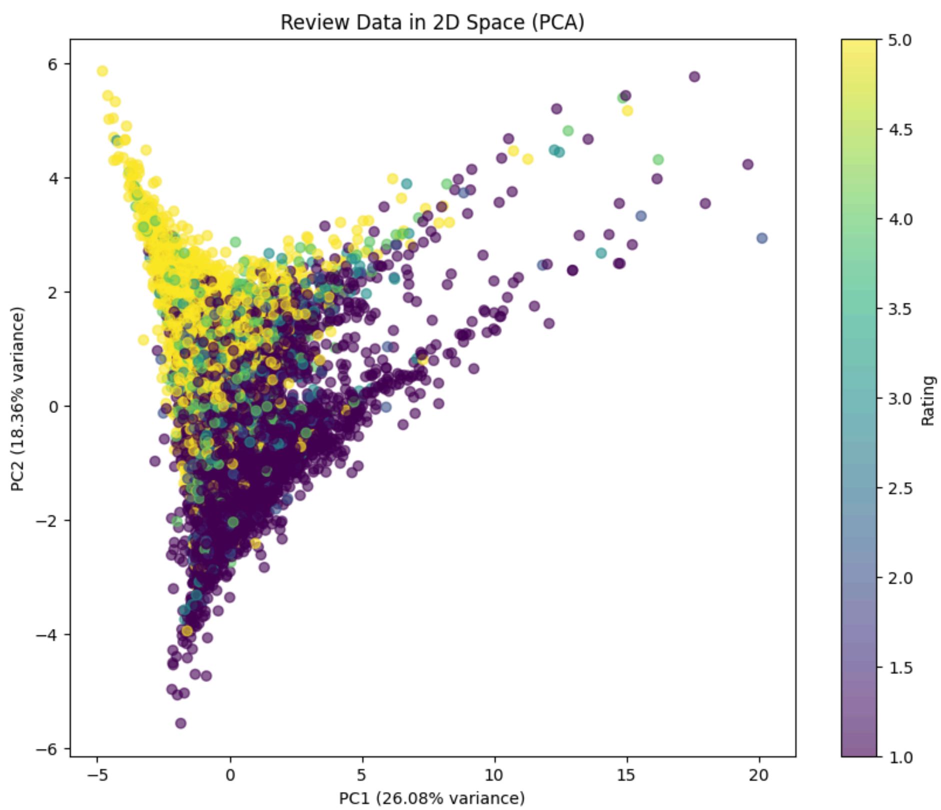
We removed platform-specific tokens **'temu'**, **'order'**, **'item'** from the visualisation because they were overwhelmingly dominant.

Sentiment-split Word Clouds

Top-30 TF-IDF terms confirms a complaint focus: **refund, package, delivery**.



A PCA retains 90 % variance in 6 components and already shows a clear 1-vs-5 separation.



5 Modelling & Evaluation

5.1 Candidates

LogReg · LinearSVC · RF · GBDT · XGBoost · k-NN · GaussianNB · DecisionTree
plus **Hard/Soft Voting** and a **Stacking** ensemble (RF + ExtraTrees as base, LogReg meta-learner). Class weights and StratifiedKFold (4×) were used to counter the 1/5 dominance.

