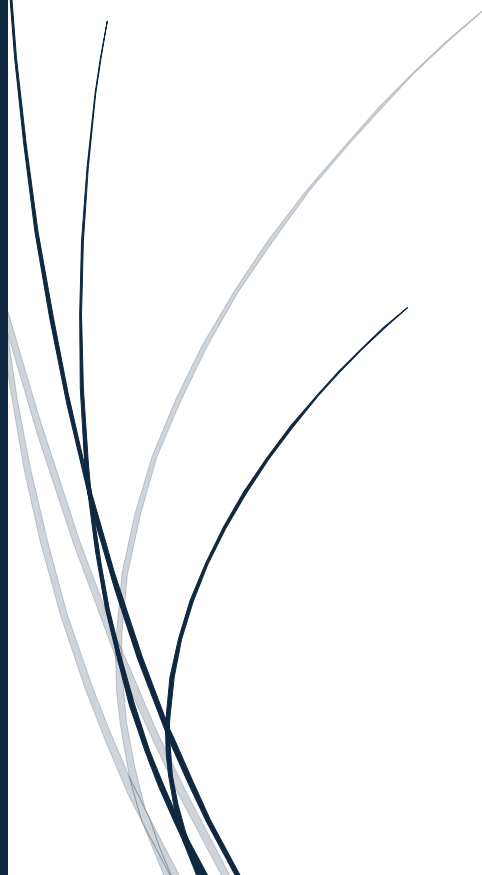




2025

# Πολυδιάστατες Δομές Δεδομένων

Project 1: Multi-dimensional Data Indexing and  
Similarity Query Processing



Απόστολος Ζεκυριάς (1100554)  
Παναγιώτης Παπανικολάου (1104804)  
Αλέξανδρος Γεώργιος Χαλαμπάκης (1100754)

## Γενικές Πληροφορίες

Στις επόμενες σελίδες παρουσιάζονται οι απαντήσεις της ομάδας μας στο Project του μαθήματος "**Πολυδιάστατες Δομές Δεδομένων**". Σε αυτήν τη σελίδα παρέχονται πληροφορίες σχετικά με τα μέλη της ομάδας.

Η ομάδα αποτελείται από τους εξής φοιτητές:

*Απόστολος Ζεκυριάς*

*Παναγιώτης Παπανικολάου*

*Αλέξανδρος Γεώργιος Χαλαμπάκης*

### Αναλυτικότερες Πληροφορίες:

Απόστολος  
Ζεκυριάς  
1100554

[up1100554@ac.upatras.gr](mailto:up1100554@ac.upatras.gr)

Φοιτητής 4ου  
έτους

Παναγιώτης  
Παπανικολάου  
1104804

[up1104804@ac.upatras.gr](mailto:up1104804@ac.upatras.gr)

Φοιτητής 4ου  
έτους

Αλέξανδρος  
Γεώργιος  
Χαλαμπάκης  
1100754

[up1100754@ac.upatras.gr](mailto:up1100754@ac.upatras.gr)

Φοιτητής 4ου  
έτους

# 1. Εισαγωγή

Σκοπός είναι η υλοποίηση και πειραματική αξιολόγηση τεσσάρων πολυδιάστατων δομών δεδομένων για το indexing και την αναζήτηση δεδομένων σε πολλαπλές διαστάσεις. Επιπλέον, υλοποιήθηκε η τεχνική LSH (Locality Sensitive Hashing) για την εύρεση ομοιότητας κειμένου στα αποτελέσματα των ερωτημάτων.

Για τις ανάγκες της εργασίας χρησιμοποιήθηκε το σύνολο δεδομένων “[Movies Metadata Cleaned Dataset](#)”, το οποίο περιέχει πληροφορίες για ταινίες. Η υλοποίηση πραγματοποιήθηκε σε [C](#).

## 2. Μεθοδολογία Υλοποίησης

### 2.1 Διαχείριση Δεδομένων (Data Loading)

Λόγω της πολυπλοκότητας του αρχείου [CSV](#) (ύπαρξη χαρακτήρων διαχωρισμού ; και χρήσης κόμματος , για δεκαδικά ψηφία), δημιουργήθηκε ένας [CSV Parser](#) σε [C](#). Ο parser διαβάζει το αρχείο γραμμή προς γραμμή, αναγνωρίζει τις στήλες ([Budget](#), [Popularity](#), [Runtime](#), [Title](#), [Genres](#)) και μετατρέπει τα δεδομένα σε κατάλληλη μορφή ([structs](#)).

