

Υλοποιητική εξαμηνιαία εργασία

# Ανάκτηση πληροφορίας



ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΠΑΤΡΩΝ  
UNIVERSITY OF PATRAS

**Κόλλιας Ιωάννης**

**1064886**

**st1064886@ceid.upatras.gr**

**2022-2023**

## Πίνακας Περιεχομένων

Περιγραφή Περιβάλλοντος Υλοποίησης.....	3
Περιγραφή Διαδικασίας Υλοποίησης.....	4
Ερώτημα Πρώτο.....	4
getDataS.py.....	4
getDataID.py.....	5
getDataID2.py.....	6
Ερώτημα Δεύτερο.....	7
fetchCluster.py.....	7
optimalK.py.....	7
dataCluster.py.....	8
Ερώτημα Τρίτο.....	9
teachUser.py.....	9
teachData.py.....	10
Περιγραφή Γραφικού Περιβάλλοντος.....	12
Περιγραφή Αποτελεσμάτων Υλοποίησης.....	13
Ερώτημα Πρώτο.....	13
Ερώτημα Δεύτερο.....	14
Ερώτημα Τρίτο.....	16

## Περιγραφή Περιβάλλοντος Υλοποίησης

Η υλοποίηση της παρούσας εργασίας έγινε στο περιβάλλον του Microsoft Visual Studio Code σε linux λειτουργικό (Pop!\_OS 22.04 LTS) ενώ η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η Python 3.10.6 64bit. Για την εκπόνηση των ερωτημάτων ήταν η απαραίτητη η χρήση των ακόλουθων βιβλιοθηκών:

```
from elasticsearch import Elasticsearch
from elasticsearch import helpers
import csv
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import MaxAbsScaler
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn import svm
from sklearn.cluster import KMeans
from matplotlib import pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import normalize, MaxAbsScaler
from kneed import KneeLocator
import numpy as np
import seaborn as sns
```

Για την χρησιμοποίηση αυτών ήταν απαραίτητη η εγκατάσταση στο λειτουργικό, με τις ακόλουθες εντολές στο terminal:

- pip install elasticsearch
- pip install scikit-learn
- pip install pandas
- pip install numpy
- pip install seaborn
- pip install kneed

Στην παρούσα εργασία έγινε και μια προσπάθεια για Graphical User Interface (GUI) για την οποία χρειάστηκαν οι βιβλιοθήκες:

```
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
from tabulate import tabulate
```

όπου για την αξιοποίηση αυτών χρειάστηκαν οι εντολές:

- pip install tkinter
- pip install tabulate

## Περιγραφή Διαδικασίας Υλοποίησης

Πριν ξεκινήσει η επεξήγηση των λύσεων των ερωτημάτων κρίθηκε απαραίτητο να υλοποιηθεί ένα στοιχειώδες menu μέσω του terminal προκειμένου να γίνει πιο εύχρηστη η εφαρμογή. Να σημειωθεί πως σε κάθε ερώτημα, όπου αυτό κρίνεται απαραίτητο, θα γίνεται αναφορά στις εισόδους που δέχεται η εκάστοτε λειτουργία.

### Ερώτημα Πρώτο

Αρχικά γίνεται η εγκατάσταση της elasticsearch στο λειτουργικό σύστημα αλλά και του kibana προκειμένου την καλύτερη επαφή με την κατανεμημένη μηχανή αναζήτησης και ανάλυσης.

Βασική προϋπόθεση είναι η σύνταξη συνάρτησης που θα επικοινωνεί το εκάστοτε πρόγραμμα της python με τη μηχανή αναζήτησης (*conn.py*). Έπειτα, χρησιμοποιώντας τη βιβλιοθήκη helpers της elasticsearch γίνεται το parse του εκάστοτε csv αρχείου έτσι ώστε να ανέβει στο περιβάλλον της elasticsearch με τον ορισμό συγκεκριμένου index (*upload.py*). Να σημειωθεί πως για την επιβεβαίωση του ανεβάσματος του csv αρχείου υλοποιήθηκε και ένα script που επιστρέφει το πλήθος των τιμών με βάση το index name που ορίστηκε στη διαδικασία του upload (*checkIndex.py*). Η περίπτωση του upload ενός csv αρχείου αλλά και η περίπτωση του check μέσω indexname, γίνεται εύκολα από την *main.py* πληκτρολογώντας το επιθυμητό choice.

```
Menu:
1. Upload a File to ElasticSearch
2. Check via index
3. Retrieve Data from ElasticSearch
4. Data Cluster
5. Quit
Enter your choice: []
```

