آرین قره محمد زاده قشقایی

4017724001

محمد برزگر

4010724001

تمرین 5

دیتاست movie_5000_t به وسیله api کگل به کولب اضاف شده و دیتاست را دانلود کردیم در ابتدا.

```
!pip install kaggle
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: kaggle in /usr/local/lib/python3.10/dist-packages (1.5.13)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.10/dist-packages (from kaggle) (1.16.0)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from kaggle) (2022.12.7)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from kaggle) (2.8.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from kaggle) (2.27.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from kaggle) (4.65.0)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.10/dist-packages (from kaggle) (8.0.1)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.10/dist-packages (from kaggle) (1.26.15)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.10/dist-packages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests->kaggle) (2.0.12)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->kaggle) (3.4)
```

```
from google.colab import files
files.upload()
```

```
Choose Files kaggle.json
• kaggle.json(application/json) - 66 bytes, last modified: 6/16/2023 - 100% done
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username":"arianghmgh","key":"33d467c6c0fb3830e865dd6fedeb6075"}'}
```

```
!mkdir ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d tmdb/tmdb-movie-metadata
```

```
Downloading tmdb-movie-metadata.zip to /content
  0% 0.00/8.89M [00:00<?, ?B/s]
100% 8.89M/8.89M [00:00<00:00, 123MB/s]
```

```
!unzip tmdb-movie-metadata.zip
```

```
Archive:  tmdb-movie-metadata.zip
  inflating: tmdb_5000_credits.csv
  inflating: tmdb_5000_movies.csv
```

```
credits = pd.read_csv('tmdb_5000_credits.csv')
```

دیتاست حاوی 2 فایل tmdb_5000_credits.csv و tmdb_5000_movies.csv بود که فایل tmdb_5000_movies.csv حاوی ستون های زیر بود

```
'budget', 'genres', 'homepage', 'id', 'keywords', 'original_language',
   'original_title', 'overview', 'popularity', 'production_companies',
        'production_countries', 'release_date', 'revenue', 'runtime',
  'spoken_languages', 'status', 'tagline', 'title', 'vote_average',
                                                'vote_count'], dtype='object
```

از بین این ستون ها، ستون های tagline , overview , title , keywords را انتخاب و عمل پری پراکسسینگ را انجام میدیم.

دو تابع زیر برای حذف کلمات stem و stopwords ها ساخته شده.

```
nltk.download('stopwords')
def RemoveStopWord(clean_tokens):
    stop_words = set(stopwords.words('english'))
    filtered_tokens = [token for token in clean_tokens if token.lower() not in stop_words and len(token) > 1]
    return filtered_tokens
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
nltk.download('punkt')
def stem_words(word_array):
    stemmer = PorterStemmer()
    stemmed_words = [stemmer.stem(word) for word in word_array]
    return stemmed_words
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

در زیر جدول داده های فیلم ها قبل از پری پراکسسینگ و بعد از آن را نمایش داده.