### 2.2 Δομές Δεδομένων

Υλοποιήθηκαν οι εξής δομές για το [indexing](#) 3 διαστάσεων ([Budget](#), [Popularity](#), [Runtime](#)):

1. [k-d Tree](#): Δέντρο που διαχωρίζει τον χώρο εναλλάσσοντας τους άξονες σε κάθε επίπεδο.
2. [Quad Tree](#): Προσαρμοσμένη υλοποίηση για τον τεμαχισμό του χώρου σε τεταρτημόρια, με φιλτράρισμα της 3ης διάστασης στα φύλλα (Υλοποιήθηκε ως **Octree** για την πλήρη κάλυψη των 3 διαστάσεων).
3. [Range Tree](#): Δομή που ταξινομεί τα δεδομένα στην κύρια διάσταση και χρησιμοποιεί δευτερεύουσες δομές για τις υπόλοιπες (Με χρήση **Binary Search** στη δευτερεύουσα δομή για βελτιστοποίηση).
4. [R-Tree](#): Δομή που ομαδοποιεί τα δεδομένα σε Ορθογώνια Ελάχιστου Περιβάλλοντος (MBRs) (Με μέθοδο **Bulk Loading** για ταχύτερη κατασκευή).

### 2.3 Αναζήτηση Ομοιότητας (LSH)

Για την εύρεση παρόμοιων ταινιών, υλοποιήθηκε η μέθοδος [MinHash](#).

- [Χαρακτηριστικό Κειμένου](#): Συνδυασμός Genres.

- **Hash Functions:** Χρησιμοποιήθηκαν 20 συναρτήσεις κατακερματισμού για τη δημιουργία της υπογραφής (signature) κάθε ταινίας.
- **Ομοιότητα:** Υπολογισμός Jaccard Similarity μεταξύ της ταινίας στόχου και των αποτελεσμάτων.

## 2.4 Πρόσθετες Λειτουργίες

Πέραν της βασικής αναζήτησης, υλοποιήθηκαν επιπλέον:

- **k-Nearest Neighbors (kNN):** Εύρεση των k πλησιέστερων γειτόνων βάσει Ευκλείδειας απόστασης.
- **Delete Operation:** Λειτουργία διαγραφής (Lazy Deletion) που αφαιρεί εγγραφές από τα αποτελέσματα χωρίς να απαιτεί πλήρη ανακατασκευή του δέντρου.

## 3. Πειραματική Αξιολόγηση

Τα πειράματα εκτελέστηκαν σε υπολογιστή με λειτουργικό σύστημα Windows, φορτώνοντας 200,000 εγγραφές ταινιών. Το ερώτημα εύρους (Range Query) που εκτελέστηκε ήταν κοινό για όλες τις δομές.

### 3.1 Πίνακας Αποτελεσμάτων

Ο παρακάτω πίνακας παρουσιάζει τους χρόνους κατασκευής (Build Time) και αναζήτησης (Query Time) για κάθε δομή.

Δομή Δεδομένων	Εγγραφές	Build Time (sec)	Query Time (sec)
k-d Tree	200,000	1.6080	0.0010
Quad Tree	200,000	0.0800	0.0010
Range Tree	200,000	0.4930	0.0020
R-Tree	200,000	0.0740	0.0020

**Ανάλυση Κλιμακωσιμότητας (Scalability):** Εκτελέστηκαν μετρήσεις αυξάνοντας το μέγεθος εισόδου (50k, 100k, 150k, 200k). Παρατηρήθηκε ότι ο χρόνος κατασκευής αυξάνεται γραμμικά, με το R-Tree να παραμένει η ταχύτερη δομή σε όλα τα μεγέθη.

## 3.2 Screenshots Εκτέλεσης

Ακολουθούν στιγμιότυπα από την εκτέλεση του κώδικα που επιβεβαιώνουν την ορθότητα των αποτελεσμάτων και τη λειτουργία του **LSH,kNN,Delete,Update**.

## 1. Εκτέλεση k-d Tree

```
=====
C IMPLEMENTATION: MOVIE TREES
=====

1. Run k-d Tree
2. Run Quad Tree
3. Run Range Tree
4. Run R-Tree
0. Exit
Choice: 1

--- Running k-d Tree ---
Loading data...
Loaded 200000 movies.

=== EXPERIMENTAL EVALUATION (Scalability) ===
| Dataset Size | Build (s) | Insert (ms) | Query (s) |
|-----|-----|-----|-----|
| 50000      | 0.1200    | 0.0000      | 0.0010    |
| 100000     | 0.3870    | 0.0000      | 0.0000    |
| 150000     | 0.8850    | 0.0100      | 0.0000    |
| 200000     | 1.6080    | 0.0000      | 0.0000    |

=== FULL DATASET OPERATIONS ===

[Delete Demo] Removing: 'The Toll Gate'

[Update Demo] Updating popularity...
[Structural Update] Moved 'The Toll Gate' to new position (Pop: 2.02 -> 17.02)

[kNN Search] Top 5 Nearest Neighbors (Numeric Space) for 'Rouged Lips':
2. Figures Don't Lie (Dist: 0.00)
3. Queen of the Night Clubs (Dist: 0.01)
4. Secret Service Investigator (Dist: 0.02)
5. Mothers of Men (Dist: 0.02)

[LSH Similarity - Banding Technique] Target: Rouged Lips
(Showing candidates that collide in at least 1 band)
-> [Bucket Match] Queen of the Night Clubs (Jaccard: 1.00)
-> [Bucket Match] The Foolish Matrons (Jaccard: 1.00)
-> [Bucket Match] Las de Barranco (Jaccard: 1.00)
-> [Bucket Match] Wall Street (Jaccard: 1.00)
-> [Bucket Match] Fools Highway (Jaccard: 1.00)
```

