The following project will go over the applications of Machine Learning (coded with Python) and a model of Sentiment Analysis applied on Tweets.

IS 6713 - Data Foundations:

Final Project

Sentiment Analysis

IS 6713 - Data Foundations
Final Project Report
December 2nd, 2021
MSDA – Fall 2021

Team:
Nicholas Anderson
Diego Aldo Pettorossi
Aranzazu Chaparro Molina

Introduction

Throughout the course (IS – Data Foundations) we learned the basics of Python, and then we moved to the applications of it. While the applications are many, including applying mathematical formulas and reading or creating files, one of the main applications is Machine Learning (supervised and unsupervised learning. The objective of supervised learning is to make predictions based on annotations contained in a given dataset

For our final project, we were requested to create an offensive language detection model.

Dataset information

We were provided with two data sets: Train.tsv (10,591 rows) and Test.tsv (2,647 rows).
Each data set consists of 3 columns with: ID – Tweet (text) – Classification.

Classifications for each Tweet: NOT, TIN, and UNT.

- **Not Offensive (NOT):** Posts that do not contain offense or profanity
- **Targeted Insult (TIN):** Posts containing insult/threat to an individual, a group, or others
- **Untargeted (UNT):** Posts containing nontargeted profanity and swearing. Posts with general profanity are not targeted, but they contain non-acceptable language.

Both the Test.tsv and Train.tsv already have annotations and place, but our model should only train in the “Train” Annotations and substitute the Test Annotations with our Predictions.

	86426	@USER She should ask a few native Americans what their take on this is.	UNT
1	90194	@USER @USER Go home you're drunk!!! @USER #MAGA #Trump2020 👊us👊 URL	TIN
2	16820	Amazon is investigating Chinese employees who are selling internal data to third-party sellers looking for a…	NOT
3	62688	@USER Someone should'veTaken" this piece of shit to a volcano. 😂"	UNT

Model and Features

For our Machine Learning model, as we learned during the course, we will use the scikit-learn approach, and we will import the modules needed from the sklearn package. We will start using the LinearSVC model.

Steps

1. **Load in the Data**

The data was loaded row by row and appended to a list. Both datasets contained a column with a tweet identification number, the tweet as a string, and then a prediction. This process was applied to both Train.tsv and Test.tsv files.

Once the data was loaded in, we split the train.tsv data into a Training and Test set. This allowed the algorithm to train on the dataset with actual meaningful predictions and benchmark the effectiveness of those predictions.

2. Run Initial Model (no Features added)

We decided to run the LinearSVC model with no features added to be able to decide whether the features we add are improving the model or not. LinearSVC is generally useful for attempting category prediction.

```
C:\Users\User1\anaconda3\lib\site-packages\sklearn\svm\_base.py:985: ConvergenceWarning: Liblinear failed to c
onverge, increase the number of iterations.
  warnings.warn("Liblinear failed to converge, increase "
Initial Validation F1: 0.7254
Initial Precision: 0.4664
Initial Recall: 0.5541
Initial F1: 0.4717
```

3. Features Engineering

The next step was to add extra features, which we defined to improve the model.

1) Find names (name detection)

The name detection feature aims to determine whether a tweet has a specific target or not. We used REGEX for this purpose.

Names detection involved looping three steps for each tweet:

- Compute the number of sentences.
- Compute the number of capitalized words.
- Comparison: if the number of capitalized words if greater of the number of sentences, we detect a name.

2) Exclamation Marks

We added the number of exclamations marks because we think it is positively related to aggressive language.

The count of exclamation marks was obtained looping over the dataset, tweet by tweet and checking whether they appeared or not.

3) Ellipses

Same as the Exclamation Marks, we looped over each tweet to find Ellipses.
Even when the Ellipses might not suggest a positive or negative intent, we believe usually when ellipses are added they might have an opinion or feeling behind it, and therefore an intention. We believe that it is a good predictor to if a sentiment is in the tweet and train over the other tweets that have them.

