

HOMEWORK 6: TEXT CLASSIFICATION

In this homework, you will create models to classify texts from TRUE call-center. There are two classification tasks:

1. Action Classification: Identify which action the customer would like to take (e.g. enquire, report, cancel)
2. Object Classification: Identify which object the customer is referring to (e.g. payment, true money, internet, roaming)

We will focus only on the Object Classification task for this homework.

In this homework, you are asked to compare different text classification models in terms of accuracy and inference time.

You will need to build 3 different models.

1. A model based on tf-idf
2. A model based on MUSE
3. A model based on wangchanBERTa

You will be asked to submit 3 different files (.pdf from .ipynb) that do the 3 different models. Finally, answer the accuracy and runtime numbers in MCV.

This homework is quite free form, and your answer may vary. We hope that the processing during the course of this assignment will make you think more about the design choices in text classification.

```
In [1]: !wget --no-check-certificate https://www.dropbox.com/s/37u83g55p19kvr1/clean-pho
```

```
--2025-02-16 14:31:25-- https://www.dropbox.com/s/37u83g55p19kvrl/clean-phone-da
ta-for-students.csv
Resolving www.dropbox.com (www.dropbox.com)... 162.125.1.18, 2620:100:6016:18::a2
7d:112
Connecting to www.dropbox.com (www.dropbox.com)|162.125.1.18|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://www.dropbox.com/scl/fi/8h8hvsW9uj6o0524lfe4i/clean-phone-data-f
or-students.csv?rlkey=lwv5xbf16jerehnv3lfgq5ue6 [following]
--2025-02-16 14:31:25-- https://www.dropbox.com/scl/fi/8h8hvsW9uj6o0524lfe4i/cle
an-phone-data-for-students.csv?rlkey=lwv5xbf16jerehnv3lfgq5ue6
Reusing existing connection to www.dropbox.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://uce9b4d9ce07540704f730162084.dl.dropboxusercontent.com/cd/0/inl
ine/CkNwY6dA0mIUMFAMX-HKzPI5krHfWCSD1PL4FmlWgrPqx-9wx0E1EOYeYbzGwZxsq54tbI1550tjn
CAqKzphbFIpKUyLGINC8t5FPYeeCrdtalrPlTbDFuEmG5BT2KP0lb8/file# [following]
--2025-02-16 14:31:25-- https://uce9b4d9ce07540704f730162084.dl.dropboxuserconte
nt.com/cd/0/inline/CkNwY6dA0mIUMFAMX-HKzPI5krHfWCSD1PL4FmlWgrPqx-9wx0E1EOYeYbzGwZ
xsq54tbI1550tjnCAqKzphbFIpKUyLGINC8t5FPYeeCrdtalrPlTbDFuEmG5BT2KP0lb8/file
Resolving uce9b4d9ce07540704f730162084.dl.dropboxusercontent.com (uce9b4d9ce07540
704f730162084.dl.dropboxusercontent.com)... 162.125.1.15, 2620:100:6016:15::a27d:
10f
Connecting to uce9b4d9ce07540704f730162084.dl.dropboxusercontent.com (uce9b4d9ce0
7540704f730162084.dl.dropboxusercontent.com)|162.125.1.15|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2518977 (2.4M) [text/plain]
Saving to: 'clean-phone-data-for-students.csv'

clean-phone-data-fo 100%[=====] 2.40M --.-KB/s in 0.04s

2025-02-16 14:31:26 (54.9 MB/s) - 'clean-phone-data-for-students.csv' saved [2518
977/2518977]
```

```
In [2]: !pip install pythainlp
```

```
Collecting pythainlp
  Downloading pythainlp-5.0.5-py3-none-any.whl.metadata (7.5 kB)
Requirement already satisfied: requests>=2.22.0 in /usr/local/lib/python3.10/dist
-packages (from pythainlp) (2.32.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python
3.10/dist-packages (from requests>=2.22.0->pythainlp) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-pac
kages (from requests>=2.22.0->pythainlp) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/di
st-packages (from requests>=2.22.0->pythainlp) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/di
st-packages (from requests>=2.22.0->pythainlp) (2025.1.31)
Downloading pythainlp-5.0.5-py3-none-any.whl (17.9 MB)
----- 17.9/17.9 MB 84.4 MB/s eta 0:00:00:0
0:0100:01
Installing collected packages: pythainlp
Successfully installed pythainlp-5.0.5
```

