Assignment 2 - DS4Biz Y63

TextScraping_Classification

Team Detail

Team Name: Lnwza7377

Student 1

Student ID: 61070273
Student Full Name: กนกกาญจน์ เหล่าประเสริฐศรี

Student 2

Student ID: 61707277
Student Full Name: กิตติธรรม ผดุงเวียง

In [1]:
```
import numpy as np #เกี่ยวกับตัวเลข ใช้แปลงค่า null
import pandas as pd #ใช้เกี่ยวกับ dataframe เป็นส่วนใหญ่
import seaborn as sns #ใช้ plot กราฟ
import requests #ใช้เรียก response จาก web
from bs4 import BeautifulSoup # scape ข้อมูลมาจาก web
from sklearn.metrics import confusion_matrix #ใช้สร้าง confution matrix
from sklearn.model_selection import cross_val_score #ใช้ในการ cross validation ค
import sklearn.neighbors as nei #เป็นส่วนของโมดูล KNN โดยตั้งชื่อสำหรับเรียกโมดูลสั้นๆว่า n
import matplotlib.pyplot as plt #ใช้ plot กราฟ
from sklearn import metrics #ใช้เกี่ยวกับค่า matrics ต่าง
from sklearn.metrics import accuracy_score, confusion_matrix #ใช้คำนวณ ค่า accur
from sklearn.naive_bayes import MultinomialNB #สำหรับเรียกใช้โมดูล ในโมเดลของ Naive
import sklearn.model_selection as mod #ใช้สำหรับเลือกโมดูล
from sklearn.neighbors import KNeighborsClassifier #ใช้เรียกโมดูลสำหรับโมเดล KNN
from sklearn.model_selection import train_test_split #ใช้สหรับสร้าง ข้อมูลสำหรับ tra
from sklearn.linear_model import LogisticRegression #ใช้สำหรับเรียกโมดูล ของโมเดล L
import operator #โมดูลสำหรับ set ค่าการ sort
from sklearn.feature_extraction.text import TfidfVectorizer #ใช้สำหรับ weight to
from sklearn.feature_extraction import text #ใช้หาคำหยุดในภาษาอังกฤษ
from sklearn.feature_extraction.text import CountVectorizer #ใช้สำหรับตัดคำเพื่อทำ
from sklearn import preprocessing #ใช้ในการทำ binarization
from sklearn import tree #ใช้สำหรับโมดูลของ Decision Tree
import operator #ใช้สำหรับ sort ข้อมูลตอนนับ
```

In [2]:
```
#เก็บลิ้งหลักของแต่ละ page ของ web แสดง quote
link = []
for i in range(1,11):
    p_link = f'https://quotes.toscrape.com/page/{i}/'
    link.append(p_link)
link
```

Out[2]:
```
['https://quotes.toscrape.com/page/1/',
 'https://quotes.toscrape.com/page/2/',
 'https://quotes.toscrape.com/page/3/',
 'https://quotes.toscrape.com/page/4/',
 'https://quotes.toscrape.com/page/5/',
 'https://quotes.toscrape.com/page/6/',
 'https://quotes.toscrape.com/page/7/',
 'https://quotes.toscrape.com/page/8/',
 'https://quotes.toscrape.com/page/9/',
 'https://quotes.toscrape.com/page/10/']
```

In [3]:
```
#quote หรือข้อความ ทั้งหมดที่ได้จากการ scraping จาก web
text = []
for q_link in link:
    response = requests.get(q_link)
    html_page = BeautifulSoup(response.content, 'lxml')
    selector = 'body > div > div > div.col-md-8 > div > span.text'
    # select return เป็น list ของ tag
    tags = html_page.select(selector)
    for txt in tags:
        text.append(txt.text)
text
```

