Luddy School of Informatics, Computing, and Engineering

Predictive Modeling for Multi-Label Tagging in Algorithmic Challenges

Team Members:

Laxmikant Kabra

Meet Palod

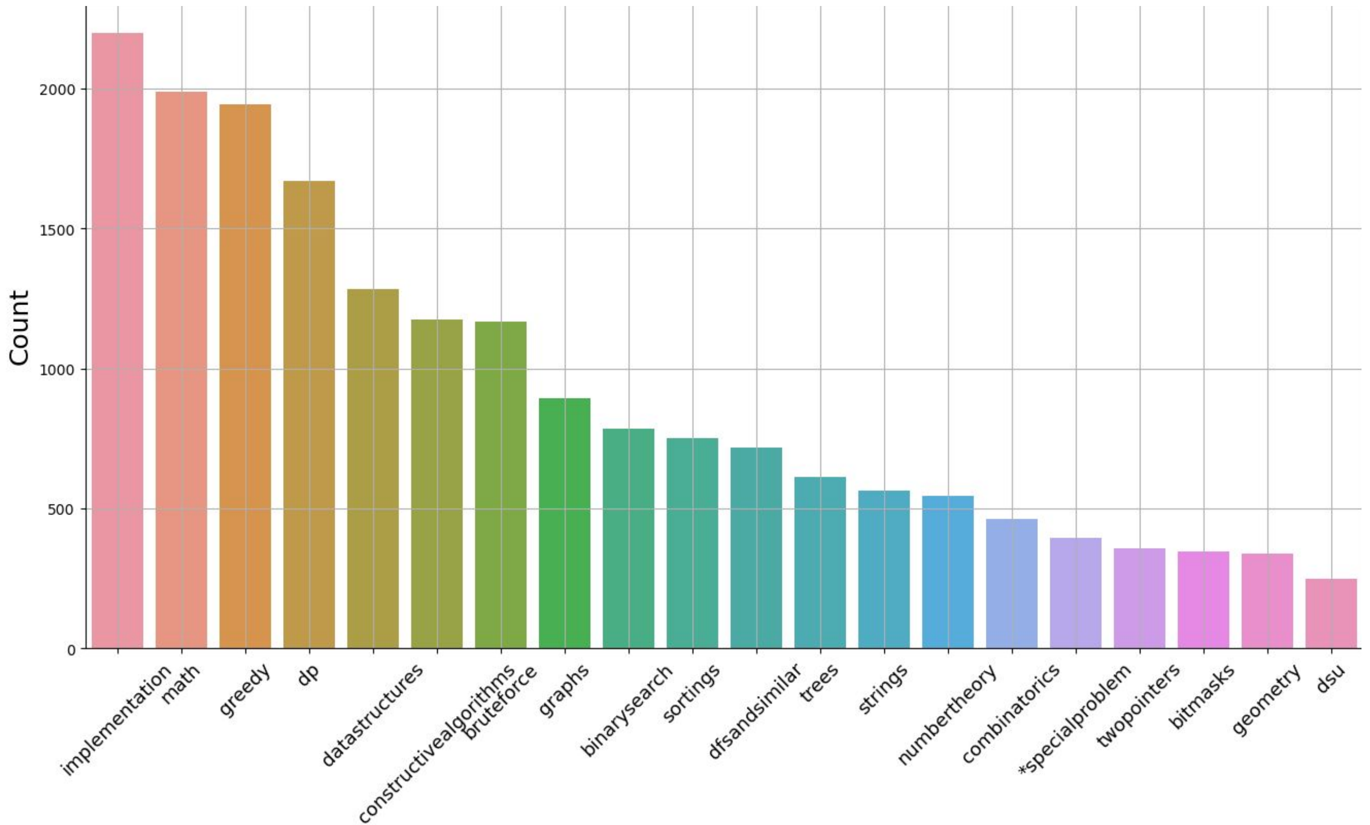
Objective

1. To generate all the tags for a algorithmic question.
2. Explore and compare various Machine Learning and Deep Learning approaches to solve this problem
3. Hyper-parameter optimization to enhance the precision, recall, f1-score, and hamming loss of different models.