Import Libs

```
In [3]: %matplotlib inline
import pandas
import sklearn
import numpy as np
```

```
import matplotlib.pyplot as plt
import pandas as pd

from torch.utils.data import Dataset
from IPython.display import display
from collections import defaultdict
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from pythainlp.tokenize import word_tokenize
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from pythainlp.corpus.common import thai_stopwords
import time
```

Loading data

First, we load the data from disk into a Dataframe.

A Dataframe is essentially a table, or 2D-array/Matrix with a name for each column.

```
In [4]: data_df = pd.read_csv('clean-phone-data-for-students.csv')
```

Let's preview the data.

```
In [5]: # Show the top 5 rows
display(data_df.head())
# Summarize the data
data_df.describe()
```

	Sentence Utterance	Action	Object
0	<PHONE_NUMBER_REMOVED> ผมไปจ่ายเงินที่ Counte...	enquire	payment
1	internet ยังความเร็วอยู่เท่าไร ครับ	enquire	package
2	ตะกี้ไปชำระค่าบริการไปแล้ว แต่ยังใช้งานไม่ได้...	report	suspend
3	พี่คะยังใช้ internet ไม่ได้เลยคะ เป็นเครือ...	enquire	internet
4	ฮาโหล คะ พอดีว่าเมื่อวานเปิดซิมทรูฟ แต่มันโ...	report	phone_issues

```
Out[5]:
```

	Sentence Utterance	Action	Object
count	16175	16175	16175
unique	13389	10	33
top	บริการอื่นๆ	enquire	service
freq	97	10377	2525

Data cleaning

We call the DataFrame.describe() again. Notice that there are 33 unique labels/classes for object and 10 unique labels for action that the model will try to predict. But there are

unwanted duplications e.g. Idd,idd,lotalty_card,Lotalty_card

Also note that, there are 13389 unqiue sentence utterances from 16175 utterances. You have to clean that too!

#TODO 0.1:

You will have to remove unwanted label duplications as well as duplications in text inputs. Also, you will have to trim out unwanted whitespaces from the text inputs. This shouldn't be too hard, as you have already seen it in the demo.

```
In [6]: display(data_df.describe())
display(data_df.Object.unique())
display(data_df.Action.unique())
```

	Sentence Utterance	Action	Object
count	16175	16175	16175
unique	13389	10	33
top	บริการอื่นๆ	enquire	service
freq	97	10377	2525

```
array(['payment', 'package', 'suspend', 'internet', 'phone_issues',
      'service', 'nonTrueMove', 'balance', 'detail', 'bill', 'credit',
      'promotion', 'mobile_setting', 'iservice', 'roaming', 'truemoney',
      'information', 'lost_stolen', 'balance_minutes', 'idd',
      'TrueMoney', 'garbage', 'Payment', 'IDD', 'ringtone', 'Idd',
      'rate', 'loyalty_card', 'contact', 'officer', 'Balance', 'Service',
      'Loyalty_card'], dtype=object)
array(['enquire', 'report', 'cancel', 'Enquire', 'buy', 'activate',
      'request', 'Report', 'garbage', 'change'], dtype=object)
```

```
In [7]: data_df.columns
```

```
Out[7]: Index(['Sentence Utterance', 'Action', 'Object'], dtype='object')
```

```
In [8]: start = time.time()
cols = ["Sentence Utterance", "Object"]
data_df = data_df[cols]
data_df.columns = ["input", "raw_label"]

data_df["clean_label"] = data_df["raw_label"].str.lower().copy()
data_df.drop("raw_label", axis=1, inplace=True)

data_df["input"] = data_df["input"].str.strip()

data_df = data_df.drop_duplicates(subset=['input'], keep='first')
```

```
In [9]: display(data_df["clean_label"].unique())
display(data_df.describe())
display(data_df.head())
```

```
array(['payment', 'package', 'suspend', 'internet', 'phone_issues',
      'service', 'nontruemove', 'balance', 'detail', 'bill', 'credit',
      'promotion', 'mobile_setting', 'iservice', 'roaming', 'truemoney',
      'information', 'lost_stolen', 'balance_minutes', 'idd', 'garbage',
      'ringtone', 'rate', 'loyalty_card', 'contact', 'officer'],
      dtype=object)
```

