CHATGPT - SENTIMENT ANALYSIS AND REVIEW SCORE PREDICTIONS

Text Mining - for Business Uses

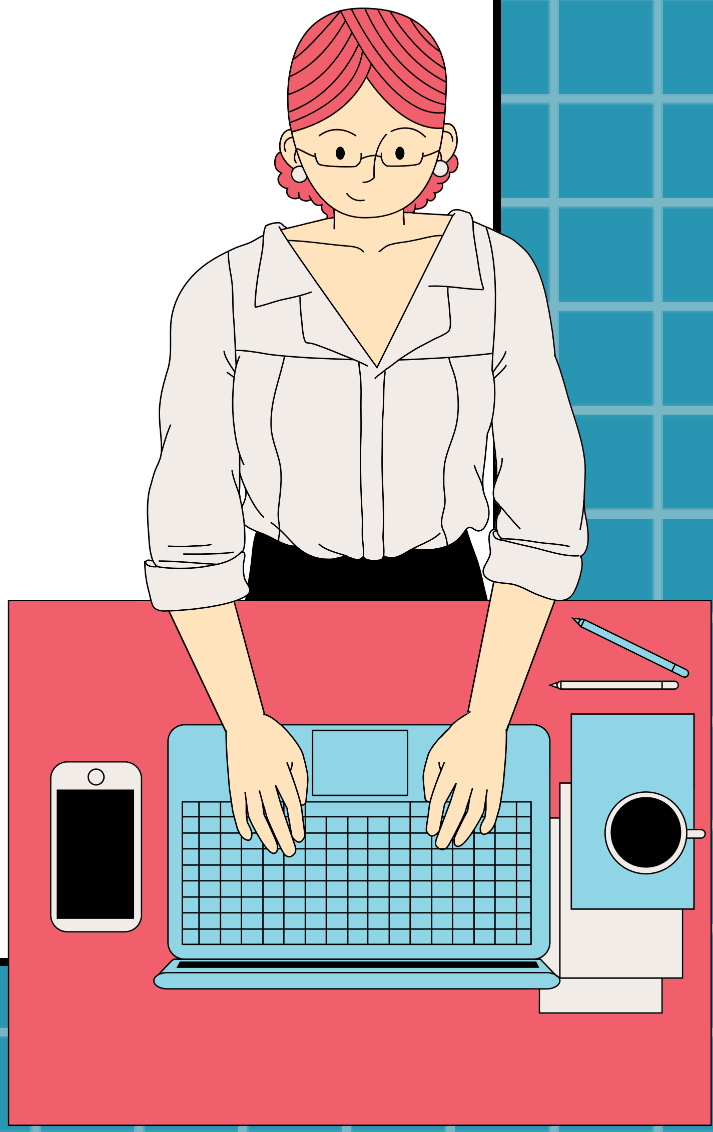
Uriel Bender - 205837131
Tal Lavi - 208662395

RESEARCH QUESTION

How can we predict the rating score (scale of 1-5) of user reviews for the ChatGPT app on Android devices to better understand and enhance customer satisfaction?