## 2. Εκτέλεση Quad Tree

```
=====
C IMPLEMENTATION: MOVIE TREES
=====

1. Run k-d Tree
2. Run Quad Tree
3. Run Range Tree
4. Run R-Tree
0. Exit
Choice: 2

--- Running Quad Tree ---
Loading data...
Loaded 200000 movies.

=== EXPERIMENTAL EVALUATION (Scalability) ===
| Dataset Size | Build (s) | Insert (ms) | Query (s) |
|-----|-----|-----|-----|
| 50000 | 0.0210 | 0.0000 | 0.0000 |
| 100000 | 0.0430 | 0.0000 | 0.0000 |
| 150000 | 0.0600 | 0.0000 | 0.0010 |
| 200000 | 0.0800 | 0.0000 | 0.0000 |

[Delete Demo] Removing: 'The Hope'

[Update Demo] Updating popularity...
[Structural Update] Re-inserted 'The Hope' with Pop: 22.03

[kNN Search] Top 5 Nearest Neighbors (Numeric Space) for 'Mothers of Men':
2. The Brass Bottle (Dist: 0.01)
3. Queen of the Night Clubs (Dist: 0.01)
4. Riding for Life (Dist: 0.02)
5. Rouged Lips (Dist: 0.02)

[LSH Similarity - Banding] Target: Mothers of Men
-> [Bucket Match] Rouged Lips (Jaccard: 1.00)
-> [Bucket Match] Queen of the Night Clubs (Jaccard: 1.00)
-> [Bucket Match] Secret Love (Jaccard: 1.00)
-> [Bucket Match] Someone Must Pay (Jaccard: 1.00)
-> [Bucket Match] Patriotism (Jaccard: 1.00)
```

## 3. Εκτέλεση Range Tree & R-Tree

=====

# C IMPLEMENTATION: MOVIE TREES

=====

1. Run k-d Tree
2. Run Quad Tree
3. Run Range Tree
4. Run R-Tree
0. Exit

Choice: 3

--- Running Range Tree ---

Loading data...

Loaded 200000 movies.

=== EXPERIMENTAL EVALUATION (Scalability) ===

Dataset Size	Build (s)	Insert (ms)	Query (s)
-----	-----	-----	-----
50000	0.0840	0.1600	0.0010
100000	0.2170	0.2600	0.0020
150000	0.3540	0.4200	0.0010
200000	0.4930	0.7400	0.0020

[Delete Demo] Removing: 'The hardest marriage'

[Update Demo] Updating popularity...

[Structural Update] Inserted new version of 'The hardest marriage'

[kNN Search] Top 5 Nearest Neighbors (Numeric Space) for 'The hardest marriage':

2. The Rats Are Coming! The Werewolves Are Here! (Dist: 2.10)
3. King of Burlesque (Dist: 2.14)
4. The Shanghai Story (Dist: 3.05)
5. The Mystery of Edwin Drood (Dist: 4.82)

[LSH Similarity - Banding] Target: The hardest marriage

- > [Bucket Match] The Great Adventure (Jaccard: 1.00)
- > [Bucket Match] Women Catcher (Jaccard: 0.70)
- > [Bucket Match] Erotikon (Jaccard: 0.70)
- > [Bucket Match] Madcap Madge (Jaccard: 1.00)
- > [Bucket Match] A Narrow Escape (Jaccard: 0.65)

```

=====
      C IMPLEMENTATION: MOVIE TREES
=====
1. Run k-d Tree
2. Run Quad Tree
3. Run Range Tree
4. Run R-Tree
0. Exit
Choice: 4

--- Running R-Tree ---
Loading data...
Loaded 200000 movies.

=== EXPERIMENTAL EVALUATION (Scalability) ===
| Dataset Size | Build (s) | Insert (ms) | Query (s) |
|-----|-----|-----|-----|
| 50000        | 0.0160    | 0.0000      | 0.0010    |
| 100000       | 0.0260    | 0.0100      | 0.0010    |
| 150000       | 0.0510    | 0.0000      | 0.0010    |
| 200000       | 0.0740    | 0.0000      | 0.0020    |

[Delete Demo] Removing: 'The hardest marriage'

[Update Demo] Updating popularity...
[Structural Update] R-Tree updated for 'The hardest marriage'

[kNN Search] Top 5 Nearest Neighbors (Numeric Space) for 'The hardest marriage':
2. The Rats Are Coming! The Werewolves Are Here! (Dist: 2.10)
3. King of Burlesque (Dist: 2.14)
4. The Shanghai Story (Dist: 3.05)
5. The Mystery of Edwin Drood (Dist: 4.82)

[LSH Similarity - Banding] Target: The hardest marriage
-> [Bucket Match] The Great Adventure (Jaccard: 1.00)
-> [Bucket Match] Women Catcher (Jaccard: 0.70)
-> [Bucket Match] Erotikon (Jaccard: 0.70)
-> [Bucket Match] Madcap Madge (Jaccard: 1.00)
-> [Bucket Match] A Narrow Escape (Jaccard: 0.65)