### getDataS.py

Πρωτού εξηγηθεί το ερώτημα της εύρεσης βιβλίων με βάση αλφαριθμητικού-keyword, αριθμού-user\_id για search είναι αναγκαία η επεξήγηση της “απλής” εύρεσης δηλαδή της συνάρτησης *getdata* λόγω του ότι είναι απαραίτητη για την επίλυση του ερωτήματος. Στο menu πλοήγησης του χρήστη ζητείται ένα keyword και έπειτα ζητά την 1<sup>η</sup> επιλογή που είναι αυτή της απλής αναζήτησης.

```
match_query = {
    "match": {
        "book_title": keyword
    }
}
result = cn.search(index=indexname, query=match_query)
```

Ο παραπάνω κώδικας ορίζει ένα ερώτημα αντιστοίχισης που αναζητά την παρεχόμενη λέξη-κλειδί στο πεδίο “book\_title” του ευρετηρίου elasticsearch. Στη συνέχεια εκτελεί την αναζήτηση χρησιμοποιώντας το καθορισμένο ερώτημα και αποθηκεύει το αποτέλεσμα.

```
books = []
score = []
for hit in result['hits']['hits']:
    books.append(hit['_source'])
    score.append(hit['_score'])
```

Έπειτα για κάθε “χτύπημα” – αποτέλεσμα από τον πίνακα `result` που τέθηκε, εξάγει τα αποτελέσματα πηγής (στο πεδίο `_source`). Επίσης ανακτά το `score` (`_score`) για κάθε εύρημα. Τα εξαγόμενα αποτελέσματα αποθηκεύονται σε νέες λίστες που ορίστηκαν (`books`, `score`) και αυτό για να καθαρίσουμε το `response` της `elasticsearch` από μεταδεδομένα.

Τέλος δημιουργείται ένα `dataframe` από τα εξαγόμενα `source data` και προστίθεται μία στήλη με την ονομασία `score` στο `dataframe`.

## getDataID.py

Σε αυτό το `script` υλοποιείται ο κώδικας για την απάντηση στο 1<sup>ο</sup> ερώτημα της εργασίας. Στο `menu` πλέον μετά την καταχώρηση του `keyword` επιλέγεται η 2<sup>η</sup> επιλογή που είναι η ανάκτηση αποτελεσμάτων με βάση ΚΑΙ το `id` του χρήστη, όπου εισάγεται και αυτό (υπάρχει και το βήμα της ενεργοποίησης νευρωνικού δικτύου αλλά προς το παρόν το αφήνουμε απενεργοποιημένο).

Πρώτα, δημιουργείται η συνάρτηση `getdataisbnUid` η οποία παίρνει τα δεδομένα που εισήγαγε ο χρήστης και αφετέρου φιλοξενεί την `isbnRatingAvg` η οποία επιστρέφει το μέσο όρο βαθμολογιών του βιβλίου. Ξεκινώντας από την αμφωλευμένη συνάρτηση, ένα γενικό σχόλιο για αυτήν θα ήταν ο υπολογισμός του μέσου όρου ενός βιβλίου με βάση το αναγνωριστικό του βιβλίου και το αναγνωριστικό του χρήστη.

```
query = {
  "bool": {
    "must": [
      {
        "match": {
          "isbn": book_code
        }
      },
      {
        "match": {
          "uid": userId
        }
      }
    ]
  }
}
```

Ο παραπάνω κώδικας ουσιαστικά πετυχαίνει το ταίριασμα μεταξύ κωδικού βιβλίου και χρήστη, με αποτέλεσμα να προωθεί το βιβλίο αν ο χρήστης το έχει διαβάσει (μιας και έχει βαθμολογία).

