# ΔΙΟΙΚΗΣΗ ΨΗΦΙΑΚΗΣ ΕΠΙΧΕΙΡΗΣΗΣ

## ΕΡΓΑΣΙΑ 4 ΑΝΑΖΗΤΗΣΗ ΚΑΙ RECOMMENDER SYSTEMS

Κουστένης Χρίστος | el20227
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ | ΣΗΜΜΥ | Εργαστήριο Διοίκησης Πληροφοριακών Συστημάτων | Ακ. Έτος 2023-2024

# 1. ΑΝΑΖΗΤΗΣΗ – TFIDF

**term_frequency = tf(t, d) = (the occurencies of term t in d) / (total number of words in d)**

d : ο αριθμός του εξεταζόμενου κειμένου

t : ο εξεταζόμενος όρος

tf("earth",1) = 1/161 = 0.0062112

tf("earth", 2) = 2/128 = 0.0156250

tf("earth", 3) = 0/206 = 0.0000000

tf("earth", 4) = 2/169 = 0.0118343

tf("earth", 5) = 1/104 = 0.0096154

tf("mission",1) = 1/161 = 0.0062112

tf("mission", 2) = 2/128 = 0.0156250

tf("mission", 3) = 1/206 = 0.0048544

tf("mission", 4) = 0/169 = 0.0000000

tf("mission", 5) = 2/104 = 0.0192308

tf("astro",1) = 1/161 = 0.0062112

tf("astro", 2) = 1/128 = 0.0078125

tf("astro", 3) = 0/206 = 0.0000000

tf("astro", 4) = 2/169 = 0.0118343

tf("astro", 5) = 1/104 = 0.0096154

---

**idft(t,D) = log( N /df(t,D) )**

N = |D| : Πλήθος εξεταζόμενων κειμένων

df(t,D) = the number of papers in which the word is present is df.

idf("earth", D) = log(5 / 4) = 0.3219281

idf("mission", D) = log(5 / 4) = 0.3219281

idf("astro", D) = log(5 / 4) = 0.3219281

**tf-idf(t, d) = tf(t, d) * idf(t)**

tf-idft(“earth”, 1) = tf(“earth”, 1) * idf(“earth”) = 0.0019996
tf-idft(“earth”, 2) = tf(“earth”, 2) * idf(“earth”) = 0.0050301
tf-idft(“earth”, 3) = tf(“earth”, 3) * idf(“earth”) = 0.0000000
tf-idft(“earth”, 4) = tf(“earth”, 4) * idf(“earth”) = 0.0038098
tf-idft(“earth”, 5) = tf(“earth”, 5) * idf(“earth”) = 0.0030955

tf-idft(“mission”, 1) = tf(“mission”, 1) * idf(“mission”) = 0.0019996
tf-idft(“mission”, 2) = tf(“mission”, 2) * idf(“mission”) = 0.0050301
tf-idft(“mission”, 3) = tf(“mission”, 3) * idf(“mission”) = 0.0015628
tf-idft(“mission”, 4) = tf(“mission”, 4) * idf(“mission”) = 0.0000000
tf-idft(“mission”, 5) = tf(“mission”, 5) * idf(“mission”) = 0.0061909

tf-idft(“astro”, 1) = tf(“astro”, 1) * idf(“astro”) = 0.0019996
tf-idft(“astro”, 2) = tf(“astro”, 2) * idf(“astro”) = 0.0025151
tf-idft(“astro”, 3) = tf(“astro”, 3) * idf(“astro”) = 0.0000000
tf-idft(“astro”, 4) = tf(“astro”, 4) * idf(“astro”) = 0.0038098
tf-idft(“astro”, 5) = tf(“astro”, 5) * idf(“astro”) = 0.0030955

