# Ανάκτηση Πληροφορίας
# 2022-2023

*Υλοποιητικό Project*

## ‘Elasticsearch, K-means Clustering and Neural Networks implemented on Book Rating Dataset’


Μέλη Ομάδας

Λεκαράκος Αλέξιος
AM:1069367
Email: st1069367@ceid.upatras.gr

Κωνσταντίνος Παναγιώτης Κωστόπουλος
AM:1067482
Email: st1067482@ceid.upatras.gr


Github code repository:
https://github.com/allexxlekk/IR-2023-PROJECT

# Elasticsearch:

## Εισαγωγή Δεδομένων στην Elasticsearch

Η υλοποίηση αυτού του ερωτήματος εμπεριέχεται στο αρχείο elastic-search.py
Η εγκατάσταση της Elasticsearch έγινε τοπικά σε περιβάλλον linux ubuntu.
Το παραπάνω αρχείο περιέχει συναρτήσεις για την:

- Σύνδεση στην Elasticsearch

```
def es_connect():
    """Connects to the cluster and notifies user."""

    client = Elasticsearch("http://localhost:9200")
    if client.ping():
        print("Successfully connected to Elasticsearch cluster.")
        print(
            f" Cluster Name: {client.info().body['cluster_name']}\nCluster UUID: {client.info().body['cluster_uuid']}"
        )
    else:
        print("Could not connect to Elasticsearch cluster.")

    return client
```

- Δημιουργία κατάλληλου index (books)

```
def create_index(client, index_name="books"):
    """Initializes specified index in the cluster."""

    if client.indices.exists(index=index_name):
        client.indices.delete(index=index_name)

    client.indices.create(index=index_name)
    print(f"New index created: -{index_name}")
```

- Εισαγωγή Δεδομένων (βιβλίων)

```
def add_data(client, index_name="books"):
    """Parses data from BX-Books.csv as Elasticsearch documents
    and adds them to the specified index."""

    df = pd.read_csv("BX-Books.csv")

    for i, r in df.iterrows():

        # Creates document for each row and adds it to index.
        doc = {
            "isbn": r["isbn"],
            "book_title": r["book_title"],
            "book_author": r["book_author"],
            "year_of_publication": r["year_of_publication"],
            "publisher": r["publisher"],
            "summary": r["summary"],
            "category": r["category"],
        }
        client.index(index=index_name, id=i, document=doc)
```

- Main:

```
if __name__ == "__main__":
    es = es_connect()
    create_index(es)
    add_data(es)
```

## Υλοποίηση ερωτημάτων στην Elasticsearch

Η υλοποίηση αυτού του ερωτήματος εμπεριέχεται στο αρχείο query.py
Η διαδικασία υποβολής ερωτήματος στην Elasticsearch είναι η εξής:

- Σύνδεση με Elasticsearch

- Υποβολή ερωτήματος και ανάκτηση αποτελεσμάτων

```python
def es_search(es_client, user_query):
    """Queries Elastisearch client based on user input."""
    # Create and submit the query
    query_body = {"bool": {"should": {"match": {"book_title": user_query}}}}
    response = es_client.search(index="books", query=query_body, size=1000)
    # Create list of results
    es_results = []
    for hit in response["hits"]["hits"]:
        result = {
            "isbn": hit["_source"]["isbn"],
            "title": hit["_source"]["book_title"],
            "author": hit["_source"]["book_author"],
            "year_of_publication": hit["_source"]["year_of_publication"],
            "rank": float(hit["_score"]),
        }
        es_results.append(result)

    return es_results
```

Το query μπορεί να εμπλουτιστεί περισσότερο ανάλογα τις ανάγκες της εφαρμογής. Σε αυτη την περίπτωση δίνουμε μια απλή εκδοχή για να κάνουμε showcase την λειτουργία της Elasticsearch.

