Pree Thiengburanathum

pree.t@cmu.ac.th

workshop part 1

We use wisesight sentiment corpus, can be downloaded at

https://github.com/PyThaiNLP/wisesight-sentiment/ or

https://www.kaggle.com/c/wisesight-sentiment/data

Perform text classification on Thai social media posts

train.txt - the training data. Each line is a piece of text.
train_label.txt - the label of the training data. Each line is the label corresponding to the same line in train.txt.
test.txt - the test data. Each line is a piece of text.
test_majority.csv - a sample submission file using the majority class in the correct format.
The Id in test_majority.csv is the line number in test.txt

no feature engineering in this notebook, teaching proposed only!

```
%matplotlib inline
%reload_ext autoreload
%autoreload 2
```

```
import re
import pandas as pd
from matplotlib import pyplot as plt

!pip install https://github.com/PyThaiNLP/pythainlp/archive/dev.zip
!pip install emoji
from pythainlp import word_tokenize

```

```
Collecting https://github.com/PyThaiNLP/pythainlp/archive/dev.zip
  Downloading https://github.com/PyThaiNLP/pythainlp/archive/dev.zip
     \ 11.7 MB 140 kB/s
Collecting python-crfsuite>=0.9.6
  Downloading python_crfsuite-0.9.7-cp37-cp37m-manylinux1_x86_64.whl (743 kB)
     |████████████████████████████████| 743 kB 7.2 MB/s
Requirement already satisfied: requests>=2.22.0 in /usr/local/lib/python3.7/dist-packag
Collecting tinydb>=3.0
  Downloading tinydb-4.5.2-py3-none-any.whl (23 kB)
```

```
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/li
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-pack
Collecting typing-extensions<4.0.0,>=3.10.0
  Downloading typing_extensions-3.10.0.2-py3-none-any.whl (26 kB)
Building wheels for collected packages: pythainlp
  Building wheel for pythainlp (setup.py) ... done
  Created wheel for pythainlp: filename=pythainlp-3.0.0.dev0-py3-none-any.whl size=1150
  Stored in directory: /tmp/pip-ephem-wheel-cache-wajjhmpy/wheels/87/69/e8/234fe11c5f2e
Successfully built pythainlp
Installing collected packages: typing-extensions, tinydb, python-crfsuite, pythainlp
  Attempting uninstall: typing-extensions
    Found existing installation: typing-extensions 3.7.4.3
    Uninstalling typing-extensions-3.7.4.3:
      Successfully uninstalled typing-extensions-3.7.4.3
ERROR: pip's dependency resolver does not currently take into account all the packages
tensorflow 2.6.0 requires typing-extensions~=3.7.4, but you have typing-extensions 3.10
Successfully installed pythainlp-3.0.0.dev0 python-crfsuite-0.9.7 tinydb-4.5.2 typing-e
Collecting emoji
  Downloading emoji-1.6.0.tar.gz (168 kB)
     |████████████████████████████████| 168 kB 6.6 MB/s
Building wheels for collected packages: emoji
  Building wheel for emoji (setup.py) ... done
  Created wheel for emoji: filename=emoji-1.6.0-py3-none-any.whl size=168256 sha256=11b
  Stored in directory: /root/.cache/pip/wheels/f7/d7/74/c720aaf345a042b0c2d74361873258c
Successfully built emoji
Installing collected packages: emoji
Successfully installed emoji-1.6.0
```

```
import emoji
from google.colab import drive
drive.mount('/content/gdrive')
data_path = '/content/gdrive/My Drive/Colab Notebooks/WiseSight Sentiment/kaggle-competiti
```

```
Mounted at /content/gdrive
```

```
!ls '/content/gdrive/My Drive/Colab Notebooks/WiseSight Sentiment/kaggle-competition'
```

```
competition.ipynb  test_majority.csv  text_generation.ipynb  train.txt
README.md          test_solution.csv  train_label.txt
test_label.txt     test.txt           train_model.py
```

```
# we open the given training set, namely train.txt and the target feature, namely train_la
# then, we save as csv to the current directory.
features = []
targets = []

with open(data_path + "train.txt") as f:
    for line in f:
        features.append(line.strip())

```

```
with open(data_path+ "train_label.txt") as f:
    for line in f:
        targets .append(line.strip())

df = pd.DataFrame({ "targets": targets, "features": features })

df.to_csv("train.csv", index=False)
df.shape
```

```
(24063, 2)
```

```
# we do the sampe for testing
features = []
targets = []

with open(data_path + "test.txt") as f:
    for line in f:
        features.append(line.strip())

with open(data_path+ "test_label.txt") as f:
    for line in f:
        targets .append(line.strip())

df_test = pd.DataFrame({ "targets": targets, "features": features })

df_test.to_csv("test.csv", index=False)
df_test.shape
```

```
(2674, 2)
```

```
df.head(3)
```