Query : “earth, mission, astro”
tf(“earth”, q) = 0.3333333
tf(“mission”, q) = 0.3333333
tf(“astro”, q) = 0.3333333
idf = 0.3219281
tf-idf[“earth”, q] = tf-idf[“mission”, q] = tf-idf[“astro”, q] = 0.1073094
Πρέπει να μετατρέψουμε το παραπάνω Query σε διάνυσμα της μορφής :
Q = [0.1073094 0.1073094 0.1073094]
tf-idft_d1 = [0.0019996 0.0019996 0.0019996]
tf-idft_d2 = [0.0050301 0.0050301 0.0025151]
tf-idft_d3 = [0.0000000 0.0015628 0.0000000]

tf-idft_d4 = [0.0038098  0.0000000  0.0038098]

tf-idft_d5 = [0.0030955  0.0061909  0.0030955]

**Cosine Similarity(Q, Di)** = $\frac{Q \cdot Di}{\|Q\| * \|Di\|}$

cos_sim[d1] =  1

cos_sim[d2] =  0.962252

cos_sim[d3] =  0.577350

cos_sim[d4] =  0.816497

cos_sim[d5] =  0.942812

Άρα, η κατάταξη των αποτελεσμάτων θα είναι :

1.

2.

5.

4.

3.

Σχόλια

Αυτή είναι μια απλοποιημένη προσέγγιση TF-IDF. Άλλοι παράγοντες, όπως ονόματα οντοτήτων (π.χ., πλανήτες, διαστημικές υπηρεσίες) ή συνώνυμα που δεν εξετάζονται ρητά εδώ, θα μπορούσαν να βελτιώσουν περαιτέρω την κατάταξη. Ανάλογα με τη μηχανή αναζήτησης, πρόσθετοι παράγοντες όπως η δημοτικότητα ή οι κριτικές χρηστών μπορεί επίσης να επηρεάσουν τα τελικά αποτελέσματα που παρουσιάζονται στον χρήστη. Συνολικά, η υπολογισμένη κατάταξη φαίνεται λογική βάσει των παρεχόμενων περιγραφών ταινιών και της προσέγγισης TF-IDF με ίση στάθμιση για τους όρους αναζήτησης.

# 2.ΑΝΑΖΗΤΗΣΗ-PRECISION/RECALL

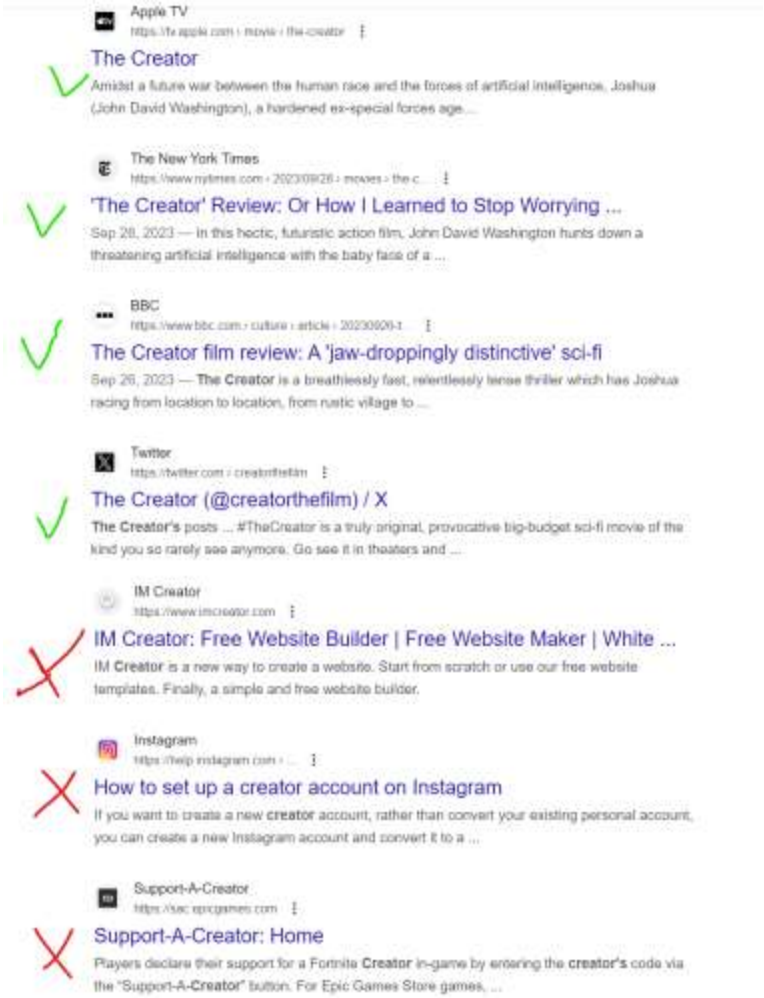
α.

