

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Διοίκηση της Ψηφιακής Επιχείρησης
8^ο Εξάμηνο – Ακ. Έτος 2023-2024
Κουράκου Σοφία 03120869
el20869@mail.ntua.gr / sofiakrk12@gmail.com

Εργασία 4 - ANAZHTHSEH KAI RECOMMENDER SYSTEMS

1. ANAZHTHSEH – TFIDF

Το query που έχουμε είναι (earth, mission, astro). Βρίσκουμε την συχνότητα εμφάνισης της (TF, Term Frequency) σε κάθε έγγραφο σημειώνοντας τη συχνότητα κάθε όρου σε κάθε ταινία (σε τίτλο και περιγραφή), λαμβάνοντας υπόψη και τις παράγωγες ή σύνθετες λέξεις (stemming).

1. Contact (1997) 2h 30min | Drama, Mystery, Sci-Fi

After graduating from Harvard University, Ellie receives a doctorate from Caltech supervised by David Drumlin, a well known radio **astronomer**. She eventually becomes the director of "Project Argus", a radiotelescope array in New Mexico dedicated to the search for extraterrestrial intelligence. The project discovers a signal containing a series of prime numbers coming from the Vega system 26 light years away. Further analysis reveals information in the polarization modulation of the signal. This message is a **retransmission** of Adolf Hitler's opening speech at the 1936 Summer Olympics in Berlin; the first television signal powerful enough to escape **Earth's** ionosphere. Ellie feels vindicated. But that vindication is short lived when others, including politicians, the military, religious leaders and other scientists such as Drumlin, try to take over her work. When the messages received from space are decoded, the project takes on a whole new dimension, which strengthens for Ellie the quest for the truth.

2. Sunshine (2007) 1h 47min | Adventure, Sci-Fi, Thriller

As the sun begins to dim along with humankind's hope for the future, it's up to a desperate crew of eight **astronauts** to reach the dying star and reignite the fire that will bring life back to planet **Earth** in this tense psychological sci-fi thriller that reteams 28 Days Later director Danny Boyle with writer Alex Garland and producer Andrew Macdonald. The skies are darkening, and the outlook for planet **Earth** is grim. A crew of eight men and women has been given a nuclear device designed to literally reignite the sun and sent hurtling through infinity on the most crucial space **mission** ever attempted. Suddenly, as the crew loses radio contact with **mission** control, everything begins to fall apart.

3. Gattaca (1997) 1h 46min | Drama, Sci-Fi, Thriller

In a futuristic society where commerce has overridden more humanistic concerns, the rich and successful, eager to obtain physical and mental perfection, have taken to genetically engineering their off-spring. Such lab-created babies are known as Valids, while those conceived in the normal, loving fashion are In-Valids and are considered second-class citizens at best. Vincent is an In-Valid while his brother Anton is a Valid. The former brother is short, sickly, and bespectacled, while the latter brother is handsome, healthy and born to succeed. But though Anton seems close to perfection, he lacks the emotional flaws, passion, determination, desire and faith that motivate Vincent, whose strongest desire is to become a space navigator for the Gattaca Aerospace Corporation and travel on an upcoming **mission** to the moons of Saturn. Unfortunately, his birth status and a heart defect, relegate him to menial jobs. Unwilling to abandon hope, Vincent determinedly visits DNA broker German (Tony Shalhoub) who is able to create false identities for similar In-Valids. Set in an oppressive, bureaucratic and chillingly plausible early-21st-century world, Andrew Niccol's sci-fi thriller differs from others in its focus on a morally ambiguous world and on characters rather than gizmos, technobabble and special effects.

4. Moon (2009) 1h 37min | Drama, Mystery, Sci-Fi

An **astronaut** miner extracting the precious moon gas that promises to reverse the **Earth's** energy crisis nears the end of his three-year contract, and makes an ominous discovery in this psychological sci-fi film starring Sam Rockwell and Kevin Spacey. For three long years, Sam Bell has dutifully harvested Helium 3 for Lunar, a company that claims it holds the key to solving humankind's energy crisis. As Sam's contract comes to an end, the lonely **astronaut** looks forward to returning to his wife and daughter down on **Earth**, where he will retire early and attempt to make up for lost time. His work on the Selene moon base has been enlightening -- the solitude helping him to reflect on the past and overcome some serious anger issues -- but the isolation is starting to make Sam uneasy. With only two weeks to go before he begins his journey back to our planet, Sam starts feeling strange: he's having inexplicable visions, and hearing impossible sounds.

