In [ ]:

```
import pandas as pd
import re
```

In [ ]:

```
train=pd.read_csv("train.csv")
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-2-bd81a0548298> in <module>()
----> 1 train=pd.read_csv("train.csv")

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers.py in read_csv(filepath_or_buffe
r, sep, delimiter, header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols,
dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfoo
ter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates
, infer_datetime_format, keep_date_col, date_parser, dayfirst, cache_dates, iterator, chu
nksize, compression, thousands, decimal, lineterminator, quotechar, quoting, doublequote,
escapechar, comment, encoding, dialect, error_bad_lines, warn_bad_lines, delim_whitespace
, low_memory, memory_map, float_precision)
    686     )
    687 
--> 688     return _read(filepath_or_buffer, kwds)
    689 
    690 

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers.py in _read(filepath_or_buffer,
kwds)
    452 
    453     # Create the parser.
--> 454     parser = TextFileReader(fp_or_buf, **kwds)
    455 
    456     if chunksize or iterator:

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers.py in __init__(self, f, engine,
**kwds)
    946             self.options["has_index_names"] = kwds["has_index_names"]
    947 
--> 948         self._make_engine(self.engine)
    949 
    950     def close(self):

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers.py in _make_engine(self, engine)
   1178     def _make_engine(self, engine="c"):
   1179         if engine == "c":
-> 1180             self._engine = CParserWrapper(self.f, **self.options)
   1181         else:
   1182             if engine == "python":

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers.py in __init__(self, src, **kwds
)
   2008         kwds["usecols"] = self.usecols
   2009 
-> 2010         self._reader = parsers.TextReader(src, **kwds)
   2011         self.unnamed_cols = self._reader.unnamed_cols
   2012 

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader.__cinit__()

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._setup_parser_source()

FileNotFoundError: [Errno 2] No such file or directory: 'train.csv'
```

In [ ]:

```
train.head()
```

Out[ ]:

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

In [ ]:

```
# drop col 'id' (as it is of no use) and replace it in the same variable
train.drop("id",inplace=True,axis=1)
```

In [ ]:

```
train.head()
```

Out[ ]:

	label	tweet
0	0	@user when a father is dysfunctional and is s...
1	0	@user @user thanks for #lyft credit i can't us...
2	0	bihday your majesty
3	0	#model i love u take with u all the time in ...
4	0	factsguide: society now #motivation

In [ ]:

```
temp = train.groupby("label").size()
temp
```

Out[ ]:

```
label
0    20109
1     1493
dtype: int64
```

In [ ]:

```
import nltk
#nltk.download()
```

In [ ]:

```
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()

def clean_sentences(text):
    text = text.lower()   # convert text to lower case
    text = re.sub(r"[^a-z0-9^,!.\/']", " ", text)    # remove special char's
    text = " ".join(text.split())
    text = " ".join(stemmer.stem(word) for word in text.split())   # do stemming
    return text
```

In [ ]:

```
x = train['tweet']
y = train['label']
```

In [ ]:

```
x = x.map(lambda a: clean_sentences(a))
```

```
x = x.map(lambda u: clean_sentences(u))
```

In [ ]:

```
x.head()
```

Out[ ]:

```
0    user when a father is dysfunct and is so selfi...
1    user user thank for lyft credit i can't use ca...
2                                  bihday your majesti
3    model i love u take with u all the time in ur !!!
4                          factsguid societi now motiv
Name: tweet, dtype: object
```

In [ ]:

```
pip install sklearn
```

```
Requirement already satisfied: sklearn in /usr/local/lib/python3.7/dist-packages (0.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.7/dist-packages (fr
om sklearn) (1.0.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-pack
ages (from scikit-learn->sklearn) (3.0.0)
Requirement already satisfied: numpy>=1.14.6 in /usr/local/lib/python3.7/dist-packages (f
rom scikit-learn->sklearn) (1.19.5)
Requirement already satisfied: scipy>=1.1.0 in /usr/local/lib/python3.7/dist-packages (fr
om scikit-learn->sklearn) (1.4.1)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (fr
om scikit-learn->sklearn) (1.1.0)
```

In [ ]:

```
from sklearn.model_selection import train_test_split

 # split the dataset into training set & testing set
 # data is split in a stratified fashion
x_train, x_test, y_train, y_test = train_test_split(x,y,stratify=y,random_state=42)
```

In [ ]:

```
x_train.head()
```

Out[ ]:

```
1036     user like the spread of peanut butter on white...
2380     watch made in america o.j. simpson..... 30for3...
31605             franci underwood seen leav marseil nojok
23437    get up get get enjoy music today free app free...
2669     my 1st juic experience! notsobad healthyliv ea...
Name: tweet, dtype: object
```

In [ ]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [ ]:

```
vectorizer = TfidfVectorizer(stop_words='english')
```

In [ ]:

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

In [ ]:

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

In [ ]:

```
from sklearn.svm import LinearSVC
```

In [ ]:

```
model = LinearSVC(C=1.05, tol=0.5)
```

In [ ]:

```
model.fit(x_train,y_train)
```

Out[ ]:

```
LinearSVC(C=1.05, tol=0.5)
```

In [ ]:

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, f1_score,
recall_score
confusion_matrix(y_test,model.predict(x_test))
```

Out[ ]:

```
array([[7372,   58],
       [ 228,  333]])
```

In [ ]:

```
accuracy_score(y_test,model.predict(x_test))
```

Out[ ]:

```
0.964209735952947
```

In [ ]:

```
recall_score(y_test,model.predict(x_test))
```

Out[ ]:

```
0.5935828877005348
```

In [ ]:

```
precision_score(y_test,model.predict(x_test))
```

Out[ ]:

```
0.8516624040920716
```

In [ ]:

```
f1_score(y_test,model.predict(x_test))
```

Out[ ]:

```
0.6995798319327731
```

In [ ]:

```
# sample_text = ['I hate you']
# sample_text = ['I dont hate you']
sample_text = ['you are a bad person.']
sample_text = list(map(lambda a: clean_sentences(a), sample_text))
sample_text
```

Out[ ]:

```
['you are a bad person.']
```

In [ ]:

```
sample_text = vectorizer.transform(sample_text)
```

In [ ]:

```
model.predict(sample_text)[0]
```

Out[ ]:

```
0
```