Συνολικά έχουμε 31 αποτελέσματα

True Positive : 12 (Σχετίζονταν με την ταινία και βρέθηκαν)

False Positive : 19 (Δε σχετίζονταν με την ταινία και βρέθηκαν)


Roblox Studio
Roblox Creator Hub
Unlock your imagination with Roblox Creator Hub - the ultimate platform for building and publishing games for free. Join a vibrant community of creators ...
The Guardian
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 ...
Cambridge Dictionary
CREATOR | English meaning - Cambridge Dictionary
creator definition: 1. someone who has invented something: 2. someone who has invented something: 3. a person who.... Learn more.
Book Creator
Book Creator - Love Learning - Book Creator app
Teachers love it. Students love it. Book Creator is the simplest, most inclusive way to create content in the classroom.
Google Play
Creator Studio - Εφαρμογές στο Google Play
Με μια σειρά εργαλείων που έχουν σχεδιαστεί για να κάνουν τον προγραμματισμό, την ανάλυση και τη δημιουργία εσόδων από το βίντεό σας πιο εύκολη από ποτέ, ...
Rating: 4.3 · 686,443 votes · Free · Android · Business/Productivity
Vocabulary.com
Creator - Definition, Meaning & Synonyms
A person who invents, produces, or makes things is called a creator. If you are an author, you are the creator of the characters in your books.
TikTok Creator Marketplace
TikTok Creator Marketplace
All your creator marketing tools in one place. Collaborate with creators based on your industry, budget, and business goals. Creator discovery. Campaign ...


Apple TV
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...
The New York Times
'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 ...
BBC
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 ...
Twitter
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 ...
IM Creator
IM Creator: Free Website Builder | Free Website Maker | White ...
IM Creator is a new way to create a website. Start from scratch or use our free website templates. Finally, a simple and free website builder.
Instagram
How to set up a creator account on Instagram
If you want to create a new creator account, rather than convert your existing personal account, you can create a new Instagram account and convert it to a ...
Support-A-Creator
Support-A-Creator: Home
Players declare their support for a Fortnite Creator in-game by entering the creator's code via the "Support-A-Creator" button. For Epic Games Store games, ...


Τα σωστά αποτελέσματα είναι αυτά στα οποία παρατηρήσαμε στην περιγραφή ότι αναφέρουν είτε τον σκηνοθέτη της ταινίας είτε τον πρωταγωνιστή είτε αναφέρουν τον όρο "movie". Επίσης, ιστοσελίδες όπως Rotten Tomatoes ή TMDB είναι προφανές ότι θα βγάζανε αποτελέσματα που αφορούν τη συγκεκριμένη ταινία. Τα υπόλοιπα αποτελέσματα αφορούσαν την κυριολεκτική σημασία της λέξης creator και απευθυνόντουσαν σε εικαστικής φύσεως απαιτήσεις ή σε σχεδιασμό ιστοσελίδων κτλ. Επίσης υπήρχαν και μερικά αποτέλεσματα που αφορούσαν την ετυμολόγική ανάλυση της λέξης creator. Αν μαζί με τον

όρο “creator”αναφερόταν και όρος “movie” ή το όνομα του σκηνοθέτη ή του πρωταγωνιστή θα παρατηρούσαμε σημαντικά καλύτερα αποτελέσματα.