5. The Martian (2015) 2h 24min | Adventure, Drama, Sci-Fi

During a manned **mission** to Mars, **Astronaut** Mark Watney (Matt Damon) is presumed dead after a fierce storm and left behind by his crew. But Watney has survived and finds himself stranded and alone on the hostile planet. With only meager supplies, he must draw upon his ingenuity, wit and spirit to subsist and find a way to signal to **Earth** that he is alive. Millions of miles away, NASA and a team of international scientists work tirelessly to bring "the Martian" home, while his crewmates concurrently plot a daring, if not impossible rescue **mission**.

Στις παραπάνω περιγραφές ταινιών που δόθηκαν από την εκφώνηση, σημειώθηκαν με **bold** οι όροι που αναζητούμε. Βάση αυτών των δεδομένων που συλλέχθηκαν, προέκυψε ο ακόλουθος πίνακας :

Ταινία	earth	mission	astro
Contact (1997)	1	1	1
Sunshine (2007)	2	2	1
Gattaca (1997)	0	1	0
Moon (2009)	2	0	2
The Martian (2015)	1	2	1

Έπειτα, βρίσκουμε το IDF (Inverse Document Frequency) από τον τύπο:

$$IDF = \log \frac{|D|}{|d : t_i \in d|}$$

Όπου D το σύνολο των ταινιών επομένως το $|D| = 5$ και $|d : t_i \in d|$: Number of documents where the term t_i appears at least once.

Σύμφωνα με τα παραπάνω, ακολουθούν οι υπολογισμοί για το IDF του εκάστοτε όρου:

Όρος	IDF
earth	0.09691
mission	0.09691
astro	0.09691

Έχοντας υπολογίσει το IDF, μπορούμε να υπολογίσουμε το TFIDF ως εξής:

$$TFIDF = TF \cdot IDF$$

Στον ακόλουθο πίνακα βλέπουμε τα αποτελέσματα για κάθε ταινία:

Ταινία	TFIDF διάνυσμα (earth, mission astro)		
Contact (1997)	0.09691	0.09691	0.09691
Sunshine (2007)	0.19382	0.19382	0.09691
Gattaca (1997)	0	0.09691	0
Moon (2009)	0.19382	0	0.19382
The Martian (2015)	0.09691	0.19382	0.09691

Στη συνέχεια, υπολογίζεται το TFIDF διάνυσμα του query(earth, mission, astro):

$$TFIDF = TF \cdot IDF = (1,1,1) \cdot (0.09691, 0.09691, 0.09691) = (0.09691, 0.09691, 0.09691)$$

Στη συνέχεια, θα συγκρίνουμε τα διανύσματα των μετρικών ανάμεσα στα TFIDF των περιγραφών των ταινιών και στο TFIDF του query με τη χρήση του Cosine Similarity και έτσι θα βρούμε μια κατάταξη.

Ταινία	Cosine Similarity
Contact (1997)	1
Sunshine (2007)	0.96225
Gattaca (1997)	0.57735
Moon (2009)	0.8165
The Martian (2015)	0.94281

Σύμφωνα με τα παραπάνω αποτελέσματα του Cosine Similarity, η τελική κατάταξη είναι :

1. Contact (1997)
2. Sunshine (2007)
3. The Martian (2015)
4. Moon (2009)
5. Gattaca (1997)

Για τους παραπάνω υπολογισμούς χρησιμοποιήθηκε python μέσω του Google Colab. Συγκεκριμένα, ο κώδικας και τα αποτελέσματα είναι τα ακόλουθα:

Κώδικας Αναζήτησης TFIDF

```
import numpy as np
from numpy.linalg import norm

# query, tfs
query = ['earth', 'mission', 'astro']
tfs = {
    "Contract(1997)": [1, 1, 1],
    "Sunshine(2007)": [2, 2, 1],
    "Gattaca(1997)": [0, 1, 0],
    "Moon(2009)": [2, 0, 2],
    "The Martian(2015)": [1, 2, 1]
}
movies = list(tfs.keys())

# idf for a query term
def calculate_idf(term_freqs):
    return np.log10(len(movies) / np.count_nonzero(term_freqs))

idsf = {term: calculate_idf([tfs[movie][i] for movie in movies]) for i, term in enumerate(query)}

# tfidf for a movie
def calculate_tfidf(movie_freqs):
    return [tf * idf[term] for tf, term in zip(movie_freqs, query)]

tfidf = {movie: calculate_tfidf(tfs[movie]) for movie in movies}
tfidf = {"Query": calculate_tfidf([1, 1, 1]), **tfidf}

# Cosine Similarity (TFIDFquery, TFIDFmovies)
def calculate_cosine_similarity(movie_tfidf):
    query_tfidf = calculate_tfidf([1, 1, 1])
    return np.dot(query_tfidf, movie_tfidf) / (norm(query_tfidf) * norm(movie_tfidf))

cosine_similarities = {movie: calculate_cosine_similarity(tfidf[movie]) for movie in movies}

# results
def print_results(identifier, dictionary, is_list):
    print(identifier)
    for key, result in dictionary.items():
        if is_list:
            print(f"{key} - {[round(element, 5) for element in result]}")
        else:
            print(f"{key} - {round(result, 5)}")

identifiers = ["- TFS", "\n- IDFs", "\n- TFIDFs", "\n- Cosine Similarities"]
dictionaries = [tfs, idf, tfidf, cosine_similarities]
is_lists = [True, False, True, False]

for identifier, dictionary, is_list in zip(identifiers, dictionaries, is_lists):
    print_results(identifier, dictionary, is_list)
```