5.2 Comparison

CLASSIFICATION MODEL COMPARISON

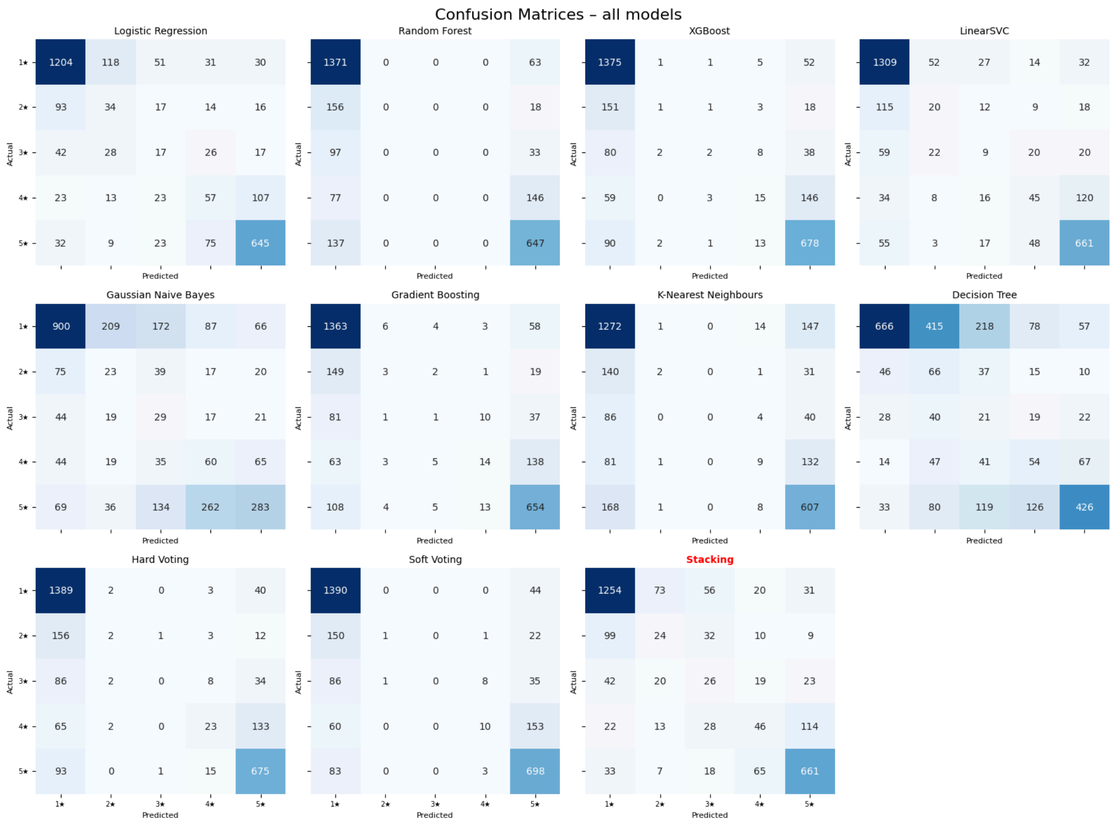
Rank	Model	Accuracy	Weighted F1	Macro F1	W. Precision	W. Recall
1	Stacking	0.733	0.725	0.452	0.719	0.733
2	Logistic Regression	0.713	0.715	0.447	0.717	0.713
3	LinearSVC	0.745	0.719	0.432	0.701	0.745
4	Hard Voting	0.761	0.695	0.371	0.673	0.761
5	XGBoost	0.754	0.687	0.361	0.667	0.754
6	Gradient Boosting	0.741	0.678	0.355	0.652	0.741
7	Soft Voting	0.765	0.690	0.353	0.689	0.765
8	Decision Tree	0.449	0.522	0.334	0.675	0.449
9	Random Forest	0.735	0.656	0.321	0.593	0.735
10	K-Nearest Neighbours	0.689	0.624	0.318	0.607	0.689
11	Gaussian Naive Bayes	0.472	0.523	0.308	0.612	0.472

Rank	Model	Accuracy	Weighted F1
🥇	**Stacking**	**0.733**	**0.725**
2	Soft Voting	0.765	0.690
3	XGBoost	0.754	0.687

Metrics throughout the classification track use **Weighted F1** (handles class imbalance better than Macro F1).