Out[3]:
```
or us.”',
 "“Life isn't about finding yourself. Life is about creating yourself.”",
 "“That's the problem with drinking, I thought, as I poured myself a drin
k. If something bad happens you drink in an attempt to forget; if somethin
g good happens you drink in order to celebrate; and if nothing happens you
drink to make something happen.”",
 '“You don’t forget the face of the person who was your last hope.”',
 "“Remember, we're madly in love, so it's all right to kiss me anytime you
feel like it.”",
 '“To love at all is to be vulnerable. Love anything and your heart will b
e wrung and possibly broken. If you want to make sure of keeping it intact
you must give it to no one, not even an animal. Wrap it carefully round wi

th hobbies and little luxuries; avoid all entanglements. Lock it up safe i
n the casket or coffin of your selfishness. But in that casket, safe, dar
k, motionless, airless, it will change. It will not be broken; it will bec
ome unbreakable, impenetrable, irredeemable. To love is to be vulnerabl
e.”',
 '“Not all those who wander are lost.”',
 '“Do not pity the dead, Harry. Pity the living, and, above all those who
```

In [4]:
```
text_df = pd.DataFrame(text, columns=['Quote']) #สร้าง dataframe ของ quote
```

In [5]:
```
text_df['ID'] = range(1, len(text_df) + 1) #ตั้งค่า ID ให้เริ่มจาก 1
text_df = text_df.set_index('ID') #set ค่า ID ให้เป็น index
```

In [6]:
```
text_df.tail() #แสดง quote 5 ตัวสุดท้ายจาก dataframe ทั้งหมด
```

Out[6]:

ID	Quote
96	“You never really understand a person until yo...
97	“You have to write the book that wants to be w...
98	“Never tell the truth to people who are not wo...
99	“A person's a person, no matter how small.”
100	“... a mind needs books as a sword needs a whe...

In [7]:
```
text_df.to_csv(r'datastore/quote.csv', index = False)
```

In [8]:
```
#tags ทั้งหมดที่ได้จากการ scraping จาก web
tag = []
for q_link in link:
    response = requests.get(q_link)
    html_page = BeautifulSoup(response.content, 'lxml')
    selector = 'body > div > div > div.col-md-8 > div > div meta'
    tags = html_page.select(selector)
    for txt in tags:
        t = txt['content'].split(',')
        tag.append(t)
tag
```

Out[8]:
```
[['change', 'deep-thoughts', 'thinking', 'world'],
 ['abilities', 'choices'],
 ['inspirational', 'life', 'live', 'miracle', 'miracles'],
 ['aliteracy', 'books', 'classic', 'humor'],
 ['be-yourself', 'inspirational'],
 ['adulthood', 'success', 'value'],
 ['life', 'love'],
 ['edison', 'failure', 'inspirational', 'paraphrased'],
 ['misattributed-eleanor-roosevelt'],
 ['humor', 'obvious', 'simile'],
 ['friends', 'heartbreak', 'inspirational', 'life', 'love', 'sisters'],
 ['courage', 'friends'],
 ['simplicity', 'understand'],
 ['love'],
 ['fantasy'],
 ['life', 'navigation'],
 ['activism',
  'apathy',
  'hate',
```

In [9]:
```
all_tag = np.array(tag) #แสดง tags ให้อยู่ในรูปของ numpy array
```

In [10]:
```
tags_df = pd.DataFrame(all_tag, columns=['tags']) #สร้าง dataframe สำหรับเก็บ tags
```

In [11]:
```
tags_df['ID'] = range(1, len(tags_df) + 1) #ตั้งค่า ID ให้เริ่มจาก 1
tags_df = tags_df.set_index('ID') #set ค่า ID ให้เป็น index
```

In [12]:
```
tags_df.tail() #แสดง tags 5 ตัวสุดท้ายจาก dataframe
```

Out[12]:

	tags
ID	
96	[better-life-empathy]
97	[books, children, difficult, grown-ups, write,...
98	[truth]
99	[inspirational]
100	[books, mind]

In [13]:
```
tags_df.to_csv(r'target/tag.csv', index = False) #save tags เป็นไฟล์ csv
```