Αποτελέσματα Αναζήτησης TFIDF



- TFs

Contract(1997) - [1, 1, 1]
Sunshine(2007) - [2, 2, 1]
Gattaca(1997) - [0, 1, 0]
Moon(2009) - [2, 0, 2]
The Martian(2015) - [1, 2, 1]

- IDFs

earth - 0.09691
mission - 0.09691
astro - 0.09691

- TFIDFs

Query - [0.09691, 0.09691, 0.09691]
Contract(1997) - [0.09691, 0.09691, 0.09691]
Sunshine(2007) - [0.19382, 0.19382, 0.09691]
Gattaca(1997) - [0.0, 0.09691, 0.0]
Moon(2009) - [0.19382, 0.0, 0.19382]
The Martian(2015) - [0.09691, 0.19382, 0.09691]


- Cosine Similarities

Contract(1997) - 1.0
Sunshine(2007) - 0.96225
Gattaca(1997) - 0.57735
Moon(2009) - 0.8165
The Martian(2015) - 0.94281

2.ΑΝΑΖΗΤΗΣΗ-PRECISION/RECALL

Παρατηρούμε πως από τα 31 συνολικά αποτελέσματα που εμφανίστηκαν, τα 12 αφορούν την ταινία The Creator. Μάλιστα, τα συγκεκριμένα αποτελέσματα είναι τα ακόλουθα:

1.

 IMDb
[https://www.imdb.com > title](https://www.imdb.com/title)


The Creator (2023)

A man, once assigned to bring down the leader of the AI robots, experiences a devastating loss of his wife and child during an undercover mission. The ...

★★★★★ Rating: 7.1/10 · 47,612 votes ⓘ

[The Creator](#) · [Official Teaser](#) · [Gareth Edwards](#) · [User Reviews \(529\)](#)

2.


 Wikipedia
[https://en.wikipedia.org > wiki > The_Creator_\(2023_...](https://en.wikipedia.org/wiki/The_Creator_(2023_film))

The Creator (2023 film)

Set in 2070, 15 years after a nuclear detonation in Los Angeles and a war against artificial intelligence, an ex-special forces agent is recruited to hunt down ...

[The Creator \(soundtrack\)](#) · [Gareth Edwards \(director\)](#) · [John David Washington](#)


3.

 20th Century Studios
[https://www.20thcenturystudios.com > movies > the-cr...](https://www.20thcenturystudios.com/movies/the-cr...)

The Creator

Sep 29, 2023 — Amidst a future war between the human race and the forces of artificial intelligence, Joshua (Washington), a hardened ex-special forces ...

4.

 Rotten Tomatoes
[https://www.rottentomatoes.com > the_creator_2023](https://www.rottentomatoes.com/the_creator_2023)

The Creator

From writer/director Gareth Edwards ("Rogue One," "Godzilla") comes an epic sci-fi action thriller set amidst a future war between the human race and the forces ...

★★★★★ Rating: 67% · 296 votes ⓘ

5.

 The Guardian
[https://www.theguardian.com/film/sep/the-creator...](https://www.theguardian.com/film/sep/the-creator)

The Creator review – vast and exhilarating sci-fi actioner ...

Sep 26, 2023 — Director Gareth Edwards draws together the many strands of our current AI debate with tremendous boldness, conjuring up an intriguing and ...

6.

 Apple TV
<https://tv.apple.com/movie/the-creator>

The Creator

Amidst a future war between the human race and the forces of artificial intelligence, Joshua (John David Washington), a hardened ex-special forces age...

7.

 The New York Times
[https://www.nytimes.com/2023/09/28/movies/the-c...](https://www.nytimes.com/2023/09/28/movies/the-creator)

'The Creator' Review: Or How I Learned to Stop Worrying ...

