a-task-twitter-sentiment-analysis

February 25, 2025

[ ]:
```
# installing kaggle library
! pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.11/dist-packages
(1.6.17)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.11/dist-
packages (from kaggle) (1.17.0)
Requirement already satisfied: certifi>=2023.7.22 in
/usr/local/lib/python3.11/dist-packages (from kaggle) (2025.1.31)
Requirement already satisfied: python-dateutil in
/usr/local/lib/python3.11/dist-packages (from kaggle) (2.8.2)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-
packages (from kaggle) (2.32.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages
(from kaggle) (4.67.1)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.11/dist-
packages (from kaggle) (8.0.4)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.11/dist-
packages (from kaggle) (2.3.0)
Requirement already satisfied: bleach in /usr/local/lib/python3.11/dist-packages
(from kaggle) (6.2.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.11/dist-
packages (from bleach->kaggle) (0.5.1)
Requirement already satisfied: text-unidecode>=1.3 in
/usr/local/lib/python3.11/dist-packages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests->kaggle) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-
packages (from requests->kaggle) (3.10)
```

Upload your Kaggle.json file

[ ]:
```
#configuring the path of kaggle.json file
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

```
cp: cannot stat 'kaggle.json': No such file or directory
chmod: cannot access '/root/.kaggle/kaggle.json': No such file or directory
```

Import Twitter Sentiment dataset

[ ]:
```
#API to fetch data from kaggle
!kaggle datasets download -d kazanova/sentiment140
```

```
Dataset URL: https://www.kaggle.com/datasets/kazanova/sentiment140
License(s): other
Downloading sentiment140.zip to /content
 89% 72.0M/80.9M [00:00<00:00, 253MB/s]
100% 80.9M/80.9M [00:00<00:00, 247MB/s]
```

[ ]:
```
#etract the compressed dataset

from zipfile import ZipFile
dataset='/content/sentiment140.zip'

with ZipFile(dataset,'r') as zip:
  zip.extractall()
  print('The dataset is extracted')
```

```
The dataset is extracted
```

Import the dependencies

[ ]:
```
import pandas as pd
import numpy as np
import re
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

[ ]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

[ ]:
```
True
```

[ ]:
```
#printing the stopwords in English
print(stopwords.words('english'))
```

```
['a', 'about', 'above', 'after', 'again', 'against', 'ain', 'all', 'am', 'an',
'and', 'any', 'are', 'aren', "aren't", 'as', 'at', 'be', 'because', 'been',
'before', 'being', 'below', 'between', 'both', 'but', 'by', 'can', 'couldn',
"couldn't", 'd', 'did', 'didn', "didn't", 'do', 'does', 'doesn', "doesn't",
```

```
'doing', 'don', "don't", 'down', 'during', 'each', 'few', 'for', 'from',
'further', 'had', 'hadn', "hadn't", 'has', 'hasn', "hasn't", 'have', 'haven',
"haven't", 'having', 'he', "he'd", "he'll", 'her', 'here', 'hers', 'herself',
"he's", 'him', 'himself', 'his', 'how', 'i', "i'd", 'if', "i'll", "i'm", 'in',
'into', 'is', 'isn', "isn't", 'it', "it'd", "it'll", "it's", 'its', 'itself',
"i've", 'just', 'll', 'm', 'ma', 'me', 'mightn', "mightn't", 'more', 'most',
'mustn', "mustn't", 'my', 'myself', 'needn', "needn't", 'no', 'nor', 'not',
'now', 'o', 'of', 'off', 'on', 'once', 'only', 'or', 'other', 'our', 'ours',
'ourselves', 'out', 'over', 'own', 're', 's', 'same', 'shan', "shan't", 'she',
"she'd", "she'll", "she's", 'should', 'shouldn', "shouldn't", "should've", 'so',
'some', 'such', 't', 'than', 'that', "that'll", 'the', 'their', 'theirs',
'them', 'themselves', 'then', 'there', 'these', 'they', "they'd", "they'll",
"they're", "they've", 'this', 'those', 'through', 'to', 'too', 'under', 'until',
'up', 've', 'very', 'was', 'wasn', "wasn't", 'we', "we'd", "we'll", "we're",
'were', 'weren', "weren't", "we've", 'what', 'when', 'where', 'which', 'while',
'who', 'whom', 'why', 'will', 'with', 'won', "won't", 'wouldn', "wouldn't", 'y',
'you', "you'd", "you'll", 'your', "you're", 'yours', 'yourself', 'yourselves',
"you've"]
```

