Project: Text Classification Competition

Team: Subhasish Bose (sbose4) and Soumya Kanti Dutta (skdutta2)

This documentation is created during CS 410: Text Information System Final Project and it contains the details of the project.

Table of Contents

Introduction

As final project for CS 410 Text Information System, we participated in Text Classification Competition to detect Twitter Sarcasm. We were given both Training and Test datasets.

The Training dataset Content

label: SARCASM or NOT_SARCASM

response: The Tweet to be classified

context: The conversation context of the response

example:

{"label": "SARCASM", "response": "@USER @USER @USER I don't get this .. obviously you do care or you would've moved right along .. instead you decided to care and troll her ..", "context": ["A minor child deserves privacy and should be kept out of politics . Pamela Karlan , you should be ashamed of your very angry and obviously biased public pandering , and using a child to do it .", "@USER If your child isn't named Barron ... #BeBest Melania couldn't care less . Fact . 💯"]}

The Test dataset content

id: String identifier for sample. This id is required for project submission and grading.

response: The Tweet to be classified

context: The conversation context of the response

example:

{"id": "twitter_1", "response": "@USER @USER @USER My 3 year old , that just finished reading Nietzsche and then asked me : \" ayo papa why these people always trying to cancel someone on Twitter , trying to pretend like that makes them better themselves ? \" . To which I replied \" idk \" , and he just \" cuz hoes mad \" . Im so proud . <URL>", "context": ["Well now that \u2019 s problematic AF <URL>", "@USER @USER My 5 year old ... asked me why they are making fun of Native Americans ..", "@USER @USER @USER I will take shit that didn't happen for $ 100", "@USER @USER @USER No .. he actually in the gifted program and reads on second grade level . ... and he knows Kansas City is in Missouri"]}

Dataset size statistics

Train	Test
5000	1800

Project Objective

Our project objective is to learn from the Training dataset and predict the labels of Test dataset (SARCASM or NOT_SARCASM).

Approach and Workflow

Data Preprocessing and Feature Engineering

- First, we read the Training and Test Data of jsonl format to Pandas data frame.
  `Function: read_jsonl_to_dataFrame`

- Then we applied the following data cleaning and feature engineering steps on the Training and Test Data.
  `Function: simple_feature_engieering_and_data_cleansing`

1) Combined Response and Context Tweets in the data frame both in training and test data.
2) Converted the dataset to lower case.
3) Got rid of '@USER', '<URL>', Web URL Links, Hashtags.
4) Next, we got rid of stop words. We used `nltk.corpus.stopwords` for this purpose.
5) We removed the emojis as well.
6) We removed all punctuations and special characters.
7) Lastly, we stripped each word to get rid of additional 'space'.

- We also used `sklearn.feature_extraction.text.TfidfVectorizer` to incorporate additional feature engineering with the following parameters:
  - `max_features =20000`
  - `min_df=1`
  - `max_df=0.5`
  - `binary=1`
  - `use_idf=1`
  - `smooth_idf=1`
  - `sublinear_tf=1`
  - `ngram_range=(1,3)`

Training Models

We have tried the following algorithms on training data.

a) Linear SVC
b) Naïve Bayes
c) Logistic Regression
d) Random Forest
e) Neural network – BERT

Among these Logistic Regression provided us the best performance. So, we designed our final code with Logistic Regression. We used `sklearn.linear_model.LogisticRegression`, with the following parameters.

- `class_weight='balanced'`
- `solver='newton-cg'`
- `C=1`

Validation of Training Data

We got the following performance matrix, doing a train test split of 80/20:

```
Classification Result for Logistic Regression

              precision    recall  f1-score   support

 NOT_SARCASM       0.78      0.72      0.75       519

     SARCASM       0.72      0.78      0.75       481


    accuracy                           0.75      1000
```

```
   macro avg       0.75      0.75      0.75      1000
weighted avg       0.75      0.75      0.75      1000
Overall accuracy for Logistic Regression
 0.75
```

Running Code on Test Data and Leaderboard Score

Once we validated the performance of Logistic Regression on training data, we applied it on Test Dataset.

`Functions: write_prediction_results_in_list and final_prediction_calculation.`

We created `answer.txt` file with test dataset labels which we uploaded for grading. We were able to beat the baseline. We tried with multiple times adjusting the feature vector.

Leaderboard snapshot:

47	subhasishb-coder	30	0.5975869410929737	0.9355555555555556	0.72932005197055	1
48	Soumya	80	0.5989992852037169	0.9311111111111111	0.7290126141800782	1

Contribution

Data Preprocessing – Subhasish

Feature Engineering – Soumya

Model Training – Soumya

Validation and Adjustment of feature vector – Subhasish

Setup and Usage Instructions

Software Dependencies

- Python==3.8.3
- nltk==3.5
- pandas==1.0.5
- scikit_learn==0.23.2

Setup and Usage Instructions

1) Apply ‘pip install requirements.txt’
2) Make sure ‘data’ folder has train and test jsonl files
3) Apply ‘python TestClassficationCompetion_Sarcasm_Detection.py’
4) answer.txt file will be created in the same directory

References

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

https://towardsdatascience.com/sarcasm-detection-step-towards-sentiment-analysis-84cb013bb6db