Sep 28, 2023 — In this hectic, futuristic action film, John David Washington hunts down a threatening artificial intelligence with the baby face of a ...

8.

 BBC
[https://www.bbc.com/culture/article/20230926-t...](https://www.bbc.com/culture/article/20230926-the-creator)

The Creator film review: A 'jaw-droppingly distinctive' sci-fi

Sep 26, 2023 — **The Creator** is a breathlessly fast, relentlessly tense thriller which has Joshua racing from location to location, from rustic village to ...


9.

 Twitter
<https://twitter.com/creatorthefilm>

The Creator (@creatorthefilm) / X

The Creator's posts ... #TheCreator is a truly original, provocative big-budget sci-fi movie of the kind you so rarely see anymore. Go see it in theaters and ...

10.

 FilmAffinity
<https://www.filmaffinity.com/film397878>

The Creator (2023)

The **Creator** is a film directed by Gareth Edwards with John David Washington, Madeleine Yuna Voyles, Gemma Chan, Allison Janney, Ken Watanabe .

★★★★☆ Rating: 6.1/10 · 4,388 votes ⓘ

11.

 The Movie Database
<https://www.themoviedb.org/movie/670292-the-cr...>

The Creator (2023) — The Movie Database (TMDb)

Sep 29, 2023 — Amid a future war between the human race and the forces of artificial intelligence, a hardened ex-special forces agent grieving the ...

12.

 Vanity Fair
<https://www.vanityfair.com/hollywood/2023/10/i...>

It's Not Too Late to Make 'The Creator' a Sleeper Hit

Oct 2, 2023 — Edwards blends his dark speculative fiction with the sweeping sentiment of an old-fashioned epic. Like Avatar, **The Creator** is a stranger-in-a- ...

Άρα, έχουμε 12 σωστά και 19 λάθος αποτελέσματα.

Σύμφωνα με την εκφώνηση επίσης, έχουμε και άλλα 550 ακόμη αποτελέσματα που σχετίζονται με την ταινία, δηλαδή σωστά, τα οποία δεν βρέθηκαν.

Άρα, βάση των παραπάνω πληροφοριών έχουμε:

- True Positive (TP)= 12
- False Positive (FP)= 19
- False Negative (FN)= 550

Υπολογίζουμε τα Precision, Recall και F-Measure ως εξής:

$$\text{➤ Precision} = \frac{TP}{TP+FP} = \frac{12}{12+19} = 0.387096774 \approx \mathbf{0.3871}$$

$$\text{➤ Recall} = \frac{TP}{TP+F} = \frac{12}{12+550} = 0.021352313 \approx \mathbf{0.0214}$$

$$\text{➤ F-Measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Re}} = 2 \cdot \frac{0.387096774 \cdot 0.021352313}{0.387096774 + 0.021352313} = 0.020236087$$

$$\Rightarrow \text{F-Measure} \approx \mathbf{0.0202}$$

Παρατηρούμε ότι η **Precision**, δηλαδή η ακρίβεια των αποτελεσμάτων, είναι περίπου 0.3871, αυτό σημαίνει ότι περίπου το 38.71% των στοιχείων που ταξινομήθηκαν ως θετικά ήταν πράγματι θετικά. Είναι ένα σχετικά χαμηλό Precision το οποίο υποδεικνύει ότι το σύστημα μπορεί να κάνει πολλές λανθασμένες θετικές προβλέψεις. Εάν θέλουμε να βελτιώσουμε τη Precision, μπορούμε να εξετάσουμε διάφορες τεχνικές, όπως η βελτιστοποίηση των παραμέτρων του μοντέλου, η επιλογή ή η εξαγωγή καλύτερων χαρακτηριστικών, η χρήση διαφορετικών αλγορίθμων μάθησης, κλπ.

Επίσης, όταν η **Recall** είναι 0.0214, αυτό σημαίνει ότι περίπου το 2.14% των πραγματικά θετικών στοιχείων εντοπίστηκαν σωστά από το σύστημα ταξινόμησης. Η Recall είναι ένα μέτρο που δείχνει πόσα από τα πραγματικά θετικά στοιχεία εντοπίζονται σωστά από το σύστημα ταξινόμησης, δηλαδή εξετάζει την πληρότητα. Έτσι, ένα χαμηλό ποσοστό Recall υποδεικνύει ότι το σύστημα μπορεί να χάνει πολλά από τα πραγματικά θετικά στοιχεία κατά την ταξινόμηση. Εάν θέλουμε να βελτιώσουμε την Recall, μπορούμε να εξετάσουμε διάφορες προσεγγίσεις, όπως την προσαρμογή του κατωφλίου απόφασης, την επιλογή ή την προσθήκη επιπλέον χαρακτηριστικών, την χρήση διαφορετικών μοντέλων μηχανικής μάθησης, κλπ.