Data Processing

[ ]:
```
# loading the data from csv file to pandas dataframe
twitter_data=pd.read_csv('/content/training.1600000.processed.noemoticon.
  ↪csv',encoding="ISO-8859-1")
```

[ ]:
```
#Checking the number pf rows and columns
twitter_data.shape
```

[ ]:
```
(1599999, 6)
```

[ ]:
```
twitter_data.head(2)
```

[ ]:
```
   0  1467810369  Mon Apr 06 22:19:45 PDT 2009  NO_QUERY _TheSpecialOne_  \
0  0  1467810672  Mon Apr 06 22:19:49 PDT 2009  NO_QUERY   scotthamilton
1  0  1467810917  Mon Apr 06 22:19:53 PDT 2009  NO_QUERY        mattycus

  @switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got
David Carr of Third Day to do it. ;D
0  is upset that he can't update his Facebook by …
1  @Kenichan I dived many times for the ball. Man…
```

[ ]:
```
#naming the columns and reading the dataset again

column_names=['target','id','date','flag','user','text']
twitter_data=pd.read_csv('/content/training.1600000.processed.noemoticon.csv',␣
  ↪names=column_names, encoding="ISO-8859-1")
```

[ ]:
```
twitter_data.shape
```

[ ]:
```
(1600000, 6)
```

[ ]:
```
twitter_data.head(5)
```

[ ]:
```
   target          id                          date      flag  \
0       0  1467810369  Mon Apr 06 22:19:45 PDT 2009  NO_QUERY
1       0  1467810672  Mon Apr 06 22:19:49 PDT 2009  NO_QUERY
2       0  1467810917  Mon Apr 06 22:19:53 PDT 2009  NO_QUERY
3       0  1467811184  Mon Apr 06 22:19:57 PDT 2009  NO_QUERY
4       0  1467811193  Mon Apr 06 22:19:57 PDT 2009  NO_QUERY

              user                                                text
0  _TheSpecialOne_  @switchfoot http://twitpic.com/2y1zl - Awww, t…
1    scotthamilton  is upset that he can't update his Facebook by …
2         mattycus  @Kenichan I dived many times for the ball. Man…
3          ElleCTF    my whole body feels itchy and like its on fire
4           Karoli  @nationwideclass no, it's not behaving at all…
```

[ ]:
```
#counting the number of missing values in the datset
twitter_data.isnull().sum()
```

[ ]:
```
target    0
id        0
date      0
flag      0
user      0
text      0
dtype: int64
```

[ ]:
```
#checking the distribution of target
twitter_data['target'].value_counts()
```

[ ]:
```
target
0    800000
4    800000
Name: count, dtype: int64
```

convert the target “4” to “1”

[ ]:
```
twitter_data.replace({'target':{4:1}},inplace=True)
```

[ ]:
```
twitter_data['target'].value_counts()
```

[ ]:
```
target
0    800000
1    800000
Name: count, dtype: int64
```

0—->Negative Tweeet

1—->Positive Tweet

Stemming

[ ]:
```
port_stem=PorterStemmer()
```

[ ]:
```
def stemming(content):

  stemmed_content=re.sub('[^a-zA-Z]',' ',content)
  stemmed_content=stemmed_content.lower()
  stemmed_content=stemmed_content.split()
  stemmed_content=[port_stem.stem(word) for word in stemmed_content if not word␣
 ↪in stopwords.words('english')]
  return stemmed_content
```

[ ]:
```
twitter_data['stemmed_content']=twitter_data['text'].apply(stemming)
```

[ ]:
```
twitter_data.head()
```

[ ]:
```
   target          id                          date      flag  \
0       0  1467810369  Mon Apr 06 22:19:45 PDT 2009  NO_QUERY
1       0  1467810672  Mon Apr 06 22:19:49 PDT 2009  NO_QUERY
2       0  1467810917  Mon Apr 06 22:19:53 PDT 2009  NO_QUERY
3       0  1467811184  Mon Apr 06 22:19:57 PDT 2009  NO_QUERY
4       0  1467811193  Mon Apr 06 22:19:57 PDT 2009  NO_QUERY

              user                                                text  \
0  _TheSpecialOne_  @switchfoot http://twitpic.com/2y1zl - Awww, t…
1    scotthamilton  is upset that he can't update his Facebook by …
2         mattycus  @Kenichan I dived many times for the ball. Man…
3          ElleCTF    my whole body feels itchy and like its on fire
4           Karoli  @nationwideclass no, it's not behaving at all…

                                      stemmed_content
0  [switchfoot, http, twitpic, com, zl, awww, bum…
1  [upset, updat, facebook, text, might, cri, res…
2  [kenichan, dive, mani, time, ball, manag, save…
3             [whole, bodi, feel, itchi, like, fire]
4                 [nationwideclass, behav, mad, see]
```

