

# Assignment 2 - DS4Biz Y63

## TextScraping\_Classification

### Team Detail

Team Name: sompinandsomshine

### Student 1

Student ID: 61070278

Student Full Name: นายกิตติภณ สุรุ่งเรืองสกุล

### Student 2

Student ID: 61070330

Student Full Name: นางสาวอิงฟ้า ภูมิวนานา

import package ที่ต้องใช้

- requests, bs4 ใช้ scrape ข้อมูลจากหน้าเว็บ
- pandas(pd), numpy(np) ใช้จัดการข้อมูลที่ scrape มา เช่น ทำเป็น DataFrame, แปลงเป็น csv/txt
- warnings ใช้ปิด warning ที่แจ้งต่อ
- CountVectorizer, TfidfTransformer, TfidfVectorizer ใช้เตรียมข้อมูล เช่น ทำ term weighting
- nltk ใช้ในการประมวลผลภาษาธรรมชาติ เช่น กำจัด stopwords (คำที่เกิดขึ้นบ่อยและไม่มีอิทธิพลในการจำแนก)
- Pipeline ทำให้การทำงานเป็นลำดับและนาโน้ะ กับ GridSearchCV
- train\_test\_split, cross\_val\_score, StratifiedKFold, GridSearchCV เป็นการทำ model selection และหา best hyperparameter
- neighbors(KNeighborsClassifier), xgboost(xgb), ensemble(RandomForestClassifier) model ที่จะนำมายืนยันว่าแต่ ข้าเป็นประเภทไหน

```
In [1]: import requests
import bs4

import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings('ignore')

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

import nltk
from nltk.corpus import stopwords
from sklearn.pipeline import Pipeline

from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score

from sklearn.neighbors import KNeighborsClassifier
import xgboost as xgb
from sklearn.ensemble import RandomForestClassifier

import matplotlib.pyplot as plt
```

## Data Collection

link: [http://www.it.kmitl.ac.th/~teerapong/news\\_archive/index.html](http://www.it.kmitl.ac.th/~teerapong/news_archive/index.html)

ใช้ `requests.get` ตรวจสอบว่าลิงก์ที่ต้องการ scrape ข้อมูลสามารถเข้าได้หรือไม่

```
In [3]: response = requests.get('http://www.it.kmitl.ac.th/~teerapong/news_archive/index.html') # ใช้ลิงก์ที่เราต้องการ scrape
print(response) # response 200 คือสามารถเข้าถึงได้ พร้อมดึงข้อมูล

<Response [200]>
```

ใช้ `bs4.BeautifulSoup` ในการอ่าน code ของลิงก์ที่เราได้มา

```
In [4]: html_page = bs4.BeautifulSoup(response.content, 'html.parser') # เก็บ code ที่ได้ไว้ในตัวแปร html_page
print(html_page)

<!DOCTYPE html>
```

```

<html lang="en">
<head>
<title>Online News Archive</title>
<meta charset="utf-8"/>
<meta content="width=device-width, initial-scale=1" name="viewport"/>
<meta content="noindex" name="robots"/>
<meta content="news,articles,news" name="keywords">
<meta content="Breaking News | International Headlines" property="og:title"/>
<meta content="News Archive" property="og:site_name"/>
<meta content="Latest news and more from the definitive brand of quality news." property="og:description"/>
<link href="css/bootstrap.min.css" rel="stylesheet"/>
<script src="js/jquery-3.2.1.slim.min.js"></script>
<script src="js/popper.min.js"></script>
<script src="js/tether.min.js"></script>
<script src="js/jquery-3.2.1.slim.min.js"></script>
<script src="js/popper.min.js"></script>
<script src="js/tether.min.js"></script>
<script src="js/bootstrap.min.js"></script>
<style>
    .main{ padding: 0; text-align: center; }
    .footer{ padding: 6px;text-align: center; margin-top: 1em; }

    h1
    {
        font-size: 180%;
        margin-top: 15px;
        margin-bottom: 15px;
    }
    ul {list-style-type: none;}
    li { margin-top: 5px; }

</style>
</meta></head>
<body>
<div class="container" style="margin-top: 2em;">
<div class="main">

<h1>News Article Archive</h1>
<p>Archive of all news headlines and stories, organised per month.</p>
<ul>
<li>Articles - <a href="month-jan-2017.html">January</a> [118]</li>
<li>Articles - <a href="month-feb-2017.html">February</a> [124]</li>
<li>Articles - <a href="month-mar-2017.html">March</a> [116]</li>
<li>Articles - <a href="month-apr-2017.html">April</a> [118]</li>
<li>Articles - <a href="month-may-2017.html">May</a> [115]</li>
<li>Articles - <a href="month-jun-2017.html">June</a> [115]</li>
<li>Articles - <a href="month-jul-2017.html">July</a> [122]</li>
<li>Articles - <a href="month-aug-2017.html">August</a> [116]</li>
<li>Articles - <a href="month-sep-2017.html">September</a> [113]</li>
<li>Articles - <a href="month-oct-2017.html">October</a> [124]</li>
<li>Articles - <a href="month-nov-2017.html">November</a> [122]</li>
<li>Articles - <a href="month-dec-2017.html">December</a> [115]</li>
</ul>
</div>
<div class="footer">
<span><a href="#">Terms & Conditions</a> | <a href="#">Privacy Policy</a> | <a href="#">Cookie Information</a> </span><br/>
<span>© <span class="thisyear">2019-2020</span> - Original rights holders</span>
</div>
</div>
</body>
</html>

```

คำสั่ง **select** ใช้ในการแสดง code ที่ตรงตาม selector ที่เรากำหนดไว้ โดยจะ return ค่าที่ตรงกับที่เรากำหนดไว้กลับมาทั้งหมด

In [5]: #ด้องการดึงข้อมูลของแต่ละเดือน  
 selector = 'body > div > div.main > ul > li > a' #สังเกตว่าข้อมูลจะต้องมีอยู่ใน tags ul>li>a ห้องกากหนดให้ไปดึง code ที่อยู่ในส่วนนี้ของภาษา  
 tags = html\_page.select(selector) #โดยที่จะ return code ที่ตรงกับ selector ที่กำหนดไว้ทั้งหมดและเก็บไว้ในตัวแปร tags  
 tags #เชิญกดูผลลัพธ์ว่าตรงกับที่ด้องการรึปะ

Out[5]: [[January](month-jan-2017.html),  
[February](month-feb-2017.html),  
[March](month-mar-2017.html),  
[April](month-apr-2017.html),  
[May](month-may-2017.html),  
[June](month-jun-2017.html),  
[July](month-jul-2017.html),  
[August](month-aug-2017.html),  
[September](month-sep-2017.html),  
[October](month-oct-2017.html),  
[November](month-nov-2017.html),  
[December](month-dec-2017.html)]

ได้ code ที่เป็นลิงก์มาหมดแล้วแต่ยังไม่พร้อมใช้งานเพราะ ลิงก์ที่ต้องการมีหน้าแบบนี้

[http://www.it.kmit.ac.th/~teerapong/news\\_archive/month-jan-2017.html](http://www.it.kmit.ac.th/~teerapong/news_archive/month-jan-2017.html) จึงจะทำการสร้าง **list\_archive** เพื่อรอเก็บลิงก์ที่พร้อมใช้งาน

In [6]: list\_archive = [] #list หรือเก็บลิงก์ที่จะนำไปใช้งานต่อ  
 for tag in tags: #ทำการวน for Loop จากค่าวา列表 tags ที่เก็บ code ที่มีลิงก์อยู่
 list\_archive.append('http://www.it.kmit.ac.th/~teerapong/news\_archive/' + tag['href'])
 '''นี่ tag['href'] ซึ่งคือส่วนที่อยู่ใน href เม่านั้น (ex. month-jan-2017.html) มาต่อคำว่า http://www.it.kmit.ac.th/~teerapong/news\_archive/
 เพื่อให้ได้ลิงก์ตามที่เราต้องการ และลิงก์นั้นไปเก็บไว้ใน list\_archive ที่สร้างเอาไว้'''
 list\_archive #เชิญกดูผลลัพธ์ว่าตรงกับที่ต้องการรึปะ

```
Out[6]: ['http://www.it.kmitl.ac.th/~teerapong/news_archive/month-jan-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-feb-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-mar-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-apr-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-may-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-jun-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-jul-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-aug-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-sep-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-oct-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-nov-2017.html',
 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-dec-2017.html']
```

