MOVIE COMMENT SENTIMENT ANALYSIS PROJECT

Uğur Selim ÖZEN

03/16/2022

PROBLEM

- Classifying user’s comment that made for IMDB movies.

METHODOLOGY

- Writing/Reading Dataset with MongoDB
- Data Cleaning & Data Processing
- Data Visualization for EDA via Plotly
- Different NLP Techniques
- Model Building & Evaluation
- Streamlit App & Deployment

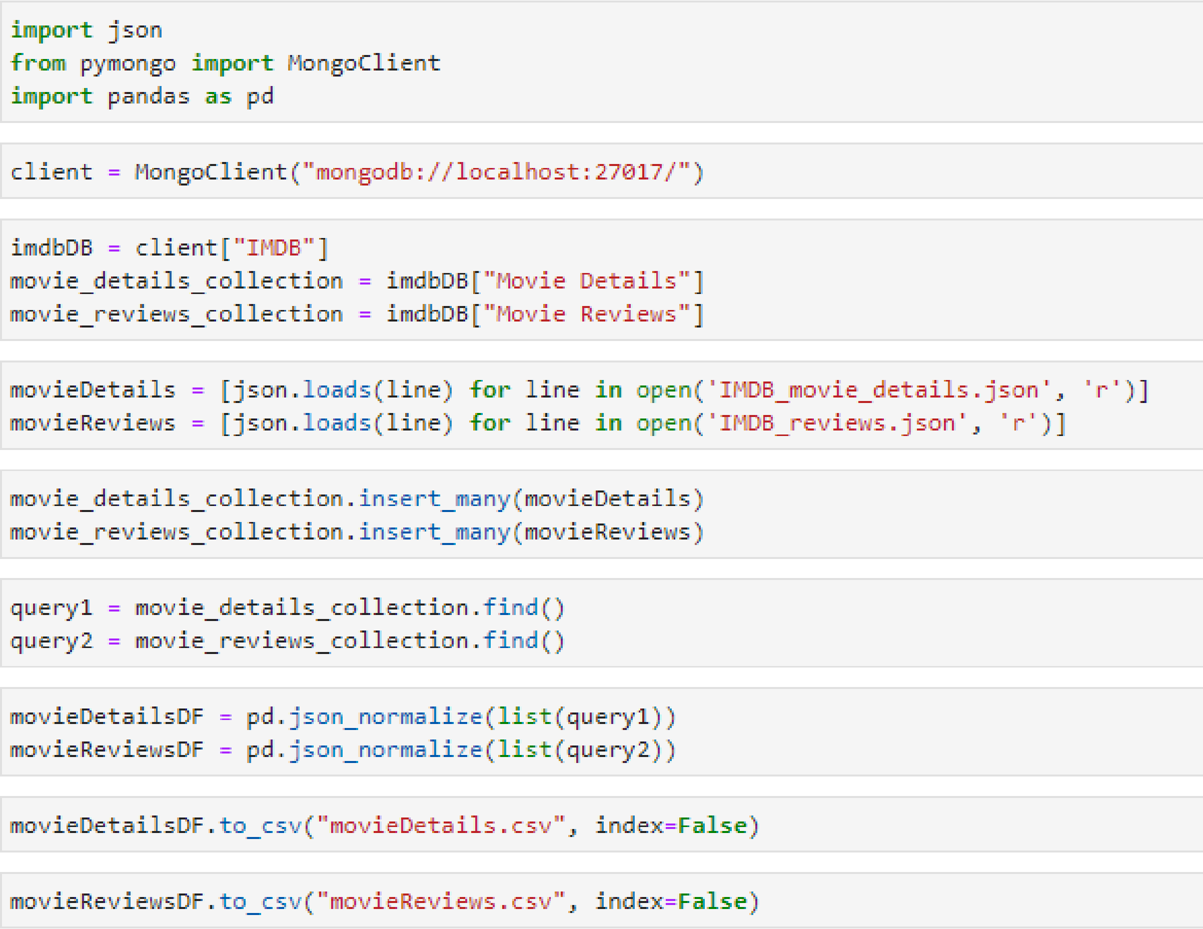
UTILIZED TECHNOLOGIES

Dataset Overview

- Reviews : **573913**
- Target1 : **Sentiment**
- Target2 : **is_spoiler**
- Data Size : **~ 900 MB**