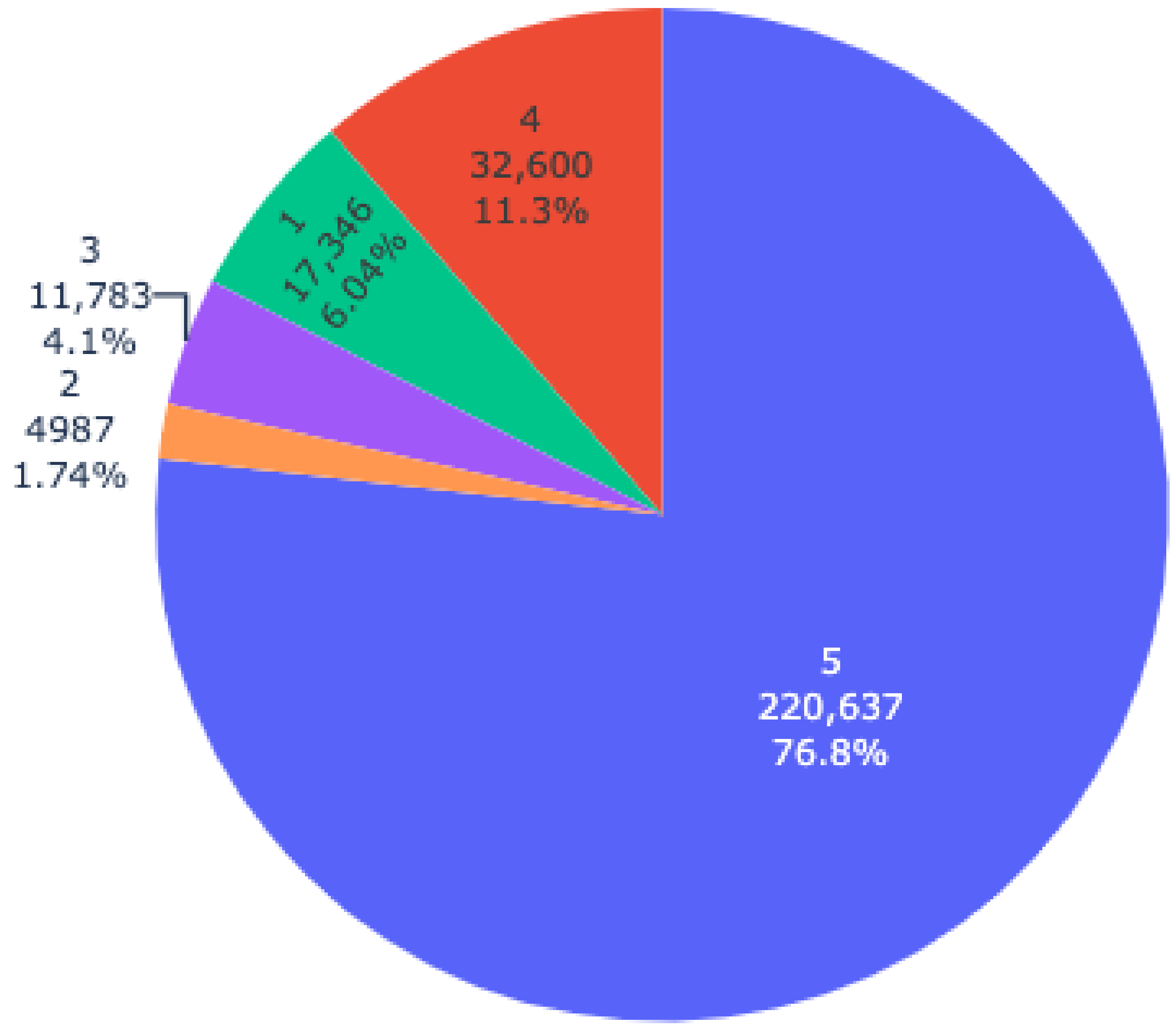
Business Use Cases:

1. **Real-Time Monitoring:** Track sudden changes in user ratings to identify unusual trends.
2. **Bug Detection:** Detect UI issues, algorithm malfunctions, or performance problems through rating patterns.
3. **Rapid Response:** Enable quick action on technical or user experience issues identified from rating trends.
4. **Continuous Improvement:** Drive app enhancements using insights from user ratings.