## Create function

สร้าง Function ที่ใช้ในการ scrape

- category ประเภทข่าว -> extract\_category()
- title หัวข้อข่าว -> extract\_title()
- link ลิงค์ข่าว -> extract\_link\_title()

```
In [7]: # สร้าง Function ที่ใช้ในการ scrape category
def extract_category(html_page):
    selector = 'body > div > div.main > table > tbody > tr > td.category' # กำหนดให้ดึงค่าที่อยู่ในตารางในส่วนของคอลัมน์ Article Category
    tags = html_page.select(selector)
    category = [] # สร้าง List สำหรับ category ไว้รอดึงค่าที่ scrape ได้
    for tag in tags: # ทำการวน for Loop จากตัวแปร tags
        if tag.text.strip() != 'N/A': # จะเอาข้อมูล category ใส่ใน List ที่มี category ดำเนินการไม่เป็น N/A (ค่าว่าง)
            category.append(tag.text.strip()) # เพิ่มข้อมูลที่ scrape ได้ใส่ไว้ใน List ที่มี category
    return category
```

```
In [8]: # สร้าง Function ที่ใช้ในการ scrape title
def extract_title(html_page):
    selector = 'body > div > div.main > table > tbody > tr > td.title > a' # กำหนดให้ดึงค่าที่อยู่ในตารางในส่วนของคอลัมน์ Article Title และดึงไปที่ tag a เพราะเราควรที่มีข้อมูลจะเป็น tag a แต่ค่าที่มีข้อมูลจะเป็น tag i
    tags = html_page.select(selector)
    title = [] # สร้าง List สำหรับ title ไว้รอดึงค่าที่ scrape ได้
    for tag in tags: # ทำการวน for Loop จากตัวแปร tags
        title.append(tag.text.strip()) # เพิ่มข้อมูลที่ scrape ได้ใส่ไว้ใน List ที่มี title
    return title
```

```
In [9]: # สร้าง Function ที่ใช้ในการ scrape link
def extract_link_title(html_page):
    selector = 'body > div > div.main > table > tbody > tr > td.title > a' # กำหนดให้ดึงค่าที่อยู่ในตารางในส่วนของคอลัมน์ Article Title และดึงไปที่ tag a เพราะลิงก์ของข่าวอยู่ tag a
    tags = html_page.select(selector)
    link_title = [] # สร้าง List สำหรับ link_title ไว้รอดึงค่าที่ scrape ได้
    for tag in tags: # ทำการวน for Loop จากตัวแปร tags
        link_title.append('http://www.it.kmitl.ac.th/~teerapong/news_archive/' + tag['href']) # ทำการแทรก URL ให้เป็นลิงก์ที่พร้อมใช้งาน และคือเพิ่มข้อมูลใส่ไว้ใน List ที่มี Link_title
    return link_title
```

```
In [10]: # ตัวอย่าง
url = 'http://www.it.kmitl.ac.th/~teerapong/news_archive/month-jan-2017.html'
response = requests.get(url)
html_page = bs4.BeautifulSoup(response.content, 'html.parser')
print('example:\n', extract_category(html_page)[0], '\n', extract_title(html_page)[0], '\n', extract_link_title(html_page)[0])

example:
technology
21st-Century Sports: How Digital Technology Is Changing the Face Of The Sporting Industry
http://www.it.kmitl.ac.th/~teerapong/news_archive/article-jan-0418.html
```

สร้าง Function (extract\_news\_archive\_info()) สำหรับเรียกใช้งาน Function ในการ scrape category ประเภทข่าว, title หัวข้อข่าว, link ลิงค์ข่าว ทั้งหมด โดยใช้ค่าเดือนที่เดือนที่ต้อง category ประเภทข่าว, title หัวข้อข่าว, link ลิงค์ข่าว ในเดือนนั้นออก มาทั้งหมด และจะ return กลับมาในรูปของ DataFrame

```
In [11]: def extract_news_archive_info(url): # รับลิงก์ของเดือนที่ต้องการดึงข้อมูล
    response = requests.get(url)
    html_page = bs4.BeautifulSoup(response.content, 'html.parser') # ดึงหน้าเว็บของลิงก์นั้น

    # เอาหน้าเว็บที่ดึงออกมาไว้ใน argument แต่ละ Function ที่ได้สร้างไว้เพื่อดึงข้อมูล ผลลัพธ์ที่ได้ก็ลืมมาจ่อๆในรูปของ List
    category = extract_category(html_page)
    title = extract_title(html_page)
    link_title = extract_link_title(html_page)

    # เอาข้อมูลที่อยู่ในรูปของ List ที่ได้ก็ลืมมาแปลงให้อยู่ในรูปของ pandas Series เพื่อที่จะได้อาช้อมูลทั้ง 3 ตัวมา concat กันให้เป็น DataFrame
    category = pd.Series(category)
    title = pd.Series(title)
    link_title = pd.Series(link_title)

    # เอาข้อมูลทั้ง 3 ตัวที่อยู่ในรูปของ pandas Series และมา concat กันในแนวคอลัมน์ (เอากลั่นมาต่อๆกัน)
```

```

result = pd.concat([category, title, link_title], axis=1)
#rename ชื่อ column ให้ตรงกับที่ค้องการใช้งาน โดยจะเปลี่ยนชื่อเป็น category-ประเภทข่าว, title-หัวข้อข่าว, link-ลิงก์ข่าว
77 result = result.rename(columns={0: "category", 1: "title", 2: "link"})
return result #return DataFrame

```

In [12]: #ดูว่าอย่าง บันทึก DataFrame ที่ Function extract\_news\_archive\_info() return กลับมา  
#ข้อมูลที่ return กลับมามีเป็นชื่อทั้งหมดในเดือนที่เราสนใจ  
url = 'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-jan-2017.html'  
extract\_news\_archive\_info(url)

Out[12]:

	category	title	link
0	technology	21st-Century Sports: How Digital Technology Is...	http://www.it.kmit.ac.th/~teerapong/news_arch...
1	business	Asian quake hits European shares	http://www.it.kmit.ac.th/~teerapong/news_arch...
2	technology	BT offers free net phone calls	http://www.it.kmit.ac.th/~teerapong/news_arch...
3	business	Barclays shares up on merger talk	http://www.it.kmit.ac.th/~teerapong/news_arch...
4	sport	Barkley fit for match in Ireland	http://www.it.kmit.ac.th/~teerapong/news_arch...
...	...	...	...
113	business	Worldcom boss 'left books alone'	http://www.it.kmit.ac.th/~teerapong/news_arch...
114	technology	Xbox 2 may be unveiled in summer	http://www.it.kmit.ac.th/~teerapong/news_arch...
115	technology	Xbox power cable 'fire fear'	http://www.it.kmit.ac.th/~teerapong/news_arch...
116	business	Yangtze Electric's profits double	http://www.it.kmit.ac.th/~teerapong/news_arch...
117	business	Yukos loses US bankruptcy battle	http://www.it.kmit.ac.th/~teerapong/news_arch...

118 rows × 3 columns

## Create DataFrame

สร้าง DataFrame เพื่อรับข้อมูลที่ scrape ได้

ซึ่งข้อมูลที่เราจะ scrape คือลิงก์ข่าวในแต่ละเดือนใน list\_archive โดยเราจะ scrape category ประเภทข่าว, title หัวข้อข่าว, link ลิงก์ข่าว

In [13]: #สร้าง DataFrame ที่รับข้อมูลที่ scrape ได้และกำหนดชื่อ column ตามสิ่งที่จะ scrape  
df = pd.DataFrame(columns=['category', 'title', 'link'])  
df #ดู df เป็นๆที่รับข้อมูลที่ scrape ได้

Out[13]:

category	title	link
----------	-------	------

## เอา list\_archive (ลิงก์เดือนทั้งหมด) ไปเข้า Function extract\_news\_archive\_info()

ขั้นตอนนี้ทำเพื่อให้ได้ข้าทั้งหมดในทุกๆเดือนมารวบกันเป็น DataFrame อันเดียว

In [14]: list\_archive

Out[14]: ['http://www.it.kmit.ac.th/~teerapong/news\_archive/month-jan-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-feb-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-mar-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-apr-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-may-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-jun-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-jul-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-aug-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-sep-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-oct-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-nov-2017.html',  
'http://www.it.kmit.ac.th/~teerapong/news\_archive/month-dec-2017.html']

เอาแต่ละลิงก์ใน list\_archive ไปเข้า function extract\_news\_archive\_info() และเพิ่มเข้าไปใน DataFrame ที่เราเตรียมไว้

In [15]: for url in list\_archive: #วน for Loop เพื่อเอาริงก์เดือนที่ลอกขึ้นไปเข้า function  
 result = extract\_news\_archive\_info(url) #เอาสิ่งที่เดือนไปเข้า function และเป็น DataFrame ที่ return กลับมาไว้  
 df = df.append(result, ignore\_index=True) #เอา result ที่ได้จาก function ไปเพิ่มใน DataFrame ที่สร้างไว้

In [16]: df #หน้าตาของ DataFrame ที่มีข้อมูล category ประเภทข่าว, title หัวข้อข่าว, Link ลิงก์ข่าว ทั้งหมด

Out[16]:

	category	title	link
0	technology	21st-Century Sports: How Digital Technology Is...	http://www.it.kmit.ac.th/~teerapong/news_arch...
1	business	Asian quake hits European shares	http://www.it.kmit.ac.th/~teerapong/news_arch...
2	technology	BT offers free net phone calls	http://www.it.kmit.ac.th/~teerapong/news_arch...
3	business	Barclays shares up on merger talk	http://www.it.kmit.ac.th/~teerapong/news_arch...
4	sport	Barkley fit for match in Ireland	http://www.it.kmit.ac.th/~teerapong/news_arch...
...	...	...	...
1403	sport	Woodward eyes Brennan for Lions	http://www.it.kmit.ac.th/~teerapong/news_arch...
1404	business	WorldCom trial starts in New York	http://www.it.kmit.ac.th/~teerapong/news_arch...
1405	business	Yukos accused of lying to court	http://www.it.kmit.ac.th/~teerapong/news_arch...
1406	business	Yukos drops banks from court bid	http://www.it.kmit.ac.th/~teerapong/news_arch...
1407	sport	Zonkis confident and cautious	http://www.it.kmit.ac.th/~teerapong/news_arch...

