**بسمه تعالی**

**پروژه درس بازیابی اطلاعات**

**سید علی هدایی نیا**

**9723200004**

**خرداد 1401**

**سایت تسنیم**

**مرحله یک(سایت انتخابی تسنیم):**

**خواندن لینک ارشیو صفحه و پیدا کردن فرمت ان :**

<https://www.tasnimnews.com/fa/latest-news?date=1401%2f03%2f01&sub=-1&service=-1&page=3>

همانطور که دیده میشود بخش ارشیو دارای فرمت تاریخ و شماره صفحه است در نتیجه ما از همین فرمت استفاده کرده و بخشی از ارشیو سایت تسنیم را میخوانیم

**مرحله دوم(خزش ارشیو سایت):**

پس از چک کردن source کد سایت تسنیم مشخص شد که این وبسایت برای نمایش خبرهای یافت شده در ارشیو از تگ article استفاده میکند در نتیجه ما در خزشگر خود با رعایت فرمت ورودی سال و صفحه ارشیو را خوانده و تگ های مورد نظر را جدا کردیم .

page\_url = f"https://www.tasnimnews.com/fa/latest-news?date={year}%2f01%2f01&sub=-1&service=-1&page={page}"

html = requests.get(page\_url).text

soup = BeautifulSoup(html, features='lxml')

links = soup.find\_all('article')

در ادامه کار بعد از جداسازی article ها ما تمامی لینک های تگ داخلی article که تگ a میباشد را جدا کرده و به ابتدای ان یک ادرس ثابت سایت را میچسبانیم به فرمت زیر .

news\_url = 'https://www.tasnimnews.com' + link.a['href']

url\_list.append(link.a['href'])

در نهایت هم خبر ها را با لینک ساخته parse میکنیم و پس از پایان در فرمت csv ذخیره میکنیم

شرط پایان هم خواندن 5 صفحه از سایت است .

**مرحله سوم (تست قبل از vectorize کردن):**

برای اماده سازی فایل خزش شده پس از خواندن ان با استفاده از کتابخانه pandas .مجموعه کلمات را از کل داکیومنت های ذخیره شده استخراج کرده و در متغییر vocab میریزیم تا در مرحله tf-idf استفاده کنیم .البته این مرحله صرفا یک تست برای vectorization است و ما برای محاسبه tf-idf خود از کتابخانه TfidfVectorizer استفاده خواهیم کرد که نتایج قابل اتکا تری داشته باشیم .

**مرحله چهارم (tf-idf vectorization):**

ابتدا یک شئی از کلاس TfidfVectorizer ساخته و مجموعه داکیومنت های خود را بر ان fit\_transform میکنیم . که مجموعه vocab ها را یادگرفته و tfidf را نیز استخراج کنیم .

در نتیجه پس از انجام این عملیات اماده محاسبه فاصله کسینوسی برای کوئری کاربران هستیم و به مرحله بعد میرویم .

**مرحله پنجم (گرفتن کوئری و محاسبه فاصله کسینوسی):**

در این مرحله یک کوئری در نظر میگیریم (با توجه به مجموعه vocab که مطمئن شویم جوابی پیدا خواهیم کرد ) و برای ان کوئری tfidf محاسبه میکنیم .

بعد از محاسبه tfidf کوئری با داشتن هر دو tfidf به محاسبه شباهت پرداخته و جواب های نهایی را مرتب میکنیم و به ترتیب شباهت بر میگردانیم.

cosines.append(float(cosine\_similarity(d, tfidf\_query)))

sorted\_ids = np.argsort(cosines)

sorted\_ids

for i in range(k):

cur\_id = sorted\_ids[-i-1]

print(cur\_id)

print(docs['text'][cur\_id], cosines[cur\_id])

**لینک کد :**

https://github.com/todalex/IR.git