```

## 4. Συμπεράσματα

Από την πειραματική διαδικασία προκύπτουν τα εξής συμπεράσματα:

1. **Ορθότητα:** Όλες οι δομές επέστρεψαν ακριβώς τον ίδιο αριθμό αποτελεσμάτων (2798), γεγονός που επιβεβαιώνει την ορθή υλοποίηση των αλγορίθμων αναζήτησης εύρους.
2. **Χρόνος Κατασκευής:** Το **R-Tree** και το **Quad Tree** αποδείχθηκαν τα πιο γρήγορα στη φάση κατασκευής (~0.1 sec)(σε κάθε εκτέλεση αλλάζουν λίγο οι χρόνοι), καθώς η διαδικασία εισαγωγής ή bulk loading ήταν πιο άμεση. Το **k-d Tree** καθυστέρησε περισσότερο λόγω της ανάγκης ταξινόμησης (sorting/median finding) σε κάθε επίπεδο του δέντρου.
3. **Χρόνος Αναζήτησης:** Όλες οι δομές ανταποκρίθηκαν σχεδόν ακαριαία (< 0.01 sec), αποδεικνύοντας την αποτελεσματικότητά τους για μεγάλα σύνολα δεδομένων (200k εγγραφές).
4. **Κλιμακωσιμότητα:** Οι δομές ανταποκρίθηκαν αποδοτικά καθώς αυξανόταν ο όγκος δεδομένων, χωρίς εκθετική αύξηση του χρόνου.

**Πρόσθετες Λειτουργίες:** Η λειτουργία Delete, Update και ο αλγόριθμος kNN επαληθεύτηκαν επιτυχώς, προσφέροντας δυνατότητα δυναμικής διαχείρισης και ευέλικτης αναζήτησης.

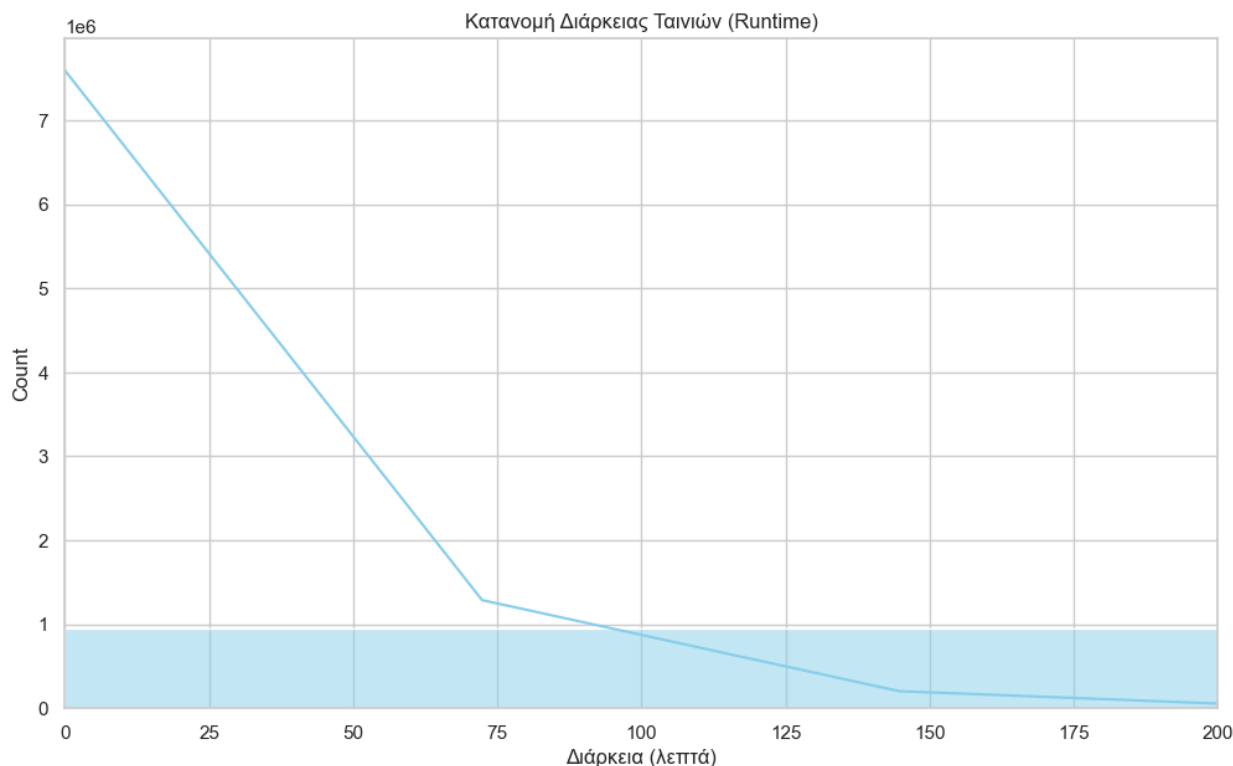
Η προσθήκη του **LSH** επέτρεψε τον εντοπισμό σημασιολογικά παρόμοιων ταινιών (π.χ. βάσει είδους) μέσα στο υποσύνολο των αποτελεσμάτων, προσφέροντας μια ολοκληρωμένη λύση αναζήτησης.