- Σε περίπτωση που ο χρήστης εισάγει και κάποιο user id

1) Φόρτωση ratings για το συγκεκριμένο id

```python
def get_ratings(user_id):
    """Get all the ratings of a certain user."""
    # Parse the data file to get ratings
    df = pd.read_csv("BX-Book-Ratings.csv")
    result = df[df["uid"] == user_id][["uid", "isbn", "rating"]]

    user_ratings = {}  # Create dict isbn : user-rating
    for _, row in result.iterrows():
        user_ratings[row["isbn"]] = float(row["rating"])

    return user_ratings
```

2) Επαναξιολόγηση σειράς παρουσίασης αποτελεσμάτων βάση των κριτικών. Η μετρική με την οποία συνδυάζονται η βαθμολογία του χρήστη και της Elasticsearch είναι ο σταθμισμένος μέσος όρος. Με αυτή την τεχνική συνδυάζουμε βαθμολογίες που πιθανόν να είναι σε διαφορετικές κλίμακες καθώς και μπορούμε να δώσουμε περισσότερη βαρύτητα σε κάποια βαθμολογία( στην δική μας περίπτωση έχουν την ίδια βαρύτητα).

```python
def calibrate_results(elastic_results, user_ratings):
    """Combine user ratings and elastic search response into the final list."""
    # Final result calculated using weighted average.
    for res in elastic_results:
        rating = (
            float(user_ratings[res["isbn"]]) if res["isbn"] in user_ratings else None
        )
        if rating is not None:
            # 0 rated books go to the bottom of the result list.
            if rating == 0:
                res["rank"] = 0
            else:
                res["rank"] = rating * RATING_COEFF + res["rank"] * RANK_COEFF
        else:
            # calculate rank only based on the Elasticsearch result.
            res["rank"] = res["rank"] * RANK_COEFF

    return elastic_results
```

RATING_COEFF = RANK_COEFF = 0.5

- Παρουσίαση αποτελεσμάτων

```python
def print_results(final_results):
    """Prints the final result in a readable format"""
    # Sort the results based on rank if user id was provided.
    if len(argv) == 3:
        final_results = sorted(final_results, key=lambda x: x["rank"], reverse=True)

    # Keep 10% of the resutls
    final_results = final_results[: int(len(final_results) / 10) :]

    print(f"\n{len(final_results)} Results: \n")
    for res in final_results:
        print(
            f' Title: {res["title"]}, Author: {res["author"]}, Year: {res["year_of_publication"]}\n Matching rank: {res["rank"]}
        )
        print("----------------------------------------------------------------------")
```

Παρουσίαση λειτουργίας

- Εκκίνηση Elasticsearch τοπικά

```
elasticsearch.service - Elasticsearch
     Loaded: loaded (/lib/systemd/system/elasticsearch.service; disabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-01-27 20:12:37 EET; 23s ago
       Docs: https://www.elastic.co
   Main PID: 13579 (java)
      Tasks: 90 (limit: 9146)
     Memory: 3.5G
        CPU: 42.409s
```

- Υποβολή ερωτήματος
  example1

```
lek@lek:~/Desktop/IR-2023$ python3 query.py Mythology 2
Successfully connected to Elasticsearch cluster.

            Cluster Name: elasticsearch

            Cluster UUID: Gp3QsXGhSbS33_moKdTtxQ


5 Results:

 Title: Mythology, Author: Edith Hamilton, Year: 1991
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1982
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1982
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Classical mythology, Author: Mark P.O Morford, Year: 1985
 Matching rank: 5.433545
----------------------------------------------------------------------
 Title: Comparative Mythology, Author: Jaan Puhvel, Year: 1989
 Matching rank: 5.433545
----------------------------------------------------------------------
```

example2

```
lek@lek:~/Desktop/IR-2023$ python3 query.py cooking 31299
Successfully connected to Elasticsearch cluster.

            Cluster Name: elasticsearch

            Cluster UUID: Gp3QsXGhSbS33_moKdTtxQ


27 Results:

 Title: Microwave Cooking, Author: Bookthrift, Year: 1991
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Texas Cooking, Author: Lisa Wingate, Year: 2003
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Home Cooking, Author: Laurie Colwin, Year: 1993
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Vegetarian Cooking, Author: Lalita Ahmed, Year: 1993
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Vegetarian Cooking, Author: Sunset Editors, Year: 1981
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Home Cooking, Author: Laurie Colwin, Year: 2000
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Cooking Essentials, Author: Mary Berry, Year: 1997
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Professional Cooking, Author: Wayne Gisslen, Year: 1999
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Home Cooking With Amy Coleman (Pbs Cooking Series), Author: Amy Coleman, Year: 2000
 Matching rank: 3.9905485
----------------------------------------------------------------------
 Title: Mother Nature's Garden: Healthy Vegan Cooking (Vegetarian Cooking), Author: Florence Bienenfeld, Year: 1994
 Matching rank: 3.9905485
----------------------------------------------------------------------
 Title: Mexican Low-Fat Cooking (Cole's Cooking Companion Series), Author: Cole Publishing Group, Year: 1996
 Matching rank: 3.9905485
```

