```
import json

with open("hw2.json", "r") as f:
    data = json.load(f)


data
```


```
[{'doc_id': 1,
  'cname': '一世狂野',
  'ename': 'Blow',
  'labels': ['劇情', '犯罪', '歷史/傳記'],
  'intro': '喬治戎格一生都在追求所謂的美國夢，也就是享受美好富裕的生活，但是他卻不願像他父親那樣一輩子都只是個出賣勞力的建築工人。於是他搬到陽光明媚的加州，靠著販賣大麻賺錢，起初，他販毒只是為了享受自由自在的生活，但是當他野心越來越大，他的勢力也日益坐大之際，卻在此時被捕入獄。\n\n他在牢裡認識一個能言善道，自稱熟識哥倫比亞販毒集團的牢友狄亞哥，他出獄後果真把當時勢力最大的毒梟艾斯科巴介紹給喬治認識，艾斯科巴計畫將古柯鹼大量引進美國的迪斯可舞廳，希望能引領一股吸毒狂歡的風潮。除了毒品供應商之外，狄亞哥也介紹了一個美艷又狂野的女人瑪莎給喬治，他們瘋狂相愛，之後馬莎還替他生下一個可愛的女兒克莉絲汀娜，也是喬治一生的最愛。\n\n喬治很快就靠著販毒發大財，他還得買一棟大房子專門存放每天賺進來的大把鈔票，但是日進斗金卻整天提心吊膽的生活卻讓喬治開始省思，到底他要繼續過著揮霍富裕的生活，還是為了自己心愛的女兒應該轉性投資正當的事業？可是這時聯邦調查局的探員，也開始盯上毒源禍首的喬治……',
  'release_date': '2001-10-12',
  'pagerank': 4.088307440719542e-05,
  'links': [2]},
 {'doc_id': 2,
  'cname': '玩命關頭',
  'ename': 'The Fast and the Furious',
  'labels': ['動作', '劇情', '犯罪', '懸疑/驚悚'],
  'intro': '唐米尼杜洛托是洛城街頭賽車界的老大哥，他身邊有一群忠心耿耿的手下，他白天忙著組裝高性能跑車，晚上則是開著他的愛車，動輒以一次一萬美元的賭注和別人軋車。\n\n布萊恩也渴望接受極速的挑戰，他對自己的駕駛技術很有信心，但是在旁觀者的眼中他只是一個菜鳥，他開了一輛超炫的跑車想和唐老大一較高下，也希望得到他的青睞，當比賽結束，布萊恩輸得一塌塗地之後，警方接獲風聲前來取締，布萊恩在無意間從一名心狠手辣的幫派份子強尼手中救了唐老大一命，於是他就被納入唐老大的權力核心，唐老大的妹妹蜜雅也對布萊恩產生好感，但是他們都不知道布萊恩其實是一名臥底警探。\n\n布萊恩滲入賽車圈的目的是調查一連串的卡車搶案，嫌犯都是開著跑車的蒙面人，警方和聯邦調查局希望能儘早逮到搶匪，以免卡車司機採取激烈的手段對這些搶匪進行報復行動，其中最有嫌疑的就是唐老大和強尼。正當唐老大和強尼形成水火不相容的情勢。\n\n布萊恩和唐老大兄妹的關係卻越來越深，他不但和唐老大結為好友，更忍不住對蜜雅產生好感，但是他也同時承受來自警方和ＦＢＩ的壓力，必須儘快查出誰才是真正的搶匪，他在天人交戰之際，在法律和友情之間，必須做出困難的決定。',
  'release_date': '2001-10-13',
  'pagerank': 0.00012264922322158625,
  'links': [1, 3]},
 {'doc_id': 3,
  'cname': '戰雲密佈',
  'ename': 'Storm Catcher',
  'labels': ['動作', '犯罪', '懸疑/驚悚', '戰爭'],
  'intro': '美國空軍最高機密的隱形戰機驚傳失蹤！祕密訓練的飛行軍官傑克，被誣陷勾結恐怖組織，參與竊取戰機陰謀，以叛國罪名被捕入獄。他為洗刷冤屈，想盡辦法逃獄，希望重獲自由後，直搗虎穴親自討還清白。\n\n越獄後，他透過好友帕克中校的幫助暗中調查，發現這不僅是一個單純的陷阱，背後更隱藏著更大的、更難掌握的恐怖主義陰謀網，而且更令人出乎意料的是──幕後黑手竟然是五角大廈的高層官員。\n\n美國政府相關人員在傑克逃獄後，緊鎖定其妻女作為人質，逼使他必須出面投案。傑克不但必須衝破重重危機拯救妻女，又得在時限之內取得機密資料，以挽回他的名譽。\n\n然而，就在他循線直搗恐怖組織大本營，即將揭發真相之際，更迫切的是──恐怖組織已發動攻
```

```
勢，企圖以最先進的隱形戰機，攻擊美國政府的權力核心白宮，一場美國有史以來最大的征戰，將一觸
即發…',
  'release_date': '2001-10-13',
  'pagerank': 8.176614881439084e-05,
  'links': [2, 4]},
 {'doc_id': 4,
  'cname': '騎士風雲錄',
  'ename': "A Knight's Tale",
  'labels': ['動作', '冒險', '喜劇'],
  'intro': '14世紀中古時期的社會階級分明，出身卑微的平民不論如何努力和奮鬥，都無法跨越
階級制度而翻身致富，當時正興起一種只有貴族騎士才能參加的運動「長槍比武大賽」。\n\n一位出
身卑賤的年輕人威廉(希斯萊傑飾) ，從小就想成為一位騎士，雖然這個夢想對於貧苦的威廉是個難以
```


```python
from nltk.corpus import stopwords
import string
import jieba

def tokenize_sentence(sentence):
    stop_words = stopwords.words("chinese")
    stop_words2 = ("\n", ", ", " ", "\r\n", "，", "。", "…", "★", "、", "《", "》"

    tokens = jieba.cut(sentence, cut_all = False, HMM = True)
    tokens = [i for i in tokens if i not in string.punctuation]
    tokens = [i for i in tokens if i not in stop_words]
    tokens = [i for i in tokens if i not in stop_words2]

    return tokens


import collections

record = set()
idf_count = collections.defaultdict(int)
lengthOfArcticles = 0

for d in data:
    if (len(d["labels"]) != 0):
        tokens = tokenize_sentence(d["cname"] + d["ename"] + d["intro"])
        for token in set(tokens):
            record.add(token)
            idf_count[token] += 1
        lengthOfArcticles += 1
    if (lengthOfArcticles == 5000):
        break
```

```
Building prefix dict from the default dictionary ...
Loading model from cache /var/folders/ly/4jgkxntx463ghnpv_mjkrb600000gn/T/j
Loading model cost 0.268 seconds.
Prefix dict has been built successfully.
```

```python
import pandas as pd
```

```
import math

x = pd.DataFrame(columns = list(record))
y = pd.DataFrame(columns = ["label"])

x_data, y_data = [], []
for d in data:
    if (len(d["labels"]) != 0):
        chart = collections.Counter(tokenize_sentence(d["cname"] + d["ename"] +
        temp = {}
        for w, n in chart.items():
            tf = round(n / sum(chart.values()), 4)
            idf = round(lengthOfArcticles / idf_count[w], 4)
            temp[w] = round(tf * math.log(idf, 10), 4)

        info_x = pd.DataFrame(temp, index = [len(x_data)])
        x_data.append(info_x)

        info_y = pd.DataFrame({"label": d["labels"][0]}, index = [len(y_data)])
        y_data.append(info_y)


        if (len(y_data) % 100 == 0):
            print(f"{len(y_data)} / 5000")
            if (len(y_data) == 5000):
                break

print("Concating x...")
x = pd.concat([x] + x_data, axis = 0)
x = x.fillna(0)
print("Concating y...")
y = pd.concat([y] + y_data, axis = 0)
```

```
100 / 5000
200 / 5000
300 / 5000
400 / 5000
500 / 5000
600 / 5000
700 / 5000
800 / 5000
900 / 5000
1000 / 5000
1100 / 5000
1200 / 5000
1300 / 5000
1400 / 5000
1500 / 5000
1600 / 5000
1700 / 5000
1800 / 5000
1900 / 5000
2000 / 5000
2100 / 5000
2200 / 5000
2300 / 5000
2400 / 5000
2500 / 5000
2600 / 5000
2700 / 5000
2800 / 5000
2900 / 5000
3000 / 5000
3100 / 5000
3200 / 5000
3300 / 5000
3400 / 5000
3500 / 5000
3600 / 5000
3700 / 5000
3800 / 5000
3900 / 5000
4000 / 5000
4100 / 5000
4200 / 5000
4300 / 5000
4400 / 5000
4500 / 5000
4600 / 5000
4700 / 5000
4800 / 5000
4900 / 5000
5000 / 5000
Concating x...
Concating y...
```

