PREPROCESO

TRAIN xml

TRAIN

768213876278165504	P	Me gusta mucho esta película.
8383838388383838	N	No me gusta la película @pepe.
	…	
	…	

#tokenitzar_tweet

from nltk.tokenize import **TweetTokenizer**

train_data <- TweetTokenizer(strip_handles=False, reduce_len=True, preserve_case=False).tokenize(tweet))

train_data=[‘Me gusta mucho esta película .’, ’No me gusta la película @pepe .’ , …]
train_labels=[‘P,’ ‘N’, …]
dev_data=[]
dev_labels=[]
test_data=[]

from sklearn.feature_extraction.text import CountVectorizer

VECTORIZACIÓN

vectorizer = CountVectorizer(tokenizer=mi_tokenizador)
train_vectors = vectorizer.**fit_transform**(train_data)

Obtener si tiene palabras positivas y/o negativas

	f1	f2	f4	f5	f6	…	…	…	fn	pos	neg
t1	1	0	0	2	0	0	0	0	2	2	1
t2	0	2	0	0	0	0	0	0	0	0	1
…										3	0
tn	0	0	1	2	0	0	0	0	0	0	0

train_vectors

train_palabras_con_polaridad

JUNTAR LAS DOS MATRICES:

```
import scipy
M=scipy.sparse.hstack((train_vectors, train_palabras_con_polaridad)
```

dev_vectors = **vectorizer.transform**(dev_data)

```
def mi_tokenizador(s):
    xx=[]
    x=s.split()
    for t in x:
        t = re.sub('@.*', "arroba", t)
        t = re.sub('#(.*)', "hashtag",t)
        t = re.sub('http.*', "http", t)
        t = re.sub('[0-9].*', "num", t)
        xx.append(t)
    return (xx)
```

vectorizer = CountVectorizer(tokenizer=mi_tokenizador)

Ejemplo de mi_tokenizador: para reducir el vocabulario

ENTRENAR MODELO y EVALUAR

```
from sklearn import svm
from sklearn.metrics import classification_report

classifier_liblinear = svm.LinearSVC(C=0.1)
classifier_liblinear.fit(train_vectors, train_labels)
prediction_liblinear = classifier_liblinear.predict(dev_vectors)

print(classification_report(dev_labels, prediction_liblinear))
```

	precision	recall	f1-score	support
N	0.60	0.78	0.68	219
NEU	0.24	0.09	0.13	69
NONE	0.24	0.13	0.17	62
P	0.58	0.60	0.59	156
accuracy			0.55	506
macro avg	0.41	0.40	0.39	506
weighted avg	0.50	0.55	0.51	506