[ ]:
```
print(twitter_data['stemmed_content'])
```

```
0          [switchfoot, http, twitpic, com, zl, awww, bum…
1          [upset, updat, facebook, text, might, cri, res…
2          [kenichan, dive, mani, time, ball, manag, save…
3                     [whole, bodi, feel, itchi, like, fire]
4                         [nationwideclass, behav, mad, see]
```

```
                                   …
1599995                      [woke, school, best, feel, ever]
1599996    [thewdb, com, cool, hear, old, walt, interview…
1599997                    [readi, mojo, makeov, ask, detail]
1599998    [happi, th, birthday, boo, alll, time, tupac, …
1599999    [happi, charitytuesday, thenspcc, sparkschar, …
Name: stemmed_content, Length: 1600000, dtype: object
```

[ ]:
```
print(twitter_data['target'])
```

```
0          0
1          0
2          0
3          0
4          0
          ..
1599995    1
1599996    1
1599997    1
1599998    1
1599999    1
Name: target, Length: 1600000, dtype: int64
```

[ ]:
```
# Separating the data and label
X = twitter_data['stemmed_content'].apply(lambda words: " ".join(words)).values
Y = twitter_data['target'].values

# Check the output
print(X[:5])
print(Y[:5])
```

```
['switchfoot http twitpic com zl awww bummer shoulda got david carr third day'
 'upset updat facebook text might cri result school today also blah'
 'kenichan dive mani time ball manag save rest go bound'
 'whole bodi feel itchi like fire' 'nationwideclass behav mad see']
[0 0 0 0 0]
```

[ ]:
```
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.
↪2,stratify=y,random_state=2)
```

[ ]:
```
print(x.shape,x_train.shape,x_test.shape)
```

```
(1600000,) (1280000,) (320000,)
```

[ ]:
```
print(x_train)
```

```
['watch saw iv drink lil wine' 'hatermagazin'
 'even though favourit drink think vodka coke wipe mind time think im gonna find
```

```
new drink'
 … 'eager monday afternoon'
 'hope everyon mother great day wait hear guy store tomorrow'
 'love wake folger bad voic deeper']
```

[ ]:
```
print(x_test)
```

```
['mmangen fine much time chat twitter hubbi back summer amp tend domin free
time'
 'ah may show w ruth kim amp geoffrey sanhueza'
 'ishatara mayb bay area thang dammit' …
 'destini nevertheless hooray member wonder safe trip' 'feel well'
 'supersandro thank']
```

[ ]:
```
#converting textual data to numerical data

vectorizer=TfidfVectorizer()

x_train=vectorizer.fit_transform(x_train)
x_test=vectorizer.transform(x_test)
```

[ ]:
```
print(x_train)
```

```
  (0, 436713)   0.27259876264838384
  (0, 354543)   0.3588091611460021
  (0, 185193)   0.5277679060576009
  (0, 109306)   0.3753708587402299
  (0, 235045)   0.41996827700291095
  (0, 443066)   0.4484755317023172
  (1, 160636)   1.0
  (2, 109306)   0.4591176413728317
  (2, 124484)   0.1892155960801415
  (2, 407301)   0.18709338684973031
  (2, 129411)   0.29074192727957143
  (2, 406399)   0.32105459490875526
  (2, 433560)   0.3296595898028565
  (2, 77929)    0.31284080750346344
  (2, 443430)   0.3348599670252845
  (2, 266729)   0.24123230668976975
  (2, 409143)   0.15169282335109835
  (2, 178061)   0.1619010109445149
  (2, 150715)   0.18803850583207948
  (2, 132311)   0.2028971570399794
  (2, 288470)   0.16786949597862733
  (3, 406399)   0.29029991238662284
  (3, 158711)   0.4456939372299574
  (3, 151770)   0.278559647704793
  (3, 56476)    0.5200465453608686
```

```
:     :
(1279996, 318303)     0.21254698865277744
(1279996, 434014)     0.27189450523324465
(1279996, 390130)     0.2206474219107611
(1279996, 373144)     0.35212500999832036
(1279996, 238077)     0.5249170684084672
(1279996, 238078)     0.5606696159563151
(1279997, 5685)       0.48650358607431304
(1279997, 273084)     0.4353549002982409
(1279997, 112591)     0.7574829183045267
(1279998, 412553)     0.2816582375021589
(1279998, 93795)      0.21717768937055476
(1279998, 169461)     0.2659980990397061
(1279998, 124765)     0.32241752985927996
(1279998, 435463)     0.2851807874350361
(1279998, 153281)     0.28378968751027456
(1279998, 156297)     0.3137096161546449
(1279998, 162047)     0.34691726958159064
(1279998, 275288)     0.38703346602729577
(1279998, 385313)     0.4103285865588191
(1279999, 242268)     0.19572649660865402
(1279999, 31410)      0.248792678366695
(1279999, 435572)     0.31691096877786484
(1279999, 433612)     0.3607341026233411
(1279999, 135384)     0.6130934129868719
(1279999, 96224)      0.5416162421321443
```