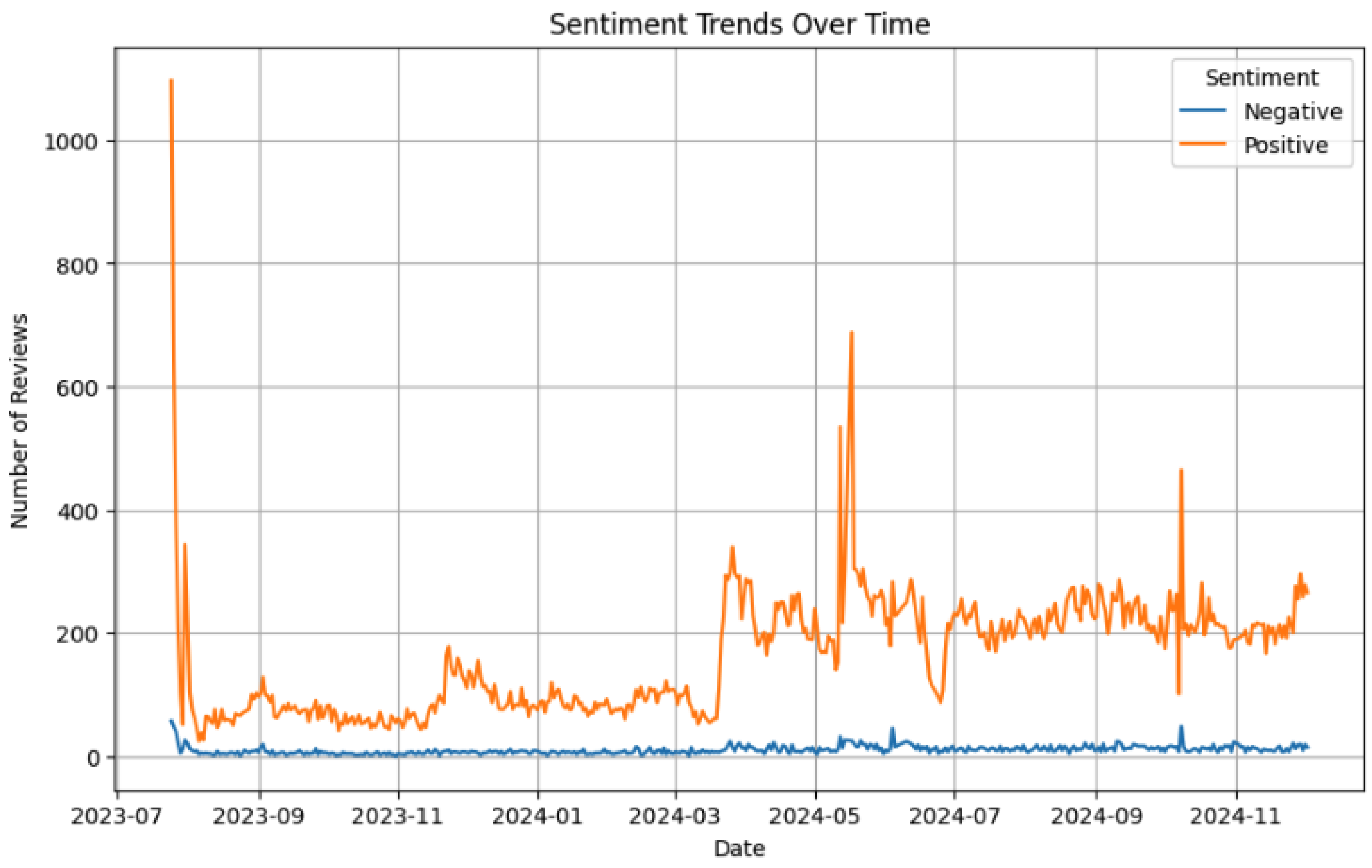


SCORE DISTRIBUTION

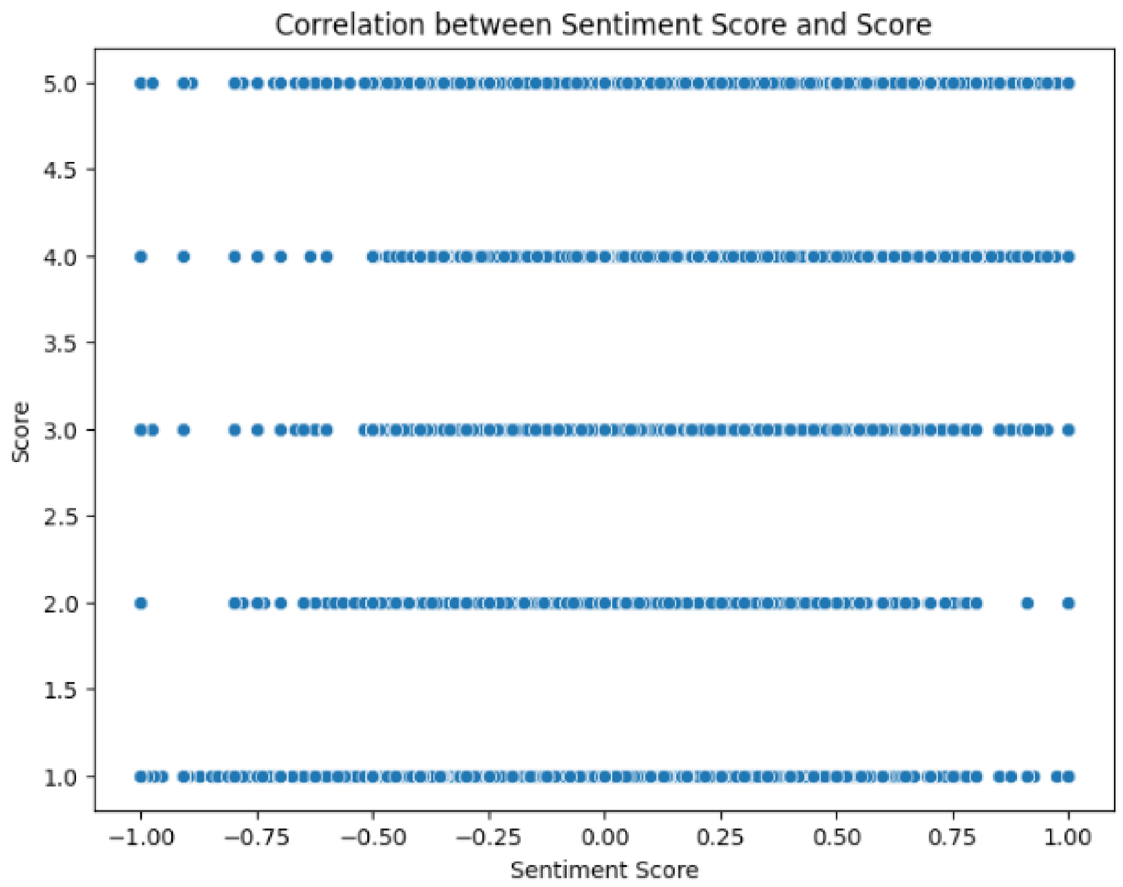
The graph shows a Bar Chart illustrating the **distribution of the scores** in the dataset. **Most reviews receive a score of 5 (very positive)**, highlighting a **significant data imbalance.** This imbalance may bias the model towards predicting higher scores more accurately, while lower scores might be less well-predicted.



