Recommendation Engine

Trevor Judd, Maria McConkey, Kristen Patterson

Introduction

Online judge platforms for programming problems need an effective recommendation engine to be able to recommend new engaging and challenging problems to the users. However, it is difficult for online judges to determine how challenging a new problem will be to each user. Recommending problems that are not challenging enough will make a user become unengaged. Recommending problems that are too challenging will make a user want to give up.

Evaluation Metric

The F1 score was used as the evaluation metric for the models.

$$F1score = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Based on the F1 scores of similar models that have been built for this issue, an F1 score of 0.4 has been deemed minimally acceptable for a model to address this issue.

Methodology

Multi-Classification

The objective of the recommendation engine is to predict which category of the range of attempts a user will make on a problem given data about users such as Rank and Submission Count and data about problems such as Points and Tags. The attempts are categorized as such:

Number of Attempts	Attempts Range
1 - 1	1
2- 3	2
4 - 5	3
6 - 7	4
8 - 9	5
≥ 10	6

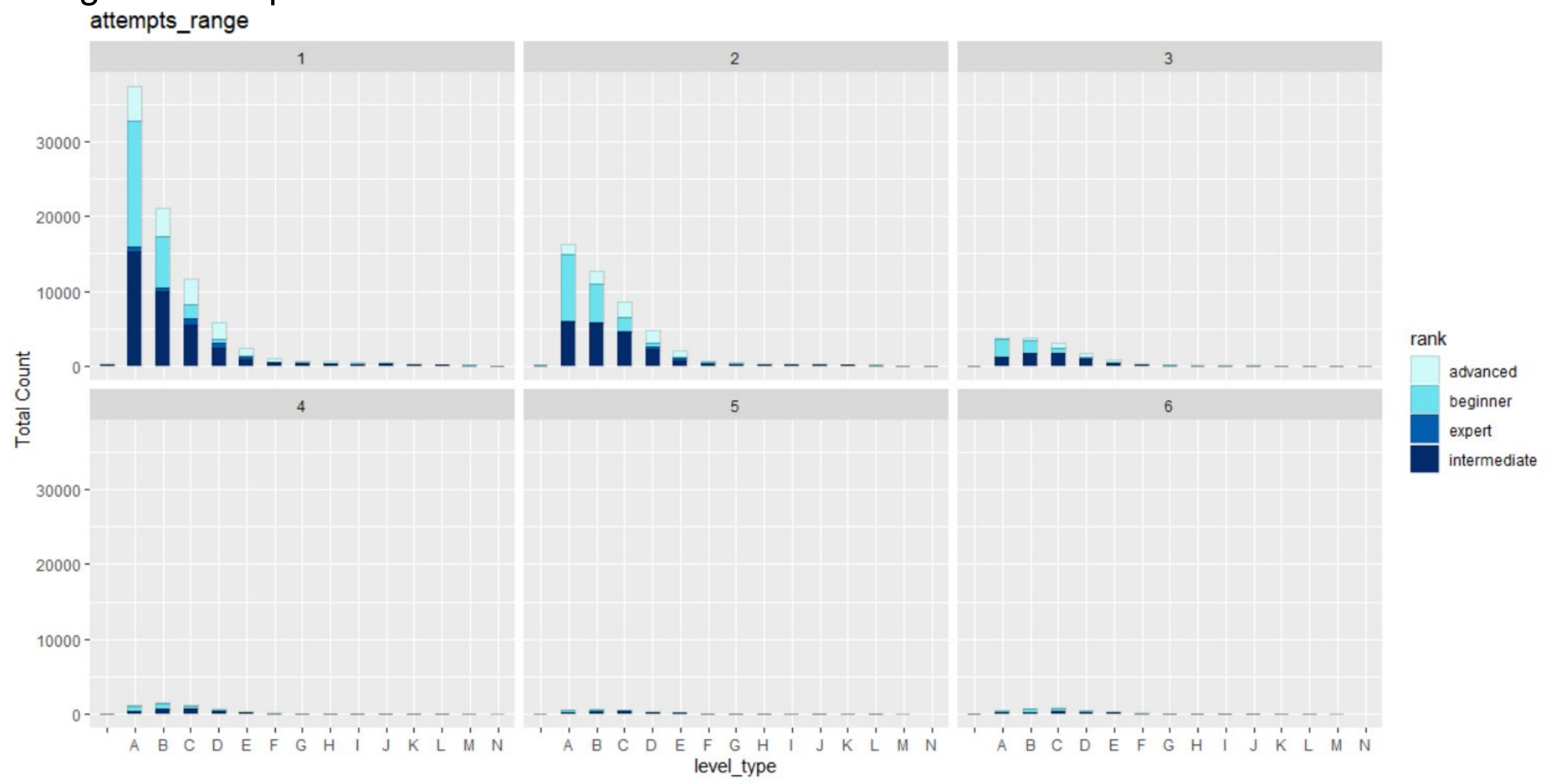
Data Analysis

Through graphical and numerical means, variables were explored to determine strong predictors in relation with Attempts Range.

Results of Data Analysis

Data Analysis

From the exploration of the data, Rank and Level Type have been determined to be strong predictors of the range of attempts.



Techniques

Random Forest

Determine which features hold the most predictive power and generate the lowest Out-of-Bag error rate by creating several uncorrelated decision trees. The result of this produces a more accurate prediction than a single decision tree when it performs its classification.

10-Fold Cross Validation

Confirm that the Random Forest Model is as accurate as the model suggests. To ensure overfitting of the model has not occured, the data set is broken into ten parts, and the model is applied with one part being the test set and the others, the training sets. For better results, this process can be repeated multiple times.

K-Nearest Neighbors

Compare labeled data to unlabeled data to designate labels to the unlabeled data using majority vote of k similar label objects. Similarity is decided by Euclidean Squared Distance:

$$d\left(\left(x_1, \ldots, x_n\right), \left(y_1, \ldots, y_n\right)\right) = \left(x_1 - y_1\right)^2 + \ldots + \left(x_n - y_x\right)^2$$

Results of Modeling

Random Forest - F1 Score = 0.403

level_type, points.imp.mean (NAs replaced by the average of all the known points), rank, and submission_count produced the lowest Out-of-Bag estimate error rate of 46.5%.