Τέλος, το **F-Measure** είναι 0.0202, αυτό υποδεικνύει ότι το σύστημα ταξινόμησης έχει πολύ χαμηλή απόδοση. Το F-Measure είναι ένας συνδυασμός της Precision και της Recall και χρησιμοποιείται για να αξιολογήσει την ισορροπία μεταξύ της ακρίβειας και της πληρότητας του συστήματος ταξινόμησης. Είναι δηλαδή ο σταθμισμένος αρμονικός μέσος όρος των Precision και Recall. Στην περίπτωσή μας, το χαμηλό F-Measure υποδεικνύει ότι το σύστημα ταξινόμησης πραγματοποιεί τόσο λίγες σωστές θετικές προβλέψεις (χαμηλή ακρίβεια) όσο και χάνει πολλά από τα πραγματικά θετικά στοιχεία (χαμηλή πληρότητα).

Το χαμηλό F-Measure είναι ένα σημάδι ότι πρέπει να επανεξετάσουμε το σύστημα ταξινόμησης και να αξιολογήσουμε τις τεχνικές που χρησιμοποιούμε για την ταξινόμηση, ώστε να βελτιώσουμε την απόδοσή του.

3. RECOMMENDER SYSTEMS

$$* X = 9 + 1 = 10$$

a.

Ο ακόλουθος πίνακας αποτελείται από τις βαθμολογίες 6 διαφορετικών χρηστών για τις 5 ταινίες του ερωτήματος 1 με βαθμολογία από 1 (καθόλου καλή) έως 10 (εξαιρετική).

Χρήστης/Ταινία	Sunshine	The Martian	Moon	Contact	Gattaca
1	10	5		1	7
2	4	5		9	5
3	6	2	4	2	10
4	7	3	6	7	3
5		4	10	8	6
6	9	9	10	6	8

Στη συνέχεια, θα υπολογίσουμε την ομοιότητα (similarity) μεταξύ των 6 χρηστών χρησιμοποιώντας δυο μεθόδους: Ευκλείδεια απόσταση και Pearson Correlation.

➤ Ευκλείδεια Απόσταση

Αρχικά θα υπολογίσουμε για κάθε ζεύγος χρηστών το άθροισμα των τετραγώνων των διαφορών ανάμεσα στις βαθμολογίες, σε όσες ταινίες έχουν βαθμολογήσει και οι δύο.

Σύμφωνα με αυτό το άθροισμα θα βγάλουμε συμπεράσματα για την ομοιότητα.

Θα βασιστούμε στους ακόλουθους τύπους από τη θεωρία:

- For Item j rated by User 1 and User 2:

$$\text{sum} = \text{sum} + (\text{rating}(\text{User 1, Item j}) - \text{rating}(\text{User 2, Item j}))^2$$

και

- $$\text{similarity} = \frac{1}{1 + \sqrt{\text{sum}}}$$

Για τους παραπάνω υπολογισμούς χρησιμοποίησα τον ακόλουθο κώδικα σε python:

```
import numpy as np
from itertools import combinations

# movie ratings and users
ratings = {
    "user 1": {"Sunshine": 10, "The Martian": 5, "Contract": 1, "Gattaca": 7},
    "user 2": {"Sunshine": 4, "The Martian": 5, "Contract": 9, "Gattaca": 5},
    "user 3": {"Sunshine": 6, "The Martian": 2, "Moon": 4, "Contract": 2, "Gattaca": 10},
    "user 4": {"Sunshine": 7, "The Martian": 3, "Moon": 6, "Contract": 7, "Gattaca": 3},
    "user 5": {"The Martian": 4, "Moon": 10, "Contract": 8, "Gattaca": 6},
    "user 6": {"Sunshine": 9, "The Martian": 9, "Moon": 10, "Contract": 6, "Gattaca": 8}
}
users = list(ratings.keys())

# Euclidean distance or similarity
def euclidean_similarity(user1, user2):
    common_movies = set(ratings[user1].keys()) & set(ratings[user2].keys())
    summary = sum((ratings[user1][movie] - ratings[user2][movie]) ** 2 for movie in common_movies)
    return 1 / (1 + np.sqrt(summary))

user_combinations = combinations(users, 2)
euclidean_similarities = {(user1, user2): euclidean_similarity(user1, user2) for user1, user2 in user_combinations}

# Print results
for (user1, user2), similarity in euclidean_similarities.items():
    print(f"Euclidean similarity between {user1} and {user2}: {similarity}")
```

