Detecting Fake Reviews: Interim Report

Team SPC : Santhosh, Pratham, Chanukya

1 Introduction

Online reviews significantly influence consumer behavior and product visibility. However, the increasing prevalence of **computer-generated (CG)** reviews threatens their reliability. In this project, we aim to detect fake (CG) reviews using a multi-model architecture that combines various heuristics and representations.

We utilize the publicly available Fake Reviews Dataset from Kaggle, which consists of the following columns:

- `category` – The product category
- `rating` – User-assigned rating (1–5)
- `label` – `CG` (computer-generated) or `OR` (original)
- `text_` – The actual review text

Our primary objective is to predict the `label` of a review as either `CG` or `OR`.

2 Approach

Our approach consists of three models, each grounded in a unique hypothesis. We aim to eventually integrate them using a DAG-based inference architecture.

2.1 Model 1: Category-Text Relevance Model

Hypothesis: A CG review is less likely to be contextually relevant to the product category.

We implemented a category-text relevance model using Latent Dirichlet Allocation (LDA) and TF-IDF. For each review, we compute its relevance score of text with the category.

Observation: Lower similarity values are often associated with CG reviews, validating our assumption. This score can serve as a feature for downstream classification.

2.2 Model 2: Text-Based Rating Regressor

Hypothesis: CG reviews may not semantically align with the rating they are associated with.

We train a regressor (e.g., linear regression, MLP) to predict the `rating` based on the `text_`. For each review, we compute the absolute difference between the predicted and actual rating.

Observation: CG reviews tend to have a larger deviation, supporting our assumption. This difference score is also used as a feature in the final classification.

2.3 Model 3: CBOW + Decision Tree Classifier

Hypothesis: CG reviews have different stylistic patterns than OR reviews.

We use a CBOW-style embedding trained on the dataset averaged over the tokens in the review. These embeddings are used as input features to a Decision Tree classifier trained to distinguish between CG and OR labels.

Observation: Initial results show that decision trees can capture shallow but meaningful patterns differentiating CG from OR reviews.

3 Models

3.1 Model 1: Category-Text Relevance Model

Methodology: We implemented a category-text relevance model using Latent Dirichlet Allocation (LDA) and TF-IDF:

- **Text Preprocessing:** All review texts were lowercased, cleaned to retain only alphabetic characters, tokenized, lemmatized, and stopwords were removed.
- **Topic Modeling with LDA:** An LDA model with 10 topics was trained on the entire corpus of tokenized reviews. Each review was then represented as a 10-dimensional topic distribution vector.
- **Category Representation with TF-IDF:** Each category label (mapped to a human-readable name) was vectorized using a TF-IDF model trained over all category names. This resulted in a 10-dimensional vector for each category.
- **Relevance Score:** For each review, we computed the cosine similarity between its LDA topic vector and the TF-IDF vector of its corresponding category. This similarity score was interpreted as a measure of semantic relevance.

Classification using Thresholds:

- For each category, we selected the optimal decision threshold from the training set that maximized classification accuracy.
- During validation, the relevance score was converted into a probability using a sigmoid function centered at the category-specific threshold.
- Final classification was based on a probability cutoff of 0.5 — scores above this were labeled `OR`, and below this as `CG`.

3.2 Model 2: Text-Based Rating Regression Model

Methodology: We implemented a regression model using BERT embeddings to predict normalized review ratings:

- **Text Encoding:** Each review was tokenized using the `bert-base-uncased` tokenizer. The text was padded/truncated to a maximum length of 128 tokens.
- **Model Architecture:** We fine-tuned a pre-trained BERT model where the `[CLS]` token's output (a 768-dimensional vector) was passed through a feedforward regressor:

  - Linear layer: $768 \rightarrow 128$
  - ReLU activation
  - Dropout layer with 0.3 rate
  - Linear layer: $128 \rightarrow 1$ (final scalar output)

- **Training Details:** The model was trained using MSE loss and Adam optimizer. The output values were scaled to the range [0, 1] using MinMaxScaler.
- **Evaluation Setup:** The model was trained on GPU and the predictions were inverse-transformed to restore the original 0–5 rating scale. Predictions were clipped to ensure values remained in this valid range.