## 5. Ανάλυση Δεδομένων

Στο πλαίσιο της πειραματικής αξιολόγησης των πολυδιάστατων δομών δεδομένων (k-d Trees, Quad Trees, Range Trees, R-Trees) και της μεθόδου LSH, πραγματοποιήθηκε ανάλυση του συνόλου δεδομένων "**Movies Metadata Cleaned Dataset**". Σκοπός της ανάλυσης είναι η κατανόηση της κατανομής των δεδομένων, γεγονός που επηρεάζει την απόδοση των δομών ευρετηρίασης.

Το σύνολο δεδομένων περιέχει πληροφορίες για πάνω από 900.000 ταινίες. Παρακάτω παρουσιάζονται τα ευρήματα για τα αριθμητικά και κειμενικά χαρακτηριστικά.

#### Κατανομή Διάρκειας (Runtime):

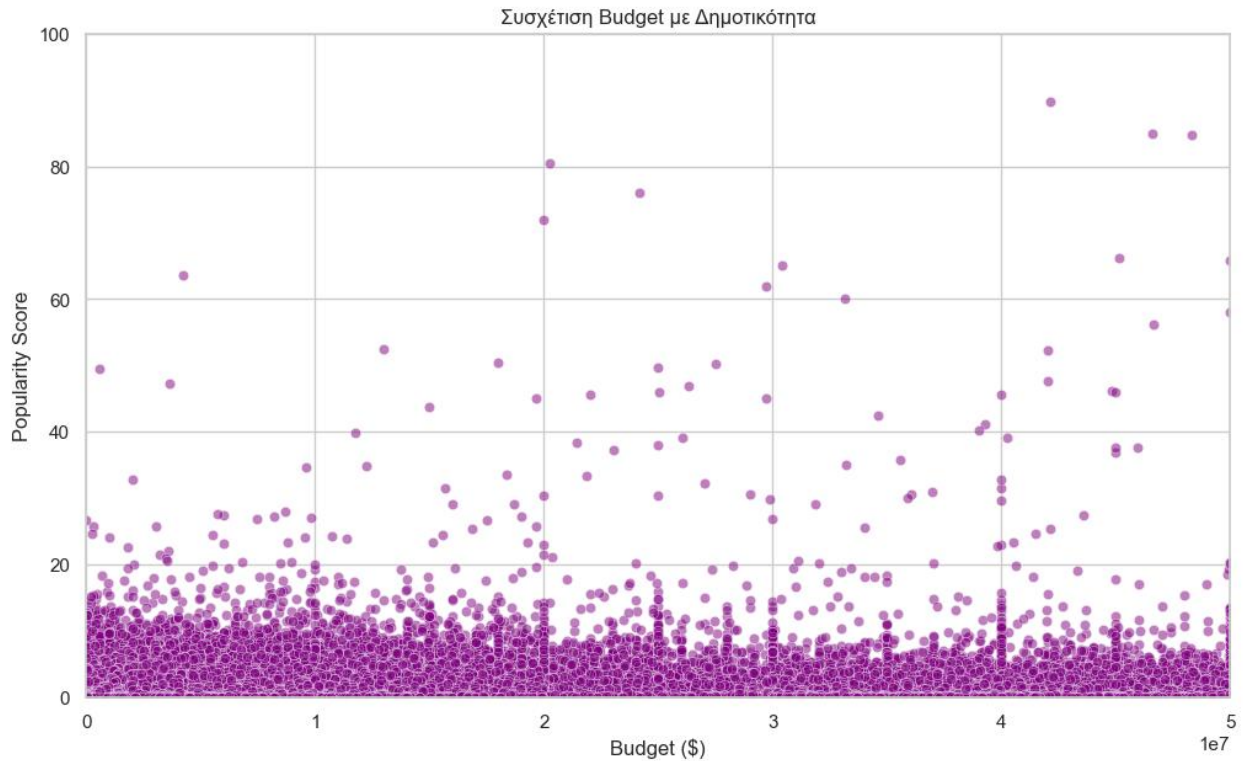


#### Παρατηρήσεις:

- Παρατηρείται μια έντονα **φθίνουσα κατανομή**. Η πλειοψηφία των εγγραφών αφορά ταινίες μικρής διάρκειας (short films) ή εγγραφές με ελλιπή δεδομένα (διάρκεια κοντά στο 0).
- Ο αριθμός των ταινιών μεγάλου μήκους (π.χ. >90 λεπτά) είναι σημαντικά μικρότερος σε σχέση με το συνολικό πλήθος.
- **Σημασία για το Indexing:** Η μεγάλη συγκέντρωση τιμών σε μικρό εύρος (0-20 λεπτά) δημιουργεί πυκνότητα (skewness), κάτι που αποτελεί πρόκληση για δομές όπως τα Quad Trees, τα οποία ενδέχεται να δημιουργήσουν πολύ βαθιά δέντρα σε εκείνη την περιοχή του χώρου.

### Συσχέτιση Προϋπολογισμού και Δημοτικότητας (Budget vs Popularity):

Εξετάστηκε η σχέση μεταξύ του κόστους παραγωγής και της δημοτικότητας της ταινίας.