	input	clean_label
count	13367	13367
unique	13367	26
top	สอบถามโปรโมชั่นปัจจุบันที่ใช้อยู่ค่ะ	
freq	1	2108

	input	clean_label
0	<PHONE_NUMBER_REMOVED> ผมไปจ่ายเงินที่ Counter...	payment
1	internet ยังความเร็วอยู่เท่าไรครับ	package
2	ตะกี้ไปชำระค่าบริการไปแล้ว แต่ยังใช้งานไม่ได้ ค่ะ	suspend
3	พี่คะยังใช้ internet ไม่ได้เลยคะ เป็นเครื่อง...	internet
4	ฮาโหล ค่ะ พอตีหัวเมื่อวานเปิดซิมทรูฟ แต่มันโท...	phone_issues

Split data into train, validation, and test sets (normally the ratio will be 80:10:10 , respectively). We recommend to use `train_test_split` from `scikit-learn` to split the data into train, validation, test set.

In addition, it should split the data that distribution of the labels in train, validation, test set are similar. There is **stratify** option to handle this issue.

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Make sure the same data splitting is used for all models.

```
In [10]: data_x = np.array(list(data_df["input"]))
data_y_tmp = np.array(list(data_df["clean_label"]))
data_y = []

map_label_num = {y.strip():i for i,y in enumerate(list(data_df["clean_label"].unique()))}
map_num_label = {i:y.strip() for i,y in enumerate(list(data_df["clean_label"].unique()))}

for i in range(len(data_y_tmp)):
    data_y.append(int(map_label_num[data_y_tmp[i]]))
data_y = np.array(data_y)
print(len(data_y))
```

13367

```
In [11]: unique, counts = np.unique(data_y, return_counts=True)
valid_classes = unique[counts >= 10]
valid_indices = np.isin(data_y, valid_classes)
data_x, data_y = data_x[valid_indices], data_y[valid_indices]
```

```
In [12]: X_train, X_temp, y_train, y_temp = train_test_split(data_x, data_y, test_size=0.
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50,

print("Train size:", len(X_train))
print("Validation size:", len(X_val))
print("Test size:", len(X_test))
```

Train size: 10690

Validation size: 1336

Test size: 1337

Model 1 TF-IDF

Build a model to train a tf-idf text classifier. Use a simple logistic regression model for the classifier.

For this part, you may find this [tutorial](#) helpful.

Below are some design choices you need to consider to accomplish this task. Be sure to answer them when you submit your model.

What tokenizer will you use? Why?

Ans:

Will you ignore some stop words (a, an, the, to, etc. for English) in your tf-idf? Is it important? Pythainlp provides a list of stopwords if you want to use (https://pythainlp.org/docs/2.0/api/corpus.html#pythainlp.corpus.common.thai_stopwords)

Ans:

The dictionary of TF-IDF is usually based on the training data. How many words in the test set are OOVs?

Ans:

```
In [13]: def thai_tokenizer(text):
return word_tokenize(text, keep_whitespace=False)
```

```
In [14]: from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from pythainlp.corpus.common import thai_stopwords
import numpy as np
import time

# Define stopwords
stopwords = list(thai_stopwords())

# Initialize TF-IDF Vectorizer
vectorizer = TfidfVectorizer(
    tokenizer=None, # Use default tokenization
    stop_words=stopwords, # Remove Thai stopwords
    max_features=10000 # Limit vocabulary size to avoid overfitting
```

```

)

# Create a logistic regression pipeline
model = Pipeline([
    ("tfidf", vectorizer),
    ("classifier", LogisticRegression(max_iter=1000, random_state=42))
])

# Train model
start = time.time()
model.fit(X_train, y_train)
end = time.time()
# Evaluate model
print(f"Training time: {end - start:.4f} seconds")
train_acc = model.score(X_train, y_train)
val_acc = model.score(X_val, y_val)
test_acc = model.score(X_test, y_test)

print(f"Train Accuracy: {train_acc:.4f}")
print(f"Validation Accuracy: {val_acc:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")

# Check OOV words (words in test set not seen in train set)
train_vocab = set(vectorizer.get_feature_names_out())
test_vocab = set(word for sentence in X_test for word in sentence.split())
oov_words = test_vocab - train_vocab
print(f"Number of OOV words: {len(oov_words)}")

