Zastosowanie analizy SHAP w analizie sentymentu metodami NLP

[Metody Inteligencji Obliczeniowej]

Mariusz Marszałek, Monika Kidawska, Wiktoria Zaczyk

1. Opis projektu

Celem projektu było zbudowanie modelu służącego do analizy sentymentu genialnych tweetów Donalda Trumpa z czerwca 2020 r. zebranych w zbiorze danych: https://www.kaggle.com/datasets/austinreese/trump-tweets

Dla zbudowanego modelu przeprowadziliśmy analizę SHAP oraz przedyskutowaliśmy otrzymane wyniki.

2. Teoria

Projekt opiera się o sztuczne sieci neuronowe, które są uogólnionym modelem obliczeniowym wzorowanym na zachowaniu komórek nerwowych, a więc czerpią z analogii do budowy mózgu. Ta analogia nie jest jednak celem samym w sobie, a przeradza się w budowę obliczeniowych modeli matematycznych opartych na idei tak zwanego uczenia maszynowego.

Analiza semantyczna i składniowa tweetów jest przeprowadzana w celu zminimalizowania utraty informacji podczas procesu klasyfikacji tweetów. Po precyzyjnej klasyfikacji i analizie nastrojów system buduje profil oparty na zainteresowaniach użytkownika, analizując jego post na Twitterze.

Analiza sentymentu słów wykonana została przy pomocy biblioteki SHAP (SHapley Additive exPlanations) wykorzystującej w praktyczny sposób podejście z teorii gier do wyjaśnienia wyniku dowolnego modelu uczenia maszynowego. Wspiera ona modele oparte o drzewa decyzyjne, sieci neuronowe oraz obsługuje też inne metody nie wpadające w te dwie grupy kosztem czasu obliczeń.

SHAP przypisuje każdej z cech wartość ważności dla danego przewidywania. Jego nowatorskie komponenty obejmują:

1) identyfikację nowej klasy addytywnych miar ważności cech
2) wyniki teoretyczne wskazujące na istnienie unikalnego rozwiązania w tej klasie, posiadającego zestaw pożądanych właściwości. Nowa klasa ujednolica sześć istniejących metod, co jest godne uwagi, ponieważ kilka najnowszych metod w tej klasie nie posiada proponowanych pożądanych właściwości.

3. Realizacja projektu

Projekt jest zrealizowany w środowisku Python. Potrzebne biblioteki do zaimportowania znajdują się w pliku *requirements.txt*.

Link do repozytorium: https://github.com/Saffons/shapProjekt

Poszczególne kroki realizacji klasyfikacji nastrojów pozytywnych i negatywnych:
Najważniejszym etapem pracy z tak wielką bazą tweetów jest ich odpowiednie przygotowanie przed ich obróbką.
Do klasyfikacji wykorzystano model distilbert-base-uncased-finetuned-sst-2-english wytrenowany na datasecie sst2 wprowadzonym przez Pang i Lee w 2005, zawierającym 11855 recenzji filmowych.

1. Importujemy potrzebne biblioteki, m.in. *transformers, datasets, shap, nltk, re, pandas, numpy.*

2. Definiujemy funkcję pomocnicze:
    a. "czyszczącą" tweet-y, aby zapobiec wczytywaniu przez program niepotrzebnych linków, znaków nowej linii, znaków przestankowych, liczb, itp.:

```
def clean_text(text):
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = re.sub('\'','', text)

    return text
```

    b. do konwertowania danych z typu *string* do obiektu typu *datetime*, pomocniczą do kolejnej funkcji

```
def datte():
    for i in range(len(train)):
        train['date'][i] = datetime.datetime.strptime(train['date'][i], '%Y-%m-%d %H:%M:%S')
```

    c. do wylosowania *n* randomowych tweetów w podanym przedziale czasowym

```
def choose_tweets_from_date(start, end, n):
    start = datetime.datetime.strptime(start, '%Y-%m-%d')
    end = datetime.datetime.strptime(end, '%Y-%m-%d')
    choices = []
    for i in range(len(train)):
        if start <= train['date'][i] <= end:
            choices.append(train['content'][i])


    chosen = []
    chosen.append(random.sample(choices, n))

    return chosen
```