β.

Συνολικά έχουμε 31 αποτελέσματα ιστοσελίδων

False Negative : 550 (Σχετίζονταν με την ταινία και δεν βρέθηκαν)

True Positive : 12 (Σχετίζονταν με την ταινία και βρέθηκαν)

False Positive : 19 (Δε σχετίζονταν με την ταινία και βρέθηκαν)

i. **Precision = TruePositives / (TruePositives + FalsePositives)**

Precision = 12/(12+19) = **0.3871**

ii. **Recall = TruePositives / (TruePositives + FalseNegatives)**

Recall = 12/(12+550) = **0.0214**

iii. **F-Measure = (2 * Precision * Recall) / (Precision + Recall)**

F-Measure = **0.0406**

Σχόλια

Το precision είναι σχετικά χαμηλό, πράγμα που σημαίνει ότι η μηχανή αναζήτησης επιστρέφει πολλά άσχετα αποτελέσματα.

Το recall είναι επίσης πολύ χαμηλό, πράγμα που σημαίνει ότι από τη μηχανή αναζήτησης λείπουν πολλά σχετικά αποτελέσματα.

Το F-Measure είναι επίσης χαμηλό, πράγμα που σημαίνει ότι η συνολική απόδοση της μηχανής αναζήτησης δεν είναι πολύ καλή.

Συνολικά, η μηχανή αναζήτησης δεν κάνει πολύ καλή δουλειά στον εντοπισμό αποτελεσμάτων που σχετίζονται με την ταινία. Αυτό μπορεί να οφείλεται σε διάφορους παράγοντες, όπως η χρήση ενός υπερβολικά ευρέος όρου αναζήτησης ή η έλλειψη σχετικών πληροφοριών στο ευρετήριο.

# 3. RECOMMENDER SYSTEMS

**A. M = 03120227 => X = 8**

α.
Για την υλοποιήση του υπολογιστικού κομματιού χρησιμοποιούμε το παρακάτω script σε Python :

```
import numpy as np

# Sample review score matrix (6 users, 5 movies)
review_scores = np.array([
    [8, 5, np.nan, 1, 7],  # User 1 has not reviewed movie 3
    [4, 5, np.nan, 9, 5],  # User 2 has not reviewed movie 3
    [6, 2, 4, 2, 8],
    [7, 3, 6, 7, 3],
    [np.nan, 4, 8, 8, 6],  # User 5 has not reviewed movie 1
    [9, 9, 10, 6, 8]
])

# Initialize a matrix to store Euclidean distances
num_users, num_movies = review_scores.shape
euclidean_distances = np.zeros((num_users, num_users))

# Calculate Euclidean distances between pairs of users
for i in range(num_users):
    for j in range(num_users):
        mask = ~np.isnan(review_scores[i]) & ~np.isnan(review_scores[j])  # Mask
for common ratings
        if np.any(mask):  # If there are common ratings
            common_ratings_i = review_scores[i][mask]
            common_ratings_j = review_scores[j][mask]
            squared_diff = np.sum((common_ratings_i - common_ratings_j) ** 2)  #
Calculate squared differences
            euclidean_distances[i, j] = squared_diff  # Store the squared
difference
            euclidean_distances[j, i] = squared_diff  # Distance matrix is
symmetric

# Calculate similarity using the formula: 1 / (1 + sqrt(sum of squared
differences))
eucl_similarities = 1 / (1 + np.sqrt(euclidean_distances))

print(eucl_similarities)

#Calculate Pearson correlation
pearson_correlations = np.zeros((num_users, num_users))

# Calculate Pearson correlation between pairs of users
```

```
for i in range(num_users):
    for j in range(num_users):
       mask = ~np.isnan(review_scores[i]) & ~np.isnan(review_scores[j])  # Mask
for common ratings
       if np.any(mask):  # If there are common ratings
        common_ratings_i = review_scores[i][mask]
        common_ratings_j = review_scores[j][mask]
        #First find the sum of the ratings for each user
        sum_i = np.sum(common_ratings_i)
        sum_j = np.sum(common_ratings_j)
        #Then find the sum of the squared ratings for each user
        sum_squared_i = np.sum(common_ratings_i**2)
        sum_squared_j = np.sum(common_ratings_j**2)

        #Then find the sum of the product of the ratings for each user
        pSum = np.dot(common_ratings_i, common_ratings_j)
        #Then find the number of movies that both users have rated
        num_movies = np.sum(mask)
        # numerator
        numerator = pSum - (sum_i*sum_j/num_movies)
        # denominator
        denominator = np.sqrt((sum_squared_i -
sum_i**2/num_movies)*(sum_squared_j - sum_j**2/num_movies))
        # Pearson correlation
        pearson_correlations[i, j] = numerator / denominator
        pearson_correlations[j, i] = numerator / denominator
# find the similarity
pearson_similarities = (1 + pearson_correlations)/2

print(pearson_similarities)
```