4) Profanity Words (list)

We looped over a profanity words list, so that we could get a “bad_word” label over the words that matched with the tweets.
We choose this list over the “Negative words” we saw on the classes’ models before, because this list contains words that have the purpose to offend, which will help us with predicting properly between the TIN and the UNT categories.

4. Load Features (#1 , #2, #3)

After generating the features, we stored me in lists of lists, so we could convert them into arrays.
We vectorized the tweets using .fit_transform and .transform methods, so we merge the arrays one by one, using the “hstack” in a cumulative way.
As last preparatory step we trained the model with the new dataset containing all the features added (keeping the other parameters unchanged).

Analysis

5. Compare F1 between Initial Model and Model with Features

F1 – Model with Features Added

```
{'C': 0.51}
Validation F1: 0.7080
Precision: 0.4683
Recall: 0.5230
F1 Micro: 0.6947
F1 Macro: 0.4800

The F1 Macro Score has improved in the new model!
F1 Macro score has improved by 0.0083
```

Our model showed improvements when the new features were added. We proceeded to apply this model to the TEST.tsv to generate our final predictions.

F1 – TEST.tsv file (fake labels)

```
{'C': 0.51}
Validation F1: 0.7080
Precision: 0.2480
Recall: 0.3333
F1 Micro: 0.7440
F1 Macro: 0.2844
```

Interestingly, when we applied the final model to TEST.tsv file, we can see F1 Micro improved, but the F1 Macro did not. This is because the F1 Macro is the accuracy of the model for each class (NOT, TIN, UNT), while the F1 Micro is the accuracy across all the classes. In other words, since most of the tweets are classified as “NOT” and the fake labels are “NOT”, the F1 micro is really high.

False positives

The model detects names in a tweet when the user input a word incorrectly capitalized. For instance, slang initialisms are detected as names, which will cause a false positive in our predictions

False negatives

The model doesn't detect names if the first letter is a name and contains the only capitalized letter in the sentence, which cause a false negative.
Furthermore, profanity or insults are not detected if not explicitly included in the txt we loaded. For instance, some users replace some letters with asterisks, which will cause a false negative.

6. **Add Predictions to TSV using PANDAS**

For a final Analysis we will add the predictions we made to the TEST.tsv file with PANDAS.

7. **Last step was to print some tweets from the train and the test dataset to visually see our predictions VS the model without features and the original annotations.**

Conclusions

Overall, analyzing tweet by tweet, we can see that some of the Ground-Truth Classes don't align with our personal interpretation.

```
Tweet: @USER @USER @USER @USER @USER @USER @USER @USER And when logic FAILS, you aren't a twitterer like ME! I
am sooo special, I have followers!  I believe, I believe!!!! Stop it with your dumb facts!!!!!" MEGY you are e
ntertaining!!!🤣🤣🤣🤣🤣🤣🤣🤣🤣"
Ground-Truth Class: NOT
SVM + Features Added: TIN

Tweet: @USER @USER @USER Where is VAN? She is ARMY who need to shazam the most you know... 🔍🐾🐶🐾🔎
Ground-Truth Class: NOT
SVM + Features Added: NOT

Tweet: @USER why do conservatives have such a hard time testifying?
Ground-Truth Class: NOT
SVM + Features Added: NOT

Tweet: @USER @USER @USER Fk River you are a sick individual. Get a life or make the world a better place or en
d yours. No loss trash IMHO
Ground-Truth Class: NOT
SVM + Features Added: TIN
```

In this case, our model was actually able to detect the "intention to offend" in both tweets, due to the Parameters we defined in our Functions (def). On the other hand, the Ground-Truth Class defined them as NOT. This misalignment had a negative impact on the performance of our model, which still had a good F1 macro score. With that in mind, we think that a better specified standard the Ground-Truth Classes would greatly increase the accuracy in the predictions.