Έπειτα εκτελείται εύρεση χρησιμοποιώντας την `search` συνάρτηση της `elasticsearch` όπως και προηγουμένως, με την ιδιαιτερότητα πως στην περίπτωση που δεν υπάρχουν επιτυχίες (ταιριάσματα) στα αποτελέσματα αναζήτησης (δλδ δεν υπάρχουν βαθμολογίες για το βιβλίο), η συνάρτηση επιστρέφει 0 ως μέση βαθμολογία και βαθμολογία χρήστη. Αυτό γιατί στην περίπτωση που δεν υπολογιζόταν αυτή η πιθανότητα προέκυπτε σφάλμα υπερχείλησης.

Ορίζεται άθροισμα αξιολογήσεων (`sum`) και μετρητής του πλήθους των αξιολογήσεων (`counter`). Ανακτά τη βαθμολογία του χρήστη από το πρώτο (`[0]`) “χτύπημα” στα αποτελέσματα αναζήτησης που έγινε παραπάνω και την εκχωρεί στη μεταβλητή `userRating`. Στη συνέχεια, επαναλαμβάνει τις επιτυχίες στα αποτελέσματα αναζήτησης και ελέγχει αν κάθε βαθμολογία είναι μεγαλύτερη από 0. Αν ναι, προσθέτει την αξιολόγηση στο άθροισμα `sum` και αυξάνει τον μετρητή.

Τέλος, με μία if ελέγχεται το ενδεχόμενο για μετρητή ίσο με το μηδέν ώστε να μην διαιρεθεί ο μέσος όρος με μηδέν.

Αφού εξετάστηκε με επιτυχία ο μέσος όρος των αξιολογήσεων, η ροή μεταφέρεται στην εξωτερική συνάρτηση *getdataisbnUid*, στην οποία γίνεται σύνδεση με την *elasticsearch* και ανακτώνται τα βιβλία μέσω της συνάρτησης *getdata* της προηγούμενη υποενότητα (*getDataS.py*)· προκειμένου να φέρουμε όλα τα βιβλία με το συγκεκριμένο keyword. Καλείται η προηγούμενη συνάρτηση *isbnRatingAvg* για το isbn του βιβλίου, συμπληρώνονται οι λίστες των μέσο όρων και των ratings και ενημερώνεται η συμπληρωμένη λίστα.

```
score = []
for i in range(len(books)):
    score += [(0.65 * books.loc[i, 'score']) + (0.25 * books.loc[i, 'rating']) + (0.1 * avgs[i])]
```

Στον παραπάνω κώδικα επαναλαμβάνεται κάθε βιβλίο στο πλαίσιο δεδομένων *books* DataFrame και υπολογίζει μια βαθμολογία με βάση έναν σταθμισμένο συνδιασμό της υπάρχουσας βαθμολογίας του βιβλίου (*books.loc[i, 'score']*), της βαθμολογίας του χρήστη (*books.loc[i, 'rating']*) και της μέσης βαθμολογίας (*avgs[i]*). Τα υπολογισμένα αποτελέσματα αποθηκεύονται στη λίστα αποτελεσμάτων.

```
return books.sort_values(by=['score'], ascending=False)
```

## getDataID2.py

Στο αρχείο αυτό έγινε προσπάθεια για την ίδια παραπάνω υλοποίηση (retrieve data by id) με τη διαφορά ότι έγινε με τη χρήση aggregation.

```
query = {
    "query": {
        "match": {
            "book_title": {
                "query": keyword,
                "operator": "and"
            }
        }
    },
    "aggs": {
        "avg_rating": {
            "avg": {
                "field": "rating",
            }
        }
    }
}

# Collecting all the data using Scan function.
rel = scan(
    client=cn,
    query=query,
    scroll='1m',
    index='merged',
    raise_on_error=True,
    preserve_order=False,
    clear_scroll=True
)
```

## Ερώτημα Δεύτερο

Για την απάντηση αυτού του ερωτήματος χρησιμοποιήσαμε τα τρία παρακάτω scripts *fetchCluster.py*, *optimalK.py*, *dataCluster.py*.

### fetchCluster.py

Η συνάρτηση του αρχείου, ανακτά δεδομένα από την Elasticsearch με βάση ένα προκαθορισμένο μέγεθος.