```
[14] movies = pd.read_csv("tmdb_5000_movies.csv")
```

movies

2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"name": "Columbia Pictures", "id": 5}, {"nam...	[{"iso_3166_1": "GB", "name": "United Kingdom"...	2015-10-26	880674609	148.0	[{"iso_63 "Fran\u00
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	[{"name": "Legendary Pictures", "id": 923}, {"...	[{"iso_3166_1": "US", "name": "United States o...	2012-07-16	1084939099	165.0	[{"iso_639 "name": "
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	[{"name": "Walt Disney Pictures", "id": 2}]	[{"iso_3166_1": "US", "name": "United States o...	2012-03-07	284139100	132.0	[{"iso_639 "name": "
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4798	220000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	NaN	9367	[{"id": 5616, "name": "united states\u2013mexi...	es	El Mariachi	El Mariachi just wants to play his guitar and ...	14.269792	[{"name": "Columbia Pictures", "id": 5}]	[{"iso_3166_1": "MX", "name": "Mexico"}, {"iso...	1992-09-04	2040920	81.0	[{"iso_639 "Espa
4799	9000	[{"id": 35, "name": "Comedy"}, {"id": 10749, "...	NaN	72766	[]	en	Newlyweds	A newlywed couple's honeymoon is upended by th...	0.642552	[]	[]	2011-12-26	0	85.0	
4800	0	[{"id": 35, "name": "Comedy"}, {"id": 18, "nam...	http://www.hallmarkchannel.com/signedsealeddel...	231617	[{"id": 248, "name": "date"}, {"id": 699, "nam...	en	Signed, Sealed, Delivered	"Signed, Sealed, Delivered" introduces a dedic...	1.444476	[{"name": "Front Street Pictures", "id": 3958}...	[{"iso_3166_1": "US", "name": "United States o...	2013-10-13	0	120.0	[{"iso_639 "name": "
								When ambitious							

✓ 0s completed at 6:55PM

Data_after_preproccesing

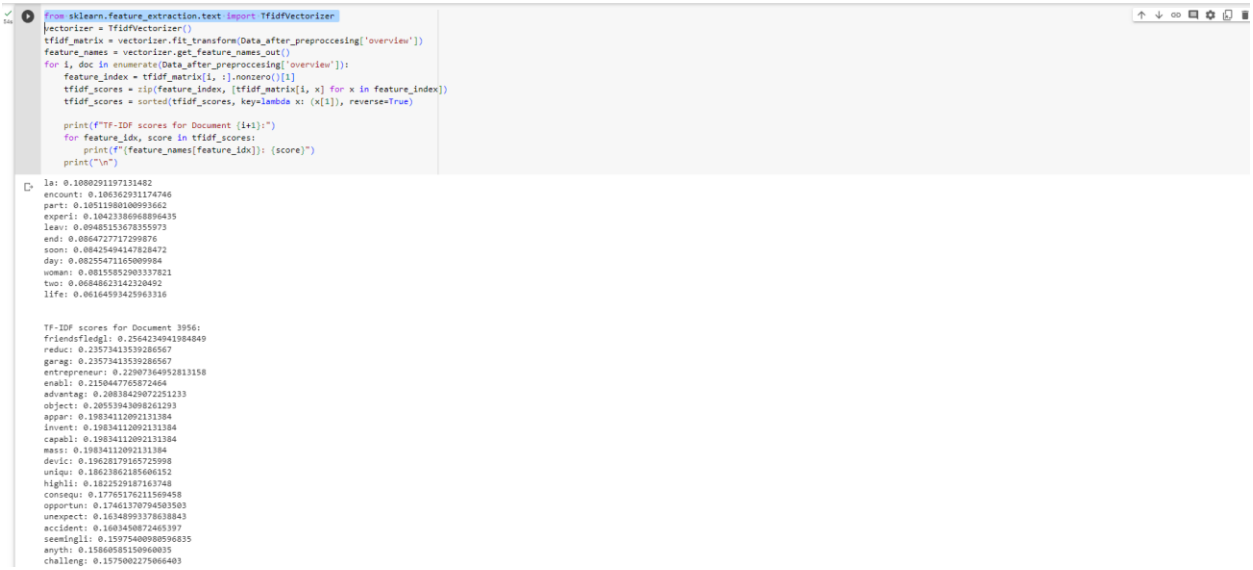
	title	tagline	overview	keywords
0	avatar	enter world pandora	22nd centuri parapleg marin dispatch moon pand...	id 1463 name cultur clash id 2964 name futur i...
1	pirat caribbean world end	end world adventur begin	captain barbossa long believ dead come back li...	id 270 name ocean id 726 name drug abus id 911...
2	spectr	plan one escap	cryptic messag bond past send trail uncov sini...	id 470 name spi id 818 name base novel id 4289...
3	dark knight rise	legend end	follow death district attorney harvey dent bat...	id 849 name dc comic id 853 name crime fighter...
4	john carter	lost world found anoth	john carter warweari former militari captain i...	id 818 name base novel id 839 name mar id 1456...
...	...	...	...	...
3954	bang	sometim got break rule	young woman la bad day evict audit end produc ...	id 10726 name gang id 33928 name audit id 1727...
3955	primer	happen actual work	friendsfledgl entrepreneur invent devic garag ...	id 1448 name distrust id 2101 name garag id 33...
3956	el mariachi	nt come look troubl troubl came look	el mariachi want play guitar carri famili trad...	id 5616 name unit statesu2013mexico barrier id...
3957	newlyw	newlyw coupl honeymoon upend arriv respect sister	newlyw coupl honeymoon upend arriv respect sister	
3958	shanghai call	new yorker shanghai	ambiti new york attorney sam sent shanghai ass...	

3959 rows × 4 columns

بعد از این که داده ها آماده شد. به وسیله کد زیر

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

از sklearn متدی برای محاسبه TfidF به پروژه اضاف میکنیم. و به وسیله آن وکتور های کلمات را بدست می آوریم.