jupyter
scikit learn
pandas
matplotlib
seaborn
plotly
Streamlit
mongoDB®

WRITING & READING DATASET FROM MONGODB



```python
import json
from pymongo import MongoClient
import pandas as pd
```

```python
client = MongoClient("mongodb://localhost:27017/")
```

```python
imdbDB = client["IMDB"]
movie_details_collection = imdbDB["Movie Details"]
movie_reviews_collection = imdbDB["Movie Reviews"]
```

```python
movieDetails = [json.loads(line) for line in open('IMDB_movie_details.json', 'r')]
movieReviews = [json.loads(line) for line in open('IMDB_reviews.json', 'r')]
```

```python
movie_details_collection.insert_many(movieDetails)
movie_reviews_collection.insert_many(movieReviews)
```

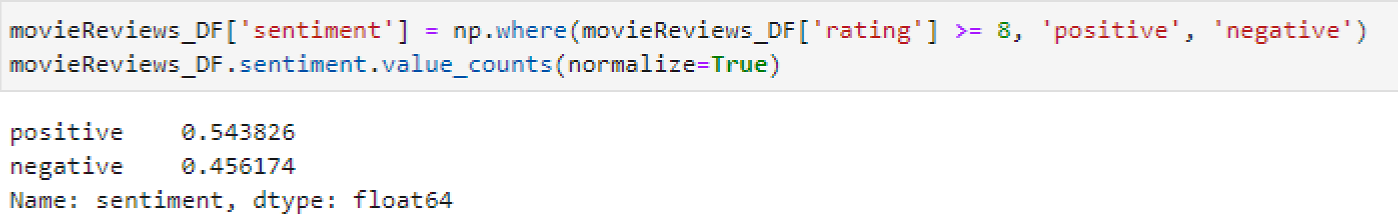
```python
query1 = movie_details_collection.find()
query2 = movie_reviews_collection.find()
```

```python
movieDetailsDF = pd.json_normalize(list(query1))
movieReviewsDF = pd.json_normalize(list(query2))
```

```python
movieDetailsDF.to_csv("movieDetails.csv", index=False)
```

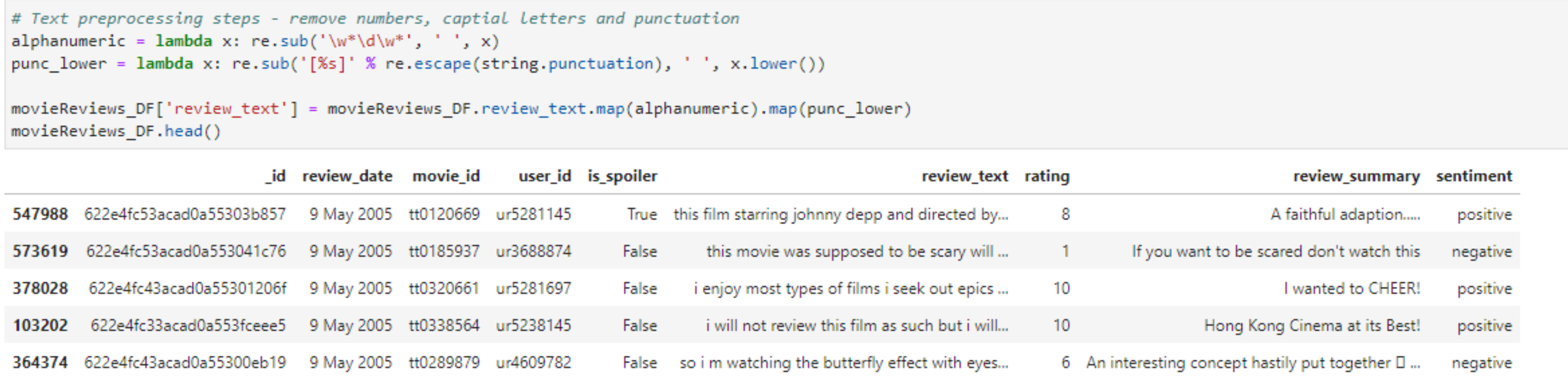
```python
movieReviewsDF.to_csv("movieReviews.csv", index=False)
```