SCORE VS SENTIMENT SCORE CORROLATION & TRENDS



```
Correlation coefficient: 0.396
```

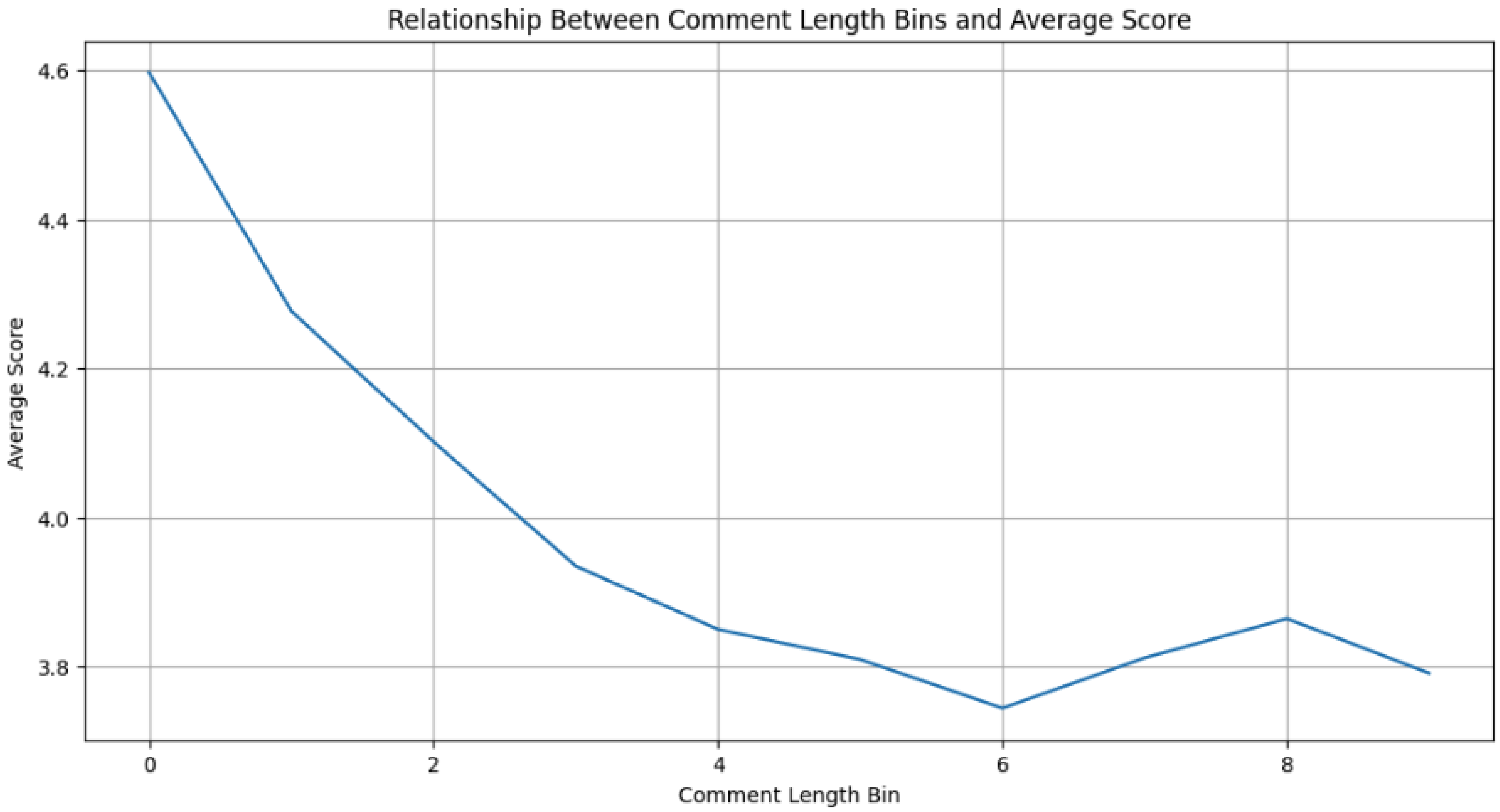


```
Correlation coefficient: 0.39604964315113345
```