5.3 Confusion matrices for all candidate models and the Stacking classification report

Detailed Stacking report – highest weighted-F1 with strong 1★ & 5★ performance, weaker on rare 2–4★ classes.




```
DETAILED ANALYSIS – Stacking
============================================================
```


```
DETAILED ANALYSIS – Stacking
============================================================
```


```
Classification report:
              precision    recall  f1-score   support

        1⭐      0.865     0.874     0.870      1434
        2⭐      0.175     0.138     0.154       174
        3⭐      0.163     0.200     0.179       130
        4⭐      0.287     0.206     0.240       223
        5⭐      0.789     0.843     0.815       784

    accuracy                         0.733      2745
   macro avg      0.456     0.452     0.452      2745
weighted avg      0.719     0.733     0.725      2745


PER-CLASS PERFORMANCE:
1⭐ : Precision = 0.865   Recall = 0.874   F1 = 0.870   Support = 1434
2⭐ : Precision = 0.175   Recall = 0.138   F1 = 0.154   Support = 174
3⭐ : Precision = 0.163   Recall = 0.200   F1 = 0.179   Support = 130
4⭐ : Precision = 0.287   Recall = 0.206   F1 = 0.240   Support = 223
5⭐ : Precision = 0.789   Recall = 0.843   F1 = 0.815   Support = 784
```

5.4 Error Analysis (Prediction Stage)

Set-up. We sampled 100 random Temu reviews from the processed corpus and evaluated the trained **Stacking** classifier on the raw 5-star task. Because Temu is highly imbalanced (many 1★/5★), we also applied our **grouped refinement** at prediction time:

- **Groups:** neg = {1,2}, neu = {3}, pos = {4,5}
- **Rule:** take the majority group according to the model's class probabilities; inside the winning group pick the higher-probability star.
- **Bias control:** reduce the prior for class **3★** by **20 %** (to counter its tendency to be over-predicted).

Headline results on the 100-sample run

- Raw (initial) 5-class prediction: **Weighted F1 = 0.603**, **Macro F1 = 0.405**, **Accuracy = 52 %**.
- After group refinement (neg/neu/pos): **Grouped accuracy = 91 %**, **Grouped Macro F1 = 0.630**.
- Quality breakdown: **52 % perfect initial**, **38 % off by one star**, **9 % corrected by refinement**, **1 % off by ≥2 stars**.

Where the model is strong / weak

- **Strong:** 1★ and 5★ (clear lexical cues like *scam, refund, broken* vs. *love, excellent, great*).
- **Weak:** 2★ and 4★ (rare classes), and **neutral 3★**, which often sits between sentiment poles. The 20 % penalty on 3★ reduces this bias without hurting 1★/5★.

Typical error patterns (from the 100-sample audit)

- **Over-prediction dominates**: 31 of 48 errors were predicted **higher** than ground truth (e.g., 1★ → 3★).
- **Most common mis-type: True 1★ / Pred 3★** (13×).

- **Why this happens:** very long "mixed" texts (lists of issues, quoted complaints, sarcasm) produce **lexical overlap** with neutral vocabulary.
- **Example (Row 84, the only ≥2-star miss):**
Ground truth **5★**, model predicted **2★**. The review mostly enumerates **other people's complaints** (delivery/carrier problems), flooding the text with negative tokens; the model (correctly) reads the text as negative even though the author's final rating is positive.

Take-away: The refinement step (group vote + 3★ penalty) materially improves practical accuracy by pulling "near-miss" predictions into the right sentiment band.

Prediction audit on 100 random reviews. Dark green = perfect initial; light green = corrected by refinement; yellow = ±1 star; red = ≥2 stars (only Row 84). Refinement consistently pulls borderline cases into the correct sentiment group.

Color Legend: Perfect Initial Prediction | Corrected by Adjustment | 1 Star Difference | 2+ Stars Difference

