Help Twitter Combat Hate Speech Using NLP and Machine Learning.

Project 2

DESCRIPTION

Using NLP and ML, make a model to identify hate speech (racist or sexist tweets) in Twitter.

Problem Statement:

Twitter is the biggest platform where anybody and everybody can have their views heard. Some of these voices spread hate and negativity. Twitter is wary of its platform being used as a medium to spread hate.

You are a data scientist at Twitter, and you will help Twitter in identifying the tweets with hate speech and removing them from the platform. You will use NLP techniques, perform specific cleanup for tweets data, and make a robust model.

Domain: Social Media

Analysis to be done: Clean up tweets and build a classification model by using NLP techniques, cleanup specific for tweets data, regularization and hyperparameter tuning using stratified k-fold and cross validation to get the best model.

Content:

id: identifier number of the tweet

Label: 0 (non-hate) /1 (hate)

Tweet: the text in the tweet

Tasks:

1. Load the tweets file using read_csv function from Pandas package.
2. Get the tweets into a list for easy text cleanup and manipulation.
3. To cleanup:
    1. Normalize the casing.
    2. Using regular expressions, remove user handles. These begin with '@'.
    3. Using regular expressions, remove URLs.
    4. Using TweetTokenizer from NLTK, tokenize the tweets into individual terms.
    5. Remove stop words.
    6. Remove redundant terms like 'amp', 'rt', etc.
    7. Remove '#' symbols from the tweet while retaining the term.
4. Extra cleanup by removing terms with a length of 1.
5. Check out the top terms in the tweets:
    1. First, get all the tokenized terms into one large list.
    2. Use the counter and find the 10 most common terms.
6. Data formatting for predictive modeling:
    1. Join the tokens back to form strings. This will be required for the vectorizers.

    2. Assign x and y.
    3. Perform train_test_split using sklearn.
7. We'll use TF-IDF values for the terms as a feature to get into a vector space model.
    1. Import TF-IDF vectorizer from sklearn.
    2. Instantiate with a maximum of 5000 terms in your vocabulary.
    3. Fit and apply on the train set.
    4. Apply on the test set.
8. Model building: Ordinary Logistic Regression
    1. Instantiate Logistic Regression from sklearn with default parameters.
    2. Fit into the train data.
    3. Make predictions for the train and the test set.
9. Model evaluation: Accuracy, recall, and f_1 score.
    1. Report the accuracy on the train set.
    2. Report the recall on the train set: decent, high, or low.
    3. Get the f1 score on the train set.
10. Looks like you need to adjust the class imbalance, as the model seems to focus on the 0s.
    1. Adjust the appropriate class in the LogisticRegression model.
11. Train again with the adjustment and evaluate.
    1. Train the model on the train set.
    2. Evaluate the predictions on the train set: accuracy, recall, and f_1 score.
12. Regularization and Hyperparameter tuning:
    1. Import GridSearch and StratifiedKFold because of class imbalance.
    2. Provide the parameter grid to choose for 'C' and 'penalty' parameters.
    3. Use a balanced class weight while instantiating the logistic regression.
13. Find the parameters with the best recall in cross-validation.
    1. Choose 'recall' as the metric for scoring.
    2. Choose a stratified 4 fold cross-validation scheme.
    3. Fit into the train set.
14. What are the best parameters?
15. Predict and evaluate using the best estimator.
    1. Use the best estimator from the grid search to make predictions on the test set.
    2. What is the recall on the test set for the toxic comments?
    3. What is the f_1 score?