Quite surprisingly, although most of the sentiments was positives the correlation between the variable created using the textblob model (sentiment_score) and the target variable can be seen to be relatively **low**.

AVERAGE SCORE VS COMMENT LENGTH

Relationship between the Avg score to the comment length made us believed that this column added in the preprocess will help the model to improve the classifications later



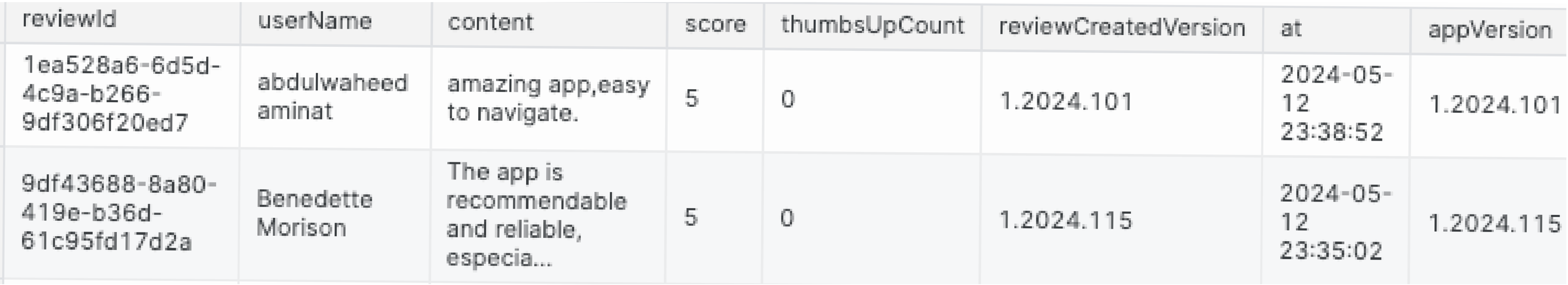
MODEL IMPLEMENTATION

EDA & Preprocess

DataSet info

	# Features	# rows	Target variable
before pre-process	7	287,352	Score (1-5)
after preprocess	12	81,947	Score (1-5)

Raw DataSet



reviewId	userName	content	score	thumbsUpCount	reviewCreatedVersion	at	appVersion
1ea528a6-6d5d-4c9a-b266-9df306f20ed7	abdulwaheed aminat	amazing app,easy to navigate.	5	0	1.2024.101	2024-05-12 23:38:52	1.2024.101
9df43688-8a80-419e-b36d-61c95fd17d2a	Benedette Morison	The app is recommendable and reliable, especia...	5	0	1.2024.115	2024-05-12 23:35:02	1.2024.115

2500 duplicate records were identified and removed.

dropping and adding new features

features dropped	reviewCreatedVersion, thumbsUpCount, appVersion, reviewId, UserName
features added	appVersion_numeric, sentiment,sentiment_score, month, user_reivew_count, comment_length_binned, day_of_week, cleaned_content (using old schoole NLP)

Handel missing values

user Name - 2	**content - 9**	**reviewCreatedVersion - 24915**	**app Version numeric 24915**
removed	removed	Due to a perfect correlation, with the appVersion column the 'reviewCreatedVersion' column was eliminated.	imputing the most frequent value

