資料探戡：分類

使用 Kaggle 資料集：

- Fake and real news dataset

In [ ]:
```
# 下載真假新聞壓縮檔
!wget --no-check-certificate "https://drive.google.com/uc?export=download&i
!unzip fakenews.zip
```

In [ ]:
```
# 讀入資料集 -- fakenews : training & test set
import pandas as pd

df_train = pd.read_csv('train.csv')
df_test = pd.read_csv('test.csv')

# 看看訓練資料欄位
print(df_train.info())
```

In [ ]:
```
# 看看 test set
print(df_test.info())
```

本日競賽：【CHT盃】真假（英文）新聞辨識

挑戰層層關卡：

- 用選擇的、或清洗後的training set，訓練模型。
- 模型可用任一個已教過的分類器：KNN、Logistic regression、Naive Bayesian Classifier
- 使用此模型，預測 test set 哪則為真？哪則為假？預測結果請存至 "predicted" 欄位。
- 測試集共有8979筆資料。請保留"id" 及 "predicted" 兩個欄位，並存成csv格式。
- 將你存好的csv，上傳到競賽網頁：http://140.119.108.249:20234

* 排名前五名的同學，將獲得精美小禮品，及好寶寶獎章一枚！

In [ ]:
```
# 不訓練模型，而使用一個極爛招：隨機猜新聞的真假！
import pandas as pd
import numpy as np
```

```
input_df = pd.read_csv('test.csv')
guess_df = input_df['id'] # 還是要欄位 id
# 真新聞:  1 ;  假新聞 : 0
df_guess = pd.DataFrame( np.random.randint(0, 2, size=len(input_df)), colum
guess_df = pd.concat( [guess_df, df_guess], axis = 1 )

# 存檔。此亂猜的結果，已作為 baseline 上傳。
guess_df.to_csv('random_guess.csv', index=False)
```

要如何進行資料前處理？

- 是否要進一步計算特徵？
- 是否要 clean data？

In [ ]:
```
# 使用 nltk 做斷詞
import nltk
from nltk.corpus import stopwords

# 必須先下載模型、語料
nltk.download('punkt')

# 導入停止詞
nltk.download('stopwords')

# Lemmatize
nltk.download('wordnet')
```

In [ ]:
```
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    # Parse the text with Spacy
    doc = word_tokenize(text)

    # Lemmatize the tokens and remove stop words
    lemmas = [lemmatizer.lemmatize(token) for token in doc if token.lower()


    # Join the lemmas back into a string and return it
    return " ".join(lemmas)
```

In [ ]:
```
from tqdm.auto import tqdm
import time
```

In [ ]:
```
# 資料多，應該會跑一段時間喔！
%%time
```

```
tqdm.pandas()

df_train['text_lemma'] = df_train['text'].progress_apply(preprocess_text)
```

In [ ]:
```
print(df_train['text_lemma'])
```

In [ ]:
```
# 因為今天有小競賽，所以 test set 已移除掉正確答案，
# 以下是從 training set，再切一塊 validate set，用它做模型評估用的資料集。

from sklearn.model_selection import train_test_split

# split the data into training and testing sets
X_train, X_val, y_train, y_val = train_test_split(df_train['text'], df_trai

# 看看切割後的 training & validation set
print('X_train shape:', X_train.shape)
print('X_Val shape:', X_val.shape)
```

建立 Bag of word 的特徵

簡言之，基於出現的字，進行詞頻統計，並視為向量化的資料。

Ref: https://en.wikipedia.org/wiki/Bag-of-words_model

In [ ]:
```
from sklearn.feature_extraction.text import CountVectorizer

# create bag-of-words features
vectorizer = CountVectorizer()
X_train_vect = vectorizer.fit_transform(X_train)
X_val_vect = vectorizer.transform(X_val)
```

In [ ]:
```
# 可以看看 vector 的內容?
print(X_val_vect)
```

貝氏分類器

- 建立模型
- 使用模型預測驗證資料集(X_val)
- 使用模型預測測試資料集，並將結果上傳

In [ ]:
```
# 從這裡開始認真訓練模型
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

# (1) train a Naive Bayes classifier
clf_NB = MultinomialNB()
```

```
clf_NB.fit(X_train_vect, y_train)
```

In [ ]:
```
#  (2) Get prediction using validation set
y_pred = clf_NB.predict(X_val_vect)

# check accuracy
acc = accuracy_score(y_val, y_pred)
print("Accuracy:", acc)
```

In [ ]:
```
# check confusion matrix
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_val, y_pred)
print(cm)
```

In [ ]:
```
# check more matrics
from sklearn.metrics import classification_report

# generate classification report
target_names = ['Fake', 'True']
print(classification_report(y_val, y_pred, target_names=target_names))
```

In [ ]:
```
# 看看 df_test
df_test.head()

# (3) 用訓練好的 clf_NB 模型，預測 df_test 的結果
# 施主，這個你得自己試試看～
```

你上傳結果並完成比賽了嗎？XD