Euclidean Similarities:

[[1.  0.09837532 0.2052131  0.11696133 0.12282857 0.13231996]
 [0.09837532 1.  0.10608785 0.17912878 0.3660254  0.11519217]
 [0.2052131  0.10608785 1.  0.11787845 0.11433842 0.08704668]
 [0.11696133 0.17912878 0.11787845 1.  0.2052131  0.0994492 ]
 [0.12282857 0.3660254  0.11433842 0.2052131  1.  0.14118785]
 [0.13231996 0.11519217 0.08704668 0.0994492  0.14118785 1.  ]]

Pearson Similarities:

[[1.  0.02046295 0.88584036 0.36012428 0.17267316 0.91876284]
 [0.02046295 1.  0.21186902 0.69528337 0.9330127  0.0216554 ]
 [0.88584036 0.21186902 1.  0.39708384 0.5  0.60114435]
 [0.36012428 0.69528337 0.39708384 1.  0.94331035 0.40347658]
 [0.17267316 0.9330127  0.5  0.94331035 1.  0.3725882 ]
 [0.91876284 0.0216554  0.60114435 0.40347658 0.3725882  1.  ]]

b.

Για την υλοποιήση του υπολογιστικού κομματιού χρησιμοποιούμε το παρακάτω script σε Python :

```
# For User 2, find the nearest neighbors using Euclidean distance and Pearson correlation
k = 2

# For User 2, find the nearest neighbors using Euclidean distance
user_2_euclidean_similarities = eucl_similarities[1]  # Similarities of User 2 with all other users
user_2_euclidean_similarities[1] = 0  # Exclude similarity with itself
nearest_neighbors_euclidean = np.argsort(user_2_euclidean_similarities)[::-1][:k]
print("Nearest neighbors for User 2 using Euclidean distance:", nearest_neighbors_euclidean+1)

# For User 2, find the nearest neighbors using Pearson correlation
user_2_pearson_similarities = pearson_similarities[1]  # Similarities of User 2 with all other users
user_2_pearson_similarities[1] = 0  # Exclude similarity with itself
nearest_neighbors_pearson = np.argsort(user_2_pearson_similarities)[::-1][:k]
print("Nearest neighbors for User 2 using Pearson correlation:", nearest_neighbors_pearson+1)

movie_index = 2
# For User 2, predict the review score for Movie 3(column index = 2) using Euclidean distance

# Filter the nearest neighbors who have rated Movie 3
```

```
nearest_neighbors_rated_movie_3 =
nearest_neighbors_euclidean[~np.isnan(review_scores[nearest_neighbors_euclidean,
2])]

# Calculate weighted average of scores for Movie 3
weighted_sum = 0
total_similarity = 0
for neighbor_index in nearest_neighbors_rated_movie_3:
    similarity = eucl_similarities[1][neighbor_index]
    weighted_sum += similarity * review_scores[neighbor_index, 2]
    total_similarity += similarity

# Predict the review score for Movie 3 for User 2
predicted_score = weighted_sum / total_similarity
print("Predicted review score for Movie 3 for User 2 using Euclidean distance:",
predicted_score)

# For User 2, predict the review score for Movie 3(column index = 2) using
Pearson correlation
nearest_neighbors_rated_movie_3 =
nearest_neighbors_pearson[~np.isnan(review_scores[nearest_neighbors_pearson, 2])]

# Calculate weighted average of scores for Movie 3
weighted_sum = 0
total_similarity = 0
for neighbor_index in nearest_neighbors_rated_movie_3:
    similarity = pearson_similarities[1][neighbor_index]
    weighted_sum += similarity * review_scores[neighbor_index, 2]
    total_similarity += similarity

# Predict the review score for Movie 3 for User 2
predicted_score = weighted_sum / total_similarity
print("Predicted review score for Movie 3 for User 2 using Pearson correlation:",
predicted_score)
```