[ ]:
```
print(x_test)
```

```
(0, 15110)    0.1719352837797837
(0, 31168)    0.1624772418052177
(0, 67828)    0.26800375270827315
(0, 106069)   0.36555450010904555
(0, 132364)   0.255254889555786
(0, 138164)   0.23688292264071406
(0, 171378)   0.2805816206356074
(0, 271016)   0.45356623916588285
(0, 279082)   0.17825180109103442
(0, 388348)   0.2198507607206174
(0, 398906)   0.34910438732642673
(0, 409143)   0.3143047059807971
(0, 420984)   0.17915624523539805
(1, 6463)     0.30733520460524466
(1, 15110)    0.211037449588008
(1, 145393)   0.575262969264869
(1, 217562)   0.40288153995289894
(1, 256777)   0.28751585696559306
(1, 348135)   0.4739279595416274
```

```
  (1, 366203)   0.24595562404108307
  (2, 22532)    0.3532582957477176
  (2, 34401)    0.37916255084357414
  (2, 89448)    0.36340369428387626
  (2, 183312)   0.5892069252021465
  (2, 256834)   0.2564939661498776
  :     :
  (319994, 443794)      0.2782185641032538
  (319995, 107868)      0.33399349737546963
  (319995, 109379)      0.3020896484890833
  (319995, 155493)      0.2770682832971669
  (319995, 213324)      0.2683969144317079
  (319995, 232891)      0.2574127854589077
  (319995, 296662)      0.3992485679384015
  (319995, 315813)      0.2848229914563413
  (319995, 324496)      0.36131679336475747
  (319995, 416257)      0.23816465111736282
  (319995, 420984)      0.22631428606830148
  (319995, 444934)      0.32110928175992615
  (319996, 397506)      0.9101400928717545
  (319996, 438709)      0.4143006291901984
  (319997, 98792)       0.4463892055808332
  (319997, 169411)      0.403381646999604
  (319997, 261286)      0.37323893626855326
  (319997, 288421)      0.48498483387153407
  (319997, 349904)      0.32484594100566083
  (319997, 416695)      0.29458327588067873
  (319997, 444770)      0.2668297951055569
  (319998, 130192)      0.6941927210956169
  (319998, 438748)      0.719789181620468
  (319999, 389755)      0.9577980203954275
  (319999, 400636)      0.2874420848216212
```

Training the Machine Learning Model

Logistic Regression

[ ]:
```
model=LogisticRegression(max_iter=1000)
```

[ ]:
```
model.fit(x_train,y_train)
```

[ ]:
```
LogisticRegression(max_iter=1000)
```

MODEL EVALUATION

ACCURACY SCORE

[ ]:
```
x_train_prediction=model.predict(x_train)
training_data_accuracy=accuracy_score(x_train_prediction,y_train)
```

[ ]:
```
print('Accuracy score of train data :',training_data_accuracy)
```

```
Accuracy score of train data : 0.79871953125
```

[ ]:
```
x_test_prediction=model.predict(x_test)
test_data_accuracy=accuracy_score(x_test_prediction,y_test)
```

[ ]:
```
print('Accuracy score of test data :',test_data_accuracy)
```

```
Accuracy score of test data : 0.77668125
```

model accuracy=77.6%

saving trained model

[ ]:
```
import pickle
```

[ ]:
```
filename='trained_model.sav'
pickle.dump(model,open(filename,'wb'))
```

using the saved model for future predictions

[ ]:
```
#loading the saved model
loaded_model=pickle.load(open('/content/trained_model.sav','rb'))
```

[ ]:
```
x_new=x_test[200]
print(y_test[200])

prediction=model.predict(x_new)
print(prediction)

if(prediction[0]==0):
  print('negative tweet')

else:
  print('positive tweet')
```

```
1
[1]
positive tweet
```

[ ]:
```

```