1408 rows × 3 columns

ลองเช็คค่า null

```
In [17]: df.isna().sum()
#ไม่มีค่า null เพราะ clean ไปตอน scrape ข้อมูลแล้ว
```

```
Out[17]: category    0
          title      0
          link      0
         dtype: int64
```

### Add id column

แต่ละข่าวมี id ของวันลงเลเยจะทำการเก็บเพิ่มอีก columน์

```
In [18]: #ในแต่ละสิ่งที่ข่าวจะมี id ของข่าวผลลัพธ์
df['link'].iloc[0]
```

```
Out[18]: 'http://www.it.kmitl.ac.th/~teerapong/news_archive/article-jan-0418.html'
```

```
In [19]: #ใช้ index slicing ในการหา id ข่าว โดยจะเริ่มนับตั้งแต่ว่าที่ 50 ถึงตัวที่ 6 จากหลังสุด
df['id'] = df['link'].str[50:-5]
df
```

```
Out[19]:
```

	category	title	link	id
0	technology	21st-Century Sports: How Digital Technology Is...	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-jan-0418
1	business	Asian quake hits European shares	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-jan-0027
2	technology	BT offers free net phone calls	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-jan-0631
3	business	Barclays shares up on merger talk	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-jan-2105
4	sport	Barkley fit for match in Ireland	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-jan-3300
...	...	...	...	...
1403	sport	Woodward eyes Brennan for Lions	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-dec-2238
1404	business	WorldCom trial starts in New York	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-dec-2334
1405	business	Yukos accused of lying to court	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-dec-2095
1406	business	Yukos drops banks from court bid	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-dec-1351
1407	sport	Zambia confident and cautious	http://www.it.kmitl.ac.th/~teerapong/news_arch...	article-dec-0068

1408 rows × 4 columns

### Get Content

ดึงข้อมูลในลิงก์ข่าว

```
In [20]: list_content = [] #สร้าง List ชื่อ list_content ไว้เก็บข้อมูล content ทั้งหมด
```

สร้าง Function ที่ใช้ในการดึงข้อมูลข่าวในสิ่งที่ใส่เข้าไป

```
In [21]: def get_content(url): #รับลิงก์ข่าวและเก็บไว้ในฟิล์ม url
    response = requests.get(url)
    html_page = bs4.BeautifulSoup(response.content, 'html.parser')

    selector = 'body > div > div.main > p' #เมื่อหาข่าวอยู่ในส่วน tag p
    #เมื่อจากนี้ให้ห้ามคลิกเมื่อหน้า (1 ลูกหน้า : 1 tag p) ซึ่งใช้ select ในการดึง คำที่ได้จะอยู่ในรูปของ List
    tags = html_page.select(selector)

    content = '' #สร้างตัวแปร content รอบเก็บเมื่อหาข่าวในรูปของ string
    for tag in tags: #วน for loop ในตัวแปร tags
        if tag.text.strip() != '': #ในเมื่อห้าว่าจะมีส่วนที่เป็นเว้นวรูปหน้าบล๊าๆ ซึ่งเราไม่ต้องการเก็บส่วนนั้น จึงจะทำการเก็บเฉพาะส่วนที่ไม่ได้เป็นวรูป
            if '.html' in str(tag): #ในส่วนล่างสุดในบางช่าวจะมีลิงก์ให้กดคลิ๊กไปที่หน้าหลัก ซึ่งเราจะไม่ต้องการเก็บส่วนนี้ จึงจะทำการซ่อนไป
                pass
            elif 'Comments are closed for this article' in str(tag): #ในส่วนล่างสุดในบางช่าวจะมีบอกว่าปิดไม่ให้มีการคอมเมนต์ ซึ่งเราไม่ต้องการเก็บเข้ากัน จึงจะทำการซ่อนไป
                pass
            else:
                content += tag.text.strip() + ' ' #หลังจากจบ Loop ก็จะทำการเพิ่มน้อห้าวไว้ในตัวแปร content
    list_content.append(content.strip()) #หลังจากจบ Loop ก็จะทำการเพิ่มน้อห้าวไว้ในตัวแปร list_content
```

```
In [22]: #df['link'] ต้องสั่งก่อนทั้งหมด
print('example:', df['link'][0])
example: http://www.it.kmitl.ac.th/~teerapong/news_archive/article-jan-0418.html
```

```
In [23]: for i in range(len(df)): #วน for loop ตามความยาวทั้งหมดของ df
    get_content(df['link'].iloc[i]) #เอาลิงก์ข่าวใส่ไปใน function get_content()
```

หลังจากวน for loop เพื่อสั่งก์ข่าวใส่ไปใน function `get_content()` ครบหมดแล้ว ในตัวแปร `list_content` จะมีเนื้อห้าวทั้งหมด เราจะแปลง `list_content` จาก list ให้เป็น pandas Series เพื่อที่จะได้เอาไป concat กับ df ให้สมบูรณ์

```
In [24]: content = pd.Series(list_content) #แปลง List_content จาก List ให้เป็น pandas Series และเก็บไว้ในชื่อ content
df = pd.concat([df, content], axis=1) #加入了 content ที่เป็น pandas Series มารวมกับ df ที่มีอยู่แล้วในแนวคอลัมน์ (ภาคล่างน้ำดองกัน)
df = df.rename(columns={0: "content"}) #rename ชื่อ column เพื่อให้สะดวกต่อการใช้งาน โดยจะเปลี่ยนชื่อเป็น content-เมื่อท้ายๆ
```

Out[24]:

	category	title	link	id	content
0	technology	21st-Century Sports: How Digital Technology Is...	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-0418	The sporting industry has come a long way sinc...
1	business	Asian quake hits European shares	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-0027	Asian quake hits European shares Shares in Eur...
2	technology	BT offers free net phone calls	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-0631	BT is offering customers free internet telepho...
3	business	Barclays shares up on merger talk	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-2105	Barclays shares up on merger talk Shares in UK...
4	sport	Barkley fit for match in Ireland	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-3300	England centre Olly Barkley has been passed fi...
...	...	...	...	...	...
1403	sport	Woodward eyes Brennan for Lions	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-dec-2238	Woodward eyes Brennan for Lions Toulouse's for...
1404	business	WorldCom trial starts in New York	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-dec-2334	The trial of Bernie Ebbers, former chief execu...
1405	business	Yukos accused of lying to court	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-dec-2095	Yukos accused of lying to court Russian oil fi...
1406	business	Yukos drops banks from court bid	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-dec-1351	Russian oil company Yukos has dropped the thre...
1407	sport	Zambia confident and cautious	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-dec-0068	Zambia's technical director, Kalusha Bwalya is...

1408 rows × 5 columns

## save DataFarame to csv/txt

เพื่อให้ง่ายต่อการใช้ในภายหลัง

```
In [17]: df.to_csv('datastore/data.csv', index=False)
```

```
In [18]: df = pd.read_csv('datastore/data.csv')
df.head()
```

Out[18]:

	category	title	link	id	content
0	technology	21st-Century Sports: How Digital Technology Is...	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-0418	The sporting industry has come a long way sinc...
1	business	Asian quake hits European shares	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-0027	Asian quake hits European shares Shares in Eur...
2	technology	BT offers free net phone calls	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-0631	BT is offering customers free internet telepho...
3	business	Barclays shares up on merger talk	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-2105	Barclays shares up on merger talk Shares in UK...
4	sport	Barkley fit for match in Ireland	http://www.it.kmit.ac.th/~teerapong/news_arch...	article-jan-3300	England centre Olly Barkley has been passed fi...