Και προέκυψαν τα ακόλουθα αποτελέσματα:

```
⇒ Euclidean similarity between user 1 and user 2: 0.08930134977850068
Euclidean similarity between user 1 and user 3: 0.14459058185587106
Euclidean similarity between user 1 and user 4: 0.11034777731716484
Euclidean similarity between user 1 and user 5: 0.122828568570857
Euclidean similarity between user 1 and user 6: 0.13231996486433337
Euclidean similarity between user 2 and user 3: 0.09682998898940481
Euclidean similarity between user 2 and user 4: 0.179128784747792
Euclidean similarity between user 2 and user 5: 0.36602540378443865
Euclidean similarity between user 2 and user 6: 0.11519216806670013
Euclidean similarity between user 3 and user 4: 0.10056040392403998
Euclidean similarity between user 3 and user 5: 0.09441387963324659
Euclidean similarity between user 3 and user 6: 0.08563786063744523
Euclidean similarity between user 4 and user 5: 0.16139047779640892
Euclidean similarity between user 4 and user 6: 0.0994491992362644
Euclidean similarity between user 5 and user 6: 0.1482675827043134
```

Τα αποτελέσματα των παραπάνω υπολογισμών της Ευκλείδειας Απόστασης μεταξύ των χρηστών βρίσκονται στον ακόλουθο πίνακα:

	Χρήστης 1	Χρήστης 2	Χρήστης 3	Χρήστης 4	Χρήστης 5	Χρήστης 6
Χρήστης 1		0.08930	0.14459	0.11035	0.12283	0.13232
Χρήστης 2	0.08930		0.09683	0.17913	0.36603	0.11519
Χρήστης 3	0.14459	0.09683		0.10056	0.09441	0.08564
Χρήστης 4	0.11035	0.17913	0.10056		0.16139	0.09945
Χρήστης 5	0.12283	0.36603	0.09441	0.16139		0.14827
Χρήστης 6	0.13232	0.11519	0.08564	0.09945	0.14827	

➤ Pearson Correlation

Αρχικά θα υπολογίσουμε το άθροισμα των ratings και του τετραγώνου των ratings για κάθε χρήστη σε ζεύγη, για όσες ταινίες έχουν βαθμολογήσει και οι δύο.

- $Sum1 = \text{sum}(\text{ratings user1}), Sum2 = \text{sum}(\text{ratings user2})$
- $Sum1Sq = \text{sum}[(\text{ratings user1})^2], Sum2Sq = \text{sum}[(\text{ratings user2})^2]$

Στη συνέχεια υπολογίζουμε το άθροισμα των γινομένων των ratings για τις n ταινίες που έχουν αξιολογήσει και οι δυο.

- $pSum = \text{sum}((\text{rating user1 for item i}) * (\text{rating user2 for item i}))$

Έπειτα, υπολογίζουμε τον αριθμητή και τον παρονομαστή ως εξής:

- $$num = psum - \frac{sum1 \cdot sum2}{n}$$
- $$den = \sqrt{(\text{sum1sq} - \frac{sum1^2}{n}) \cdot (\text{sum2sq} - \frac{sum2^2}{n})}$$

Έτσι, υπολογίζουμε την συσχέτιση ως εξής :

- $$correlation = \frac{num}{den}$$

Και την ομοιότητα ως εξής:

- $$similarity = \frac{1 + correlation}{2}$$

Για τους παραπάνω υπολογισμούς χρησιμοποίησα τον ακόλουθο κώδικα σε python (συνέχισα στο προηγούμενο script με το ευκλείδειο):

```
# Pearson correlation similarity between 2 users
def pearson_correlation_similarity(user1, user2):
    common_movies = set(ratings[user1].keys()) & set(ratings[user2].keys())
    n = len(common_movies)
    if n == 0:
        return 0 # Handle division by zero when no common movies
    sum1 = sum([ratings[user1][movie] for movie in common_movies])
    sum2 = sum([ratings[user2][movie] for movie in common_movies])
    sq_sum1 = sum([ratings[user1][movie]**2 for movie in common_movies])
    sq_sum2 = sum([ratings[user2][movie]**2 for movie in common_movies])
    pSum = sum([ratings[user1][movie] * ratings[user2][movie] for movie in common_movies])

    num = pSum - (sum1 * sum2) / n
    den = np.sqrt((sq_sum1 - sum1**2 / n) * (sq_sum2 - sum2**2 / n))
    if den == 0:
        return 0 # Handle division by zero in denominator
    correlation = num / den

    return (1 + correlation) / 2

user_combinations = combinations(users, 2)
pearson_correlation_similarities = {(user1, user2): pearson_correlation_similarity(user1, user2) for user1, user2 in user_combinations}

# Print results
for (user1, user2), similarity in pearson_correlation_similarities.items():
    print(f"Pearson correlation similarity between {user1} and {user2}: {similarity}")
```