3. Inicjalizacja zbiorów tweetów:
    - zapisujemy plik csv do pandas dataframe
    - wyrzucamy niepotrzebne kolumny
    - wypisujemy input
    - używamy *tokenizer-a* do zapisania każdego słowa osobno

```
train = pd.read_csv('WFiIS-MIO-main/datasets/realdonaldtrump.csv')
train = train.drop(['link','mentions','hashtags'], axis=1)

train['content'] = train['content'].map(lambda x: re.sub(r'\W+', ' ', x))
train['content'] = train['content'].replace(r'\W+', ' ', regex=True)
train['content'] = train['content'].apply(lambda x:clean_text(x))
train_token = train['content'].apply(lambda x:word_tokenize(x))
datte()
```

4. Wybieramy 50 losowych tweetów przed kadencją Trumpa (before) oraz w trakcie kadencji (after):

5. Zliczamy każde słowo i tworzymy top listę z rankingiem częstotliwości występowań
   Następnie wyrysowujemy naszą tabele z wynikami: [Ranking 1]

6. Klasyfikator

Ładujemy i uruchamiamy funkcję *transformers.pipeline* do analizy nastrojów.

```
classifier = transformers.pipeline('sentiment-analysis', return_all_scores=True)
classifier(before[0])
```

Utworzenie opakowanie *Transformers.pipeline*

```
pmodel = shap.models.TransformersPipeline(classifier, rescale_to_logits=True)
```

Definiujemy *explainer* do analizy nastrojów.

```
explainer2 = shap.Explainer(pmodel)
```

Przewidywania *explainer-a* dla danych *before*

```
shap_values = explainer2(before[0])
```

Wypisujemy wynik naszego explainer-a w postaci wykresów z biblioteki SHAP [wykres 1, wykres 2]

```
shap.plots.text(shap_values[:,:,1])
```

Tworzenie wykresów słupkowych: [wykres 3]

```
for i in range(n):
    shap.plots.bar(shap_values[i, :,"POSITIVE"])
shap.plots.bar(shap_values[:, :, "POSITIVE"].mean(0))
shap.plots.bar(shap_values[:, :, "POSITIVE"].mean(0), order=shap.Explanation.argsort)
```

*dla przypadku *after* jest dokładnie to samo

Inicjalizujemy analizę SHAP, która wymaga wyjścia tensorowego (obiekt będący uogólnieniem wektora) z klasyfikatora, a wyjaśnienia działają najlepiej w przestrzeniach addytywnych, więc przekształcamy prawdopodobieństwa na wartości logitowe (wartości informacyjne zamiast prawdopodobieństw).

7. Obliczyliśmy średni wynik analizy dla tweetów przed i po kadencji prezydenta Donald'a Trump'a.

```
# average_after = sum(shap_values[:, :, "POSITIVE"].mean(0).values)/len(shap_values[:, :, "POSITIVE"].mean(0).values)
# print(average)

print(average_before)
print(average_after)
```

```
0.023818865144145086
-0.01727848837024183
```