MODEL IMPLEMENTATION

NLP + Models Implementations

NLP

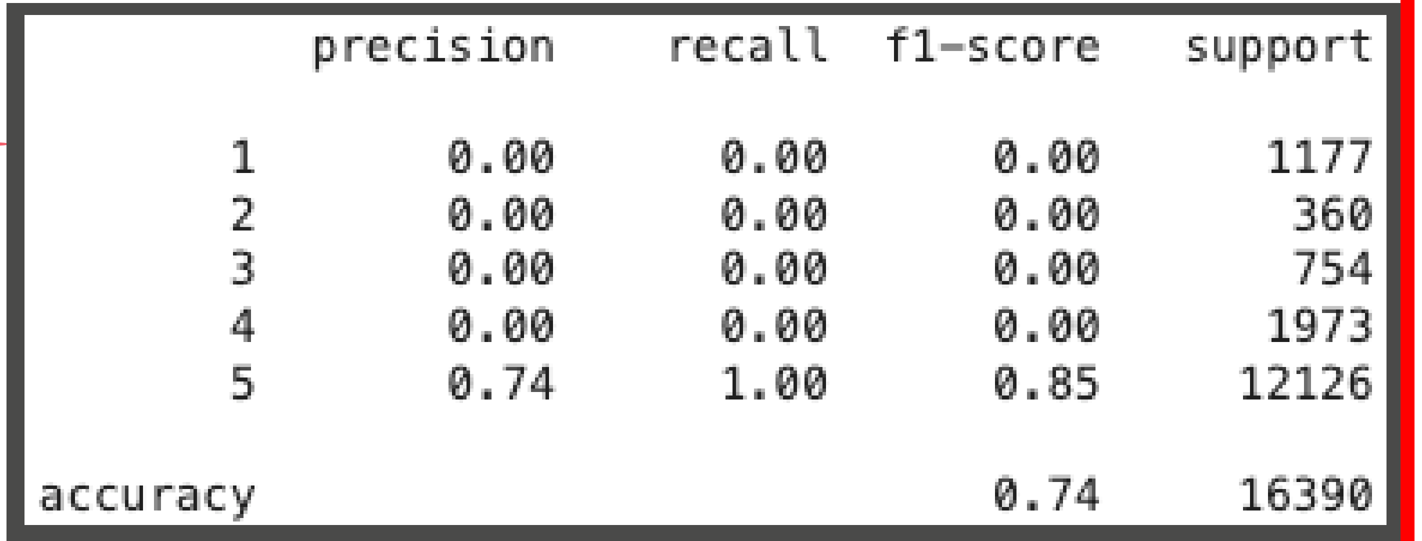
lower case	Done in the preprocessing, seems to be added that is not relevant
remove punctuation	Done, not relevant Keep punctuation marks in comments (short text) it seemed **important to leave exclamation marks (!) so as not to break the intonation.**
remove stop words + other words	Done + Additional words were added to the list of stop words such as '**app**', '**chatgpt**' which appeared with high frequency
stemming	Done, the root of a word is semantically very close in every aspect, so there is no meaning to the grammar, etc., to the task

Models Implementations

- **TextBlob** for adding sentiment and sentiment Score features to empower the untextual data
- **TF-IDF + GradientBoostingClassifier**
- **distilbert-base-uncased** (pre-traind from huggingface) + **GradientBoostingClassifier**
- Some version with xgb / random forests /logeisticregression - **poor results**
- **(almost) fine-tuned distilbert-base-uncased (pre-traind from huggingface) + GradientBoostingClassifier**

MAIN FINDINGS

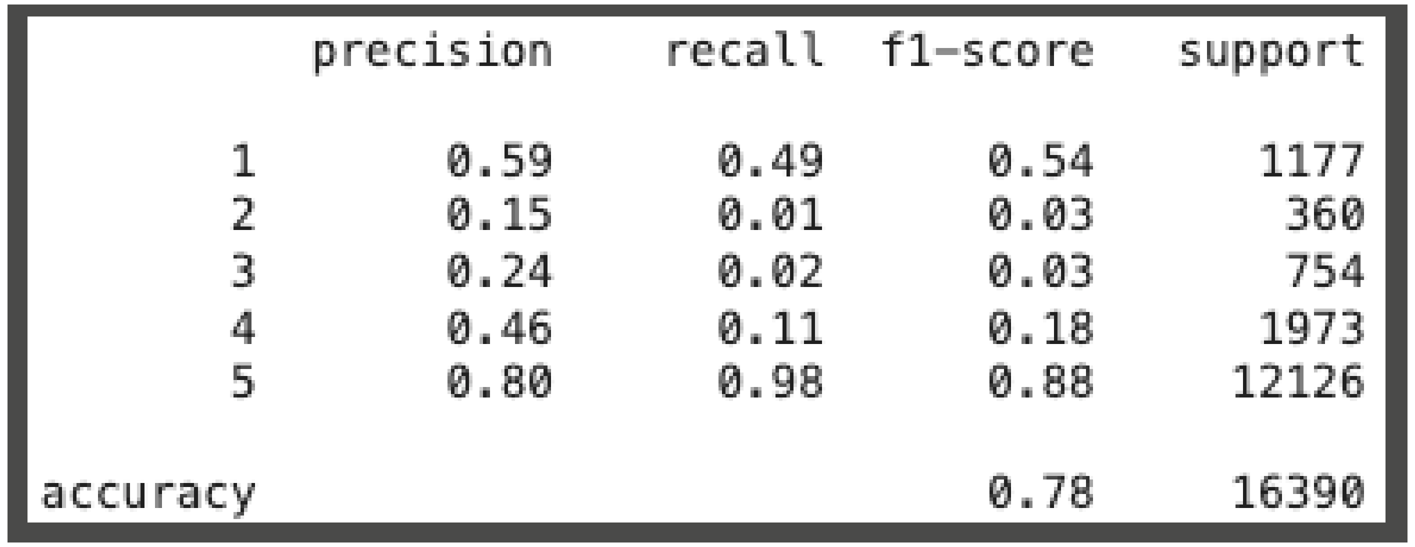
Model Benchmark prediction



```
              precision    recall  f1-score   support

           1       0.00      0.00      0.00      1177
           2       0.00      0.00      0.00       360
           3       0.00      0.00      0.00       754
           4       0.00      0.00      0.00      1973
           5       0.74      1.00      0.85     12126

    accuracy                           0.74     16390
```