About the data

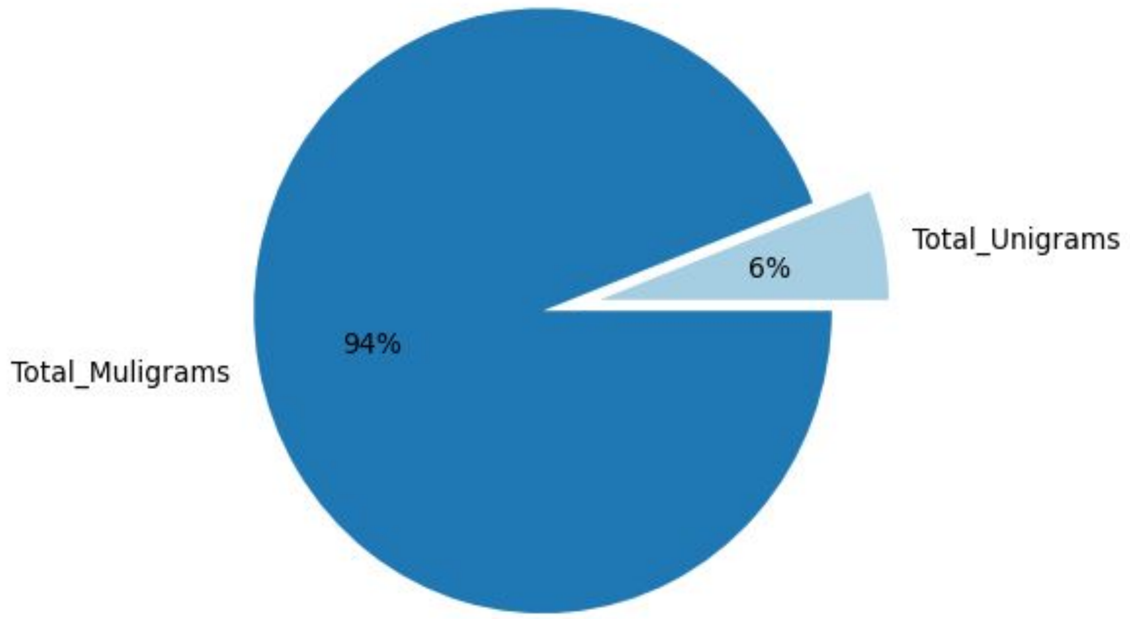
1. Source from Kaggle, it is a comprehensive dataset of contest questions from Codeforces, a popular platform for coding practice.
2. Contains a total of 8434 records, with 4 fields: 'contest' (numerical), 'problemname' (categorical), 'problem-statement' (categorical), 'problems-tags' (categorical, multi-label)
3. Our column of interest are problem statement and problem tags. Problem statement is the full text from the problem page and problem tags are comma-separated tagged classes.

Top 20 Highest occurring Tags



1. Implementation
2. Math
3. greedy
4. dp
5. datastructures
6. constructivealgorithms
7. bruteforce
8. graphs
9. binarysearch
10. sortings
11. dfandsimilar
12. trees
13. strings
14. numbertheory
15. combinatorics
16. specialproblem
17. twopointers
18. bitmasks
19. geometry
20. dsu