DATA CLEANİNG & DATA PROCESSİNG



```
movieReviews_DF['sentiment'] = np.where(movieReviews_DF['rating'] >= 8, 'positive', 'negative')
movieReviews_DF.sentiment.value_counts(normalize=True)
```

```
positive    0.543826
negative    0.456174
Name: sentiment, dtype: float64
```

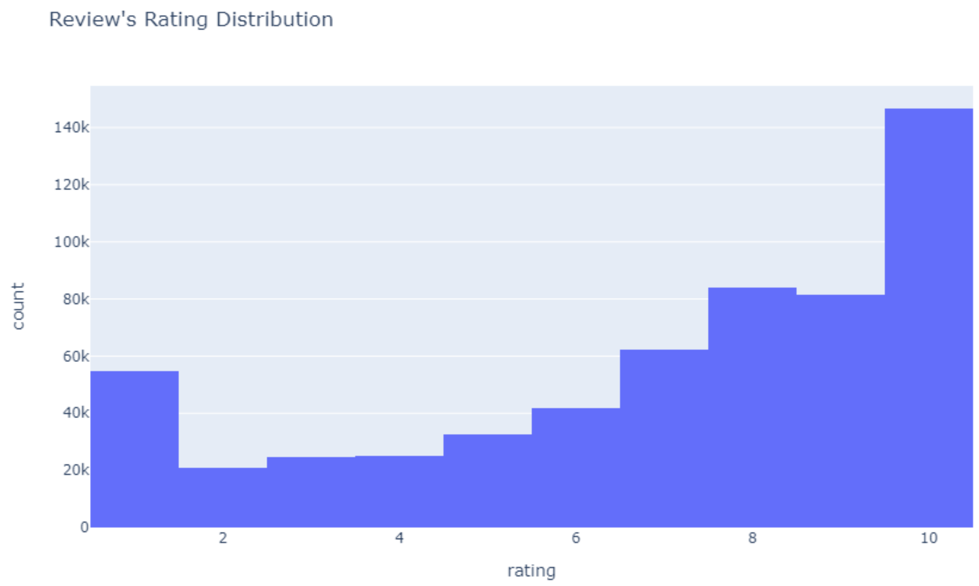


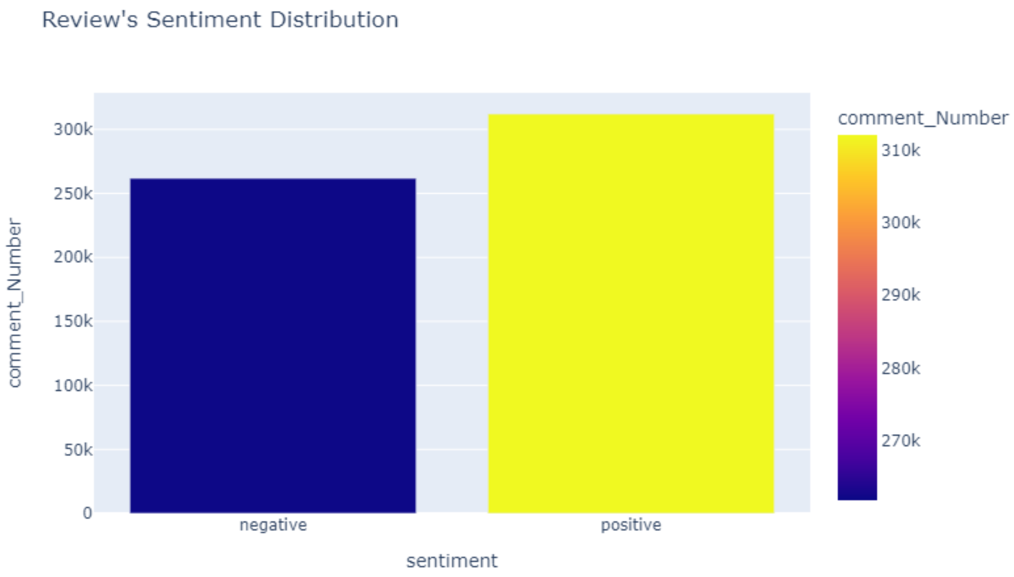
```
# Text preprocessing steps - remove numbers, captial letters and punctuation
alphanumeric = lambda x: re.sub('\w*\d\w*', ' ', x)
punc_lower = lambda x: re.sub('[%s]' % re.escape(string.punctuation), ' ', x.lower())

movieReviews_DF['review_text'] = movieReviews_DF.review_text.map(alphanumeric).map(punc_lower)
movieReviews_DF.head()
```

