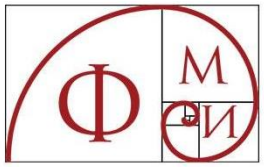
Размити множества
Класификация на сарказъм чрез размити числа

Цветан Цветанов
ф.н. 4MI3400570
Изкуствен интелект

4 февруари 2025 г.



Съдържание

1 Въведение

Анализът на емоции (sentiment analysis) е една от основните задачи в обработката на естествени езици. Сравнително лесно е да се класифицират с голяма точност основните емоции - като щастие, тъга, страх, гняв или изненада. По-трудно е да се категоризира сарказъм. Често смисълът е точно противоположен на буквалното значение на думите. Понякога не е възможно категорично да се определи дали конкретно съобщение е саркастично или не - това зависи от контекста и начина на тълкуване. За такива случаи е неточно да се използва бинарна класификация. Затова моделиране с размити числа е подходящо когато се изследва тази емоция в ежедневното писмено общуване между хора онлайн.

2 Използвани данни

Използвал съм анотирани туитове от Kaggle, разделени на два файла:

filename	tweets
train.csv	67997
test.csv	7994

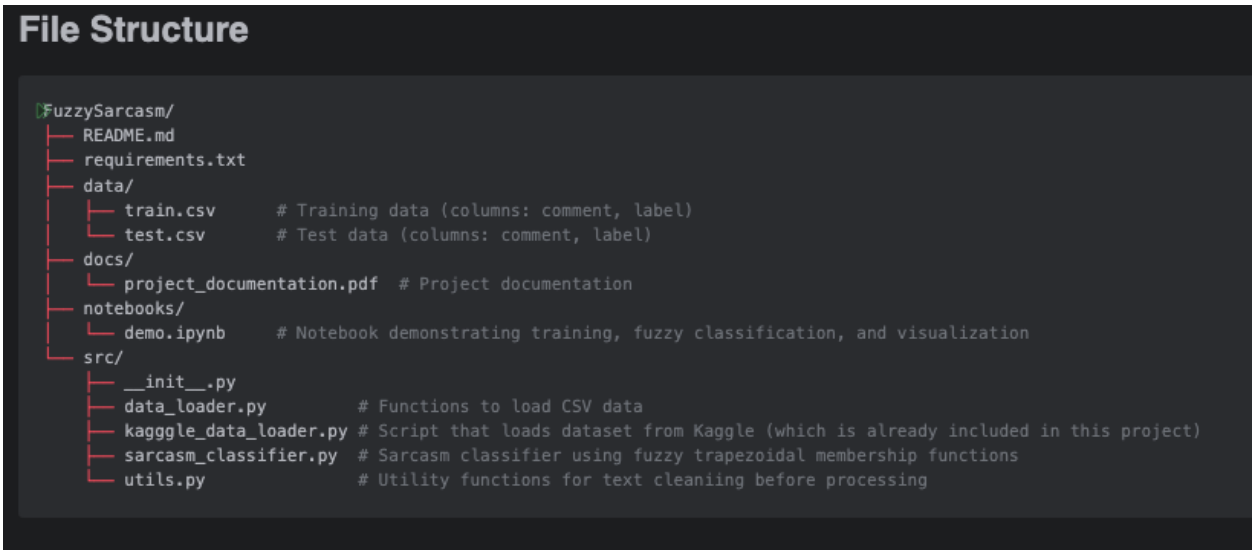
Всеки туит е класифициран в една от четири категори - regular, irony, sarcasm, figurative (sarcasm + irony). Илюстративни стойности (след обработка):

tweet	class
This is why you need friends at work	regular
sleep pills upon sleep pills and still can't sleep.	irony
Sarcastic People Are Actually Smarter, Sexier And More Successful	sarcasm
You probably just missed the text	figurative

Предварителната обработката се състои в премахване на линкове ('https://'), тагове('@user') и хаштагове (#), но запазване на текста след хаштага, тъй като носи полезна информация за обучение на модела. Например #Sarcastic или #Successful.

3 Техническа реализация

За реализацията на проекта е използван езикът python и логистична регресия за класификация на примерите.