# Ομαδοποίηση Χρηστών σε Συστάδες

Για την υλοποίηση του συγκεκριμένου ερωτήματος χρειάστηκε να γίνουν τα εξής:

## Dataset Cleaning & Preparation

1) **Καθαρισμός BX-Book-Ratings.csv**
   Στο συγκεκριμένο αρχείο δεδομένων παρατηρήθηκε ένας μεγάλος αριθμός κριτικών από UIDs τα οποία δεν ανήκουν στο αρχείο BX-Book-Users.csv επομένως δεν ήταν δυνατή η ομαδοποίηση τους. Αποφασίστηκε αυτές οι εγγραφές να καθαριστούν για τον σκοπό του clustering αντι να δημιουργηθούν με συνθετικά δεδομένα. Το καθαρό σετ κριτικών περιέχεται στο BX-Book-Ratings-clean.csv

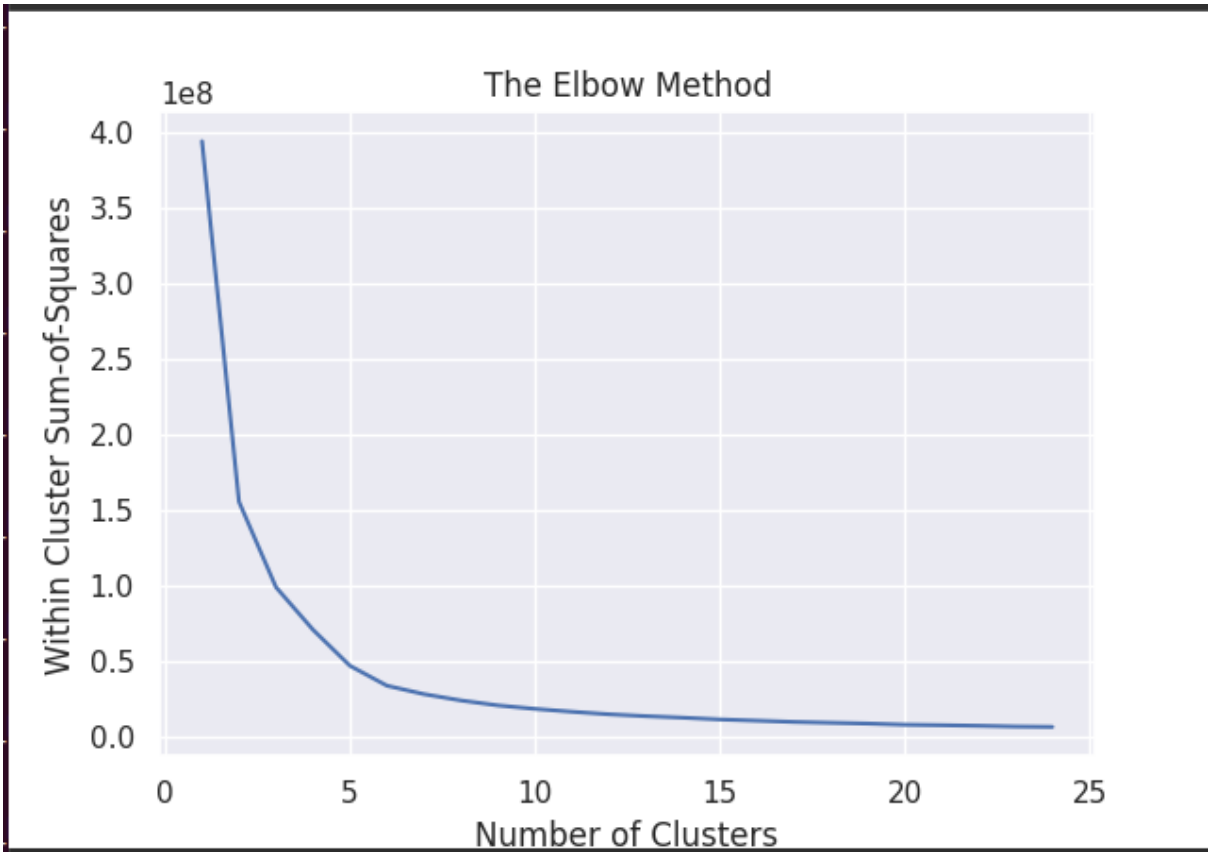
2) **Καθαρισμός και προετοιμασία BX-Book-Users.csv**
   Το συγκεκριμένο αρχείο περιέχει πολλές ελλείψεις και λάθη στα δεδομένα. Για τον σκοπό της ομαδοποίησης των χρηστών βάση χώρας και ηλικίας αρχικά έπρεπε να εξάγουμε την πληροφορία για την χώρα. Η πληροφορία αυτή βρίσκεται στη στήλη location η οποία περιέχει 3 δεδομένα εκ των οποίων το τελευταιο είναι η χώρα. Χρήστες που δεν είχαν δώσει την χώρας τους, θεωρήθηκε ώς κενό πεδίο. .Έτσι στο νέο αρχείο BX-Users-clean.csv Εμπεριέχονται οι στήλες uid,location,age οπου το location αντιπροσωπεύει μόνο την χώρα. Για την εισαγωγή κενών δεδομένων (χώρα/ηλικία) χρησιμοποιήθηκε τυχαία επιλογή από τις τιμές που υπήρχαν ήδη στο dataset με πιθανότητα ίση με αυτής της κατανομής τους, ώστε να έχουμε ένα πιο ρεαλιστικό και δίκαιο αποτέλεσμα.