## คัดให้เหลือเฉพาะที่ใช้ในการวิเคราะห์

- AllArticles\_OnlyHeading
- AllArticles\_OnlyContent
- AllArticles\_HeadingPlusContent
- Category

```
In [3]: AllArticles_OnlyHeading = df['title']
AllArticles_OnlyContent = df['content']
AllArticles_HeadingPlusContent = df[['title','content']]
Category = df['category']
```

CSV

save เป็นไฟล์ประเภท csv

```
In [4]: AllArticles_OnlyHeading.to_csv('datastore/AllArticles_OnlyHeading.csv', index=False)

In [5]: AllArticles_OnlyContent.to_csv('datastore/AllArticles_OnlyContent.csv', index=False)

In [6]: AllArticles_HeadingPlusContent.to_csv('datastore/AllArticles_HeadingPlusContent.csv', index=False)

In [7]: Category.to_csv('target/category.csv', index=False)
```

txt

save เป็นไฟล์ประเภท txt

```
In [8]: AllArticles_OnlyHeading.to_csv(r'datastore/AllArticles_OnlyHeading.txt', header=None, index=None, sep=' ', mode='a')

In [9]: AllArticles_OnlyContent.to_csv(r'datastore/AllArticles_OnlyContent.txt', header=None, index=None, sep=' ', mode='a')

In [10]: AllArticles_HeadingPlusContent.to_csv(r'datastore/AllArticles_HeadingPlusContent.txt', header=None, index=None, sep=' ', mode='a')

In [15]: Category.to_csv(r'target/category.txt', header=None, index=None, sep=' ', mode='a')

In [ ]:
```

## Modeling

ทำการ Tuning hyperparameter และหา Model ที่ดีที่สุดในการทำงาน

- K-Nearest Neighbors
- XGBoost
- Random forest

โหลดข้อมูลที่เคย save เก็บไว้

- AllArticles\_OnlyContent.txt ต้องข้อมูลที่แยกเฉพาะเนื้อหาข่าวโดยจะเก็บข้อมูลในค่าแปร raw\_documents
- target/category.txt ต้องส่วนของ target หรือเฉลยว่าข่าวข่าวดังกล่าวอยู่ในประเภทอะไร

```
In [2]: with open("datastore/AllArticles_OnlyContent.txt", "r", encoding="utf8") as f: #ตรวจสอบจำนวนบรรทัด
    raw_documents = f.read().splitlines() #ใช้เมธอด string ที่จดชื่อบรทัดใหม่เมื่อ จุดที่มี '\n' และคืนค่ากลับมาเป็นชื่อ
    # ของประเภท list
f.close() #ปิดไฟล์ทั้งหมดให้เสร็จ
print("Read %d raw text documents" % len(raw_documents)) #ตรวจสอบจำนวนบรรทัด
```

Read 1408 raw text documents

```
In [3]: raw_documents[0] #เรียกดูชื่อข้อมูลตัวแรกที่อ่านมา
```

Out[3]: '"The sporting industry has come a long way since the '60s. It has carved out for itself a niche with its roots so deep that I cannot fathom the sports industry showing any sign of decline any time soon - or later. The reason can be found in this seemingly subtle difference - other industries have customers; the sporting industry has fans. Vivek Ranadivé, leader of the ownership group of the NBA's Sacramento Kings, explained it beautifully, "Fans will paint their face purple, fans will evangelize. ... Every other CEO in every business is dying to be in our position - they're dying to have fans." While fan passion alone could almost certainly keep the industry going, leagues and sporting franchises have decided not to rest on their laurels. The last few years have seen the steady introduction of technology into the world of sports - amplifying fans' appreciation of games, enhancing athletes' public profiles and informing their training methods, even influencing how contests are waged. Also, digital technology in particular has helped to create an alternative source of revenue, besides the games themselves - corporate sponsorship. They achieved this by capitalizing on the ardor of their customer base - sorry, fan base."'

```
In [4]: with open("target/category.txt", "r", encoding="utf8") as f: #ตรวจสอบจำนวนบรรทัด
    y = f.read().splitlines() #ใช้เมธอด string ที่จดชื่อบรทัดใหม่เมื่อ จุดที่มี '\n' และคืนค่ากลับมาเป็นชื่อข้อมูลประเภท list
f.close() #ปิดไฟล์ทั้งหมดให้เสร็จ
print("Read %d" % len(y)) #ตรวจสอบจำนวนบรรทัด
```

Read 1408

```
In [ ]:
```

สร้าง dict ชื่อ accuracy\_dict เพื่อใช้เก็บค่า accuracy ของแต่ละ model

```
In [5]: accuracy_dict = dict() #สร้าง dict ชื่อ accuracy_dict เพื่อรับค่า accuracy
```

## เข้าใจว่าข้อมูล imbalance ไหน

```
In [25]: #สร้าง dataframe จากข้อมูลที่ได้เข้ามานะ
all_data = pd.DataFrame(list(zip(raw_documents, y)),
                        columns=['news', 'target'])

all_data
```

Out[25]:

	news	target
0	"The sporting industry has come a long way sin...	technology

1	"Asian quake hits European shares Shares in Eu...	business
2	"BT is offering customers free internet teleph...	technology
3	"Barclays shares up on merger talk Shares in U...	business
4	"England centre Olly Barkley has been passed f...	sport
...	...	...
1403	"Woodward eyes Brennan for Lions Toulouse's fo...	sport
1404	"The trial of Bernie Ebbers, former chief exec...	business
1405	"Yukos accused of lying to court Russian oil f...	business
1406	"Russian oil company Yukos has dropped the thr...	business
1407	"Zambia's technical director, Kalusha Bwalya i...	sport

1408 rows x 2 columns

```
In [26]: # បង្ការវាងនាមទៅតាម target និងកំណើនវិវឌីថានា/រ y_axis
y_axis = all_data['target'].value_counts()
y_axis
```

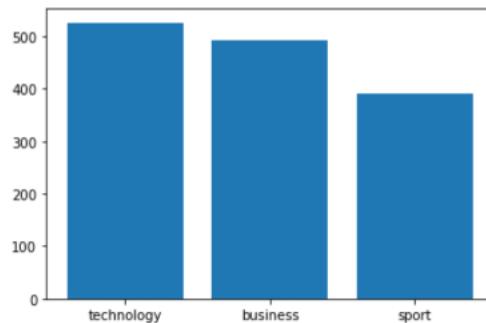
```
Out[26]: sport      526
business    491
technology   391
Name: target, dtype: int64
```

```
In [27]: # មែន target និងចំណាំ x_axis
x_axis = all_data['target'].unique()
x_axis
```

```
Out[27]: array(['technology', 'business', 'sport'], dtype=object)
```

```
In [29]: # plot រាយដឹងទូទាត់មុននៃ imbalance នៃនាមពេល
plt.bar(x_axis, y_axis)
```

```
Out[29]: <BarContainer object of 3 artists>
```



ខ្លួនឯកនៃ imbalance សារណ៍នៅក្បែង តាមការប្រើប្រាស់ upsampling ឬ downsampleing

## Text preprocessing

ស្រាវ function ដែលត្រូវបានប្រើប្រាស់ជាពាណិជ្ជកម្ម

```
In [63]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
# បង្ការនៃយុទ្ធសាស្ត្រ 2 សំនួរ 80% train 20% test
```

```
In [64]: # Tokenizer and WordNet Lemmatizer with POS (part of speech)
def lemma_tokenizer_w_pos_tag(text):

    def convert_tags(tag):
        #បង្ការ tag ដើម្បី vbd vbg vbz ឬអីមីនុយ
        if tag == 'vbd' or tag == 'vbg' or tag == 'vbz':
            return 'v'
        #បង្ការពីចំណាំទៅការប្រើប្រាស់ការបញ្ជាក់ព័ត៌មាន
        else:
            return 'n'

    standard_tokenizer = CountVectorizer().build_tokenizer() #ស្រាវគ្រប់ព័ត៌មាន text (string) ដែលមានអក្សរមិន
tokens
    tokens = standard_tokenizer(text) #បង្ការ text (string) ដែលមានអក្សរមិន
tokens_with_pos_tag = nltk.pos_tag(tokens) #តិច pos_tag នៃកូនពេលនៃ token

    lemmatizer = nltk.WordNetLemmatizer() #ស្រាវគ្រប់ព័ត៌មាននៃការបញ្ជាក់ព័ត៌មាន (រាកកា)
lemma_tokens = [] #ស្រាវ List ដែលមានការបញ្ជាក់ព័ត៌មាននៃការបញ្ជាក់ព័ត៌មាន
for token in tokens_with_pos_tag: #រាយការនៃការបញ្ជាក់ព័ត៌មាន
    new_tag = convert_tags(token[1].lower())
    '''token[1] គឺ pos_tag ដែលមានអក្សរមិនអាមេរិកទី 12 (nltk.pos_tag(tokens) ) ឲ្យ .Lower() ដើម្បីបង្ការពី
ប្រើប្រាស់ការបញ្ជាក់ព័ត៌មាន
    ដែលខ្លួនខ្លួនគឺ convert_tags() ដើម្បីបង្ការពីការបញ្ជាក់ព័ត៌មានទី 2 បានគឺតុលាមិន
VBD verb: past tense took,
VBG verb: gerund/present participle taking,
VBZ verb: 3rd person sing. present takes
ឧបមែលនៃការបញ្ជាក់ v នៃការបញ្ជាក់ព័ត៌មាននៃការបញ្ជាក់ព័ត៌មាន
ដើម្បីបង្ការពីការបញ្ជាក់ព័ត៌មាននៃការបញ្ជាក់ព័ត៌មាន
    lemma_tokens.append(lemmatizer.lemmatize(token[0], new_tag))
'''បង្ការ pos tag នៃការបញ្ជាក់ព័ត៌មាននៃការបញ្ជាក់ព័ត៌មាន ដើម្បីបង្ការពីការបញ្ជាក់ព័ត៌មាន
```

```

    เช่น token[0] = 'amplifying', new_tag = 'v ผลลัพธ์ที่ได้คือ amplify และจะเอาผลลัพธ์เพิ่มเข้าไปใน list ชื่อ lemma_tokens'

    return lemma_tokens