```



```

# Define stopwords
stopwords = list(thai_stopwords())

# Initialize TF-IDF Vectorizer
vectorizer = TfidfVectorizer(
    tokenizer=thai_tokenizer, # Use default tokenization
    stop_words=stopwords, # Remove Thai stopwords
    max_features=10000 # Limit vocabulary size to avoid overfitting
)

# Create a Logistic regression pipeline
model = Pipeline([
    ("tfidf", vectorizer),
    ("classifier", LogisticRegression(max_iter=1000, random_state=42))
])

# Train model

model.fit(X_train, y_train)
end = time.time()

# Evaluate model
train_acc = model.score(X_train, y_train)
val_acc = model.score(X_val, y_val)
test_acc = model.score(X_test, y_test)
print(f"Training time: {end - start:.4f} seconds")
print(f"Train Accuracy: {train_acc:.4f}")
print(f"Validation Accuracy: {val_acc:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")

# Check OOV words (words in test set not seen in train set)
train_vocab = set(vectorizer.get_feature_names_out())
test_vocab = set(word for sentence in X_test for word in sentence.split())
oov_words = test_vocab - train_vocab
print(f"Number of OOV words: {len(oov_words)}")

```

/usr/local/lib/python3.10/dist-packages/sklearn/feature_extraction/text.py:528: UserWarning: The parameter 'token_pattern' will not be used since 'tokenizer' is not None'

warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/feature_extraction/text.py:409: UserWarning: Your stop_words may be inconsistent with your preprocessing. Tokenizing the stop words generated tokens ['กระไร', 'กาลนาน', 'ขึ้น', 'ดั่งที่', 'ดี', 'ดีกว่า', 'ด้อย', 'ตัว', 'ต่อไป', 'ถัดไป', 'ทั่วถึง', 'ท่า', 'ที่จะ', 'ท่าน', 'ท้าย', 'นา', 'บอ', 'บัด', 'ระยะเวลา', 'ละ', 'วันวาน', 'สม', 'สมบูรณ์', 'สำ', 'หน้า', 'หรับ', 'หา', 'อย', 'เกีย', 'ยว', 'เก่า', 'เดี๋ยวนี', 'เย็น', 'เล่า', 'เสมือน', 'เหมือนกัน', 'แต่', 'มั่น', 'แหล', 'โง้น', 'โน้น', 'ใด', 'ไ', 'โหม', '\uffff'] not in stop_words.
warnings.warn(

Training time: 8.4621 seconds

Train Accuracy: 0.7650

Validation Accuracy: 0.6939

Test Accuracy: 0.6971

Number of OOV words: 2694

1.What tokenizer will you use? Why?

Ans: pythainlp.word_tokenize เพราะ เชื่อ ว่าออกแบบมาเพื่อภาษาไทยโดยเฉพาะ ดังที่เห็นว่าaccuracy ของ validation,test สูงกว่า

2. Will you ignore some stop words (a, an, the, to, etc. for English) in your tf-idf? Is it important? PythaiNLP provides a list of stopwords if you want to use (https://pythainlp.org/docs/2.0/api/corpus.html#pythainlp.corpus.common.thai_stopwords)

Ans: ใช่ ใช้ `pythainlp.thai_stopwords()` เพราะมันเป็นคำที่ไม่สื่อความหมายอะไรอยู่แล้ว

3. The dictionary of TF-IDF is usually based on the training data. How many words in the test set are OOVs?

Ans: 2694