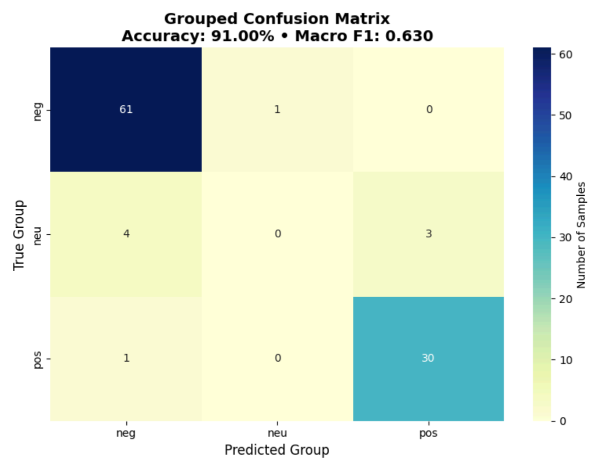
Index	True★	Pred★	Adj★	Group True	Group Pred	Group Adj
1	2★	2★	2★	neg	neg	neg
2	1★	2★	2★	neg	neg	neg
3	1★	2★	2★	neg	neg	neg
4	2★	4★	2★	neg	pos	neg
5	1★	1★	1★	neg	neg	neg
6	1★	3★	1★	neg	neu	neg
7	5★	5★	5★	pos	pos	pos
8	3★	2★	2★	neu	neg	neg
9	1★	1★	1★	neg	neg	neg
10	4★	4★	4★	pos	pos	pos
11	3★	4★	4★	neu	pos	pos
12	5★	5★	5★	pos	pos	pos
13	1★	2★	2★	neg	neg	neg
14	1★	3★	2★	neg	neu	neg
15	1★	3★	2★	neg	neu	neg
16	5★	5★	5★	pos	pos	pos
17	5★	4★	4★	pos	pos	pos
18	5★	4★	4★	pos	pos	pos
19	5★	4★	4★	pos	pos	pos
20	1★	1★	1★	neg	neg	neg
21	1★	1★	1★	neg	neg	neg
22	2★	2★	2★	neg	neg	neg
23	5★	5★	5★	pos	pos	pos
24	1★	3★	1★	neg	neu	neg
25	1★	2★	2★	neg	neg	neg
26	1★	1★	1★	neg	neg	neg
27	1★	3★	2★	neg	neu	neg
28	1★	2★	2★	neg	neg	neg
29	1★	1★	1★	neg	neg	neg
30	2★	2★	2★	neg	neg	neg
31	1★	3★	2★	neg	neu	neg
32	1★	2★	2★	neg	neg	neg

66	1⭐	1⭐	1⭐	neg	neg	neg
67	1⭐	1⭐	1⭐	neg	neg	neg
68	1⭐	1⭐	1⭐	neg	neg	neg
69	1⭐	1⭐	1⭐	neg	neg	neg
70	1⭐	1⭐	1⭐	neg	neg	neg
71	3⭐	2⭐	2⭐	neu	neg	neg
72	5⭐	4⭐	4⭐	pos	pos	pos
73	1⭐	3⭐	1⭐	neg	neu	neg
74	5⭐	4⭐	4⭐	pos	pos	pos
75	3⭐	2⭐	2⭐	neu	neg	neg
76	5⭐	5⭐	5⭐	pos	pos	pos
77	1⭐	3⭐	2⭐	neg	neu	neg
78	3⭐	4⭐	4⭐	neu	pos	pos
79	1⭐	3⭐	2⭐	neg	neu	neg
80	1⭐	1⭐	1⭐	neg	neg	neg
81	2⭐	2⭐	2⭐	neg	neg	neg
82	1⭐	1⭐	1⭐	neg	neg	neg
83	5⭐	4⭐	4⭐	pos	pos	pos
84	5⭐	2⭐	2⭐	pos	neg	neg
85	1⭐	3⭐	1⭐	neg	neu	neg
86	2⭐	2⭐	2⭐	neg	neg	neg
87	2⭐	4⭐	2⭐	neg	pos	neg
88	1⭐	1⭐	1⭐	neg	neg	neg
89	1⭐	2⭐	2⭐	neg	neg	neg
90	5⭐	5⭐	5⭐	pos	pos	pos
91	1⭐	3⭐	1⭐	neg	neu	neg
92	5⭐	5⭐	5⭐	pos	pos	pos
93	5⭐	5⭐	5⭐	pos	pos	pos
94	1⭐	1⭐	1⭐	neg	neg	neg
95	1⭐	3⭐	1⭐	neg	neu	neg
96	1⭐	1⭐	1⭐	neg	neg	neg
97	1⭐	1⭐	1⭐	neg	neg	neg
98	1⭐	1⭐	1⭐	neg	neg	neg
99	3⭐	4⭐	4⭐	neu	pos	pos
100	1⭐	2⭐	2⭐	neg	neg	neg

