Machine Learning Through Natural Language Processing With Disaster Tweets

Anuradha Tiru-Narayanan, Gavyn Gallagher, Jhoseline Vasquez, and Venkat Rajagopal

Northwestern University

MSDS 422: Practical Machine Learning

Introduction

Our company, XYZ, allocated our team to evaluate a dataset consisting of tweets that may contain content about disasters. Our goal is to accurately identify the text in the tweets that mention a disaster through machine learning techniques. The training dataset consists of 7613 tweets while the test dataset holds 3263 tweets.

TF-IDF Model #1

Developed this mode using the *TextVectorization* with the output parameter as *tf-idf*. Used *Adam* optimizer in addition to implementing *earlystopping* and *reducelearning* rate callback methods for optimal model performance. Also used *stratifiedkfold* with *epoch* size of 100 that produced accuracy scores of .78 and .79 after execution of each fold. The submission of test data in Kaggle returned a score of 0.78884.

RNN Model #2

Developed model 2 using SGD optimizer, tanh and softmax, as the parameters and also categorical_crossentropy and learning rate .001. Achieved the best accuracy for this model of 0.42966 with the training data set and loss rate of 3.9065. Running the prediction of the test data set produced the test accuracy of 0.43. Achieved kaggle score of .4296 on the test data.

RNN Model #3

Adagrad was used as the optimizer in this model, meaning low learning rates are applied to frequently occurring features while high learning rates are applied to uncommon features. Three *Dense layers* of 32, 64, and 128 with a dropout rate of 0.2 were incorporated in the RNN model. *earlystopping* and ReduceLROnPlateau were given a *patience* of 20 & 10. Using a *stratifiedkfold* of 5 combined with *epoch* size = 20, the best accuracy resulted: .79. This RNN model generated a Kaggle score of .79068.

RNN Model #4

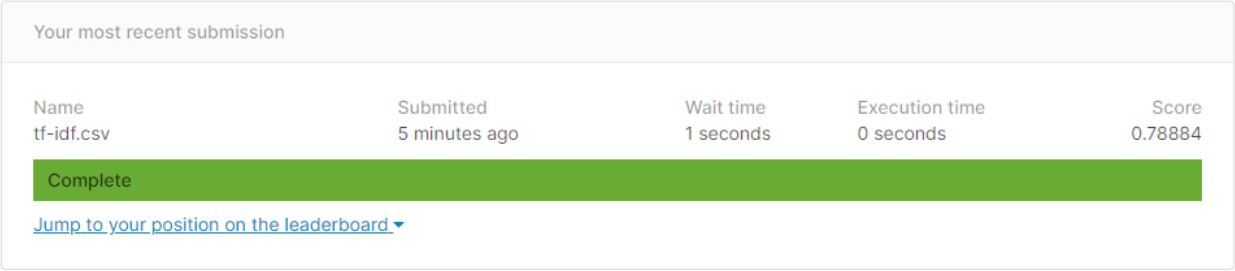
Data cleansing and preprocessing was performed to the text data prior to modeling. We removed symbols, numbers, stop words, and converted all text to lowercase. This model is similar to model #1 except we used the output parameter as multi-hot. After using *stratifiedkfold* with *epoch* size of 100, the accuracy scores improved to .79 and .80 after execution of each fold. The submission of test data in Kaggle returned the best score 0.79589.

Conclusion

Based on the submission responses from Kaggle it is determined that model #4 produced the best results.

Appendix

Model-1 TF-IDF w/Adam optimizer



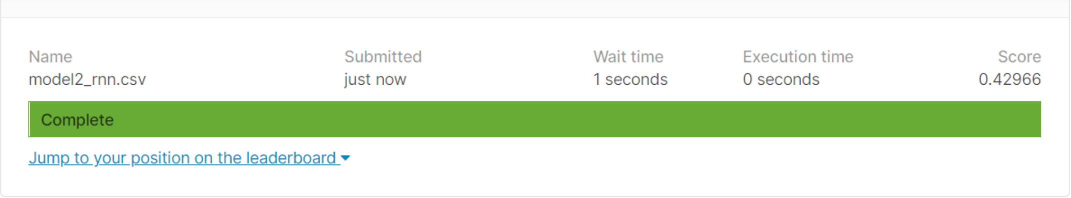
Your most recent submission

Name	Submitted	Wait time	Execution time	Score
tf-idf.csv	5 minutes ago	1 seconds	0 seconds	0.78884

Complete

Jump to your position on the leaderboard

Model 2

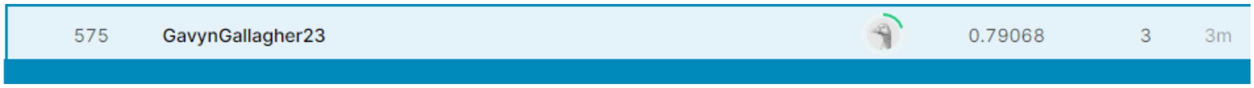


Name	Submitted	Wait time	Execution time	Score
model2_rnn.csv	just now	1 seconds	0 seconds	0.42966

Complete

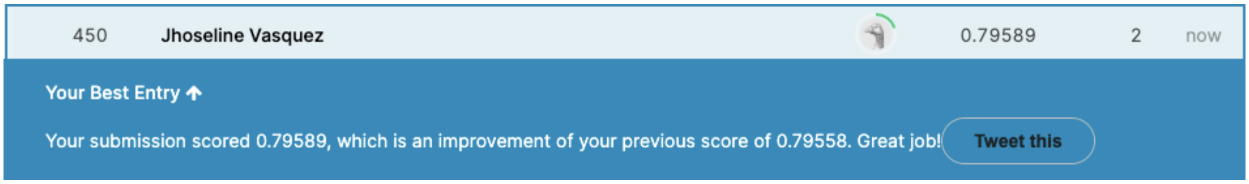
Jump to your position on the leaderboard

Model 3 w/Adagrad optimizer



575	GavynGallagher23	0.79068	3	3m

Model 4 multi-hot w/Adam optimizer

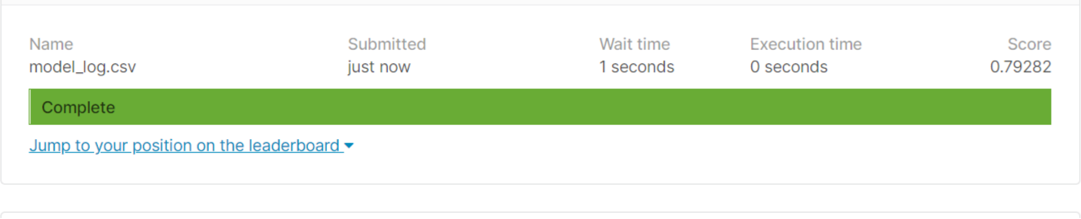


450	Jhoseline Vasquez	0.79589	2	now

Your Best Entry

Your submission scored 0.79589, which is an improvement of your previous score of 0.79558. Great job!

Extra model



Name	Submitted	Wait time	Execution time	Score
model_log.csv	just now	1 seconds	0 seconds	0.79282

Complete

Jump to your position on the leaderboard