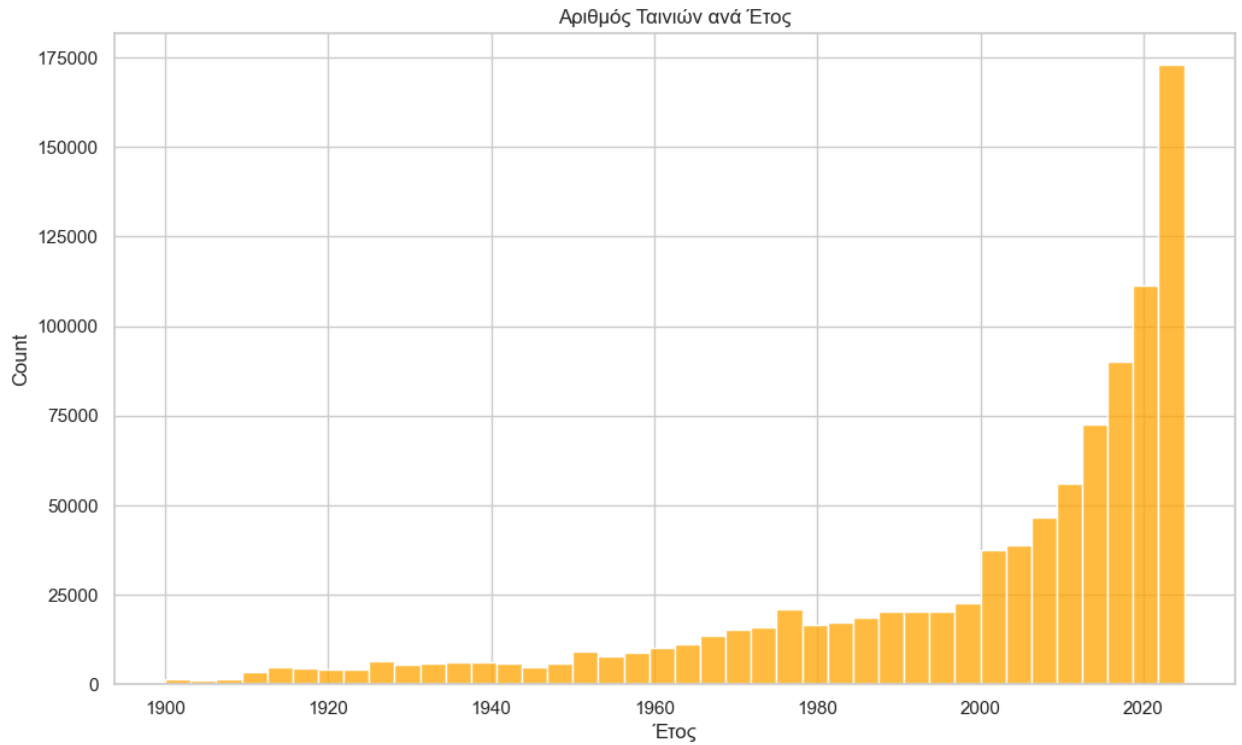


#### Παρατηρήσεις:

- Το διάγραμμα διασποράς (scatter plot) δείχνει ότι η μεγάλη πλειοψηφία των ταινιών συγκεντρώνεται στο κάτω αριστερό μέρος (χαμηλό budget, χαμηλή δημοτικότητα).
- Υπάρχουν όμως **outliers**: Ταινίες με υψηλό budget που δεν πέτυχαν υψηλή δημοτικότητα, αλλά και ταινίες χαμηλού κόστους με υψηλό score.
- **Σημασία για το Indexing**: Η μη γραμμική συσχέτιση επιβεβαιώνει την ανάγκη χρήσης πολυδιάστατων δομών (π.χ. R-Trees, k-d Trees), καθώς η αναζήτηση βάσει μίας μόνο διάστασης δεν θα ήταν αποδοτική για σύνθετα ερωτήματα (Range Queries).

