**Αναλυτικό documentation**

create virtual environment and install dependencies:

python -m venv myenv

myenv\Scripts\activate

pip install pandas numpy matplotlib seaborn scikit-learn tensorflow

#this is more more organized work and to do not affect the global environment

Bhma1:

Τι ζητάει το Ερώτημα 1 και τι έχουμε κάνει

1. Καθαρισμός δεδομένων (Data Cleaning)

✔ Ζητούμενο: Να καθαρίσουμε ελλιπή ή εσφαλμένα δεδομένα.

✔ Τι κάναμε:

Αντικαταστήσαμε τις ελλιπές τιμές στο tmdbId με -1.

Μετατρέψαμε τα timestamps σε ημερομηνίες για να είναι αναγνώσιμα.

Εξετάσαμε τα στατιστικά των βαθμολογιών (rating.describe()).

📌 Συμπλήρωση:

Δεν εντοπίσαμε σημαντικά outliers στις βαθμολογίες, αλλά αν θέλουμε πιο αυστηρό έλεγχο για ακραίες τιμές, μπορούμε να εφαρμόσουμε τον κανόνα IQR (Interquartile Range) και να αφαιρέσουμε outliers.

2. Κανονικοποίηση & Διακριτοποίηση (Normalization & Discretization)

✔ Ζητούμενο: Κανονικοποίηση συνεχών δεδομένων και διακριτοποίηση όπου χρειάζεται.

✔ Τι κάναμε:

Κανονικοποιήσαμε τις βαθμολογίες στην κλίμακα [0,1] για να είναι συγκρίσιμες.

Διακριτοποιήσαμε τις βαθμολογίες σε Χαμηλή, Μέτρια, Υψηλή.

📌 Συμπλήρωση:

Μπορούμε να ελέγξουμε αν οι κατηγορίες που ορίσαμε (Χαμηλή, Μέτρια, Υψηλή) έχουν ισορροπημένη κατανομή. Αν υπάρχουν πολύ λίγα δεδομένα σε μια κατηγορία, ίσως χρειαστεί να αλλάξουμε τα όρια.

3. Μείωση όγκου δεδομένων (Dimensionality Reduction)

✔ Ζητούμενο: Μείωση όγκου δεδομένων αν είναι απαραίτητο.

✔ Τι κάναμε:

Επιλέξαμε μόνο τις απαραίτητες στήλες (movieId, title, genres, rating, rating\_normalized, rating\_category).

Αποφύγαμε δεδομένα που δεν χρειάζονται για την ανάλυση μας (π.χ. zip code, tmdbId).

📌 Συμπλήρωση:

Η μείωση του dataset μπορεί να γίνει και με PCA (Principal Component Analysis) για clustering, αλλά αυτό είναι προαιρετικό σε αυτή τη φάση.

4. Στατιστική Ανάλυση & Οπτικοποιήσεις

✔ Ζητούμενο: Να παρουσιάσουμε στατιστικά δεδομένα με διαγράμματα.

✔ Τι κάναμε:

Ιστόγραμμα της κατανομής των βαθμολογιών.

Box plot για εντοπισμό πιθανών ακραίων τιμών.

📌 Συμπλήρωση:

Μπορούμε να προσθέσουμε ένα bar chart με τις πιο δημοφιλείς ταινίες (ταινίες με τις περισσότερες αξιολογήσεις).

🟢 Συμπληρωματικά Βήματα

Θα εφαρμόσουμε τώρα:

Έλεγχο για outliers στις βαθμολογίες με τον κανόνα IQR.

Έλεγχο της κατανομής των κατηγοριοποιημένων βαθμολογιών.

Δημιουργία ενός bar chart με τις πιο δημοφιλείς ταινίες.

**What Are We Trying to Find with Clustering? 🤔**

The goal of clustering is to **group similar users or movies** based on their rating patterns. Specifically:

1. **K-Means:**
   * Groups users/movies into **K distinct clusters** based on **rating behavior**.
   * Helps us identify **rating patterns** (e.g., users who rate movies similarly).
2. **DBSCAN:**
   * Detects **dense regions of similar ratings** without requiring a fixed number of clusters.
   * Can identify **outliers** (users with very unique rating behavior).

**By comparing these methods, we want to see:**

* Which method **better segments the users/movies**?
* Are clusters **meaningful** (e.g., separating high-rating users from low-rating users)?
* Does **DBSCAN detect noise (outliers)** that K-Means doesn’t?

**How Can We Compare Clustering Methods? 📊**

1. **Cluster Size & Distribution**
   * How many users/movies belong to each cluster?
   * Are some clusters too small or too big?
2. **Average Rating Per Cluster**
   * Do clusters separate users/movies into **low, medium, and high rating groups**?
3. **Silhouette Score Comparison**
   * Measures how **well-separated** clusters are.
4. **Genre Influence on Clusters**
   * Do some clusters have a high concentration of certain movie genres?

**Step 3: Classification Task Breakdown**

The goal of Step 3 is to **train two classification models** using **Scikit-Learn, TensorFlow, or Keras** and compare their performance.

**🔹 Defining the Problem: What Are We Classifying?**

We need to convert our dataset into:

* **Features (X)** → Input data.
* **Labels (Y)** → The category/class we want to predict.

**Classification Approach: Predict "Good" vs. "Bad" Movies**

* **Binary Classification Problem**:
  + "Good Movie" (rating **≥ 3.5**) → **Class 1**
  + "Bad Movie" (rating **< 3.5**) → **Class 0**
* **Feature Selection (X)**:
  + **User ID (Categorical Encoding)**
  + **Movie ID (Categorical Encoding)**
  + **Genres (One-Hot Encoding)**
  + **Number of Ratings Per Movie**
  + **Cluster Assignments (from K-Means)**
  + **Average Movie Rating** (if available)

**🔹 Models to Use**

1. **LS-SVM (Least Squares Support Vector Machine)**
   * Good for binary classification.
   * Works well with small datasets.
   * Robust to high-dimensional data.
2. **Neural Network (Multi-Layer Perceptron - MLP)**
   * Can model complex relationships in data.
   * Works well with categorical and numerical features.
   * Requires careful tuning.

**🔹 Model Training & Evaluation**

* **Preprocess Data:** Convert categorical features, normalize numerical data.
* **Train Both Models:** One **LS-SVM** and one **Neural Network (MLP)**.
* **Compare Performance Using:**
  + **Confusion Matrix**
  + **ROC-AUC Curve**
  + **Accuracy, Precision, Recall, F1-Score**