Αφού γίνει η σύνδεση με την Elasticsearch, ορίζεται το query ταιριάσματος (*match\_all*) για την ανάκτηση όλων των εγγράφων από το ευρετήριο “books”. Με το πέρας της εκτέλεσης αποθηκεύονται στην μεταβλητή *result*.

Οι λίστες, *summary* και *isbns* δημιουργούνται για την αποθήκευση των περιλήψεων αλλά και του *id* του βιβλίου. Η περίληψη και το ISBN κάθε βιβλίου εξάγονται από τα αποτελέσματα της αναζήτησης και προσαρτώνται στους αντίστοιχους καταλόγους.

Οι ρυθμίσεις του aggregation (*aggr*) ορίζονται για την ανάκτηση των κορυφαίων *hits* (χρηστών) για κάθε ISBN. Ο μέγιστος αριθμός *hits* ορίζεται μέχρι τα 100 και τα πεδία που συμπεριλαμβάνονται στην πηγή καθορίζονται ως [*'uid'*, *'rating'*].

```
aggr = {  
    "aggs": {  
        "top_hits": {  
            "size": 100,  
            "_source": ['uid', 'rating']  
        }  
    }  
}
```

Ο κώδικας επαναλαμβάνεται για κάθε *ISBN* και εκτελεί ένα ερώτημα αντιστοίχισης για να ανακτήσει τους *users* που βαθμολόγησαν το βιβλίο. Οι ανακληθέντες χρήστες και οι αξιολογήσεις αυτών αποθηκεύονται σε μια προσωρινή λίστα (*temp\_list*). Εάν ο χρήστης υπάρχει (στα αποτελέσματα αναζήτησης), η τοποθεσία, η ηλικία και η βαθμολογία του εξάγονται και προστίθενται στην *temp\_list*. Μετά την επανάληψη όλων των χρηστών για το τρέχον *ISBN*, η *temp\_list* προσαρτάται στην κύρια λίστα χρηστών.

```
for hit in res['aggregations']['aggs']['hits']['hits']:  
    match_user = {"match": {"uid": hit['_source']['uid']}}  
    userRes = cn.search(index='users', query=match_user, size=1)  
    if userRes['hits']['hits']:  
        temp = userRes['hits']['hits'][0]['_source']  
        temp_list.append((temp['location'], temp['age'], hit['_source']['rating']))  
    users.append(temp_list)
```

Τέλος, δημιουργείται ένα *DataFrame* για την αποθήκευση των ανακτηθέντων δεδομένων, με στήλες για τη περίληψη και τους χρήστες και επιστρέφεται ως αποτέλεσμα συνάρτησης (*return*).

```
return pd.DataFrame({"summary": [s for s in summary], "users": [user for user in users]},  
                    columns=['summary', 'users'])
```

### optimalK.py

Μέσω της συνάρτησης *find\_opt\_k* του αρχείου και καθορίζει τον βέλτιστο αριθμό συστάδων (*K*) χρησιμοποιώντας τη μέθοδο *Elbow*. Η συνάρτηση λαμβάνει μια παράμετρο *x*, η οποία αντιπροσωπεύει τα δεδομένα για την ομαδοποίηση, ένα εύρος τιμών από 1 έως 14 ορίζεται με τη χρήση του *range(1,15)*.

Αρχικοποιείται μια κενή λίστα για το άθροισμα τετραγωνικών σφαλμάτων (`Sum_Squared_Errors`) για την αποθήκευση των σφαλμάτων για κάθε τιμή του  $K$ .

Αφού γίνει η σύνδεση με την `Elasticsearch`, ορίζεται το `query` ταιριάσματος (`match_all`) για την ανάκτηση όλων των εγγράφων από το ευρετήριο “books”. Με το πέρας της εκτέλεσης αποθηκεύονται στην μεταβλητή `result`.

Στην δομή ελέγχου, Δημιουργείται ένα μοντέλο *KMeans* με την τρέχουσα τιμή  $K$  χρησιμοποιώντας το *KMeans* (`n_clusters=k`, `n_init=10`). Προσαρμόζει το μοντέλο στα δεδομένα  $x$  χρησιμοποιώντας το `km.fit(x)`. Υπολογίζει το `Sum_Squared_Errors` (αδράνεια) του αποτελέσματος της ομαδοποίησης χρησιμοποιώντας το `km.inertia_` και το προσθέτει στη λίστα του `Sum_Squared_Errors`.