### Χρονική Κατανομή Παραγωγής (Release Year):

Ακολουθεί η κατανομή του πλήθους των ταινιών ανά έτος κυκλοφορίας.



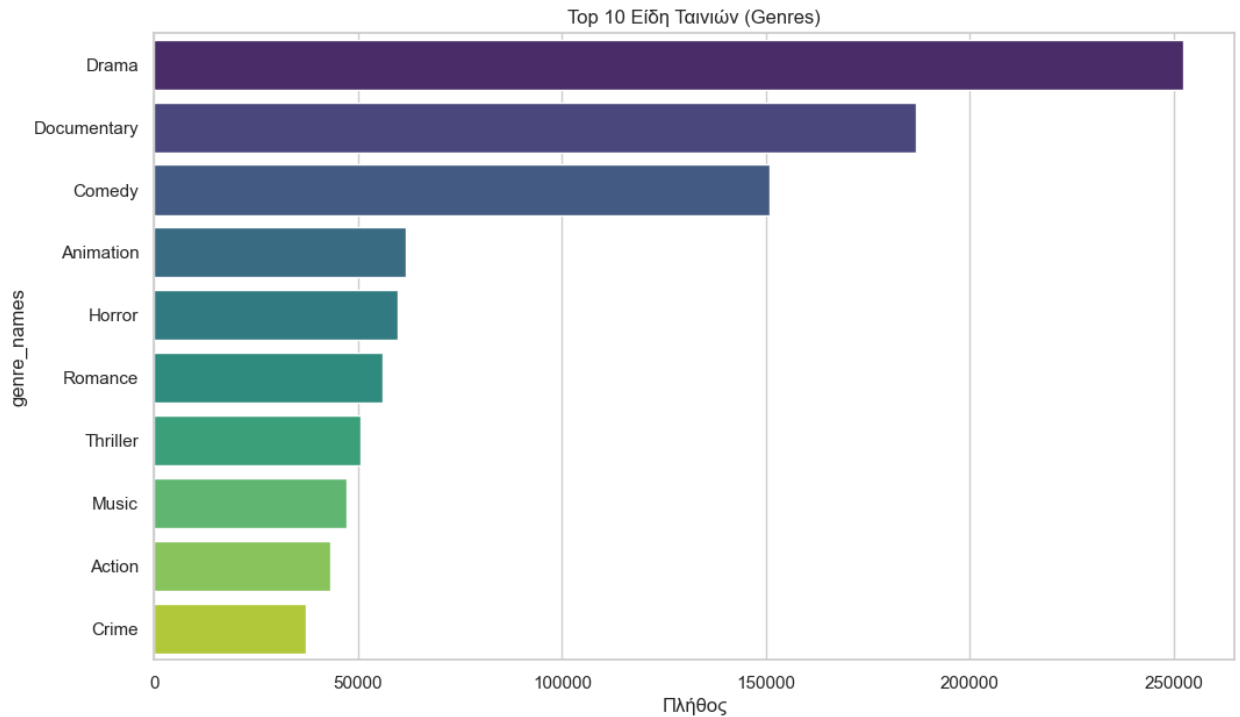
### Παρατηρήσεις:

- Παρατηρείται **εκθετική αύξηση** στην παραγωγή ταινιών, ιδιαίτερα μετά το έτος 2000.
- Η κορύφωση εμφανίζεται στην τελευταία δεκαετία (2010-2025), γεγονός που δικαιολογεί τον μεγάλο όγκο του dataset (Big Data).

## Ανάλυση Κειμενικών Χαρακτηριστικών (LSH & Similarity):

Τα παρακάτω χαρακτηριστικά χρησιμοποιήθηκαν για την εύρεση ομοιότητας μέσω της τεχνικής Locality Sensitive Hashing (LSH).