## Users Clustering (K-means)

1) **Επιλογή πλήθους ομάδων**
   Η επιλογή έγινε με βάση τις μεθόδους Elbow

```python
def elbow(data):
    """
    Using the Elbow Method to Identify
    optimal number of clusters.
    """
    wcss = []
    for i in range(1, 25):
        kmeans = KMeans(i)
        kmeans.fit(data)
        wcss_iter = kmeans.inertia_
        wcss.append(wcss_iter)

    number_of_clusters = range(1, 25)
    plt.plot(number_of_clusters, wcss)
    plt.title("The Elbow Method")
    plt.xlabel("Number of Clusters")
    plt.ylabel("Within Cluster Sum-of-Squares")
    plt.show()
```




Με βάση την συγκεκριμένη μέθοδο αποφασίστηκε πως το κατάλληλο πλήθος ομάδων ισούται με 6.

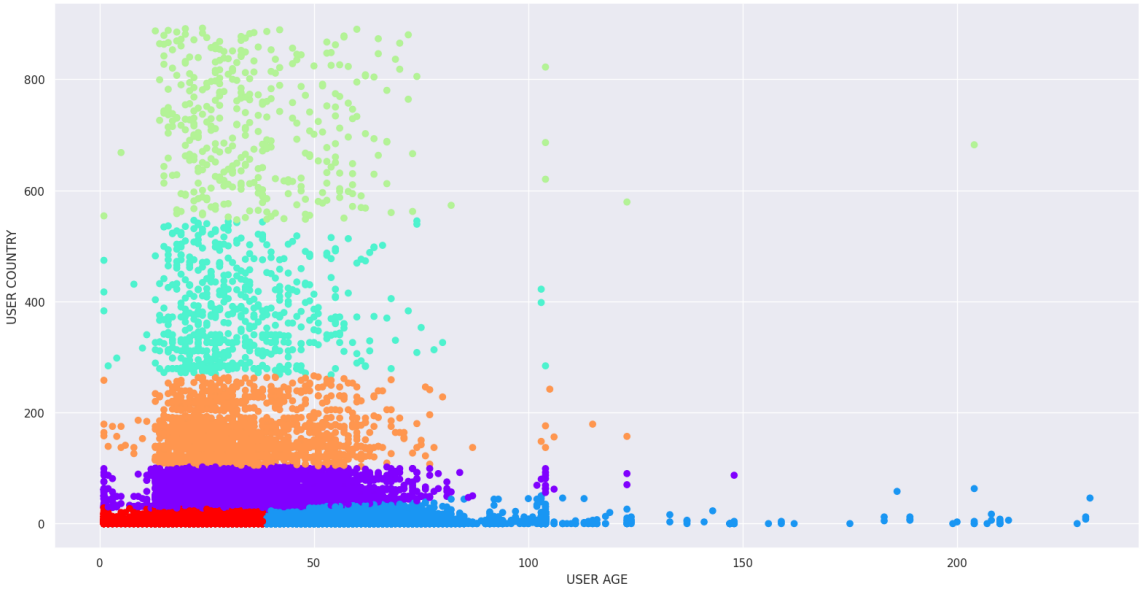
2) **Ομαδοποίηση βάση ηλικίας και χώρας**