Και προέκυψαν τα ακόλουθα αποτελέσματα:

```
Pearson correlation similarity between user 1 and user 2: 0.03705553940563833
Pearson correlation similarity between user 1 and user 3: 0.8112715087397335
Pearson correlation similarity between user 1 and user 4: 0.46176404435490637
Pearson correlation similarity between user 1 and user 5: 0.1726731646460114
Pearson correlation similarity between user 1 and user 6: 0.9058538516854008
Pearson correlation similarity between user 2 and user 3: 0.24485268160687063
Pearson correlation similarity between user 2 and user 4: 0.6952833664712358
Pearson correlation similarity between user 2 and user 5: 0.9330127018922192
Pearson correlation similarity between user 2 and user 6: 0.02165539689253948
Pearson correlation similarity between user 3 and user 4: 0.3396167430320539
Pearson correlation similarity between user 3 and user 5: 0.46590028302647635
Pearson correlation similarity between user 3 and user 6: 0.5591082804679327
Pearson correlation similarity between user 4 and user 5: 0.9070457892053472
Pearson correlation similarity between user 4 and user 6: 0.4034765821868317
Pearson correlation similarity between user 5 and user 6: 0.5377964473009227
```

Τα αποτελέσματα των παραπάνω υπολογισμών για την Pearson Correlation μεταξύ των χρηστών βρίσκονται στον ακόλουθο πίνακα:

	Χρήστης 1	Χρήστης 2	Χρήστης 3	Χρήστης 4	Χρήστης 5	Χρήστης 6
Χρήστης 1		0.03706	0.81127	0.46176	0.17267	0.90585
Χρήστης 2	0.03706		0.24485	0.69528	0.93301	0.02166
Χρήστης 3	0.81127	0.24485		0.33962	0.46590	0.55911
Χρήστης 4	0.46176	0.69528	0.33962		0.90705	0.40348
Χρήστης 5	0.17267	0.93301	0.46590	0.90705		0.53780
Χρήστης 6	0.90585	0.02166	0.55911	0.40348	0.53780	

b.

Χρησιμοποιώντας το **K-Nearest Neighbors** με $k=2$ και weighted average και με τις δύο μετρικές (Ευκλείδεια απόσταση και Pearson Correlation) θα κάνουμε μια πρόβλεψη για το πως θα αξιολογήσει την ταινία Moon ο χρήστης 2.

➤ Ευκλείδεια Απόσταση

Ανατρέχουμε στο αντίστοιχο πίνακα παραπάνω, όπου έχουμε :

	Χρήστης 1	Χρήστης 2	Χρήστης 3	Χρήστης 4	Χρήστης 5	Χρήστης 6
Χρήστης 2	0.08930		0.09683	0.17913	0.36603	0.11519

Παρατηρούμε πως ο Χρήστης 2 έχει τη μεγαλύτερη ομοιότητα με τον Χρήστη 5, καθώς η τιμή ομοιότητας μεταξύ τους είναι 0.36603, η μεγαλύτερη από όλες τις υπόλοιπες τιμές. Η επόμενη μεγαλύτερη ομοιότητα είναι με τον Χρήστη 4, με τιμή ομοιότητας 0.17913.

Αφού βρήκαμε τους $K=2$ πιο κοντινούς γείτονες του χρήστη 2, θα υπολογίσουμε το:

$\text{predictedRating}_{U2-EA} =$

$$\frac{\text{similarity}(\text{user2}, \text{user5}) \cdot \text{rating}(\text{user5}) + \text{similarity}(\text{user2}, \text{user4}) \cdot \text{rating}(\text{user4})}{\text{similarity}(\text{user2}, \text{user5}) + \text{similarity}(\text{user2}, \text{user4})}$$

$$\Rightarrow \text{predictedRating}_{U2-EA} = \frac{0.36603 \cdot 10 + 0.17913 \cdot 6}{0.36603 + 0.17913} = \frac{4.73508}{0.54516}$$

$$\Rightarrow \text{predictedRating}_{U2-EA} \approx 8.68596$$

Άρα, με μετρική την Ευκλείδεια Απόσταση περιμένουμε πως ο Χρήστης 2 θα αξιολογήσει την ταινία “Moon” με 9.