	_id	review_date	movie_id	user_id	is_spoiler	review_text	rating	review_summary	sentiment
547988	622e4fc53acad0a55303b857	9 May 2005	tt0120669	ur5281145	True	this film starring johnny depp and directed by...	8	A faithful adaption.....	positive
573619	622e4fc53acad0a553041c76	9 May 2005	tt0185937	ur3688874	False	this movie was supposed to be scary will ...	1	If you want to be scared don't watch this	negative
378028	622e4fc43acad0a55301206f	9 May 2005	tt0320661	ur5281697	False	i enjoy most types of films i seek out epics ...	10	I wanted to CHEER!	positive
103202	622e4fc33acad0a553fceee5	9 May 2005	tt0338564	ur5238145	False	i will not review this film as such but i will...	10	Hong Kong Cinema at its Best!	positive
364374	622e4fc43acad0a55300eb19	9 May 2005	tt0289879	ur4609782	False	so i m watching the butterfly effect with eyes...	6	An interesting concept hastily put together □ ...	negative

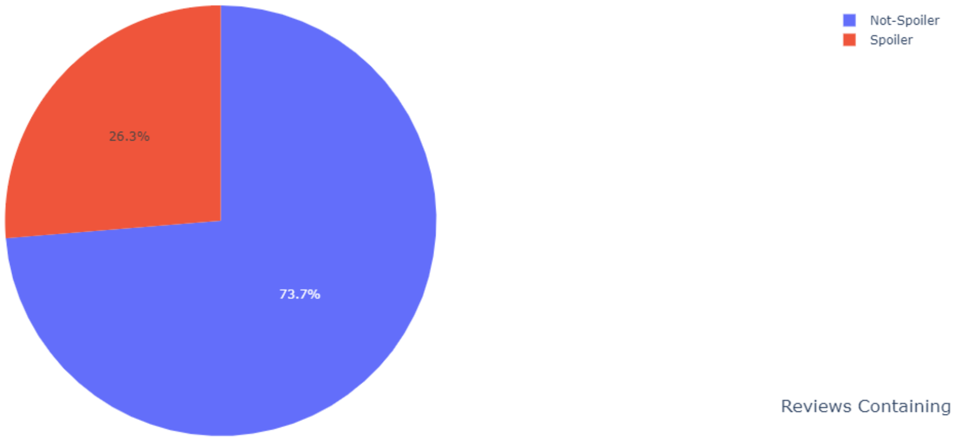
DATA VISUALIZATION FOR EDA VIA PLOTLY


Review's Rating Distribution

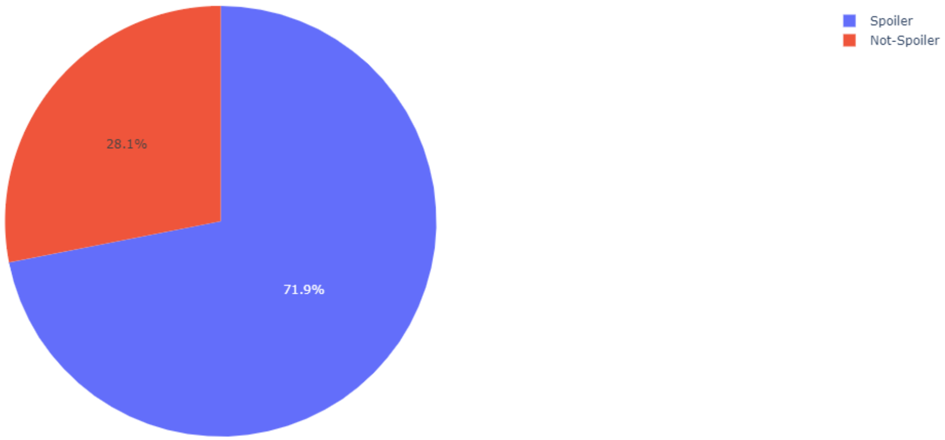

Review's Sentiment Distribution

DATA VISUALIZATION FOR EDA VIA PLOTLY

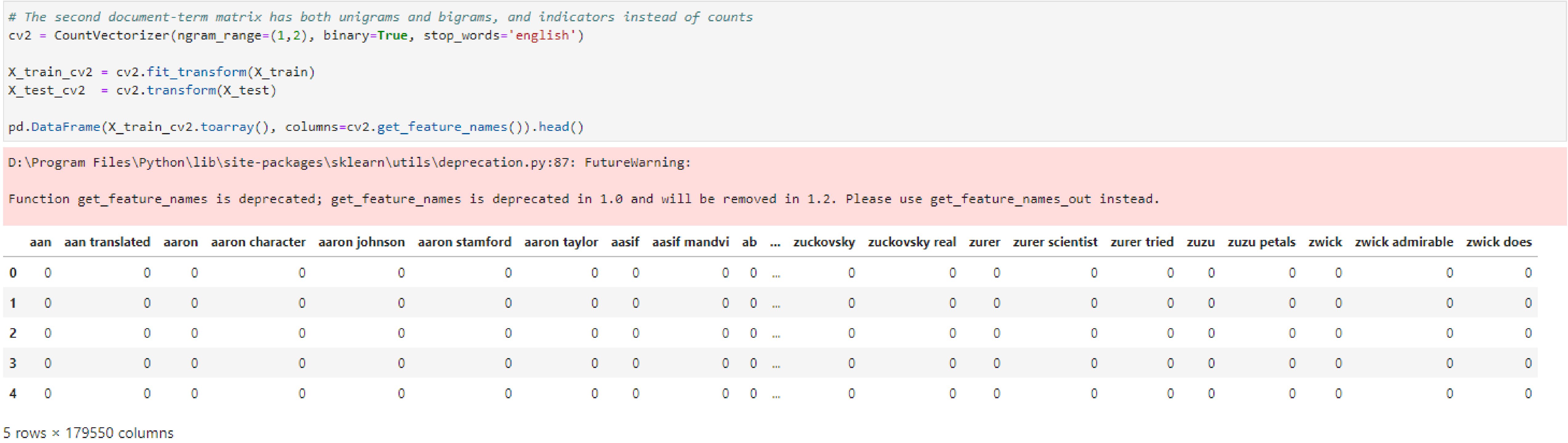
All Reviews Spoiler Distribution



Reviews Containing word 'Spoiler'



DIFFERENT NLP TECHNIQUES – COUNT VECTORIZER



```
# The second document-term matrix has both unigrams and bigrams, and indicators instead of counts
cv2 = CountVectorizer(ngram_range=(1,2), binary=True, stop_words='english')

X_train_cv2 = cv2.fit_transform(X_train)
X_test_cv2  = cv2.transform(X_test)

pd.DataFrame(X_train_cv2.toarray(), columns=cv2.get_feature_names()).head()
```