4. Podział pracy

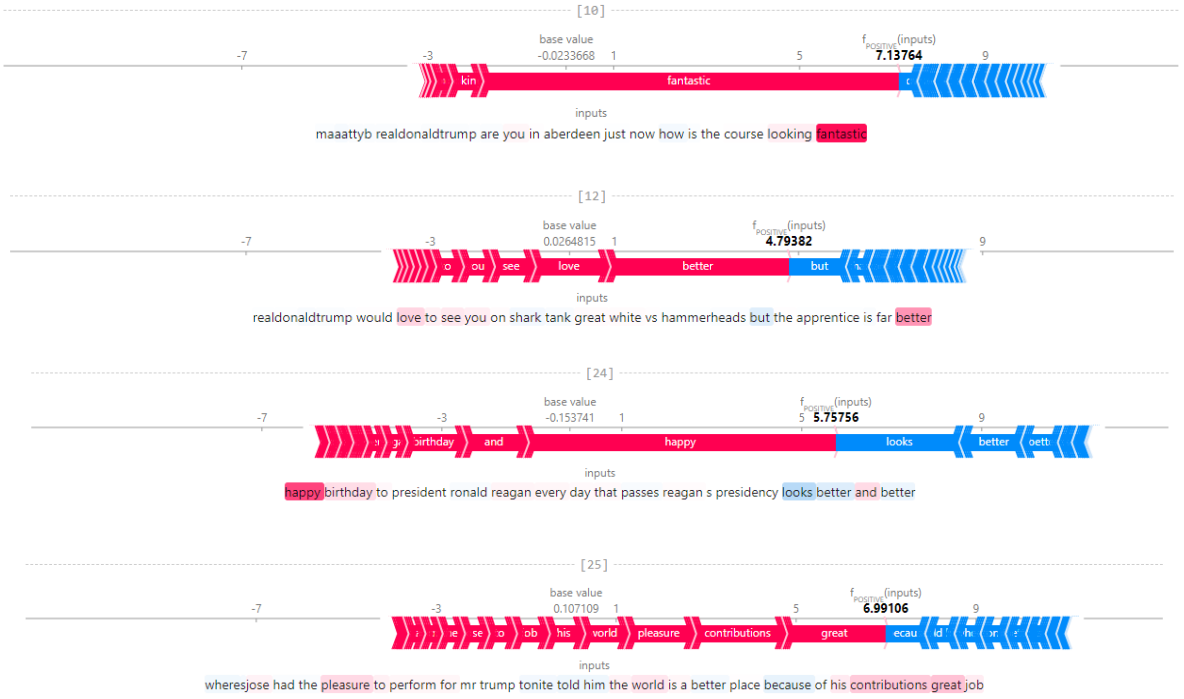
- ☐ Mariusz Marszałek: research(20%), kod(55%), dokumentacja(15%)
- ☐ Monika Kidawska: research(40%), kod(30%), dokumentacja(30%)
- ☐ Wiktoria Zaczyk: research(40%), kod(15%), dokumentacja(55%)

5. Analiza wyników

	Common_words	count
0	the	34884
1	to	20043
2	and	16007
3	a	15589
4	of	13444
5	is	12944
6	in	12156
7	i	10830
8	you	10729
9	for	10109
10	realdonaldtrump	8788
11	com	8318
12	on	8243
13	s	7443
14	be	7110
15	it	6885
16	great	6774
17	will	6701
18	trump	6493
19	that	6301

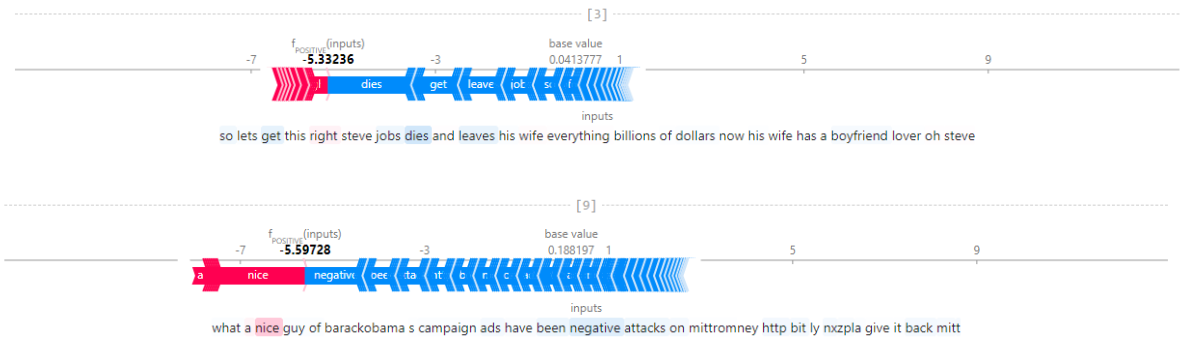
Ranking 1. Ranking [1-20] częstotliwości występowań poszczególnych słów wszystkich tweetów (w githubie: tweet.txt):