In [14]:
```
#author ทั้งหมดที่ได้จากการ scraping จาก web
author = []
for q_link in link:
    response = requests.get(q_link)
    html_page = BeautifulSoup(response.content, 'lxml')
    selector = 'body > div > div > div.col-md-8 > div > span > small'
    tags = html_page.select(selector)
    for txt in tags:
        author.append(txt.text)
author
```

```
'J.K. Rowling',
'Ernest Hemingway',
'Ralph Waldo Emerson',
'Mark Twain',
'Dr. Seuss',
'Alfred Tennyson',
'Charles Bukowski',
'Terry Pratchett',
'Dr. Seuss',
'J.D. Salinger',
'George Carlin',
'John Lennon',
'W.C. Fields',

'Ayn Rand',
'Mark Twain',
'Albert Einstein',
'Jane Austen',
'J.K. Rowling',
'Jane Austen',
'Jane Austen'
```

In [15]:
```
author_df = pd.DataFrame(author, columns=['Author']) #สร้าง dataframe ของ Autho
```

In [16]:
```
author_df['ID'] = range(1, len(author_df) + 1) #ตั้งค่า ID ให้เริ่มจาก 1
author_df = author_df.set_index('ID') #set ค่า ID ให้เป็น index
```

In [17]:
```
author_df.tail() #แสดง dataframe ของ Author เฉพาะ 5 ตัวสุดท้าย
```

Out[17]:

ID	Author
96	Harper Lee
97	Madeleine L'Engle
98	Mark Twain
99	Dr. Seuss
100	George R.R. Martin

In [18]:
```
#author detail ทั้งหมดที่ได้จากการ scraping จาก web
author_detail = []
for q_link in link:
    response = requests.get(q_link)
    html_page = BeautifulSoup(response.content, 'lxml')
    selector = 'body > div > div > div.col-md-8 > div > span > a'
    tags = html_page.select(selector)
    for txt in tags:
        url = 'https://quotes.toscrape.com'+txt['href']
        response = requests.get(url)
        html_page = BeautifulSoup(response.content, 'lxml')
        selector = 'body > div > div.author-details > div'
        tags = html_page.select(selector)
        for txt in tags:
            new_detail = txt.text.replace('\n','')
            author_detail.append(new_detail)
author_detail
```

Out[18]:
```
['        In 1879, Albert Einstein was born in Ulm, Germany. He completed
his Ph.D. at the University of Zurich by 1909. His 1905 paper explaining t
he photoelectric effect, the basis of electronics, earned him the Nobel Pr
ize in 1921. His first paper on Special Relativity Theory, also published
in 1905, changed the world. After the rise of the Nazi party, Einstein mad
e Princeton his permanent home, becoming a U.S. citizen in 1940. Einstein,
a pacifist during World War I, stayed a firm proponent of social justice a
nd responsibility. He chaired the Emergency Committee of Atomic Scientist
s, which organized to alert the public to the dangers of atomic warfare.At
a symposium, he advised: "In their struggle for the ethical good, teachers
of religion must have the stature to give up the doctrine of a personal Go
d, that is, give up that source of fear and hope which in the past placed
such vast power in the hands of priests. In their labors they will have to
avail themselves of those forces which are capable of cultivating the Goo
d, the True, and the Beautiful in humanity itself. This is, to be sure a m
ore difficult but an incomparably more worthy task . . . " ("Science, Phil
osophy and Religion, A Symposium," published by the Conference on Science,
Philosophy and Religion in their Relation to the Democratic Way of Life, I
nc., New York, 1941). In a letter to philosopher Eric Gutkind, dated Jan.
```

In [19]:
```
authorDetail_df = pd.DataFrame(author_detail, columns=['Author Detail'])#สร้าง
```

In [20]:
```
authorDetail_df['ID'] = range(1, len(authorDetail_df) + 1) #ตั้งค่า ID ให้เริ่มจาก 1
authorDetail_df = authorDetail_df.set_index('ID') #set ค่า ID ให้เป็น index
```

In [21]:
```
authorDetail_df.tail() #แสดง dataframe ของ AuthorDetail เฉพาะ 5 ตัวสุดท้าย
```

Out[21]:

Author Detail

ID	
96	Harper Lee, known as Nelle, was born i...
97	Madeleine L'Engle was an American writ...
98	Samuel Langhorne Clemens, better known...
99	Theodor Seuss Geisel was born 2 March ...
100	George R. R. Martin was born September...