Για να γίνει η ομαδοποίηση με βάση την χώρα η οποία είναι μια κατηγορική τιμή έπρεπε να επιλεγεί ο κατάλληλος τρόπος για να γίνει η απαραίτητη κωδικοποίηση σε μια αριθμητική τιμή. Για να είναι εύστοχη η κωδικοποίηση θα μπορούσαμε να λάβουμε υπόψη την γεωγραφική τοποθεσία ή ήπειρο της κάθε χώρας. Ωστόσο επειδή το dataset περιέχει ιδιάζουσες τιμές (π.χ. u. s. of a αντι για usa, fairyland κ.α.), αποφασίστηκε να γίνει η απλούστερη κωδικοποίηση δίνοντας στην κάθε χώρα μια μοναδική θετική αριθμητική τιμή. Στην συνέχεια πραγματοποιήθηκε το clustering.

```
def clustering(filename):
    """Perform Clustering"""
    df = loadData(filename)

    kmeans = KMeans(CLUSTERS, n_init=10)
    id_clusters = kmeans.fit_predict(df)
    data_with_clusters = df.copy()
    data_with_clusters["cluster"] = id_clusters

    showClusters(data_with_clusters)
```



3) **Ενημέρωση κριτικών βάση cluster.**

```python
# Update new user ratings from the cluster's average
for i,row in df_ratings.iterrows():
    current_uid = row["uid"]
    current_rating = row["rating"]
    current_isbn = row["isbn"]
    user_cluster = int(df_users[df_users["uid"] == current_uid].head(1)['cluster'])
    print(f"User ID: {current_uid} , Cluster: {user_cluster}")
    if current_rating == 0:
        # Access book ratings of the user's cluster
        if current_isbn in cluster_ratings[user_cluster].keys():
            df_ratings.at[i, 'rating'] = int(cluster_ratings[user_cluster][current_isbn])

df_ratings.to_csv('BX-Book-Ratings-Clustered.csv', index=False)
```

Με την ενημέρωση των βαθμολογιών ανα cluster, μειώθηκε στο 50% ο αριθμός των κριτικών χωρίς βαθμολογία