```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(Data_after_preproccesing['overview'])
feature_names = vectorizer.get_feature_names_out()
for i, doc in enumerate(Data_after_preproccesing['overview']):
    feature_index = tfidf_matrix[i, :].nonzero()[1]
    tfidf_scores = zip(feature_index, [tfidf_matrix[i, x] for x in feature_index])
    tfidf_scores = sorted(tfidf_scores, key=lambda x: (x[1]), reverse=True)

    print(f"TF-IDF scores for Document {i+1}:")
    for feature_idx, score in tfidf_scores:
        print(f"{feature_names[feature_idx]}: {score}")
    print("\n")
```

```
la: 0.1080291197131482
encount: 0.10036293111747046
part: 0.10511980100993662
experi: 0.10423386968896435
leav: 0.09485153678355973
end: 0.0864727717299876
soon: 0.08425494147828472
day: 0.08255471165009984
woman: 0.08155852903337821
two: 0.06848623142320492
life: 0.06164593425963316


TF-IDF scores for Document 3956:
friendsfledgl: 0.2564234941984849
reduc: 0.23573413539286567
garag: 0.23573413539286567
entrepreneur: 0.22907364952813158
enabl: 0.2150447765872464
advantag: 0.20838429072251233
object: 0.20553943098261293
appar: 0.19834112092131384
invent: 0.19834112092131384
capabl: 0.19834112092131384
mass: 0.19834112092131384
devic: 0.19628179165725998
uniqu: 0.18623862185606152
highli: 0.1822529187163748
consequ: 0.17765176211569458
opportun: 0.17461370794503503
unexpect: 0.16348993378638843
accident: 0.1603450872465397
seemingli: 0.15975400980596835
anyth: 0.15860585150960035
challeng: 0.1575002275066403
```

همین طور این متد به صورت دستی توسط ما نیز پیاده سازی شد.

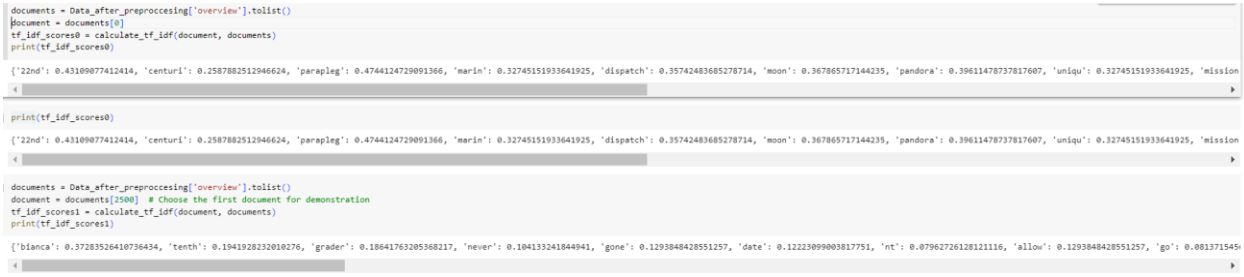
```python
import math
from collections import Counter
def calculate_tf(document):
    tf_scores = {}
    word_counts = Counter(document.split())
    total_words = len(document.split())
    for word, count in word_counts.items():
        tf_scores[word] = count / total_words
    return tf_scores

def calculate_idf(documents):
    idf_scores = {}
    total_documents = len(documents)
    all_words = set([word for document in documents for word in document.split()])
    for word in all_words:
        document_count = sum([1 for document in documents if word in document.split()])
        idf_scores[word] = math.log(total_documents / (1 + document_count))
    return idf_scores

def calculate_tf_idf(document, documents):
    tf_idf_scores = {}
    tf_scores = calculate_tf(document)
    idf_scores = calculate_idf(documents)
    for word in document.split():
        tf_idf_scores[word] = tf_scores[word] * idf_scores[word]
    return tf_idf_scores
```

در این پیاده سازی در ابتدا متد tf برای بدست آوردن ترم فریکوانسی کلمات یا همان تعداد تکرار کلمات در داکیومنت را محاسبه میکند و idf نیز متدی برای بدست آوردن معیار اهمیت کلمه بر حسب rarity هست. و در نهایت متد tf- idf برای در نظر گرفتن اسکور نهایی ساخته شده.

در اینجا دو دامیومنت را به صورت رندوم انتخاب و tf idf آن را محاسبه کردیم



وکتور های محاسبه شده را به تابع زیر دادیم

```
54] from sklearn.metrics.pairwise import cosine_similarity
    def calculate_cosine_similarity(vector1, vector2):
        vector1 = list(vector1.values())
        vector2 = list(vector2.values())
        dot_product = sum([x * y for x, y in zip(vector1, vector2)])
        magnitude1 = math.sqrt(sum([x**2 for x in vector1]))
        magnitude2 = math.sqrt(sum([x**2 for x in vector2]))
        similarity = dot_product / (magnitude1 * magnitude2)
        return similarity
```

```
55] calculate_cosine_similarity(tf_idf_scores0,tf_idf_scores1)

    0.5819957082048989
```

که یک معیار شباهت برحسب cos similarity هست دادیم و مقدار شباهت این دو وکتور را بدست آوردیم.