Threshold-Based Classification and Probability Estimation:

- For each integer rating $r \in \{1, 2, 3, 4, 5\}$, we computed the optimal decision threshold on the validation set using the ROC curve by selecting the cutoff that maximized the classification accuracy between `CG` and `OR` classes.
- Each predicted rating $\hat{y}$ was then compared against the closest integer r to determine the class-specific threshold T_r.
- The probability of a sample belonging to the `CG` class was calculated as:

$$\text{prob\_CG} = 1 - \frac{|\hat{y} - r|}{T_r}$$

- The resulting score was clipped to the [0, 1] range and interpreted as the probability of the `CG` class.
- Along with `prob_CG`, the model's predicted rating was also stored for each review instance.

3.3 Model 3: CBOW + Decision Tree Classifier

For our third model, we employed a traditional supervised learning approach by training a Decision Tree classifier on text embeddings derived from a Continous Bag-of-Words (CBOW) representation.

- We used `CountVectorizer` to convert review texts into a high-dimensional sparse vector space with a vocabulary size capped at 5000.
- Labels were binarized with `0` representing original reviews (OR) and `1` representing computer-generated reviews (CG).
- A `DecisionTreeClassifier` was trained on the BoW-encoded data using scikit-learn's `Pipeline`, streamlining both vectorization and classification.

This model serves as a direct text-based baseline for detecting fake reviews and is integrated as one of the components in our final ensemble.

4 Results

4.1 Model 1: Category-Text Relevance Model

Category-wise accuracy, precision, recall, and F1 scores were evaluated on the validation set.

Table 1: Category-wise Thresholds and Evaluation Metrics for Relevance Model

Category	Threshold	Accuracy	Precision	Recall	F1 Score
Sports and Outdoors	0.0202	0.5873	0.7167	0.2820	0.4047
Toys and Games	0.0198	0.5814	0.5906	0.5847	0.5876
Home and Kitchen	0.0375	0.5079	0.5263	0.4615	0.4918
Books	0.3469	0.7316	0.7215	0.7238	0.7227
Clothing Shoes and Jewelry	0.1341	0.7518	0.7797	0.6730	0.7224
Tools and Home Improvement	0.0381	0.5420	0.6054	0.3660	0.4562
Pet Supplies	0.0000	0.4776	0.4776	1.0000	0.6464
Kindle Store	0.0310	0.5971	0.7872	0.2202	0.3442
Movies and TV	0.0000	0.5094	0.5094	1.0000	0.6750
Electronics	0.6735	0.5565	0.5776	0.5942	0.5858

Interpretation: The performance of the category-text relevance model varies significantly across categories. Categories like *Books* and *Clothing Shoes and Jewelry* show relatively high accuracy (above 73%) and balanced precision-recall, indicating that relevance is a good indicator of review authenticity in these domains. In contrast, categories like *Pet Supplies* and *Movies and TV* have low thresholds and perfect recall but poor precision, suggesting the model tends to over-predict original (OR) reviews. These variations highlight the need for category-specific thresholding and suggest that combining this model with other signals (e.g., rating consistency or text features) will be essential for robust detection.

4.2 Model 2: Text-Based Rating Regression Model

The validation set metrics are present in the below table

Table 2: Performance Metrics per Rating for the Validation Set

Rating	Best Threshold	Accuracy	Precision	Recall	F1 Score
1.0	2.4941	0.8529	0.8963	0.7707	0.8288
2.0	1.4941	0.8824	0.8910	0.8797	0.8854
3.0	1.0107	0.7448	0.7022	0.8712	0.7776
4.0	0.0100	0.4992	0.4975	0.9983	0.6641
5.0	0.0844	0.5564	0.5368	0.8416	0.6555