4) **Υποβολή ερωτήματος στην elasticsearch με ενημερωμένο αρχείο κριτικών.**

## Τελικά Αποτελέσματα

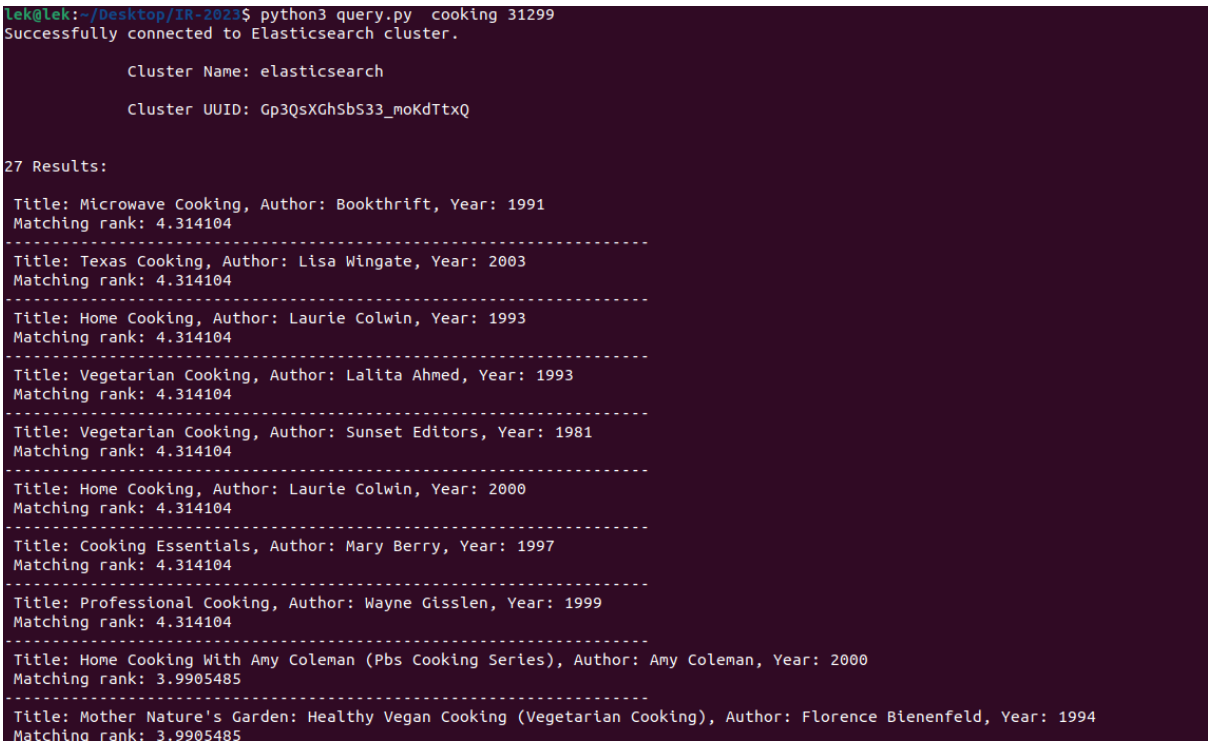
```
lek@lek:~/Desktop/IR-2023$ python3 query.py  Mythology 2
Successfully connected to Elasticsearch cluster.

            Cluster Name: elasticsearch

            Cluster UUID: Gp3QsXGhSbS33_moKdTtxQ


5 Results:

 Title: Mythology, Author: Edith Hamilton, Year: 1991
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1982
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1982
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Classical mythology, Author: Mark P.O Morford, Year: 1985
 Matching rank: 5.433545
----------------------------------------------------------------------
 Title: Comparative Mythology, Author: Jaan Puhvel, Year: 1989
 Matching rank: 5.433545
----------------------------------------------------------------------
```




# Εκπαίδευση Νευρωνικού Δικτύου στις βαθμολογίες ανα Cluster

## Dataset Cleaning & Preparation

1) **Καθαρισμός summaries των βιβλίων και δημιουργία vocabulary.**
   Για να γίνει σωστά η χρήση της τεχνικής word embedding πρέπει να δημιουργήσουμε ένα λεξιλόγιο το οποίο θα αποτελείται από κάθε λέξη που περιλαμβάνεται στα summaries όλων των βιβλίων. Αρχικά με χρήση **regex** αφαιρούμε ότι δεν αναγνωρίζεται ώς λέξεις και στην συνέχεια με την βιβλιοθήκη **nltk** αφαιρούμε τα **stopwords**. Έτσι τελικά προκύπτουν summaries τα οποία μπορούν να τεθούν ως είσοδος στο **embedding layer** του νευρωνικού δικτύου, το ενημερωμένο dataset βρίσκεται στο αρχείο **BX-Books-clean.csv** . Τέλος δημιουργείται το vocabulary.dat που περιέχει όλες τις λέξεις οι οποίες εμφανίζονται στα summaries των βιβλίων.

## Word Embedding

1) **Επιλογή τεχνικής Word Embedding**
   Embedding Layer : Απαιτεί πολλά δεδομένα εκπαίδευσης και είναι αργό αλλά είναι καλύτερα προσαρμοσμένο στο dataset στο οποίο θέλουμε να δουλέψουμε. Φτιάχτηκε vocabulary από όλα τα summaries και καθε κάθε

λέξη έγινε encoded ως vector 10000 θέσεων για καλύτερα δυνατά αποτελέσματα.

