*CricBuzz Prediction Model*

**Prediction Model Building and Troubleshooting**

After completing the initial sentiment analysis, I proceeded with building a robust prediction model to classify sentiment more effectively. Below is a detailed documentation of the steps taken, challenges faced, and solutions implemented.

*1. Initial Model Training and High Accuracy Issue*

The first models trained were:

- Logistic Regression

- Random Forest

- Support Vector Machine (SVM)

- Naïve Bayes

Each of these models achieved nearly 100% accuracy, which was unexpected and raised concerns about potential data leakage or imbalance.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a machine

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

*2. Checking for Data Leakage and Imbalance*

To investigate the high accuracy, I checked for potential overlap between training and testing sets. It turned out there was an overlap, leading to data leakage. After fixing this, the models still performed exceptionally well, suggesting possible class imbalance.

Steps taken:

- Verified training/test data split to remove overlap.

- Checked class distribution in the dataset and confirmed imbalance.

- Implemented SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset.

A graph of a class distribution

AI-generated content may be incorrect.

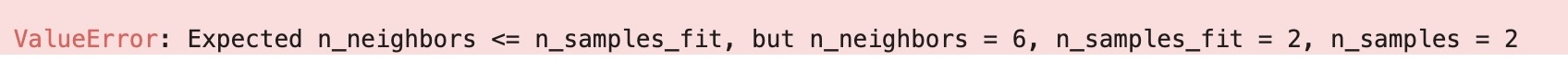


A screenshot of a data table

AI-generated content may be incorrect.

*3. Handling SMOTE Errors*

While applying SMOTE, I encountered the following error:



ValueError: Expected n\_neighbors <= n\_samples\_fit, but n\_neighbors = 6, n\_samples\_fit = 2

Solution:

- Adjusted SMOTE parameters to avoid selecting a number of neighbors higher than available samples.

- Ensured minority class had enough instances to apply SMOTE effectively.

A screenshot of a computer program

AI-generated content may be incorrect.

*4. Re-Training Models and Evaluating Performance*

With a balanced dataset, I retrained the models. The accuracy was more reasonable, with Logistic Regression and SVM still performing exceptionally well, while Naïve Bayes had a drop in accuracy due to its reliance on feature independence.

Results:

- Logistic Regression: ~100%

- Random Forest: ~97%

- SVM: ~100%

- Naïve Bayes: ~69%

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a machine

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

*5. Testing on Unseen Data*

To validate the model further, I tested it on:

- Random unseen text samples with different sentiments.

- Noisy and ambiguous text to check robustness.

Observations:

- The model struggled with highly ambiguous or sarcastic text.

- Performance remained strong for standard sentiment-based sentences.

*6. Hyperparameter Tuning*

I applied hyperparameter tuning using GridSearchCV for all models. The best parameters selected were:

- Logistic Regression: Optimal C value

- Random Forest: max\_depth, min\_samples\_split, n\_estimators

- SVM: Best kernel selection

- Naïve Bayes: Best smoothing parameter (alpha)

*Final Results:*

- Logistic Regression and SVM maintained high accuracy.

- Random Forest improved slightly.

- Naïve Bayes saw a small boost but remained the weakest performer.

*Conclusion*

Through this iterative process, I identified that:

1. Data leakage was causing unrealistically high accuracy initially.

2. Class imbalance required handling via SMOTE to create a fair model.

3. Hyperparameter tuning helped optimize model performance further.

4. The models still showed high accuracy, indicating that the dataset might not be challenging enough.

Going forward, testing on more diverse datasets would help confirm real-world effectiveness.