Plot for distribution of classes for different thresolds

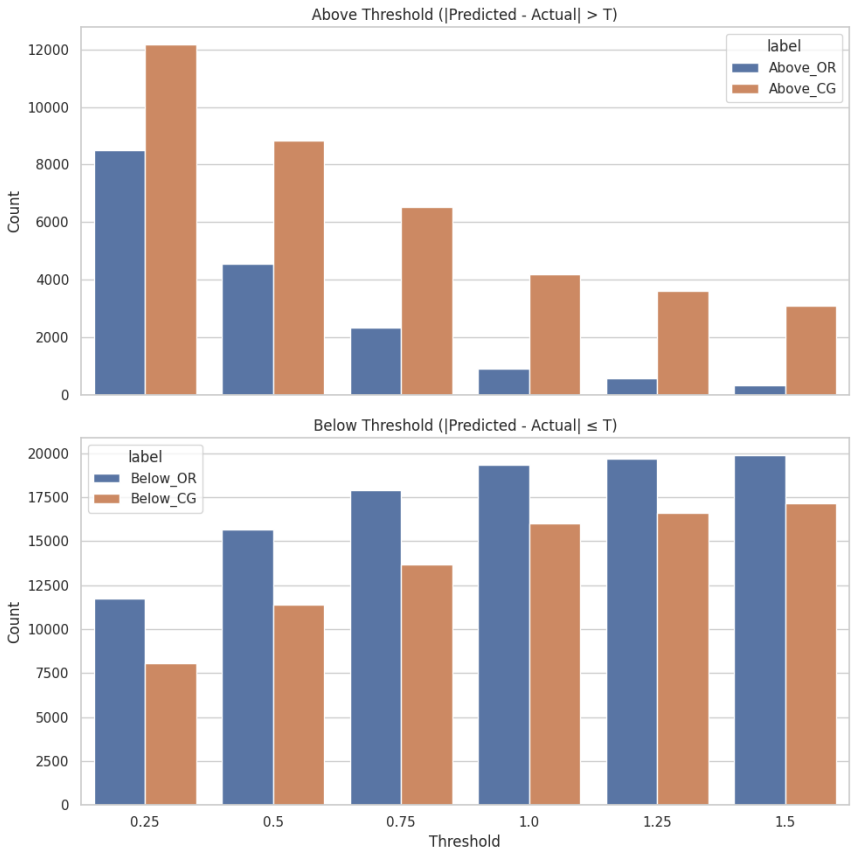

Figure 1: Distribution of classes across different thresold

Interpretation:

- The model exhibits high validation accuracy for lower rating values — particularly for ratings 1, 2, and 3 — with accuracies exceeding 74%. This indicates that the model is more confident and reliable in identifying cases likely to be classified as `CG` when the review sentiment is negative or neutral.
- For higher ratings such as 4 and 5, the validation accuracy drops significantly (around 50%–57%), suggesting that the model finds it difficult to draw a consistent boundary for classification in more favorable reviews. This may be due to high correlation between the text and the rating for both CG and OR reviews.
- Given this observation, in downstream applications or decision-making systems, it would be beneficial to assign more weight or trust to the model's `prob_CG` predictions when the predicted rating is 1, 2, or 3. Conversely, additional review or secondary heuristics might be necessary when the predicted rating is 4 or 5.

4.3 Model 3: CBOW + Decision Tree Classifier

We evaluated a Decision Tree classifier trained on Bag-of-Words (BoW) embeddings. The performance metrics on the validation set are presented below:

Metric	OR	CG
Precision	0.74	0.72
Recall	0.70	0.75
F1-Score	0.72	0.73
Overall Accuracy	0.7251	
ROC-AUC	0.7251	

Table 3: Validation Metrics for BoW + Decision Tree Model

Interpretation: The model achieves a validation accuracy of 72.51% and an ROC-AUC of 0.7251. It shows relatively balanced performance across both classes—Original (OR) and Fake (CG)—with F1-scores of 0.72 and 0.73 respectively. This indicates a decent ability to detect fake reviews but highlights potential for improvements using more powerful classifiers.