```

ดัวอย่างเมื่อเราค่าเข้าไปใน function lemma\_tokenizer\_w\_pos\_tag()

In [65]: lemma\_tokenizer\_w\_pos\_tag('amplifying')

Out[65]: ['amplify']

In [66]: #สร้าง pipeline สำหรับที่ CountVectorizer และ TfidfTransformer  
 pipe = Pipeline([
 #ทำการตัด stop\_words, แล้ว text ไปใช้ function Lemma\_tokenizer\_w\_pos\_tag โดยผลลัพธ์ที่ได้หลังจากทำ CountVectorizer คือ Term-Document Matrix
 ('vect', CountVectorizer(stop\_words="english", min\_df = 0, tokenizer=lemma\_tokenizer\_w\_pos\_tag)),
 #แปลงมาเป็น tf-idf
 ('tfidf', TfidfTransformer()),
 ]).fit(X\_train)

In [67]: #Term weighting ใน Scikit-Learn เราสามารถสร้างที่ TF-IDF weighted document-term matrix โดยใช้ TfidfVectorizer() แทน CountVectorizer()  
 #default ของ analyzer คือ word, ทำให้ตัด stop\_words  
 vectorizer = TfidfVectorizer(analyzer='word', ngram\_range=(1,3),
 min\_df = 0, stop\_words = 'english', sublinear\_tf=True, lowercase=True)

## K-Nearest Neighbors

ในการท่า Tuning parameter จะต้องกำหนด parameter ที่เราจะทำการรีเคราะห์เป็น knn จะต้องมีการกำหนดจำนวน k

In [68]: #ทำการกำหนด parameter ที่จะเอาไปรีเคราะห์ โดยค่าที่กำหนดไว้คือ parameter ของ knn  
 params = {
 'n\_neighbors' : list(np.arange(1,50,2)), #ให้ทำการทดลองหา k ที่เล็กสุดตั้ง 1-50 โดยขั้นบนที่ลง 2 เช่น 1, 3, 5
 'weights' : ['uniform', 'distance'], #weight function ที่ใช้ในการหานาย
 'metric' : ['euclidean', 'manhattan'] #คิดระยะทางด้วยวิธีไหน
 }

## KNN Tuning parameter (CountVectorizer & TfidfTransformer)

ทำการแบ่ง data ออกเป็น 2 ส่วนคือ **train set** (X\_train, y\_train) กับ **test set** (X\_test, y\_test) โดยส่วน train set จะใช้ในการท่า Tuning parameter, Model selection และ Train model โดยการท่า Tuning parameter จะใช้ **GridSearchCV**, การท่า Model selection จะใช้ **cross\_val\_score** โดยเลือก model ที่มีค่า accuracy สูงที่สุด

In [39]: X\_train, X\_test, y\_train, y\_test = train\_test\_split(raw\_documents, y, test\_size=0.2, random\_state=42)  
 #แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%

ทุกครั้งก่อนนำ train set ไปรีเคราะห์จะต้องทำการ transform ข้อมูลให้อยู่ในรูปแบบที่จะใช้ก่อน

In [40]: X\_train = pipe.transform(X\_train) #ทำการแปลง X\_train โดยใช้ pipeline ที่เราสร้างไว้

In [41]: X\_train

Out[41]: <1126x17450 sparse matrix of type '<class 'numpy.float64'>'  
 with 154876 stored elements in Compressed Sparse Row format>

เราค่าทั้งหมดที่เรากำหนดไว้มาใส่ GridSearchCV

In [127]: folds = 5 #กำหนด fold ที่จะใช้ในการท่า K-fold  
 knn=KNeighborsClassifier() #เรียกใช้ KNN  
 skf = StratifiedKFold(n\_splits=folds, shuffle = True, random\_state = 100) #ทำการกำหนดเครื่องมือ K-fold ที่จะใช้ 5 folds ทำ StratifiedKFold เป็นการแบ่งข้อมูลสุ่มแบบมีชั้น  
 #เมื่อ Classifier, params ได้ลองไปแล้ว กำหนด cv ตัวย่อ k-fold ที่เรากำหนดไว้ในตัวแปร skf  
 knn\_grid = GridSearchCV(knn, params, verbose = 3, cv=skf.split(X\_train,y\_train), n\_jobs = -1)  
 knn\_grid.fit(X\_train, y\_train) #เมื่อ GridSearchCV ไม่ fit สำหรับ train set  
 #totaling 500 fit เมื่อจาก params ที่มีค่า n\_neighbors 25 ตัว \* weights 2 ตัว \* metric 2 ตัว \* folds 5 = 25\*2\*2\*5 = 500

Fitting 5 folds for each of 100 candidates, totalling 500 fits

```

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done   8 tasks      | elapsed:   1.1s
[Parallel(n_jobs=-1)]: Done 188 tasks      | elapsed:   1.8s
[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:   9.1s finished

```

Out[127]: GridSearchCV(cv=<generator object \_BaseKFold.split at 0x0000021A83CA8200>,
 estimator=KNeighborsClassifier(), n\_jobs=-1,
 param\_grid={'metric': ['euclidean', 'manhattan'],
 'n\_neighbors': [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21,
 23, 25, 27, 29, 31, 33, 35, 37, 39, 41,
 43, 45, 47, 49],
 'weights': ['uniform', 'distance']},
 verbose=3)

In [128]: #แสดงค่า parameter ที่ดีที่สุดที่ทำได้

```
print(knn_grid.best_params_)

{'metric': 'euclidean', 'n_neighbors': 41, 'weights': 'uniform'}
```

## CountVectorizer & TfidfTransformer Best params

น่า best parameter ที่หาได้มาใส่ที่ classifier

```
In [69]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

```
In [70]: #ก่อนอ่า X ไปใช้ต้อง transform ก่อนทุกครั้ง
X_train = pipe.transform(X_train)
```

ทำการ fit model กับ parameter ที่ดีที่สุดที่เคราะห์มาได้

```
In [71]: knn=KNeighborsClassifier(metric='euclidean', n_neighbors=41, weights='uniform') #เรียกใช้ knn และกำหนด parameter ตามที่หาได้
knn.fit(X_train, y_train)
```

```
Out[71]: KNeighborsClassifier(metric='euclidean', n_neighbors=41)
```

หา accuracy ด้วย cross\_val\_score

```
In [72]: #cross_val_score จะ return accuracy ออกมากเป็น List โดยจำนวน accuracy ที่ return คือจำนวนเท่ากับ cv ที่เรากำหนด
scores = cross_val_score(knn, X_train, y_train, scoring="accuracy", cv=10)
print("knn model: ", scores.mean()) #เอา List ของ accuracy มาหา mean

knn model:  0.9733723135271809
```

```
In [73]: #เพิ่ม accuracy ที่ເອີ້ນຕົ້ນລວှນໃນ accuracy_dict ที่สร้างไว้ໄວ້ ໂດຍໃຫ້ key เป็นชื่อ model และ value เป็น accuracy
accuracy_dict['knn'] = scores.mean()
```

## Test knn model (CountVectorizer & TfidfTransformer)

เอา model ที่ใช้ parameter ที่ดีที่สุดมา test กับ test set ที่แบ่งไว้ตอนแรก

```
In [74]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ได้และเก็บไว้ในตัวแปร X_new_tfidf
X_new_tfidf = pipe.transform(X_test)

#นำ X_new_tfidf มาทำนาย
predicted = knn.predict(X_new_tfidf)
```

```
In [75]: # หา accuracy โดยเอา เฉลยของ X_test มาเทียบกับ predicted หรือค่าที่ model เรากำหนด
accuracy_score(y_test, predicted)
```

```
Out[75]: 0.9539007092198581
```

```
In [ ]:
```

## KNN Tuning parameter (weighting)

ในส่วนของ Tuning parameter โดย X ที่เราจะใช้จะทำ Term weighting ซึ่งจะคล้ายกับตอนที่เรา Tuning parameter ตอนที่ X ใช้ tfidf แต่จะด่างกันแค่ส่วนของการ transform X

```
In [36]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

Transform X ก่อนเข้าไปเคราะห์ ตรงนี้คือส่วนที่ด่างจาก tfidf

```
In [37]: X_train = vectorizer.fit_transform(X_train)
#ก่อนอ่า X ไปใช้ต้อง transform ก่อนทุกครั้ง
```

```
In [38]: X_train
```

```
Out[38]: <1126x399935 sparse matrix of type '<class 'numpy.float64'>' with 611630 stored elements in Compressed Sparse Row format>
```