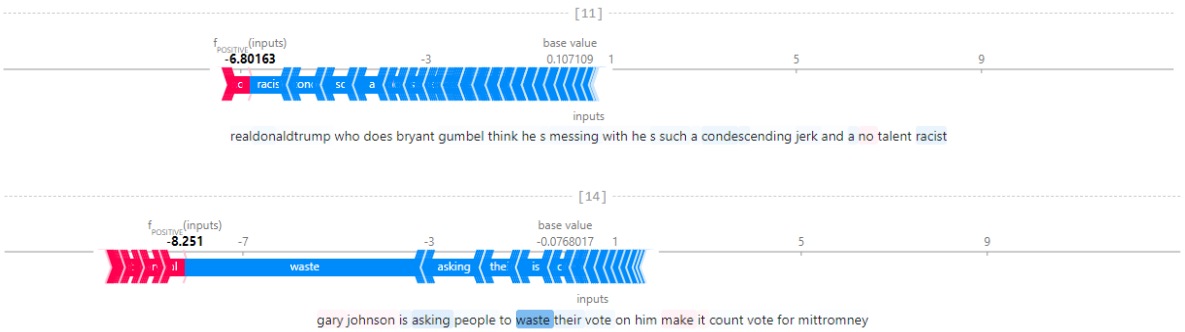
Wykres 1: Przykładowe wykresy dla tweetów o pozytywnym sentymencie



Z powyższych wykresów zauważamy poprawność wykonania analizy, ponieważ na jej wyniki wpłynęły słowa o pozytywnym brzmieniu (fantastic, better, happy, pleasure, great).

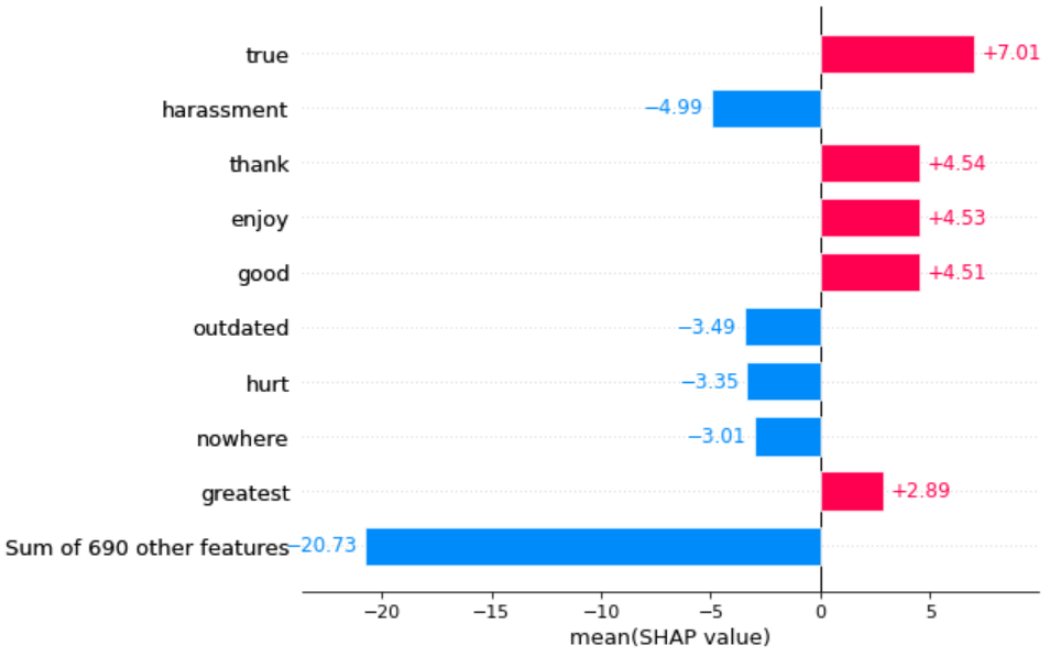
Wykres 2: Przykładowe wykresy dla tweetów o negatywnym sentymencie