4.4 Model Combination via Data-Dependent Weighting

To leverage the strengths of multiple models, we implement a weighted ensemble that dynamically assigns weights to each model's output based on the input instance's rating and category. We consider three predictive signals:

- **Category-Text Relevance (w_1):** A fixed weight is assigned to the category relevance model for specific categories where this signal is known to be informative. In particular:
  - $w_1 = 0.1$ for `Books_5` and `Kindle_Store_5`
  - $w_1 = 0.2$ for `Clothing_Shoes_and_Jewelry_5` and `Toys_and_Games_5`
  - $w_1 = 0$ for all other categories
- **Rating Consistency (w_2):** Reviews with lower ratings are more likely to be artificial. We assign:
  - $w_2 = 0.9$ for ratings 1 or 2
  - $w_2 = 0.8$ for rating 3
  - $w_2 = 0$ for ratings 4 and 5
- **Text-Based Classifier (w_3):** The remaining weight is allocated to a text-based classifier (CBOW + Decision Tree), calculated as:

$$w_3 = 1 - w_1 - w_2$$

The final probability of a review being classified as `CG` is computed using a convex combination:

$$P(\text{CG}) = w_1 \cdot \text{prob\_cat} + w_2 \cdot \text{prob\_rating} + w_3 \cdot \text{prob\_bow\_dt}$$

where the weights w_1, w_2, w_3 are determined per instance based on the review's metadata.

This data-dependent weighting scheme allows the ensemble to adaptively leverage the most reliable predictors for each review, resulting in improved classification performance.

4.5 Final Results

On validation set (using just model3)

Label	Precision	Recall	F1-Score	Support
CG	0.7160	0.7460	0.7307	3032
OR	0.7350	0.7043	0.7193	3033
Accuracy		0.7251		
Macro Avg	0.7255	0.7251	0.7250	6065
Weighted Avg	0.7255	0.7251	0.7250	6065

On validation set (using 3 models)

Label	Precision	Recall	F1-Score	Support
CG	0.7337	0.7688	0.7508	3032
OR	0.7573	0.7211	0.7387	3033
Accuracy		0.7449		
Macro Avg	0.7455	0.7449	0.7448	6065
Weighted Avg	0.7455	0.7449	0.7448	6065

On test set (using just model3)

Label	Precision	Recall	F1-Score	Support
CG	0.7028	0.7432	0.7224	3033
OR	0.7274	0.6857	0.7059	3032
Accuracy		0.7144		
Macro Avg	0.7151	0.7144	0.7142	6065
Weighted Avg	0.7151	0.7144	0.7142	6065

On test set (using 3 models)

Label	Precision	Recall	F1-Score	Support
CG	0.7177	0.7620	0.7392	3033
OR	0.7462	0.7002	0.7225	3032
Accuracy		0.7311		
Macro Avg	0.7320	0.7311	0.7308	6065
Weighted Avg	0.7320	0.7311	0.7308	6065

Validation and test accuracy for different model combinations

Model	Validation Accuracy	Test Accuracy
M3	0.7251	0.7144
M1 + M2 + M3	0.7449	0.7311

Though the improvement is slight, the ensemble of M1, M2, and M3 shows consistently higher accuracy than M3 alone on both validation and test sets, suggesting a modest gain in generalization performance.

5 Current Progress

- Dataset preprocessing and initial exploration complete
- Category-text relevance model implemented with cosine similarity
- Rating regression model trained and evaluated
- CBOW + Decision Tree model trained on review text
- Integration and analysis of intermediate outputs in progress

6 Future Work

- Explore adversarial architectures inspired by GANs (Generative Adversarial Networks) to improve fake review detection using a generator-discriminator setup.
- Evaluate additional classifiers such as:
  - Random Forests
  - Support Vector Machines (SVMs)
  - Logistic Regression
- Experiment with richer text embeddings by incorporating:
  - Term Frequency–Inverse Document Frequency (TF-IDF)
  - Bag-of-Words (BoW)
- Combine multiple classifiers and embeddings in ensemble models for better generalization across review categories.

7 Git Repository

The complete codebase, including preprocessing scripts, model training routines, and evaluation notebooks, is available on GitHub:

`https://github.com/Chanukya578/Fake_Review_Detector`

8 Conclusion

By leveraging multiple independent signals and integrating them through a DAG structure, we aim to develop a robust system for detecting fake reviews. Our current findings support the underlying assumptions of each model and motivate further exploration of ensemble strategies.