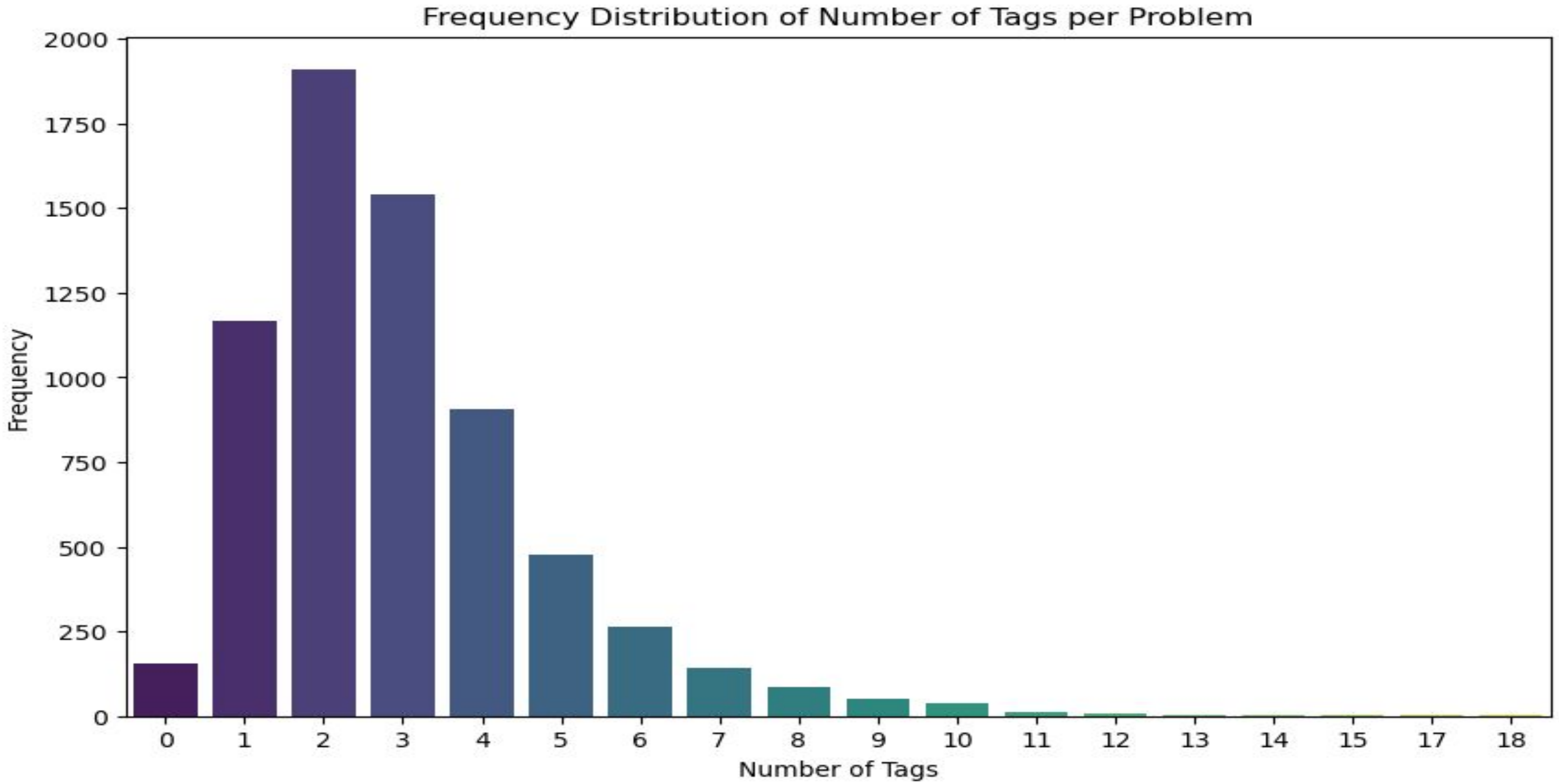
Multi-gram Analysis



Unigram tag contains 6 percent of data, i.e., tag column containing only one tag.

Multi-gram has 91 percent of data, i.e., tags column with more than two or more than two tags.

Frequency Distribution of Number of Tags per Problem



Data Preprocessing

1. Dropped rows without tags and convert text to lowercase.
2. Remove unicodes, HTML tags, and stop words.
3. Tokenize the problem statement.
4. Perform Lemmatization and Stemming.
5. Divide data in train-test split with 20% test size.

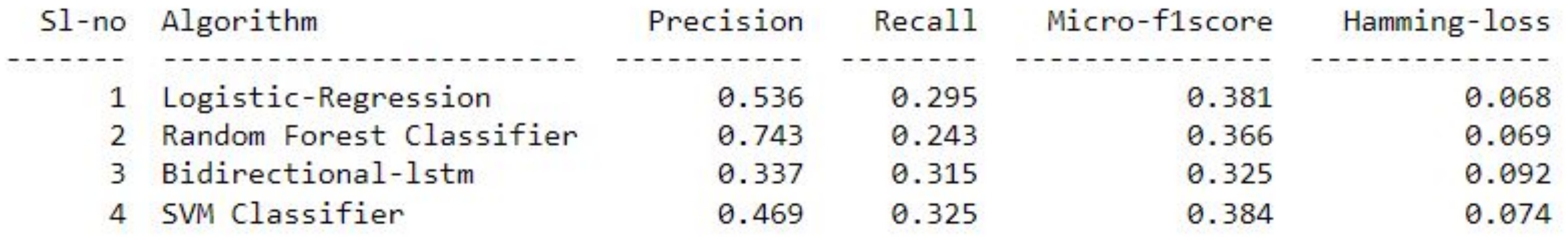
Models and their performance

1. Logistic Regression
2. Random Forest Classifier
3. Bi-LSTM
4. SGD Classifier

Models and their performance

1. Precision - True Positives / (True Positives + False Positives)
2. Recall - True Positives / (True Positives + False Negatives)
3. Micro F1-score - 2 x Precision x Recall / (Precision + Recall)
4. Hamming Loss - 1 - Accuracy

Models and their performance



```
  Sl-no  Algorithm                   Precision    Recall    Micro-f1score    Hamming-loss
-------  ------------------------  -----------  --------  ---------------  --------------
      1  Logistic-Regression             0.536     0.295            0.381           0.068
      2  Random Forest Classifier        0.743     0.243            0.366           0.069
      3  Bidirectional-lstm              0.337     0.315            0.325           0.092
      4  SVM Classifier                  0.469     0.325            0.384           0.074
```

Conclusion

1. The Random Forest Classifier outperformed other models in precision, recall, making it the most suitable model for this dataset.
2. Logistic Regression has the best recall and hamming loss compared to random forest regressor and lstm.
3. As the dataset is small, machine learning model is performing better than deep learning model.

Future Work

1. Get more data from different platforms to facilitate training.
2. Techniques like oversampling or undersampling can be applied to make the dataset with various tags more balanced.
3. Hyper-parameter optimization to enhance the precision, recall, f1-score, and hamming loss of different models.
4. Explore more complex approaches and architectures like ensemble learning, BERT to get better results.