ML project final demo

Team MaRVel

Contributions

The collaboration was so intensive and involved that exclusive contributions cannot be detailed.

Preprocessing

1. Tried Word2Vec with TF IDF:
   a. With “ngram=(1,3)”: We waited for around 5 hours but it could process only upto 90,000 rows. So, we gave up.
   b. With default value of “ngram” i.e. (1,1): gave poor score.

2. optimized hyperparameters

3. Removed the common words from sentences of minority and majority class. The score was around 0.6, which couldn’t beat our high score.

4. Implemented a “sequential model” but each epoch was taking about 4.5 hours and for effective results we needed to have at least 5 epochs. So, we couldn’t pursue it further.

Additional models

a. RandomForest with best result for tree_depth 25 (score around 0.4)

b. AdaBoost with best result for “n_estimators = 500” (score around 0.5)

Note:

We tried each training model for the data, which was preprocessed using the hyperparameter:

1. “ngram=(1,3)” of the vectorizer functions: the models were either taking too long or they gave poorer results as compared to logistic regression model. Why we are so adamant about ngram being equal to (1,3) and not (1,2), (1,4) etc.; has been explained in the project report.

2. “ngram=(1,1)”(default value)) of the vectorizer functions: All the models(except SVM with kernels) gave results. However, the scores were poor and weren’t helpful. SVM with polynomial and rbf kernels took too long to execute.

Additional models

Models that gave results using “ngram=(1,3)” but the score was poor as compared to logistic regression:

a. Bernoulli Naive Bayes
b. Perceptron
c. Linear SVC
d. XG Boost

Models that took too long to execute, using “ngram=(1,3)”:

a. SVM with polynomial and rbf kernels
b. Random forest model
c. Adaboost

Another attempt to best score

Used Bag of Words with main parameter - ngram_range, which improved our score.

ngram_range = (1,4)

Model - Logistic Regression with threshold = 0.132