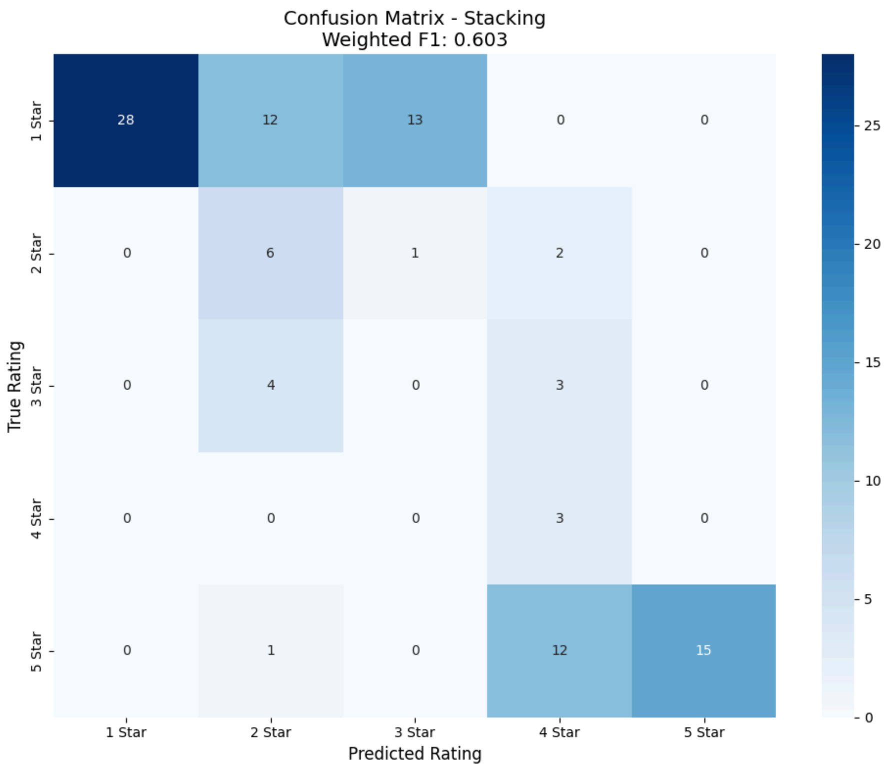
```
🔴 Worst Predictions Analysis (2+ Stars Difference)
================================================================================
Found 1 samples with 2+ star difference:
--------------------------------------------------------------------------------

🔥 Worst Prediction #1 (Sample Index: 84)
True: 5⭐ | Predicted: 2⭐ | Adjusted: 2⭐
Difference: 3 stars
Original Text: Why the full everyone here is complaining only about damaged packages or delivery problems. You handicapped or so ? You review should be over temu , and not about ups or dhl .
Processed Text: full everyone complaining damaged package delivery problem handicapped review temu ups dhl make review trustpilot look like cant count anymore...
Probabilities: 1⭐: 0.170 | 2⭐: 0.304 | 3⭐: 0.225 | 4⭐: 0.243 | 5⭐: 0.057
------------------------------------------------------------

✅ Analysis complete! Grouped accuracy: 91.00%, Macro F1: 0.630, Weighted F1: 0.882
```

```
📈 Prediction Quality Summary:
==================================================
🟢 Perfect Initial Predictions:  52 (52.0%)
🟡 Corrected by Adjustment:       9 (9.0%)
🟧 Close Predictions (1 diff):   38 (38.0%)
🔴 Poor Predictions (2+ diff):    1 (1.0%)
==================================================
📊 Total Samples: 100
```

Grouped evaluation after refinement: **91 % accuracy**, **Macro F1 = 0.630** (100-sample run). Most residual confusion is between neg and neu; pos is well separated.