ในการทำ Tuning parameter จะต้องกำหนด parameter ที่เราจะการวิเคราะห์ ซึ่งตรงนี้ก็จะเหมือนกับข้างบนเพราะเป็น classifier ด้วยกัน

```
In [139]: #code ในส่วนนี้ก็จะเหมือนกับข้างบนแต่จะด่างแค่ X_train ที่เราใช้ Term weighting
folds = 5 #กำหนด fold ที่จะใช้ในการทำ K-fold
knn=KNeighborsClassifier() #เรียกใช้ KNN
skf = StratifiedKFold(n_splits=folds, shuffle = True, random_state = 1001)
knn_grid_weight = GridSearchCV(knn, params, verbose = 3, cv=skf.split(X_train,y_train), n_jobs = -1) #ทำการ tune parameter
#เอา GridSearchCV ไป fit ภูมิ train set
knn_grid_weight.fit(X_train, y_train)
```

Fitting 5 folds for each of 100 candidates, totalling 500 fits

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done 8 tasks      | elapsed:    0.1s
[Parallel(n_jobs=-1)]: Done 184 tasks     | elapsed:   2.1s
```

```
[Parallel(n_jobs=-1)]: Done 420 tasks      | elapsed:  16.1s
[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:  21.9s finished

Out[139]: GridSearchCV(cv=<generator object _BaseKFold.split at 0x0000021A839EC510>,
                      estimator=KNeighborsClassifier(),
                      n_jobs=-1,
                      param_grid={'metric': ['euclidean', 'manhattan'],
                                  'n_neighbors': [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21,
                                                 23, 25, 27, 29, 31, 33, 35, 37, 39, 41,
                                                 43, 45, 47, 49],
                                  'weights': ['uniform', 'distance']},
                      verbose=3)
```

```
In [140]: #แสดงค่า parameter ที่ดีที่สุด ในส่วนนี้จะต่างกับข้างบน เพราะว่าเราได้ใช้ Term weighting ในส่วนนี้ซึ่งทำให้ parameter ที่ดีที่สุดนี้
          #ค่าทั้งคู่นี้
          print(knn_grid_weight.best_params_)

{'metric': 'euclidean', 'n_neighbors': 17, 'weights': 'uniform'}
```

## Term weighting Best hyperparameter

นำ best parameter ที่หาได้มาใส่ที่ classifier

```
In [127]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
          #แบ่งชื่ออยู่ลิสต์เป็น 2 ส่วน train 80% test 20%
```

```
In [128]: #ก่อนเอา X ไปใช้ต้อง transform ก่อนทุกครั้ง
          X_train = vectorizer.fit_transform(X_train)
```

```
In [129]: #กำหนด parameter ตามที่หาได้
          knn_weight = KNeighborsClassifier(metric='euclidean', n_neighbors=17, weights='uniform')
          knn_weight.fit(X_train, y_train)
```

```
Out[129]: KNeighborsClassifier(metric='euclidean', n_neighbors=17)
```

```
In [130]: #นำ accuracy ด้วย cross_val_score
          scores = cross_val_score(knn_weight, X_train, y_train, cv=10)
          print('KNN model (weight):', scores.mean())

KNN model (weight): 0.9769121365360304
```

```
In [80]: #เพิ่ม accuracy ที่เหลือกันแล้วไว้ใน accuracy_dict ที่สร้างไว้ โดยให้ key เป็นชื่อ model และ value เป็น accuracy
          accuracy_dict['knn_weight'] = scores.mean()
```

### Test knn model (weight)

นำ model ที่จุน parameter เรียบร้อยแล้วมา test กับ test set

```
In [131]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ได้และเก็บไว้ในตัวแปร X_new_tfidf
          X_new_tfidf = vectorizer.transform(X_test)

#นำ X_new_tfidf มาทำนาย
          predicted = knn_weight.predict(X_new_tfidf)
```

```
In [132]: # นำ accuracy ได้มา เฉลยของ X_test มาเทียบกับ predicted หรือค่าที่ model เราทำนาย
          accuracy_score(y_test, predicted)
```

```
Out[132]: 0.975177304964539
```

## XGBoost

ทำการค่า hyperparameter ที่เราจะทำการวัดรายหัว

```
In [83]: #ค่าที่กำหนดไว้สำหรับ parameter ของ XGBoost
          params = {
              'min_child_weight': [1, 5],
              'gamma': [0.5, 1, 1.5],
              'subsample': [0.6, 0.8, 1.0],
              'colsample_bytree': [0.6, 0.8, 1.0],
              'max_depth': [3, 4, 5]
          }
```

## XGBoost Tunning parameter (CountVectorizer & TfidfTransformer)

```
In [266]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
          #แบ่งชื่ออยู่ลิสต์เป็น 2 ส่วน train 80% test 20%
```

ก่อนเอา X ไปใช้ต้อง transform ก่อนทุกครั้ง

```
In [149]: X_train = pipe.transform(X_train)
          #ก่อนเอา X ไปใช้ต้อง transform ก่อนทุกครั้ง
```

```
In [151]: import xgboost as xgb
folds = 5 #ก้าวนดต fold ที่จะใชในการทำ K-fold
xgb = xgb.XGBClassifier() #เรียกใช้ XGBoost
skf = StratifiedKFold(n_splits=folds, shuffle = True, random_state = 1001)
xgb_grid = GridSearchCV(xgb, params, verbose = 3, cv=skf.split(X_train,y_train), n_jobs = -1) #พากากร tune parameter
#แล้ว GridSearchCV จะ fit กับ train set
xgb_grid.fit(X_train, y_train)
```

Fitting 5 folds for each of 162 candidates, totalling 810 fits

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done   8 tasks    | elapsed:   6.3s
[Parallel(n_jobs=-1)]: Done 104 tasks    | elapsed:  33.7s
[Parallel(n_jobs=-1)]: Done 264 tasks    | elapsed:  1.5min
[Parallel(n_jobs=-1)]: Done 488 tasks    | elapsed:  3.1min
[Parallel(n_jobs=-1)]: Done 776 tasks    | elapsed:  5.4min
[Parallel(n_jobs=-1)]: Done 810 out of 810 | elapsed:  5.7min finished
```

```
Out[151]: GridSearchCV(cv=<generator object _BaseKFold.split at 0x0000021A83E2C4A0>,
estimator=XGBClassifier(base_score=None, booster=None,
colsample_bylevel=None,
colsample_bynode=None,
colsample_bytree=None, gamma=None,
gpu_id=None, importance_type='gain',
interaction_constraints=None,
learning_rate=None, max_delta_step=None,
max_depth=None, min_child_weight=None,
missing=None,
n_estimators=100, n_jobs=None,
num_parallel_tree=None, random_state=None,
reg_alpha=None, reg_lambda=None,
scale_pos_weight=None, subsample=None,
tree_method=None, validate_parameters=None,
verbosity=None),
n_jobs=-1,
param_grid={'colsample_bytree': [0.6, 0.8, 1.0],
'gamma': [0.5, 1, 1.5], 'max_depth': [3, 4, 5],
'min_child_weight': [1, 5],
'subsample': [0.6, 0.8, 1.0]},
verbose=3)
```

```
In [152]: #ทดสอบค่า parameter ที่ดีที่สุด
print(xgb_grid.best_params_)

{'colsample_bytree': 0.6, 'gamma': 0.5, 'max_depth': 4, 'min_child_weight': 1, 'subsample': 0.8}
```

## CountVectorizer & TfidfTransformer Best params

```
In [84]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งชื่อมาล้วน 2 ส่วน train 80% test 20%
```

ก่อนเอา X ไปใช้ต้อง transform

```
In [85]: X_train = pipe.transform(X_train)
#ก่อนเอา X ไปใช้ต้อง transform ก่อนทุกครั้ง
```

```
In [86]: #ก้าวนดต parameter ที่ดีที่สุดตามที่เราได้
import xgboost as xgb
xgb = xgb.XGBClassifier(colsample_bytree=0.6, gamma=0.5, max_depth=4, min_child_weight=1, subsample=0.8)
xgb.fit(X_train, y_train)
```

```
Out[86]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=0.6, gamma=0.5, gpu_id=-1,
importance_type='gain', interaction_constraints='',
learning_rate=0.30000012, max_delta_step=0, max_depth=4,
min_child_weight=1, missing=None, monotone_constraints='()', n_estimators=100, n_jobs=0, num_parallel_tree=1, objective='multi:softprob', random_state=0, reg_alpha=0, reg_lambda=1, scale_pos_weight=None, subsample=0.8, tree_method='exact', validate_parameters=1, verbosity=None)
```

```
In [87]: #หา accuracy ด้วย cross_val_score
scores = cross_val_score(xgb, X_train, y_train, scoring="accuracy", cv=10)
print("XGB model: ", scores.mean())
```

XGB model: 0.9689159292035396

```
In [88]: accuracy_dict['xgb'] = scores.mean()
#เพิ่ม accuracy ที่เรียบร้อยแล้วไว้ใน accuracy_dict ที่สร้างขึ้นไว้ โดยให้ key เป็นชื่อ model และ value เป็น accuracy
```

## Test xgb model (CountVectorizer & TfidfTransformer)

นำ model ที่รุน parameter เรียบร้อยแล้วมา test กับ test set

```
In [157]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ได้และเก็บไว้ในตัวแปร X_new_tfidf
X_new_tfidf = pipe.transform(X_test)

#ดู X_new_tfidf มาก่อน
predicted = xgb.predict(X_new_tfidf)
```

```
In [158]: from sklearn.metrics import accuracy_score
# นี่ accuracy ได้มาจากการ predict ของ X_test มาก่อนแล้วก็คือ predicted หรือค่าที่ model เราร่างนาย
accuracy_score(y_test, predicted)
```

```
Out[158]: 0.9468085106382979
```

```
In [ ]:
```