In [22]:
```
#tags ทั้งหมดที่ได้จากการ scraping จาก web
c_tag = []
for q_link in link:
    response = requests.get(q_link)
    html_page = BeautifulSoup(response.content, 'lxml')
    selector = 'div.tags > a.tag'
    tags = html_page.select(selector)
    for txt in tags:
        a = str(txt).split('>')[1]
        b = str(a).split('<')[0]
        c_tag.append(b)
c_tag
```

```
 'contentment',
 'friends',
 'friendship',
 'life',
 'fate',
 'life',
 'misattributed-john-lennon',
 'planning',
 'plans',
 'love',
 'poetry',
 'happiness',
 'attributed-no-source',
 'humor',
 'religion',
 'humor',
 'comedy',
 'life',
 'yourself',
 'children',
```

In [23]:
```
#หาค่า tag มีค่า ความถี่เท่าไหร่ นับจำนวน tags ว่า ถูกใช้กี่ครั้ง
count = {}
for txt in c_tag:
    if txt in count:
        count[txt] +=1
    else:
        count[txt] = 1
```

In [24]:
```
sorted_count = sorted(count.items(), key=operator.itemgetter(1), reverse=True)
```

In [25]:
```
#นับว่าแต่ละ tags มีจำนวนเท่าไหร่ โดยนับ 15 ตัวที่มีมากที่สุด
top_tag = []
for i in range(15):
    term = sorted_count[i][0]
    count = sorted_count[i][1]
    top_tag.append(term)
    print( "%s (count=%d)" % ( term, count )  )
```

```
love (count=14)
inspirational (count=13)
life (count=13)
humor (count=12)
books (count=11)
reading (count=7)
friendship (count=5)
friends (count=4)
truth (count=4)
simile (count=3)
attributed-no-source (count=3)
death (count=3)
writing (count=3)
thinking (count=2)
classic (count=2)
```

In [26]:
```
top_tag # list ของ top15 tag ที่มีมากที่สุด
```

Out[26]:
```
['love',
 'inspirational',
 'life',
 'humor',
 'books',
 'reading',
 'friendship',
 'friends',
 'truth',
 'simile',
 'attributed-no-source',
 'death',
 'writing',
 'thinking',
 'classic']
```

In [27]:
```
# ทำให้ list อยู่ในรูป  numpy array จะได้ save เป็นไฟล์ text ได้
ttags = np.array(top_tag)
ttags
```

Out[27]:
```
array(['love', 'inspirational', 'life', 'humor', 'books', 'reading',
       'friendship', 'friends', 'truth', 'simile', 'attributed-no-source',
       'death', 'writing', 'thinking', 'classic'], dtype='<U20')
```

In [28]:
```
np.savetxt('target/top_tag.txt', ttags, delimiter=", ", encoding="UTF-8", fmt=
```

In [29]:
```
# Dataframe มา join กันตาม ID
a = text_df.join(tags_df,on='ID')
b = a.join(author_df,on='ID')
all_quote_df = b.join(authorDetail_df,on='ID')
```

In [30]:
```
all_quote_df.head() #ตาราง Dataframe ที่ได้หลังจากการ join
```

Out[30]:

ID	Quote	tags	Author	Author Detail
1	“The world as we have created it is a process ...	[change, deep-thoughts, thinking, world]	Albert Einstein	In 1879, Albert Einstein was born in U...
2	“It is our choices, Harry, that show what we t...	[abilities, choices]	J.K. Rowling	See also: Robert GalbraithAlthough she...
3	“There are only two ways to live your life. On...	[inspirational, life, live, miracle, miracles]	Albert Einstein	In 1879, Albert Einstein was born in U...
4	“The person, be it gentleman or lady, who has ...	[aliteracy, books, classic, humor]	Jane Austen	Jane Austen was an English novelist wh...
5	“Imperfection is beauty, madness is genius and...	[be-yourself, inspirational]	Marilyn Monroe	Marilyn Monroe (born Norma Jeane Morte...