Raw 5-class performance before refinement: **Weighted F1 = 0.603**, **Macro F1 = 0.405**. Sparse classes (2★/4★) remain hard.



6 Interactive Rating Demo (Notebook UI)

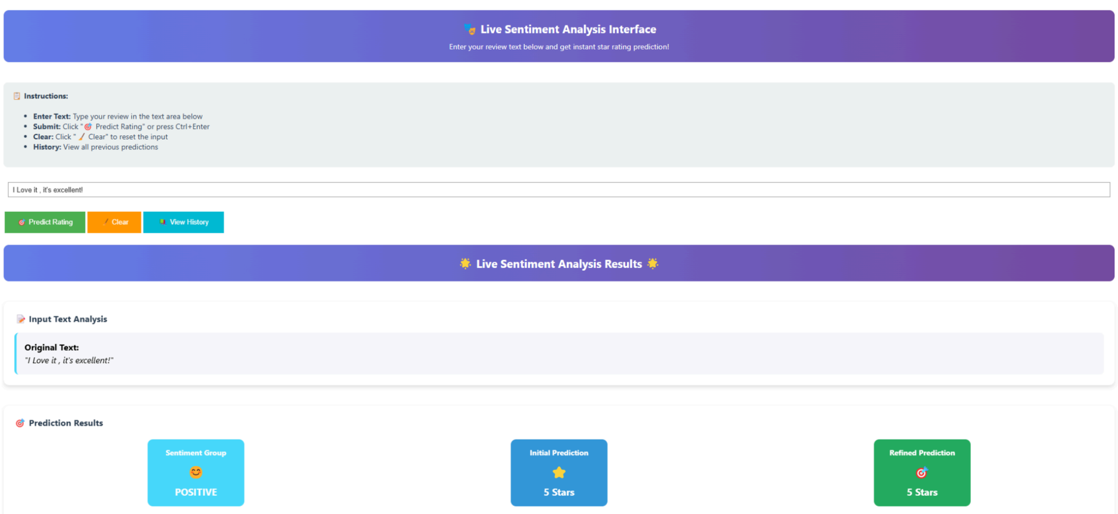
What the user sees

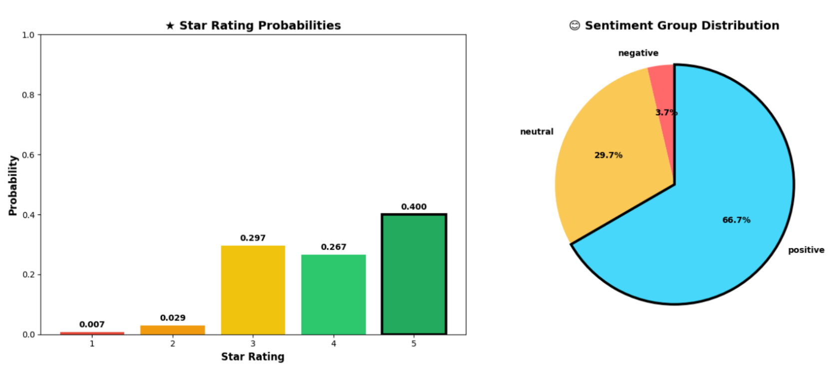
1. Enter any free-text review.
2. **Initial prediction** from the Stacking model (5-star).
3. **Refined rating** via sentiment groups (neg/neu/pos) with the **20 % 3★ penalty**.
4. Visuals:
   - Bar chart of **class probabilities (1–5★)**,
   - Pie chart of **sentiment distribution (neg/neu/pos)**,
   - **History table** of previous inputs.