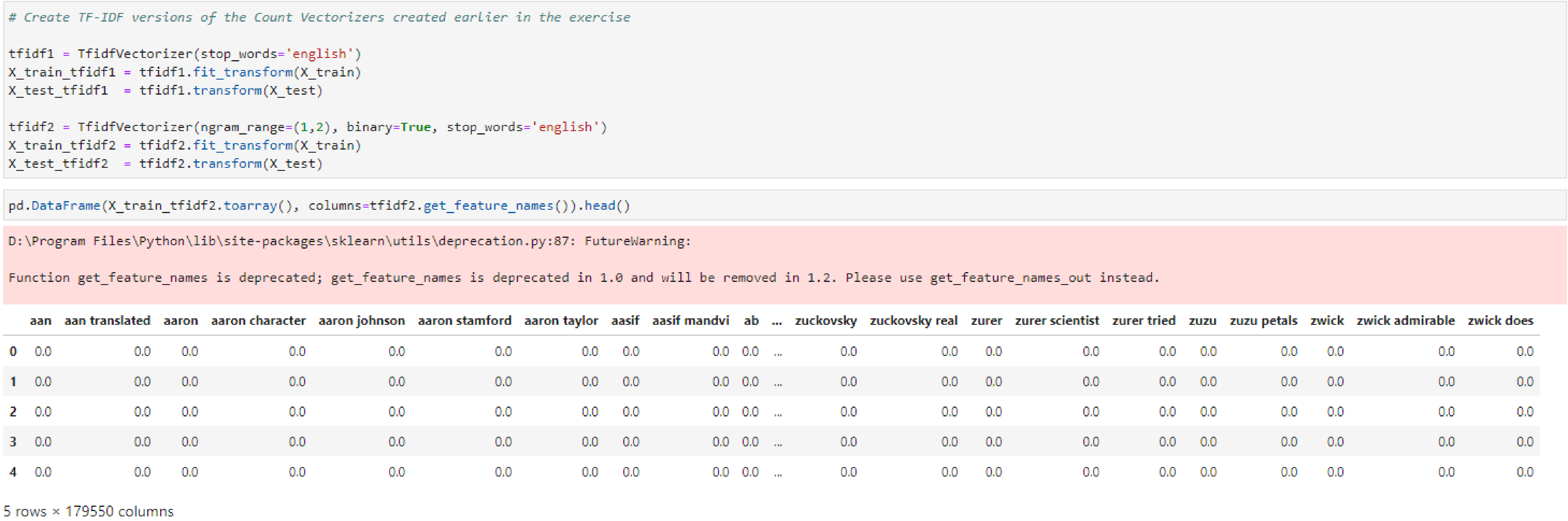
```
D:\Program Files\Python\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning:

Function get_feature_names is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out instead.
```

	aan	aan translated	aaron	aaron character	aaron johnson	aaron stamford	aaron taylor	aasif	aasif mandvi	ab	...	zuckovsky	zuckovsky real	zurer	zurer scientist	zurer tried	zuzu	zuzu petals	zwick	zwick admirable	zwick does
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 179550 columns

DIFFERENT NLP TECHNIQUES – TF-IDF



```
# Create TF-IDF versions of the Count Vectorizers created earlier in the exercise

tfidf1 = TfidfVectorizer(stop_words='english')
X_train_tfidf1 = tfidf1.fit_transform(X_train)
X_test_tfidf1  = tfidf1.transform(X_test)

tfidf2 = TfidfVectorizer(ngram_range=(1,2), binary=True, stop_words='english')
X_train_tfidf2 = tfidf2.fit_transform(X_train)
X_test_tfidf2  = tfidf2.transform(X_test)
```

```
pd.DataFrame(X_train_tfidf2.toarray(), columns=tfidf2.get_feature_names()).head()
```

```
D:\Program Files\Python\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning:

Function get_feature_names is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out instead.
```

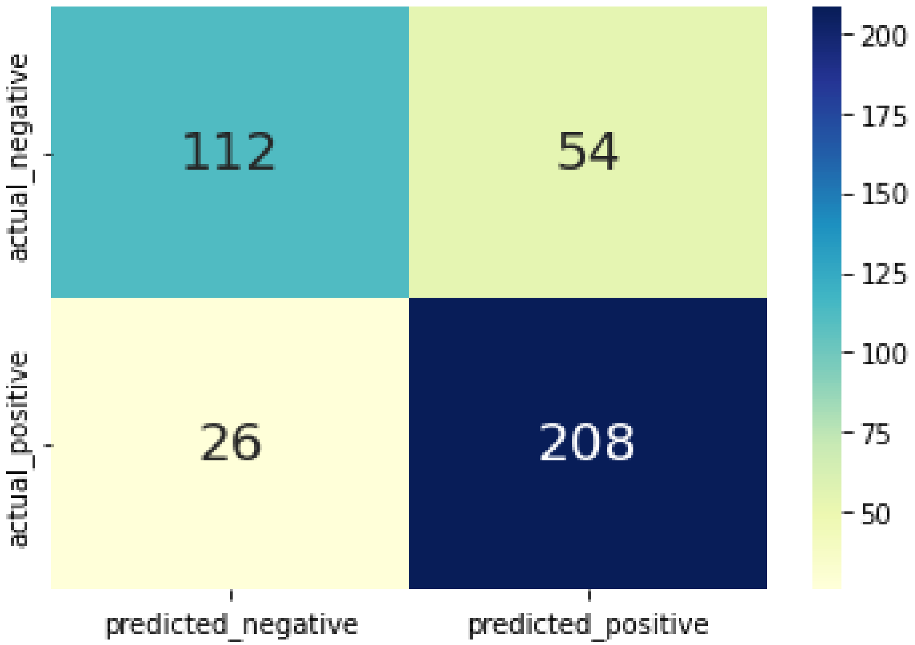
	aan	aan translated	aaron	aaron character	aaron johnson	aaron stamford	aaron taylor	aasif	aasif mandvi	ab	...	zuckovsky	zuckovsky real	zurer	zurer scientist	zurer tried	zuzu	zuzu petals	zwick	zwick admirable	zwick does
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 179550 columns