In [31]:
```
#ลูป สำหรับ เช็คว่าแต่ละ quoote มี tag ที่อยู่ใน top 15 หรือเปล่า
top15 = []
for i in all_quote_df['tags']:
    t = []
    for j in i:
        if j in top_tag:
            t.append(j)
    top15.append(t)
```

In [32]:
```
top15 #เช็คว่าแต่ละ quoote มี tag ที่อยู่ใน top 15 หรือเปล่า
```

```
['attributed-no-source'],
['humor'],
['humor'],
['life'],
[],
[],
[],
[],
['reading'],
[],
['friendship'],
[],
['death', 'inspirational'],
['humor'],
['reading'],
[],
['books'],
['inspirational'],
['reading'],
```

In [33]:
```
all_quote_df['top_tags'] = top15
```

In [34]:
```
all_quote_df #ตารางแสดงภาพรวมข้อมูลทั้งหมดที่ scrape ได้ จาก web
```

Out[34]:

ID	Quote	tags	Author	Author Detail	top_tags
1	“The world as we have created it is a process ...	[change, deep-thoughts, thinking, world]	Albert Einstein	In 1879, Albert Einstein was born in U...	[thinking]
2	“It is our choices, Harry, that show what we t...	[abilities, choices]	J.K. Rowling	See also: Robert GalbraithAlthough she...	[]
3	“There are only two ways to live your life. On...	[inspirational, life, live, miracle, miracles]	Albert Einstein	In 1879, Albert Einstein was born in U...	[inspirational, life]
4	“The person, be it gentleman or lady, who has ...	[aliteracy, books, classic, humor]	Jane Austen	Jane Austen was an English novelist wh...	[books, classic, humor]
5	“Imperfection is beauty, madness is genius	[be-yourself, inspirational]	Marilyn Monroe	Marilyn Monroe (born Norma Jeane	[inspirational]

In [35]:
```
df = all_quote_df[["Quote"]] #สร้าง dataframe สำหรับ save quote เป็น ไฟล์ text
```

In [36]:
```
df
```

ID	
73	“The trouble with having an open mind, of cour...
74	“Think left and think right and think low and ...
75	“What really knocks me out is a book that, whe...
76	“The reason I talk to myself is because I’m th...
77	“You may say I'm a dreamer, but I'm not the on...
78	“I am free of all prejudice. I hate everyone e...
79	“The question isn't who is going to let me; it...
80	“'Classic' - a book which people praise and do...
81	“Anyone who has never made a mistake has never...
82	“A lady's imagination is very rapid; it jumps ...
83	“Remember, if the time should come when you ha...
84	“I declare after all there is no enjoyment lik...
85	“There are few people whom I really love, and

In [37]:
```
np.savetxt('datastore/quote.txt', df, delimiter=", ", encoding="UTF-8", fmt="%
```

In [38]:
```
mlb = preprocessing.MultiLabelBinarizer() #เรียกใช้โมดูลสำหรับแปลงค่าเป็น 0 กับ 1
```

In [39]:
```
mlb.fit(all_quote_df.top_tags) # fit โมเดลในการแปลงเป็น binary
```

Out[39]:
```
MultiLabelBinarizer(classes=None, sparse_output=False)
```

In [40]:
```
label = mlb.transform(all_quote_df.top_tags) #ชื่อคลาส
```

In [41]:
```
list(mlb.classes_) #แสดงคลาสที่ต่างกัน 15 ตัวซึ่งมาจาก top 15 tag ที่ถูกใช้มากที่สุด
```

Out[41]:
```
['attributed-no-source',
 'books',
 'classic',
 'death',
 'friends',
 'friendship',
 'humor',
 'inspirational',
 'life',
 'love',
 'reading',
 'simile',
 'thinking',
 'truth',
 'writing']
```

In [42]:
```
bi_df = pd.DataFrame(label,columns=list(mlb.classes_))
```

In [43]:
```
bi_df #ตารางแสดงค่าว่า quote มี tag ใน 15 อันดับ tag สูงสุดหรือไม่ ถ้ามีจะแสดงเลข 1 ไม่มีจะแ
```

Out[43]:

	attributed-no-source	books	classic	death	friends	friendship	humor	inspirational	life	love	reading
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	1	0	0
3	0	1	1	0	0	0	1	0	0	0	0
4	0	0	0	0	0	0	0	1	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1	1	0
7	0	0	0	0	0	0	0	1	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0