x

	裝訂	百樂門	更揚名	週穩	愛芙羅黛蒂號	布宜諾斯艾	普法	娛樂網	索討	場電影	...	親見	基澤並	熊貓	母嚇	實同	號交響	
0	0.0	0.0	0.0	0.0	0.0	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
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4995	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0
4996	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0
4997	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0
4998	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0	0.0	0.0	0.0	0.0	0

y

	label
0	劇情
1	動作
2	動作
3	動作
4	喜劇
...	...
4995	劇情
4996	劇情
4997	劇情
4998	動作
4999	劇情

5000 rows × 1 columns

```
x_train, x_test = x[: -500], x[-500: ]
y_train, y_test = y[: -500], y[-500: ]


from sklearn.neighbors import KNeighborsClassifier

KNN = KNeighborsClassifier()
print("Training KNN...")
KNN.fit(x_train, y_train)
print("Predicting KNN...")
result1 = KNN.predict(x_test)
```

```
Training KNN...
/opt/homebrew/anaconda3/lib/python3.9/site-packages/sklearn/neighbors/_clas
  return self._fit(X, y)
Predicting KNN...
```

```
from sklearn import ensemble

RF = ensemble.RandomForestClassifier(n_estimators = 100)
print("Training RF...")
RF.fit(x_train, y_train)
print("Predicting RF...")
result2 = RF.predict(x_test)
```

```
Training RF...
/var/folders/ly/4jgkxntx463ghnpv_mjkrb600000gn/T/ipykernel_46207/2304900600
  RF.fit(x_train, y_train)
Predicting RF...
```

```
def evaluate_score(prediction, answer):
    n = 0
    for i in range(len(prediction)):
        if (prediction[i] == answer._get_value(4500 + i, "label")):
            n += 1

    return n / len(prediction)


print(f"KNN score: {evaluate_score(result1, y_test)}")
print(f"RF score: {evaluate_score(result2, y_test)}")
```

```
KNN score: 0.486
RF score: 0.536
```

Colab 付費產品 - 按這裡取消合約