Фигура 1: Описание на структурата на проекта

След трениране на модела, той е съхранен във файл и може да се прочете в последствие. Качил съм тренирана негова версия и в публичното хранилище в Github. Класификаторът смята както стандатна вероятност със стойности в интервала [0,1], така и размити числа за трите дефинирани лингвистични променливи - невероятно (unlikely), възможно (probably), силно вероятно (highly likely).

3.1 Трапецоидална функция на принадлежност

Математическата дефиниция на функцията е следната:

$$\mu(x;a,b,c,d) = \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \leq x < b \\ 1, & b \leq x \leq c \\ \frac{d-x}{d-c}, & c < x \leq d \\ 0, & x > d \end{cases}$$

Unlikely

$$\mu_{\text{unlikely}}(x) = \begin{cases} 0, & x < 0.0 \\ 1, & 0.0 \leq x \leq 0.3 \\ \frac{0.5-x}{0.5-0.3}, & 0.3 < x \leq 0.5 \\ 0, & x > 0.5 \end{cases}$$

Probably

$$\mu_{\text{probably}}(x) = \begin{cases} 0, & x < 0.3 \\ \frac{x-0.3}{0.5-0.3}, & 0.3 \leq x < 0.5 \\ 1, & 0.5 \leq x \leq 0.6 \\ \frac{0.8-x}{0.8-0.6}, & 0.6 < x \leq 0.8 \\ 0, & x > 0.8 \end{cases}$$

Highly Likely

$$\mu_{\text{highly likely}}(x) = \begin{cases} 0, & x < 0.6 \\ \frac{x-0.6}{0.8-0.6}, & 0.6 \leq x < 0.8 \\ 1, & 0.8 \leq x \leq 1.0 \\ 0, & x > 1.0 \end{cases}$$

3.2 Фрагменти от кода на проекта

Реализацицията на функцията на принадлежност със съответните стойности и параметри в кода на проекта изглежда по следния начин:

```
# code fragment from: src/sarcasm_classifier.py

import skfuzzy as fuzz
import numpy as np

# Define the universe of discourse
x_sarcasm = np.linspace(0, 1, 100)

# Define trapezoidal membership functions for sarcasm likelihood
mf_unlikely = fuzz.trapmf(x_sarcasm, [0.0, 0.0, 0.3, 0.5])
mf_probably = fuzz.trapmf(x_sarcasm, [0.3, 0.5, 0.6, 0.8])
mf_highly_likely = fuzz.trapmf(x_sarcasm, [0.6, 0.8, 1.0, 1.0])
```

4 Инсталация и употреба

Стъпките за инсталация на проекта са съвсем стандартни и са описани в README.md. След трениране на класификатора, има две възможности за употреба. Едната е чрез подаване на входен текст през конзолния скрипт run_model.py, а по-приятно е демото в тетрадката на Jupyter

Setup and Running

1. **Install dependencies:**

```
pip install -r requirements.txt
```

2. **Train the classifier:**

```
python -m src.sarcasm_classifier
```

This will train the classifier using the training data and save the model to `src/sarcasm_classifier.pkl`.

3. **Run the demo notebook:**

```
jupyter notebook notebooks/demo.ipynb
```

4. For testing/optionally you can run the model with your own message from the command line:

```
python run_model.py "What a great day!"

Input Comment: What a great day
Sarcasm Probability: 0.58
Fuzzy Classification:
Unlikely:      0.00
Probably:      1.00
Highly Likely: 0.00
```

Фигура 2: Стандартна инсталация и очакван начин на употреба на проекта

5 Jupyter Notebook & интерактивни примери

Примерно изпълнение от конзолата:

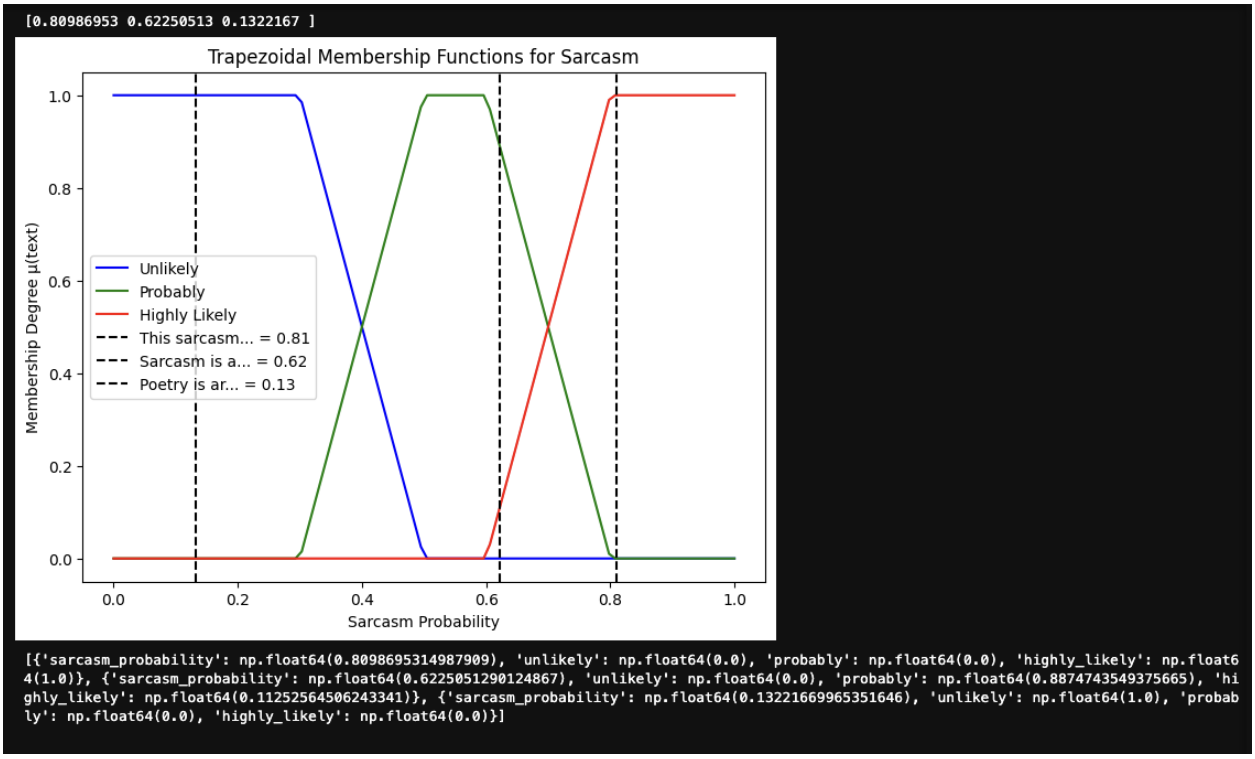
```
(.venv) ➜  FuzzySarcasmV2 git:(master) ✗ python run_model.py "Am I sarcastic or what?"

Input Comment: Am I sarcastic or what?
Sarcasm Probability: 0.73
Fuzzy Classification:
  Unlikely:      0.00
  Probably:      0.34
  Highly Likely: 0.66
```

Примерно изпълнение от Jupyter, с включени debug редове:

```
# Initialize and train the classifier
sarcasm_clf = SarcasmClassifier()
sarcasm_clf.train(train_df['clean_comment'], train_df['label'])
```

```
print("Sarcasm␣classifier␣trained.")
# ...
demo_texts = [
'This␣sarcasm␣is␣not␣sarcasm',
'Sarcasm␣is␣art',
'Poetry␣is␣art'
]
demo_probs = sarcasm_clf.predict_proba(demo_texts)
print(demo_probs)
```



Фигура 3: Вертикалните пунктирани линии показват стойностите за всеки от трите примерни текста

6 Връзка към кода в Github

Линк към проекта в Github

7 Цитати и източници

[1] Kaggle Tweets with Sarcasm and Irony

[2] Материали от курса по Размити множества в moodle

[3] Amina Ben Meriem, Lobna Hlaoua, Lotfi Ben Romdhane, A fuzzy approach for sarcasm detection in social networks, Procedia Computer Science, Volume 192, 2021, Pages 602-611, ISSN 1877-0509,