## Νευρωνικο Δίκτυο

1) **Τοπολογία νευρωνικού**
   Επιλέχθηκε η προσέγγιση ως multiple classification problem
   Τα ratings μετατρέπονται σε ανάλογο one hot vector
   π.χ 6 = [0 0 0 0 0 1 0 0 0 0 ]
   1 Embedding Layer
   2 Flatten Layer
   1 Output layer 10 νευρώνες με softmax activation function

```
# define the model
model = Sequential()
model.add(Embedding(INPUT_DIM, OUTPUT_DIM, input_length=INPUT_LENGTH, name="embeddings"))
model.add(Flatten())
model.add(Dense(10, activation='softmax'))
# compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
# summarize the model
print(model.summary())
# fit the model
model.fit(padded_summaries, labels, epochs=15, verbose=1)

df_pending = pd.read_csv('pending_reviews.csv')
# Get missing ratings for cluster
df_pending = df_pending[df_pending['cluster'] == i]
```

2) **Εύρεση βαθμολογιών**

```
    # Predict missing ratings
    pending_summaries = []
    uids = []
    isbns = []
    ratings = []
    for i,row in df_data.iterrows():
        uids.append(row['uid'])
        isbns.append(row['isbn'])
        pending_summaries.append(row['summaries'])

    encoded_summaries = [one_hot(d, INPUT_DIM) for d in pending_summaries]
    padded_summaries = pad_sequences(encoded_summaries, maxlen=INPUT_LENGTH, padding='post')

    for s in padded_summaries:
        prediction = model.predict(s)
        rating = np.argmax(prediction) + 1
        ratings.append(rating)

    final_uids.extend(uids)
    final_isbns.extend(isbns)
    final_ratings.extend(ratings)

df_final = pd.DataFrame({'uid' : final_uids, 'isbn' : final_isbns, 'rating' : final_ratings})
df_final.to_csv('temp.csv', index=False)
```

Όλες οι βαθμολογίες πλέον είναι πλήρεις.

# Τελικά Αποτελέσματα

```
lek@lek:~/Desktop/IR-2023$ python3 query.py  Mythology 2
Successfully connected to Elasticsearch cluster.

            Cluster Name: elasticsearch

            Cluster UUID: Gp3QsXGhSbS33_moKdTtxQ


5 Results:

 Title: Classical Mythology, Author: Mark P. O. Morford, Year: 2002
 Matching rank: 7.933545
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1991
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1982
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Mythology, Author: Edith Hamilton, Year: 1982
 Matching rank: 5.958295
----------------------------------------------------------------------
 Title: Classical mythology, Author: Mark P.O Morford, Year: 1985
 Matching rank: 5.433545
----------------------------------------------------------------------
```

```
lek@lek:~/Desktop/IR-2023$ python3 query.py  cooking 31299
Successfully connected to Elasticsearch cluster.

            Cluster Name: elasticsearch

            Cluster UUID: Gp3QsXGhSbS33_moKdTtxQ


27 Results:

 Title: Microwave Cooking, Author: Bookthrift, Year: 1991
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Texas Cooking, Author: Lisa Wingate, Year: 2003
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Home Cooking, Author: Laurie Colwin, Year: 1993
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Vegetarian Cooking, Author: Lalita Ahmed, Year: 1993
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Vegetarian Cooking, Author: Sunset Editors, Year: 1981
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Home Cooking, Author: Laurie Colwin, Year: 2000
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Cooking Essentials, Author: Mary Berry, Year: 1997
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Professional Cooking, Author: Wayne Gisslen, Year: 1999
 Matching rank: 4.314104
----------------------------------------------------------------------
 Title: Home Cooking With Amy Coleman (Pbs Cooking Series), Author: Amy Coleman, Year: 2000
 Matching rank: 3.9905485
----------------------------------------------------------------------
 Title: Mother Nature's Garden: Healthy Vegan Cooking (Vegetarian Cooking), Author: Florence Bienenfeld, Year: 1994
 Matching rank: 3.9905485
----------------------------------------------------------------------
 Title: Mexican Low-Fat Cooking (Cole's Cooking Companion Series), Author: Cole Publishing Group, Year: 1996
 Matching rank: 3.9905485
----------------------------------------------------------------------
 Title: Joy of Cooking, Author: Irma S. Rombauer, Year: 1997
 Matching rank: 3.9649122
----------------------------------------------------------------------
```

# Καταγραφή περιβάλλοντος υλοποίησης.

Λειτουργικό Σύστημα : Ubuntu Linux
Περιβάλλον εγγραφής κώδικα : VSCode
Βιβλιοθήκες Python:
Elasticsearch
pandas
matplotlib
seaborn
sklearn
numpy
keras
Tensorflow
Η εγκατάσταση τους γίνεται με την εντολή στο τερματικό pip install <library_name>