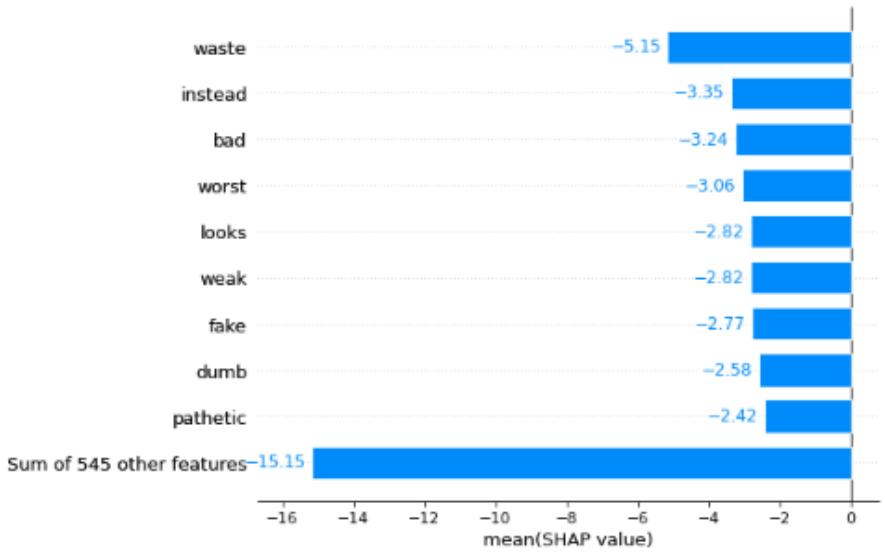


O poprawnym wykonaniu analizy świadczą również powyżej przedstawione wykresy, ponieważ o wynikach zadecydowały negatywne słowa (dies, negative, racist, waste).

Wykres 3: Przykładowe wykresy słupkowe



Wykres przedstawiający słowa najbardziej wpływające na wydźwięk (spośród tweetów w trakcie kadencji) niezależnie czy pozytywnie, czy negatywnie



Wykres przedstawiający słowa wpływające najbardziej negatywnie na wydźwięk tweeta (spośród tweetów wybranych przed kadencją)

6. Wnioski

Podsumowując, analiza SHAP w analizie sentymentu tweetów przebiegła pomyślnie. Wypowiedzi pozytywne oraz negatywne zostały poprawnie rozpoznane, jednak nie jesteśmy całkowicie zadowoleni z otrzymanych wyników. Jak mogliśmy zauważyć, sieć nie zawsze prawidłowo interpretowała słowa, czasami je łączyła ze sobą, czasami czytała linki (przy czym próbowaliśmy to uwzględnić, aby sieć nie brała linków pod uwagę, to gdzie niegdzie wkradło się samo słowo “http”, jednak nie zawsze się to udawało). Uważamy, że sieć spełniła swoje zadanie i poprawnie interpretuje większość słów oraz zlicza ich wystąpienia.

Z racji na długi czas procesowania tweetów przez CPU, zespół planował wykorzystać do tego GPU wykorzystując procesory CUDA, które przyspieszają uczenie nawet 40-krotnie. Przerosło to jednak możliwości nasze i naszego sprzętu :) (są to albo zbyt słabe karty nVidia w laptopach, albo karta AMD, która niestety nie posiada procesorów CUDA. Istnieją jednak biblioteki OpenCL i od niedawna GPUFORT, które pomagają robić podobne obliczenia, na kartach AMD.

Jak widać w tabeli częstości słów, najczęstsze słowa są niewiele znaczące, więc podczas przygotowywania inputu mogliśmy jakoś je przefiltrować, korzystając np ze ‘stopwords’ dostępnych w różnych formach.

Nawet po przeprocesowaniu jedynie po 50 tweetów sprzed kadencji i w jej trakcie, widzimy, że w trakcie kadencji tweety Trumpa były bardziej pozytywne (średni sentyment 0.02318) niż przed kadencją (średni sentyment -0.01728).

Linki:

- *https://helion.pl/ksiazki/wykorzystanie-sztucznych-sieci-neuronowych-lukasz-wordliczek,e_1x82.htm#format/e*
- *https://www.hindawi.com/journals/complexity/2020/8892552/*
- *https://coderzcolumn.com/tutorials/artificial-intelligence/shap-values-for-text-classification-tasks*
- *https://www.kaggle.com/datasets/austinreese/trump-tweets*