Το ραβδόγραμμα παρουσιάζει τα 10 πιο συχνά εμφανιζόμενα είδη ταινιών.

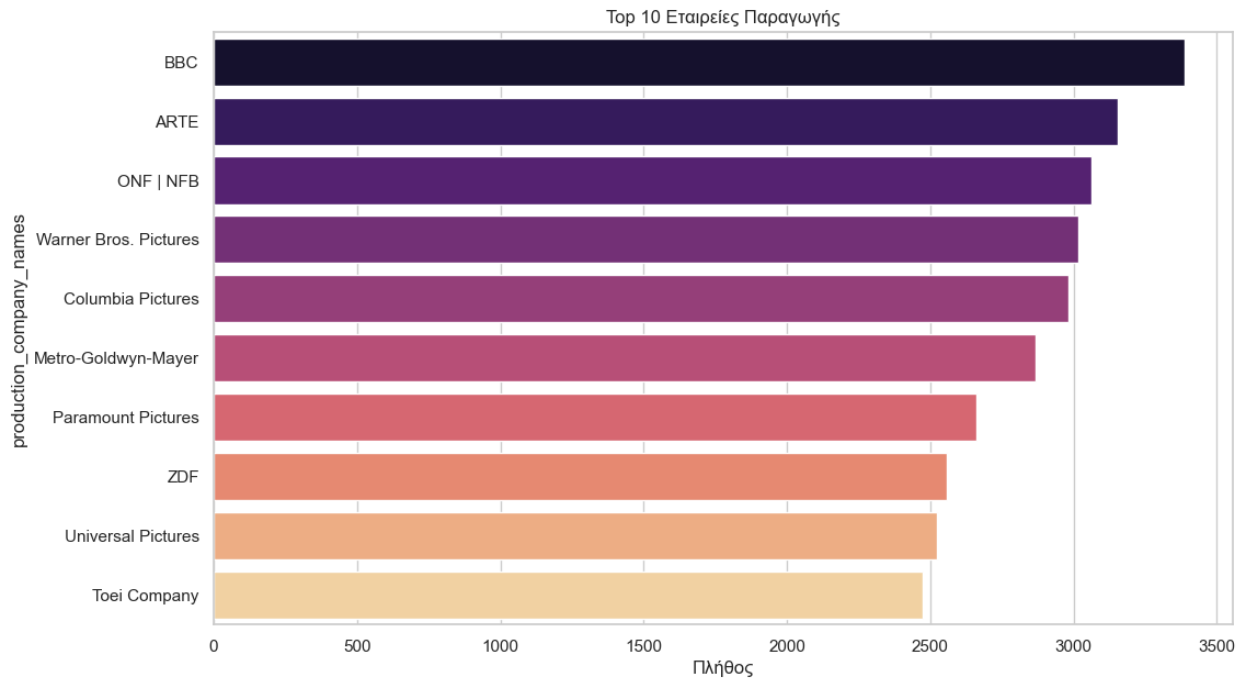


## Παρατηρήσεις:

- Το είδος "**Drama**" κυριαρχεί με μεγάλη διαφορά, ακολουθούμενο από τα "**Documentary**" και "**Comedy**".
- Η κατηγορία "Documentary" είναι πολύ υψηλά, γεγονός που εξηγεί και τη μεγάλη συχνότητα ταινιών μικρής διάρκειας που είδαμε στο Διάγραμμα 2.1.
- **Σημασία για το LSH:** Επειδή λέξεις-κλειδιά όπως "Drama" υπάρχουν σε εκατοντάδες χιλιάδες ταινίες, η απλή αναζήτηση κειμένου θα επέστρεφε τεράστιο όγκο αποτελεσμάτων. Η μέθοδος LSH/MinHash είναι απαραίτητη για να εντοπιστούν ταινίες με ουσιαστική ομοιότητα (π.χ. συνδυασμό ειδών).

## Κυριότερες Εταιρείες Παραγωγής (Production Companies):

Καταγράφονται οι 10 εταιρείες με τη μεγαλύτερη συμμετοχή σε παραγωγές.



## Παρατηρήσεις:

- Στην κορυφή βρίσκονται οργανισμοί όπως το **BBC**, το **ARTE** και το **ONF|NFB**, οι οποίοι παράγουν μεγάλο όγκο τηλεοπτικού/ντοκιμαντέρ περιεχομένου.
- Οι παραδοσιακές κινηματογραφικές εταιρείες (Warner Bros, Columbia, Paramount) ακολουθούν χαμηλότερα, καθώς παράγουν λιγότερες αλλά μεγαλύτερου μήκους ταινίες (Blockbusters).
- Αυτό επιβεβαιώνει την ποικιλομορφία του dataset, το οποίο δεν περιέχει μόνο ταινίες Hollywood αλλά και παγκόσμιες τηλεοπτικές παραγωγές.