In [44]:
```
fin = open("datastore/quote.txt","r",encoding="UTF-8") #อ่านค่าจากไฟล์ text โดยใช้
raw_documents = fin.readlines()
fin.close()
print("Read %d raw text documents" % len(raw_documents)) #จำนวนquoteทั้งหมดที่จะนำ
```

```
Read 100 raw text documents
```

In [45]:
```
X = raw_documents #อ่านค่าจากไฟล์ text ที่ save ไว้
Y = bi_df #target ของ การ ทำนาย
```

In [46]:
```
test_size = 0.5 # สร้าง set ข้อมูลสำหรับ train และ test โดยแบ่ง เป็น อย่างละ 50 50
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=test_size,
```

In [47]:
```
binary_vectorizer = CountVectorizer(binary=True) # "binary=True" means that si
binary_vectorizer.fit(X_train)
```

Out[47]:
```
CountVectorizer(analyzer='word', binary=True, decode_error='strict',
        dtype=<class 'numpy.int64'>, encoding='utf-8', input='content',
        lowercase=True, max_df=1.0, max_features=None, min_df=1,
        ngram_range=(1, 1), preprocessor=None, stop_words=None,
        strip_accents=None, token_pattern='(?u)\\b\\w\\w+\\b',
        tokenizer=None, vocabulary=None)
```

In [48]:
```
X_train_binary = binary_vectorizer.transform(X_train) #แปลง X_train ให้อยู่ในรูป b
X_test_binary = binary_vectorizer.transform(X_test) #แปลง X_test ให้อยู่ในรูป bina
```

In [49]:
```
print("Training set size is %d" % X_train_binary.shape[0] ) #จำนวนข้อมูลสำหรับการ
print("Test set size is %d" % X_test_binary.shape[0] ) # จำนวนข้อมูลสำหรับการ tes
```

```
Training set size is 50
Test set size is 50
```

In [50]:
```
model = KNeighborsClassifier(n_neighbors=9) #เรียกใช้โมดูล และ tune พารามิเตอร์โดยให้
model.fit(X_train_binary, Y_train) # fit โมเดล
print(model)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=9, p=2,
           weights='uniform')
```

In [51]:
```python
predicted = model.predict(X_test_binary) # input ค่า สำหรับ test โมเดล
predicted
```

Out[51]:
```
array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

In [52]:
```
accuracy_knn=accuracy_score(Y_test, predicted) #หาค่าความแม่นยำของโมเดล
accuracy_knn
```

Out[52]:
```
0.32
```

In [53]:
```
deci_tree = tree.DecisionTreeClassifier(criterion='gini')#ทำการ tune พารามิเตอร์ข
```

In [54]:
```
deci_tree.fit(X_train_binary, Y_train) #ทำการ fit โมเดลด้วย set ข้อมูลสำหรับการ tra
```

Out[54]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                       splitter='best')
```

```
In [55]: d_pred = deci_tree.predict(X_test_binary)#ป้อนค่า test เข้าโมเดล เพื่อให้โมเดลทายค่าออ
         d_pred
```

```
Out[55]: array([[0., 1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

In [56]:
```
accuracy_tree=accuracy_score(Y_test, d_pred) #หาค่าความแม่นยำของโมเดล
accuracy_tree
```

Out[56]: 0.3

In [57]:
```
print('Accuracy of KNN :'+str(accuracy_knn))
print('Accuracy of Decision Tree :'+str(accuracy_tree))
```

```
Accuracy of KNN :0.32
Accuracy of Decision Tree :0.3
```

สรุปผล

จากการทำการทดลองพบว่าโมเดลมีความแม่นยำที่ต่ำมากอาจมาจาก ข้อมูล ข้อความมีขนาดเล็ก การ tune model จึงออกมาได้ไม่ดี shape ของข้อมูลก็ไม่ดี ทำให้ไม่สามารถทดลองหลายๆโมเดลได้ จาก 2 โมเดลที่ได้ทำการทดลองพบว่า โมเดล KNN Classifier ให้ค่าความแม่นยำมากที่สุดที่ 0.32 หรือ 32%

In [ ]:
```

```