## XGBoost Tuning parameter (weighting)

```
In [272]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

ก่อนเรา X ไปใช้ต้อง transform

```
In [161]: X_train = vectorizer.fit_transform(X_train)
#ก่อนเรา X ไปใช้ต้อง transform ก่อนทุกครั้ง
```

```
In [163]: import xgboost as xgb
folds = 4 #กำหนด fold ที่จะใช้ในการท่า K-fold กำหนด folds = 4 เพราะว่าใช้เวลาเรียนรู้ เลยลดลงมาให้เหลือ 4
xgb = xgb.XGBClassifier() #เรียกใช้ XGBoost
skf = StratifiedKFold(n_splits=folds, shuffle = True, random_state = 1001)
xgb_grid_weight = GridSearchCV(xgb, params, verbose = 3, cv=skf.split(X_train,y_train), n_jobs = -1) #กำหนด tune parameter
#เรา GridSearchCV นี้ fit ที่ train set
xgb_grid_weight.fit(X_train, y_train)
```

Fitting 4 folds for each of 162 candidates, totalling 648 fits

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done   8 tasks      | elapsed:  1.0min
[Parallel(n_jobs=-1)]: Done 104 tasks      | elapsed: 10.2min
[Parallel(n_jobs=-1)]: Done 264 tasks      | elapsed: 26.5min
[Parallel(n_jobs=-1)]: Done 488 tasks      | elapsed: 52.3min
[Parallel(n_jobs=-1)]: Done 648 out of 648 | elapsed: 69.4min finished
```

```
Out[163]: GridSearchCV(cv=<generator object _BaseKFold.split at 0x0000021A839414A0>,
estimator=XGBClassifier(base_score=None, booster=None,
colsample_bylevel=None,
colsample_bynode=None,
colsample_bytree=None, gamma=None,
gpu_id=None, importance_type='gain',
interaction_constraints=None,
learning_rate=None, max_delta_step=None,
max_depth=None, min_child_weight=None,
missing=None,
n_estimators=100, n_jobs=None,
num_parallel_tree=None, random_state=None,
reg_alpha=None, reg_lambda=None,
scale_pos_weight=None, subsample=None,
tree_method=None, validate_parameters=None,
verbosity=None),
n_jobs=-1,
param_grid={'colsample_bytree': [0.6, 0.8, 1.0],
'gamma': [0.5, 1, 1.5], 'max_depth': [3, 4, 5],
'min_child_weight': [1, 5],
'subsample': [0.6, 0.8, 1.0]},
verbose=3)
```

```
In [164]: #แสดงค่า parameter ที่ดีที่สุด
```

```
print(xgb_grid_weight.best_params_)
```

```
{'colsample_bytree': 0.6, 'gamma': 1, 'max_depth': 5, 'min_child_weight': 1, 'subsample': 1.0}
```

## Term weighting Best hyperparameter

```
In [89]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

ก่อนเรา X ไปใช้ต้อง transform

```
In [90]: X_train = vectorizer.fit_transform(X_train)
#transform ก่อนเรา X ไปใช้
```

```
In [91]: import xgboost as xgb
#กำหนด parameter ตามที่หาได้
xgb_weight = xgb.XGBClassifier(colsample_bytree=0.6, gamma=1, max_depth=5, min_child_weight=1, subsample=1.0)
xgb_weight.fit(X_train, y_train)
```

```
Out[91]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=0.6, gamma=1, gpu_id=-1,
importance_type='gain', interaction_constraints='',
learning_rate=0.300000012, max_delta_step=0, max_depth=5,
min_child_weight=1, missing=nan, monotone_constraints='()', n_estimators=100, n_jobs=0, num_parallel_tree=1,
objective='multi:softprob', random_state=0, reg_alpha=0,
reg_lambda=1, scale_pos_weight=None, subsample=1.0,
tree_method='exact', validate_parameters=1, verbosity=None)
```

```
In [92]: #หา accuracy ตัวของ cross_val_score
scores = cross_val_score(xgb_weight, X_train, y_train, cv=10)
print('XGB model (weight):', scores.mean())
XGB model (weight): 0.9662531605562579
```

```
In [93]: #เพิ่ม accuracy ที่เหลือกันแล้วไว้ใน accuracy_dict ที่สร้างไว้ โดยให้ key เป็นชื่อ model และ value เป็น accuracy
accuracy_dict['xgb_weight'] = scores.mean()
```

### Test xgb model (weight)

```
In [94]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ได้แล้วกันไว้ในตัวแปร X_new_tfidf
X_new_tfidf = vectorizer.transform(X_test)

#นำ X_new_tfidf มาทำนาย
predicted = xgb_weight.predict(X_new_tfidf)
```

```
In [95]: from sklearn.metrics import accuracy_score
# หา accuracy โดยเอา เดลยของ X_test มาเทียบกับ predicted หรือค่าที่ model เราทำนาย
accuracy_score(y_test, predicted)
```

```
Out[95]: 0.9326241134751773
```

## Random forest

```
In [96]: #ค่าที่กำหนดไว้ต่อ parameter ของ Random forest
params = {
    'n_estimators': [200, 500],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [4,5,6,7,8],
    'criterion' :['gini', 'entropy']
}
```

### Random forest Tunning parameter (CountVectorizer & TfidfTransformer)

```
In [278]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

```
In [173]: X_train = pipe.transform(X_train)
#ก้อนเอกสาร X ไปใช้ต่อ transform ตอนทุกครั้ง
```

```
In [175]: from sklearn.ensemble import RandomForestClassifier
folds = 5 #กำหนด fold ที่จะใช้ในการท่า K-fold
rf = RandomForestClassifier() #เรียกใช้ RandomForest
skf = StratifiedKFold(n_splits=folds, shuffle = True, random_state = 1001)
rf_grid = GridSearchCV(rf, params, verbose = 3, cv=skf.split(X_train,y_train), n_jobs = -1) #ทำการ tune parameter
#เรา GridSearchCV จะ fit ที่ train set
rf_grid.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 60 candidates, totalling 300 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done   8 tasks      | elapsed:    2.1s
[Parallel(n_jobs=-1)]: Done 104 tasks      | elapsed:   9.5s
[Parallel(n_jobs=-1)]: Done 264 tasks      | elapsed:  24.8s
[Parallel(n_jobs=-1)]: Done 300 out of 300 | elapsed:  28.6s finished
```

```
Out[175]: GridSearchCV(cv=<generator object _BaseKFold.split at 0x0000021A83F41740>,
                        estimator=RandomForestClassifier(), n_jobs=-1,
                        param_grid={'criterion': ['gini', 'entropy'],
                                    'max_depth': [4, 5, 6, 7, 8],
                                    'max_features': ['auto', 'sqrt', 'log2'],
                                    'n_estimators': [200, 500]},
                        verbose=3)
```

```
In [176]: #แสดงค่า parameter ที่ดีที่สุด
print(rf_grid.best_params_)

{'criterion': 'gini', 'max_depth': 8, 'max_features': 'auto', 'n_estimators': 500}
```

### CountVectorizer & TfidfTransformer Best params

```
In [97]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

```
In [98]: X_train = pipe.transform(X_train)
#ก้อนเอกสาร X ไปใช้ต่อ transform ตอนทุกครั้ง
```

```
In [99]: #กำหนด parameter ตามที่หาได้
rf = RandomForestClassifier(criterion='gini', max_depth=8, max_features='auto', n_estimators=500)
rf.fit(X_train, y_train)
```

```
In [100]: RandomForestClassifier(max_depth=8, n_estimators=500)
```

```
In [100]: #หา accuracy ด้วย cross_val_score
scores = cross_val_score(rf, X_train, y_train, scoring="accuracy", cv=10)
print("RF model: ", scores.mean())
```

```
RF model: 0.9689238305941845
```

```
In [101]: #เพิ่ม accuracy ที่เหลือกับผลลัพธ์ใน accuracy_dict ที่สร้างไว้ โดยให้ key เป็นชื่อ model และ value เป็น accuracy
accuracy_dict['rf'] = scores.mean()
```

#### Test rf model (CountVectorizer & TfidfTransformer)

```
In [102]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ต้นและเก็บไว้ในตัวแปร X_new_tfidf
X_new_tfidf = pipe.transform(X_test)
```

```
#นำ X_new_tfidf มาทายนาย
predicted = rf.predict(X_new_tfidf)
```

```
In [103]: from sklearn.metrics import accuracy_score
# หา accuracy ได้โดย เดียวกับ X_test มาเทียบกับ predicted หรือค่าที่ model เราท่านาย
accuracy_score(y_test, predicted)
```

```
Out[103]: 0.9574468085106383
```

```
In [ ]:
```

## Random forest Tunning parameter (weighting)