*using sklearn DummyClassifier

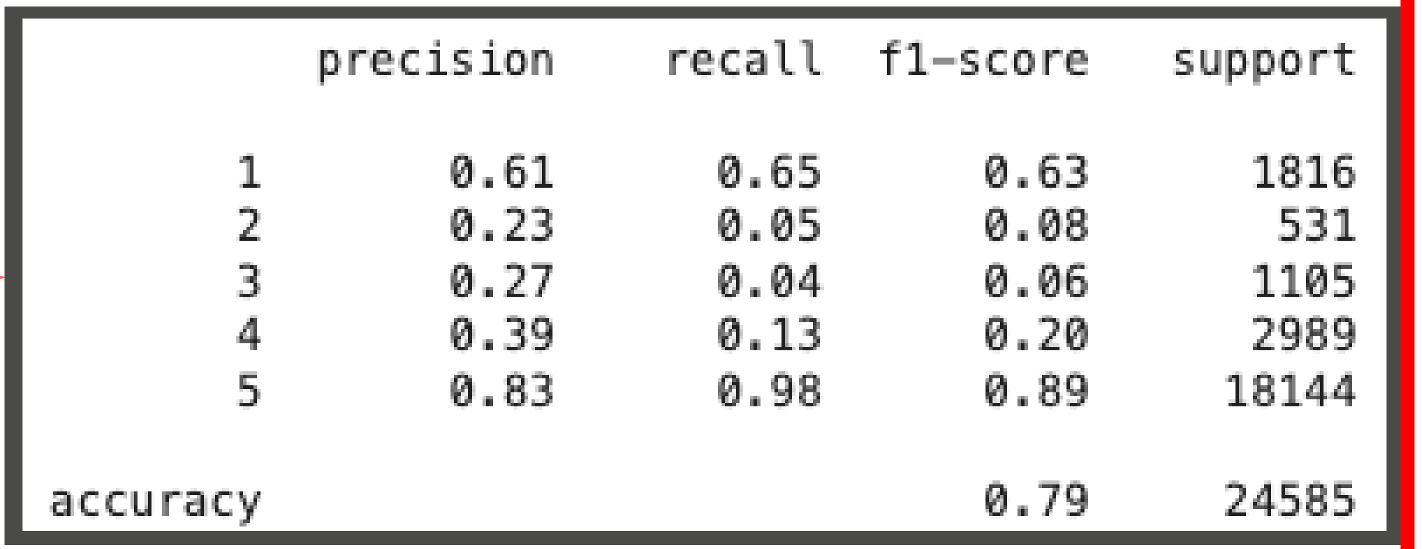
Tfidf + un-textual features



```
              precision    recall  f1-score   support

           1       0.59      0.49      0.54      1177
           2       0.15      0.01      0.03       360
           3       0.24      0.02      0.03       754
           4       0.46      0.11      0.18      1973
           5       0.80      0.98      0.88     12126

    accuracy                           0.78     16390
```