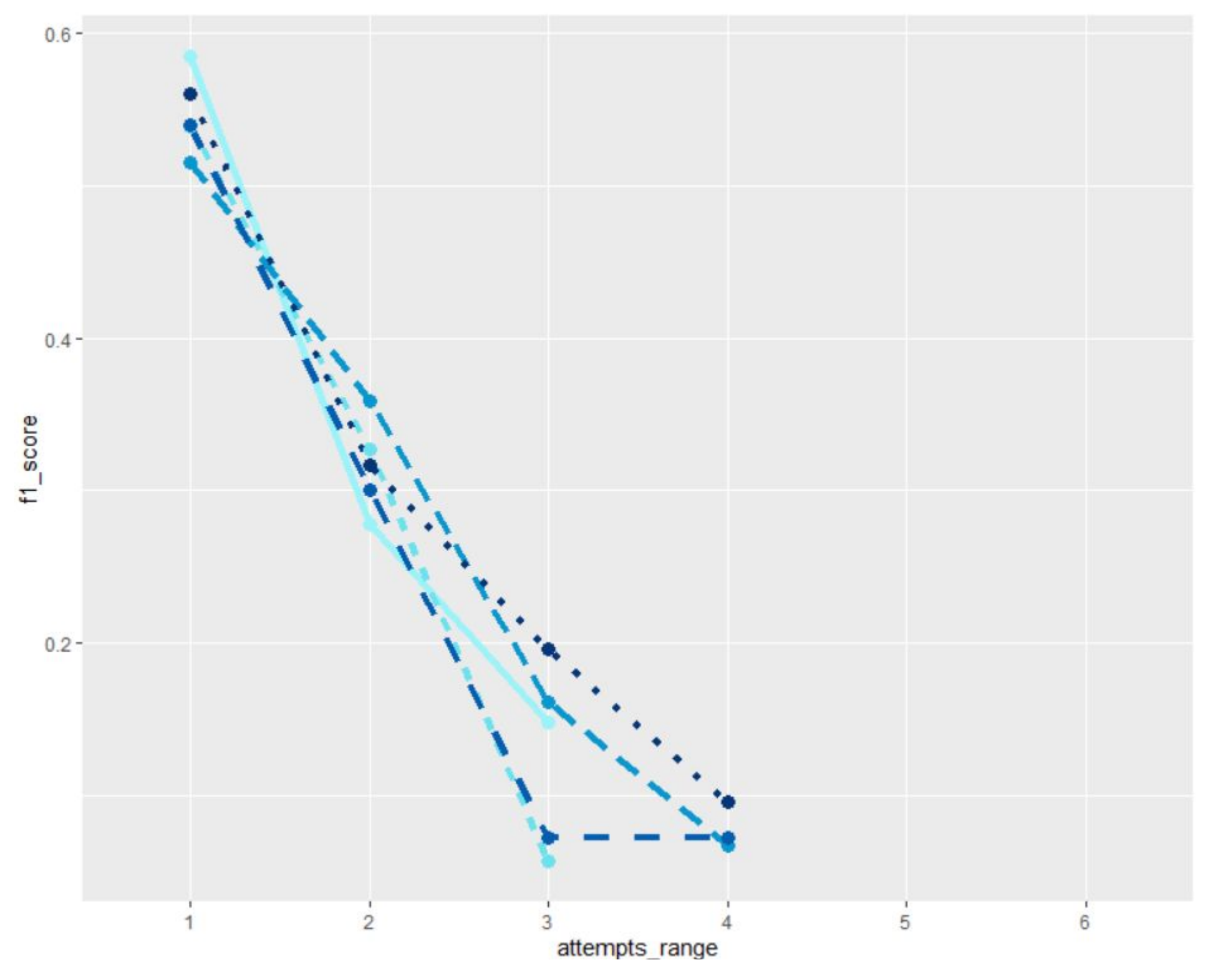
10-Fold Cross Validation

Due to the size of the data set and the lack of necessary hardware means (even with trimming to less than 10% of the original data set), the Cross Validation could not be performed.

Results of Modeling Cont'd

K-Nearest Neighbors - F1 Score = NA

Due to the lack of necessary hardware means, proper exploration of this model could not be completed. However, using a randomly selected 1% of of the original data and k=1 indicate that this modelling technique could be useful if proper hardware is available. With numeric_level_type (level_type converted to numbers; e.g. A - 1, B - 2, etc.) and points.imp.mean, repeat randomly selecting 1% of the data and performing K-NN five times gives a little insight as to how this model can predict correct labels.



Conclusions

Random Forest proves to be a good model, but some improvement as for modeling this data is still needed. 10-Fold Cross Validation could indicate the issues with the Random Forest model but could not performed without necessary hardware means. Like 10-Fold Cross Validation, K-Nearest Neighbors also requires better hardware, but it shows potential with a sample of the data.

Future Work

Better F1 Scores could possibly be achieved if other modelling techniques such as Multinomial Logistic Regression and K-Means were explored as well as obtaining the necessary hardware to execute the models.

Resources

https://datahack.analyticsvidhya.com/contest/practice-problem-recommendation-engine/#

We are thankful for our professor, Dr William Eberle, who taught us the fundamentals of data science. We are also grateful to Tennessee Tech for the change to learn and participate in events as this.