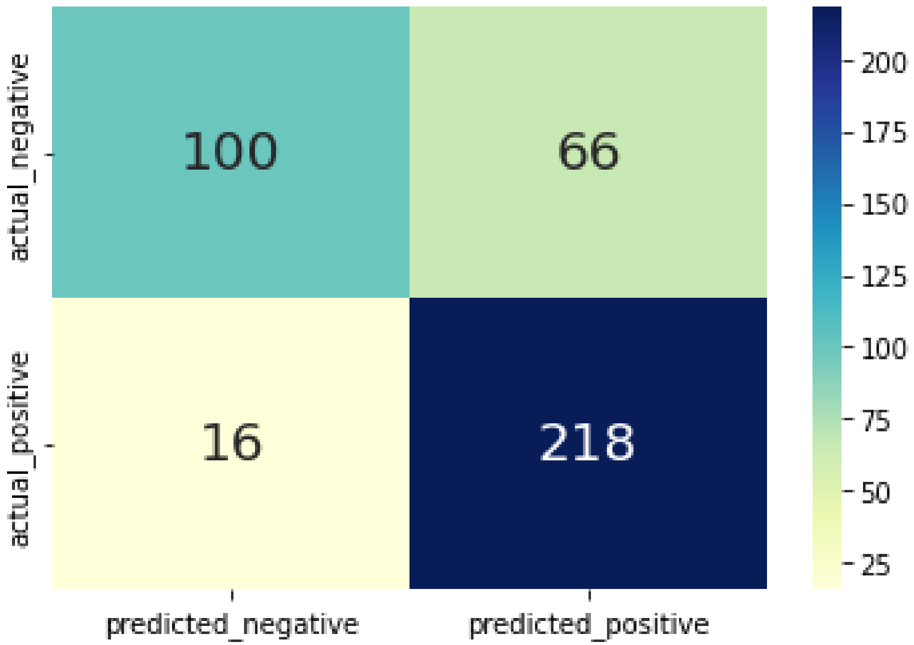
MODEL BUILDING & EVALUATION

	LogReg1	LogReg2	NB1	NB2	LR1-TFIDF	LR2-TFIDF	NB1-TFIDF	NB2-TFIDF
Accuracy	0.765	0.800	0.758	0.595	0.795	0.742	0.668	0.595
Precision	0.780	0.794	0.764	0.591	0.768	0.698	0.639	0.591
Recall	0.833	0.889	0.846	1.000	0.932	0.987	0.991	1.000
F1 Score	0.806	0.839	0.803	0.743	0.842	0.818	0.777	0.743

TWO BEST MODEL’S COMPARISON



Logistic Regression – Count Vectorizer with ngram



Logistic Regression – TF-IDF without ngram

THANK YOU !

thank you
danke
謝謝
ngiyabonga
شكراً جزيلا
teşekkür ederim
спасибо
gracias
dank je
tapadh leat
bedankt
maururu
hvala
dziękuję
mochchakkeram
go raibh maith agat
sukriya
kop khun krap
obrigado
arigatō
takk
dakujem
grazie
terima kasih
ありがとう
мерси
감사합니다
merci