$$\text{Weighted F1-Score} = \frac{11696.02}{16390} \approx 0.71$$

pre trained distilbert + un-textual features

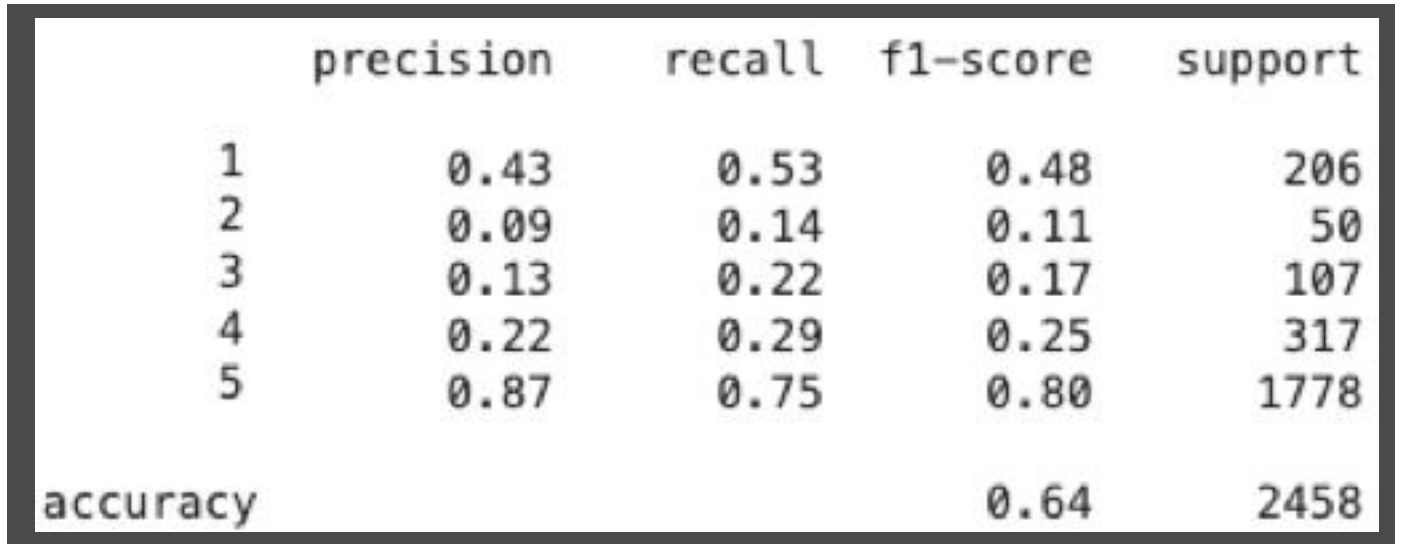


```
              precision    recall  f1-score   support

           1       0.61      0.65      0.63      1816
           2       0.23      0.05      0.08       531
           3       0.27      0.04      0.06      1105
           4       0.39      0.13      0.20      2989
           5       0.83      0.98      0.89     18144

    accuracy                           0.79     24585
```

*using distilbert-base-uncased (ignoring case sensitivity)

$$\text{Weighted F1-Score} = \frac{17998.82}{24585} \approx 0.73$$

pre trained distilbert + un-textual features + SMOTE



```
              precision    recall  f1-score   support

           1       0.43      0.53      0.48       206
           2       0.09      0.14      0.11        50
           3       0.13      0.22      0.17       107
           4       0.22      0.29      0.25       317
           5       0.87      0.75      0.80      1778

    accuracy                           0.64      2458
```

*using distilbert-base-uncased This results made on 10% of the data (time limitations)

$$\text{Weighted F1-Score} = \frac{1624.22}{2458} \approx 0.66$$

CONCLUSION

- The **model's performance measures aligned with the skewed distribution of scores**, performing less well on lower-rated responses due to limited data.

- While trained models were explored, **the simpler TF-IDF approach proved competitive**. (0.71 vs 0.73 - overall f1)

- The **inclusion of supplementary features was crucial for enhancing the model's ability to generalize to unseen data.**

$$\text{Weighted F1-Score} = \frac{\sum_{i=1}^{n}(F1_i \times \text{Support}_i)}{\text{Total Support}}$$

→ Was the best in the pre trained distilbert + un-textual features

Our performance measures will be Weighted F1-Score for overall performance evaluation which is the best avaluation for imbalnced data as we faced here .
Monitor Recall specifically for low ratings (1-2 stars) to ensure you don't miss out on unhappy users. This combination ensures that both detecting and acting on low ratings are prioritized while maintaining balanced performance.

RESEARCH LIMITATIONS

- The dynamic nature of the dataset, **updated daily**, necessitates **frequent retraining of the model to accommodate significant ranking shifts resulting from new version releases**.

- **Limiting our analysis to English** may have introduced a **language bias**, as we missed valuable insights from different linguistic contexts and cultural nuances.

- To ensure timely project completion within the given resource constraints, **we had to reduce the dataset size by approximately 50% (and even up to 90%)**

ANY QUESTIONS?

Thank you
for listening