➤ Pearson Correlation

Και για αυτή τη μετρική θα ακολουθήσουμε παρόμοια διαδικασία:

	Χρήστης 1	Χρήστης 2	Χρήστης 3	Χρήστης 4	Χρήστης 5	Χρήστης 6
Χρήστης 2	0.03706		0.24485	0.69528	0.93301	0.02166

Παρατηρούμε πως ο Χρήστης 2 έχει τη μεγαλύτερη ομοιότητα με τον Χρήστη 5 καθώς η τιμή ομοιότητας μεταξύ τους είναι 0.93301, η μεγαλύτερη από όλες τις υπόλοιπες τιμές. Η επόμενη μεγαλύτερη ομοιότητα είναι με τον Χρήστη 4, με τιμή ομοιότητας 0.69528.

Αφού βρήκαμε τους K=2 πιο κοντινούς γείτονες του χρήστη 2, θα υπολογίσουμε το:

$\text{predictedRating}_{U2-PS} =$

$$\frac{\text{similarity}(\text{user2}, \text{user5}) \cdot \text{rating}(\text{user5}) + \text{similarity}(\text{user2}, \text{user4}) \cdot \text{rating}(\text{user4})}{\text{similarity}(\text{user2}, \text{user5}) + \text{similarity}(\text{user2}, \text{user4})}$$

$$\Rightarrow \text{predictedRating}_{U2-PS} = \frac{0.93301 \cdot 10 + 0.69528 \cdot 6}{0.93301 + 0.69528} = \frac{13.50178}{1.62829}$$

$$\Rightarrow \text{predictedRating}_{U2-PS} \approx 8.29597$$

Άρα, με μετρική την Pearson Correlation περιμένουμε πως ο Χρήστης 2 θα αξιολογήσει την ταινία “Moon” με 8.

Παρατηρήσεις:

Η πρώτη παρατήρηση που μπορούμε να κάνουμε είναι ότι οι δύο μετρικές, η Ευκλείδεια απόσταση και η Pearson Correlation, παρήγαγαν ελάχιστα διαφορετικά αποτελέσματα για την πρόβλεψη της βαθμολογίας που θα έδινε ο χρήστης 2 σε μια ταινία. Κατά τη χρήση και των δυο μετρικών επιλέχθηκαν οι ίδιοι γείτονες, συγκεκριμένα οι χρήστες 4 και 5.

Επίσης παρατηρούμε και τη διαφορά ανάμεσα στις βαθμολογίες του εκάστοτε χρήστη για την ίδια ταινία. Συγκεκριμένα, ο χρήστης 4 έβαλε 6 και ο χρήστης 5 έβαλε 10.

Βλέπουμε δηλαδή ότι οι δυο μετρικές έδωσαν διαφορετικό αποτέλεσμα (με μικρή απόκλιση μεταξύ τους ωστόσο) ακόμα και αν επιλέχθηκαν οι ίδιοι γείτονες.

Φυσικά, η (μικρή) διαφορά στο αποτέλεσμα θα μπορούσε να αποδοθεί στη διαφορετική φύση των δύο μετρικών. Η Ευκλείδεια απόσταση εστιάζει αποκλειστικά στην απόσταση μεταξύ των χρηστών στον χώρο των αξιολογήσεών τους, ενώ ο συντελεστής συσχέτισης Pearson λαμβάνει υπόψη και τη συσχέτιση μεταξύ των αξιολογήσεων των χρηστών.

c.

Θα χρησιμοποιήσουμε τις προτιμήσεις των χρηστών στις ταινίες για να προτείνουμε φίλους. Στον ακόλουθο πίνακα αναγράφονται οι σχέσεις που είναι αρκετά πιθανές, ανάλογα την μετρική (Ευκλείδεια απόσταση ή Pearson Correlation).

Ευκλείδεια Απόσταση	Pearson Correlation
Χρήστης 1 & 3 (0.14459)	Χρήστης 1 & 6 (0.90585)
Χρήστης 2 & 5 (0.36603)	Χρήστης 2 & 5 (0.93301)
	Χρήστης 3 & 1 (0.81127)
Χρήστης 4 & 2 (0.17913)	Χρήστης 4 & 5 (0.90705)
Χρήστης 6 & 5 (0.14827)	

Τελικά, οι προτάσεις φιλίας για τους χρήστες είναι οι εξής :

- Χρήστες 1 & 3
- Χρήστες 2 & 5

Καθώς αυτό υποστηρίζεται και από τις δυο μετρικές όπως φαίνεται στο παραπάνω πίνακάκι.

~ Τέλος Εργασίας ~