	targets	features
0	neu	ประเทศเราผลิตและส่งออกยาสูบเยอะสุดในโลกจิงป่าวคับ
1	neu	คะ
2	neg	อิเหี้ยออมทำกูอยากกินเอ็มเค

```
# check for any missing vlue
df.isnull().sum()
```

```
targets     0
features    0
dtype: int64
```

```
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.model_selection import cross_val_score

# try different machine-learning algorithm
#knn = KNeighborsClassifier(n_neighbors=5, metric='euclidean')
logistic = LogisticRegression(C=2., penalty='l2', solver='liblinear', dual=False, multi_cl


# demonstrate w/o pipelining
coun_vect = CountVectorizer(max_features=15000)
count_matrix = coun_vect.fit_transform( df['features'] )
count_matrix_test = coun_vect.fit_transform( df_test['features'] )


# once we have the IDF values, wecan now compute the tf-idf scores for any document or set
tfidf_trans = TfidfTransformer()
vectors = tfidf_trans.fit_transform(count_matrix)
vectors_test = tfidf_trans.fit_transform(count_matrix_test)

# demonstrate using pipelining
#logis = Pipeline([("count_vectorizer", CountVectorizer(analyzer=lambda x: x)), ("tfidf",
count_matrix, vectors, count_matrix_test, vectors_test
```

```
(<24063x15000 sparse matrix of type '<class 'numpy.int64'>'
        with 251593 stored elements in Compressed Sparse Row format>,
 <24063x15000 sparse matrix of type '<class 'numpy.float64'>'
        with 251593 stored elements in Compressed Sparse Row format>,
 <2674x15000 sparse matrix of type '<class 'numpy.int64'>'
        with 35378 stored elements in Compressed Sparse Row format>,
 <2674x15000 sparse matrix of type '<class 'numpy.float64'>'
        with 35378 stored elements in Compressed Sparse Row format>)
```

```
# k-fold crossvalidtion to get generalize of model performance here.
score = cross_val_score( logistic, vectors, df['targets'].to_numpy(), cv = 5).mean()
#score = cross_val_score( logistic, df['features'].to_numpy(), df['targets'].to_numpy(), c


print(score)
```

```
0.6456379309362166
```

```
# see actual performance with test set
logistic.fit(vectors, df.targets.to_numpy())
logistic.score(vectors_test, df_test.targets.to_numpy())
```

```
0.5329094988780853
```

```
#lets pre-processing the features :(
# took it from https://github.com/PyThaiNLP/wisesight-sentiment/blob/master/kaggle-competi
def replace_url(text):
    URL_PATTERN = r"""(?i)\b((?:https?:(?:/{1,3}|[a-z0-9%])|[a-z0-9.\-]+[.](?:com|net|org|
    return re.sub(URL_PATTERN, 'xxurl', text)
```

```
 6
 7 def replace_rep(text):
 8     def _replace_rep(m):
 9         c,cc = m.groups()
10         return f'{c}xxrep'
11     re_rep = re.compile(r'(\S)(\1{2,})')
12     return re_rep.sub(_replace_rep, text)
13
14 def ungroup_emoji(toks):
15     res = []
16     for tok in toks:
17         if emoji.emoji_count(tok) == len(tok):
18             for char in tok:
19                 res.append(char)
20         else:
21             res.append(tok)
22     return res
23
24 def process_text(text):
25     #pre rules
26     res = text.lower().strip()
27     res = replace_url(res)
28     res = replace_rep(res)
29
30     #tokenize
31     res = [word for word in word_tokenize(res) if word and not re.search(pattern=r"\s+", s
32
33     #post rules
34     res = ungroup_emoji(res)
35
36     return res
```

```
1 # let's do it again properly with proper sampling :)
2 pos_df = df[df.targets == "pos"]
3 neu_df = df[df.targets == "neu"]
4 neg_df = df[df.targets== "neg"]
```

```
1 sentiment_df = pd.concat([neg_df, pos_df, neu_df])
2 sentiment_df = sentiment_df.reset_index(drop=True)
3 sentiment_df.groupby("targets").features.describe()
```

targets	count	unique	top	freq
neg	6140	6129	#ERROR!	12
neu	13105	13087	#ERROR!	19
pos	4300	4300	สงขลายังสาขาไหนมั่งนิ ไซ่ไม่บอกให้แหมด	1

```
sentiment_df['processed'] = sentiment_df.features.map(lambda x: '|'.join(process_text(x)))
```

```
# performing random hold-out method
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(sentiment_df['processed'], sentiment_d
```

```
X_train.head()
```

```
177          มัน|คือ|ฝี|หนอง|xxrep|😳|#|เจริญอาหาร|มั้ย|ล่ะ
3        ลิป|มัน|ของ|แบร์น|mistine|ราคา|กลาง|ๆ|ไม่|แพง...
17957                                        ครับ|เสี่ย|บอส
16938    ทุกสิ่ง|อย่าง|ที่|พวก|เค้า|ต้าน|เรา|ก็|แค่|ประ...
13996    จำนวน|speed|ของ|เกียร์|ไม่|ได้|บอ|กว่า|รถ|จะ|ต...
Name: processed, dtype: object
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer()
tfidf_fit = tfidf.fit(sentiment_df['processed'])
text_train = tfidf_fit.transform(X_train)
text_valid = tfidf_fit.transform(X_test)
text_train.shape, text_valid.shape
```

```
((18836, 16257), (4709, 16257))
```

```
score = cross_val_score( logistic, text_train, y_train, cv = 5).mean()
score
```

```
0.6956892518953004
```

```
1
```

✓ 2s completed at 12:06 PM