```
In [284]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

```
In [185]: X_train = vectorizer.fit_transform(X_train)
#ก้อนเอกสาร X ไปใช้ต่อ transform ก่อนทุกครั้ง
```

```
In [187]: folds = 5 #กำหนด fold ที่จะใช้ในการทำ K-fold
rf = RandomForestClassifier() #เรียกใช้'Random Forest'
skf = StratifiedKFold(n_splits=folds, shuffle = True, random_state = 1001)
rf_grid_weight = GridSearchCV(rf, params, verbose = 3, cv=skf.split(X_train,y_train), n_jobs = -1)#ท่าน
#ต้อง tune parameter
#เมื่อ GridSearchCV จะ fit ทุก train set
rf_grid_weight.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 60 candidates, totalling 300 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done   8 tasks      | elapsed:    6.9s
[Parallel(n_jobs=-1)]: Done 104 tasks      | elapsed:  58.8s
[Parallel(n_jobs=-1)]: Done 264 tasks      | elapsed:  2.5min
[Parallel(n_jobs=-1)]: Done 300 out of 300 | elapsed:  2.9min finished
```

```
Out[187]: GridSearchCV(cv=<generator object _BaseKFold.split at 0x0000021A8DE51F20>,
estimator=RandomForestClassifier(), n_jobs=-1,
param_grid={'criterion': ['gini', 'entropy'],
'max_depth': [4, 5, 6, 7, 8],
'max_features': ['auto', 'sqrt', 'log2'],
'n_estimators': [200, 500]},
verbose=3)
```

```
In [188]: #แสดงค่า parameter ที่ดีที่สุด
print(rf_grid_weight.best_params_)
```

```
{'criterion': 'entropy', 'max_depth': 8, 'max_features': 'auto', 'n_estimators': 500}
```

## Term weighting Best hyperparameter

```
In [104]: X_train, X_test, y_train, y_test = train_test_split(raw_documents, y, test_size=0.2, random_state=42)
#แบ่งข้อมูลเป็น 2 ส่วน train 80% test 20%
```

```
In [105]: X_train = vectorizer.fit_transform(X_train)
#ก้อนเอกสาร X ไปใช้ต่อ transform ก่อนทุกครั้ง
```

```
In [106]: #กำหนด parameter ตามที่หาได้
rf_weight = RandomForestClassifier(criterion='entropy', max_depth=8, max_features='auto', n_estimators=500)
rf_weight.fit(X_train, y_train)
```

```
Out[106]: RandomForestClassifier(criterion='entropy', max_depth=8, n_estimators=500)
```

```
In [107]: #หา accuracy ด้วย cross_val_score
scores = cross_val_score(rf_weight, X_train, y_train, cv=10)
print('RF model (weight):', scores.mean())
```

```
RF model (weight): 0.9431890012642224
```

```
In [108]: accuracy_dict['rf_weight'] = scores.mean()
#เพิ่ม accuracy ที่เหลือกับผลลัพธ์ใน accuracy_dict ที่สร้างไว้ โดยให้ key เป็นชื่อ model และ value เป็น accuracy
```

### Test rf model (weight)

```
In [109]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ตั้งแต่ก่อน ไม่ว่าในตัวแปร X_new_tfidf  
X_new_tfidf = vectorizer.transform(X_test)  
  
#นำ X_new_tfidf มาทำนาย  
predicted = rf_weight.predict(X_new_tfidf)  
  
In [110]: # น่า accuracy ได้มาจากการ predict ของ model หรือค่าที่ model เรากำหนด  
accuracy_score(y_test, predicted)  
Out[110]: 0.9326241134751773
```

## Conclusion

หลังจากทำการทดสอบ model ทุกแบบแล้ว จึงนำ accuracy ที่ได้จากการ cross\_val\_score มาเปรียบเทียบกัน

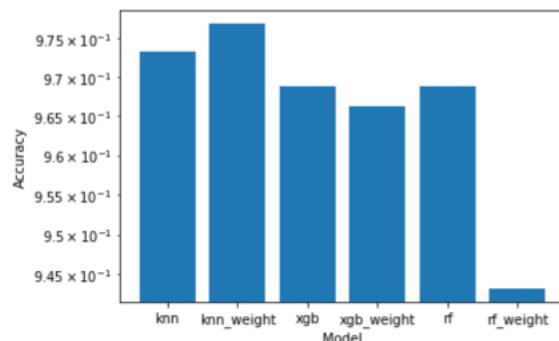
```
In [111]: accuracy_dict  
  
Out[111]: {'knn': 0.9733723135271809,  
           'knn_weight': 0.9769121365360304,  
           'xgb': 0.9689159292035396,  
           'xgb_weight': 0.9662531605562579,  
           'rf': 0.9689238305941845,  
           'rf_weight': 0.9431890012642224}  
  
In [120]: # นำ accuracy_dict มาแปลงเป็น DataFrame เพื่อให้ง่ายต่อการอ่านและเปรียบเทียบ  
accuracy_df = pd.DataFrame(list(accuracy_dict.items()), columns = ['Model', 'Accuracy'])  
accuracy_df  
Out[120]:
```

	Model	Accuracy
0	knn	0.973372
1	knn_weight	0.976912
2	xgb	0.968916
3	xgb_weight	0.966253
4	rf	0.968924
5	rf_weight	0.943189

```
In [121]: #ทำการเรียงลำดับตามค่า Accuracy โดยเรียงจากมากไปน้อย  
accuracy_df.sort_values(by=['Accuracy'], ascending=False)  
Out[121]:
```

	Model	Accuracy
1	knn_weight	0.976912
0	knn	0.973372
4	rf	0.968924
2	xgb	0.968916
3	xgb_weight	0.966253
5	rf_weight	0.943189

```
In [113]: #ทำการ plot bar chart เพื่อให้ง่ายต่อการอ่านค่าและเปรียบเทียบ โดยกำหนดให้ take Log เพื่อใช้ในการแสดงค่า  
plt.bar(accuracy_df['Model'], accuracy_df['Accuracy'], log=True)  
plt.ylabel('Accuracy')  
plt.xlabel('Model');
```



สรุปผล: Model KNN ที่ใช้วิธี Term weighting มี accuracy สูงที่สุด

```
In [ ]:
```

## Test with unseen data

ลองนำข่าวจากลิงก์ด้านล่างมาวิเคราะห์

- <https://www.60secondsnow.com/sports/>

- <https://www.60secondsnow.com/business/>
- <https://www.60secondsnow.com/technology/>

### knn model (weight)

```
In [123]: doc_cat = ['sport', 'business', 'technology', 'sport', 'business', 'technology']
docs_new = ["Ajinkya Rahane remains vice-captain in India's revised squad for the Australia Tests but Irfan Pathan feels a much more experienced Rohit Sharma has to lead the side in Virat Kohli's absence following the series opener. Kohli has been granted paternity leave after the first Test and Pathan said it is bound to have a huge impact on the team.",
            "The S&P 500 and Dow were set to open at record highs on Monday, as news of the first successful late stage COVID-19 vaccine trials stirred hopes of the economy emerging from a year of pandemic-driven crisis.",
            "Apple One More Thing event brought in a couple of new products, including the new M1 processor. Based on the 5nm architecture. The company also launched three new products based on the new Apple M1 processor, namely the MacBook Air, MacBook Pro, and the Apple Mini. Here's everything you need to know about the new offerings.",
            "The FIGC has rejected appeals from Napoli and Roma against 3-0 defeats imposed on them earlier this season. Gennaro Gattuso's men did not turn up for the game in Turin last month and were also docked a point by Lega Serie A for failing to fulfil their fixture obligations. Napoli argued they were banned from travelling by local health authorities.",
            "On Wednesday, global index provider MSCI (Morgan Stanley Capital International) announced changes to its Global Standard Indexes as part of its semi-annual index review. As part of the changes, two Indian stocks, that is Bosch and LIC Housing Finance, will be removed from the index. On the other hand, 12 stocks will be added to the index",
            "In the latest incident, a Nokia phone was charred in flames. The victim called Chandra Babu who hails from Kerala was sleeping when this mishap took place. The device that blasted was a Nokia feature phone and not any smartphone. The device is said to be not charging when this incident happened. The victim got burn injuries on the left arm and shoulder."]
]
```

แปลงข่าวใหม่ก่อนนำไปใช้ model

```
In [124]: # transform X_test ให้เหมือนกับ X_train ก่อน เพื่อที่จะได้ใช้กับ model ที่ train ไว้ได้และเก็บไว้ในตัวมาร X_new_tfidf
X_new_tfidf = vectorizer.transform(docs_new)

#นำ X_new_tfidf มาทำนาย
predicted = knn_weight.predict(X_new_tfidf)
```

```
In [125]: # sport, business, technology, sport, business, technology
print("predicted:",predicted) #แสดงผลที่ไม่เคลื่อนไหว

predicted: ['sport' 'business' 'technology' 'sport' 'business' 'technology']
```

```
In [126]: # นับ accuracy โดยเอา เดลย์ของ X_test มาเทียบกับ predicted หรือค่าที่ model เรากำหนด
accuracy_score(doc_cat, predicted)
```

Out[126]: 1.0