Nearest neighbors for User 2 using Euclidean distance: [5 4]

Nearest neighbors for User 2 using Pearson correlation: [5 4]

Predicted review score for Movie 3 for User 2 using Euclidean distance: 7.342832583823387

Predicted review score for Movie 3 for User 2 using Pearson correlation: 7.14599883893346

Σχόλια

- Τόσο η Ευκλείδεια απόσταση όσο και η μέθοδος συσχέτισης Pearson δίνουν τους ίδιους πλησιέστερους γείτονες για τον Χρήστη 2, οι οποίοι είναι οι Χρήστες 5 και 4.
- Σύγκριση: Και οι δύο μέθοδοι δίνουν αρκετά κοντινές προβλέψεις για τη βαθμολογία σχολίων της ταινίας 3 για τον χρήστη 2. Η μέθοδος της Ευκλείδειας απόστασης προβλέπει ελαφρώς υψηλότερη βαθμολογία σε σύγκριση με τη μέθοδο συσχέτισης Pearson. Αυτό συμβαίνει λόγω μεγαλύτερης ομοιότητας(και άρα βάρους) μεταξύ Χρήστη 2 και Χρήστη 5 ο οποίος αξιολογεί με 8 την ταινία.
- Γενικότερα, σε μεγαλύτερα datasets εάν οι αξιολογήσεις ταινιών έχουν σημαντικές διακυμάνσεις, η συσχέτιση Pearson μπορεί να είναι μια καλύτερη επιλογή, καθώς είναι λιγότερο ευαίσθητη σε ακραίες τιμές.

c.

Οι πιθανότερες σχέσεις φαίνεται να είναι οι παρακάτω καθώς παρουσιάζουν υψηλές τιμές similarity και για τις δύο μετρικές γεγονός που προσδίδει συνέπεια στις μετρήσεις και άρα μας οδηγεί σε ασφαλέστερες υποθέσεις.

Users 2 & 5 : Eyclidean = 0.3660254(1$^{st}$ highest) | Pearson = 0.9330127(2$^{nd}$ highest)

Users 1 & 3 : Eyclidean = 0.2052131(2$^{nd}$ highest) | Pearson = 0.88584036(4$^{th}$ highest)

Users 4 & 5 : Eyclidean = 0.2052131(2$^{nd}$ highest) | Pearson = 0.94331035 (1$^{st}$ highest)

Παρόλα αυτά το πλήθος δεδομένων είναι μικρό άρα υπάρχει μεγάλη πιθανότητα λανθασμένης εκτίμησης. Επίσης, η πρόταση φίλων γενικότερα μπορεί να εξαρτάται και από άλλα ενδιαφέροντα ή παράγοντες(πχ. ηλικία, φύλο κτλ). Στα πλαίσια όμως μιας πλατφόρμας ταινιών οι εκτιμήσεις μας ίσως είναι αρκετές για να υποδείξουν κάποιες επιτυχείς προτάσεις φίλων.