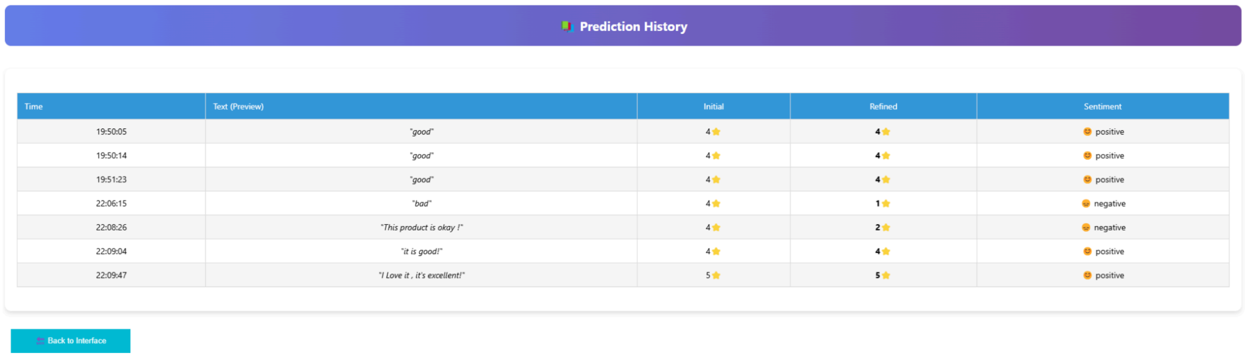
Example runs (shown in the screenshots)

- "**bad**" → initial **4★**, **refined 1★** (refinement fixes a confident over-prediction).
- "**This product is okay!**" → initial **4★**, **refined 2★** (neutral tone is treated as "neg-leaning" rather than positive).
- "**it is good!**" → **4★** initial and refined.
- "**I love it, it's excellent!**" → **5★** initial and refined.

Practical note. Everything runs in the notebook with saved pickles (`best_classification_model.pkl`, vectorizer, scaler). No Streamlit used.







Prediction History

Time	Text (Preview)	Initial	Refined	Sentiment
19:50:05	"good"	4 ⭐	**4** ⭐	😊 positive
19:50:14	"good"	4 ⭐	**4** ⭐	😊 positive
19:51:23	"good"	4 ⭐	**4** ⭐	😊 positive
22:06:15	"bad"	4 ⭐	**1** ⭐	😞 negative
22:08:26	"This product is okay !"	4 ⭐	**2** ⭐	😞 negative
22:09:04	"it is good!"	4 ⭐	**4** ⭐	😊 positive
22:09:47	"I Love it , it's excellent!"	5 ⭐	**5** ⭐	😊 positive

Back to Interface

7 Additional Experiments

A separate `first_deep_model/` folder explores deep-learning baselines (CNN and Bi-LSTM with embeddings) for star prediction. We also provide a small FastAPI demo with a `/predict` endpoint that accepts JSON {"reviewText":..,"reviewCount":.., "ReviewTitle":..} and returns the predicted star.

8 Conclusions & Recommendations

- **Stacking** delivers the best trade-off on imbalanced data (*Weighted F1 = 0.725*).
- Text length, VADER compound score and TF-IDF n-grams suffice – expensive embeddings are optional.
- Negative reviews are verbose and easier to spot; neutral (3★) remains the Achilles' heel.
- The prototype proves production viability

On a 100-review audit, the Stacking model scores **52 %** accuracy (Weighted F1 **0.603**) on raw 5-class predictions. Most errors are mild (±1 star) and over-predicted (1★→3★). Applying our grouped refinement (neg = {1,2}, neu = {3}, pos = {4,5}) and a 20 % prior penalty on 3★ lifts grouped accuracy to **91 %** (Macro F1 **0.630**). Extremes (1★/5★) are strong; rare mid classes (2★/4★) remain challenging. The single ≥2-star miss was a 5★ review whose text enumerated negative complaints, which rightly triggered a low-star prediction. The refinement step therefore improves practical reliability without changing the underlying model.

Acknowledgements

We thank **Kylian Santos** (mentor) for steering us toward classification and for the constructive Slack feedback.