Ο χρήστης μέσω της μεταβλητής `choice` καλείται αν θα σχεδιάσει το *Elbow* γράφημα. Εάν τεθεί θετικά ο κώδικας σχεδιάζει το γράφημα μέσω της `plt.plot(rng, Sum_Squared_Errors)` και εμφανίζεται.

Το *KneeLocator* από τη βιβλιοθήκη *kneed* χρησιμοποιείται για την εύρεση της βέλτιστης τιμής  $K$  (σημείο *Elbow*) με βάση τις τιμές `Sum_Squared_Errors`. Η βέλτιστη τιμή  $K$  επιστρέφεται με τη χρήση του `kneel.elbow`.

## dataCluster.py

Η συνάρτηση `clusterData` εκτελεί ομαδοποίηση και παράγει απεικονίσεις, λαμβάνοντας ως παραμέτρους `metric` και `userId`.

Η εμφολευμένη συνάρτηση `clr` ορίζεται για την επεξεργασία των δεδομένων για κάθε συστάδα. Η συνάρτηση `fetch_cluster` καλείται για να ανακτήσει τις περιλήψεις βιβλίων και να τις αποθηκεύσει στο πλαίσιο δεδομένων `DataFrame`.

Οι περιλήψεις μετατρέπονται σε διανύσματα με τη χρήση του *CountVectorizer* και αποθηκεύονται στον πίνακα  $A$ . Η μείωση της διαστατικότητας πραγματοποιείται στα διανύσματα με τη χρήση *PCA* (Ανάλυση Κύριων Συνιστωσών), με αποτέλεσμα το `aSvd`.

Η βέλτιστη τιμή  $K$  προσδιορίζεται με τη συνάρτηση `find_opt_k`. Ανάλογα με την μετρική παράμετρο, η ομαδοποίηση πραγματοποιείται είτε με ομοιότητα συνημιτόνου είτε με ευκλείδεια απόσταση με τη χρήση του *Kmeans*.

Οι ετικέτες των συστάδων αντιστοιχίζονται στο πλαίσιο δεδομένων `df`. Λαμβάνονται μοναδικές ετικέτες συστάδων και χρησιμοποιούνται για την απεικόνιση των αποτελεσμάτων. Κάθε συστάδα απεικονίζεται ως διάγραμμα διασποράς, με σημειωμένα τα κεντροειδή.

Οι χάρτες θερμότητας δημιουργούνται για κάθε συστάδα χρησιμοποιώντας τη συνάρτηση `clr` για την επεξεργασία των δεδομένων και τη συνάρτηση `heatmap` του *seaborn* για τη δημιουργία της απεικόνισης.

### Σημείωση:

`/home/jsaillok/.local/lib/python3.10/site-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning`

```
warnings.warn(
```

Λόγω του συγκεκριμένου warning προστέθηκε η αρχικοποίηση της τιμής *Kmeans*(`n_init=10`).



## Ερώτημα Τρίτο

### teachUser.py

Στο αρχείο αυτό με τον ορισμό της συνάρτησης *teach\_user(userId)* ανακτά δεδομένα από την Elasticsearch που αφορούν τον χρήστη από ένα ευρετήριο και τα μετατρέπει σε ένα πλαίσιο δεδομένων pandas DataFrame.

Εκτελεί query στη στο ευρετήριο με όνομα “bratings” για να ανακτήσει δεδομένα αξιολόγησης χρηστών με βάση το userId. Το ερώτημα αναζήτησης ταιριάζει το πεδίο uid με το userId και η παράμετρος size περιορίζει τον αριθμό των αποτελεσμάτων σε 10.000.

```
rs = cn.search(index='bratings', query={"match": {"uid": userId}}, size=10000)
```

Ο κώδικας επαναλαμβάνει τα αποτελέσματα της αναζήτησης και εξάγει τις τιμές isbn και rating από κάθε εύρημα. Δημιουργεί μια πλειάδα (isbn, rating) και την προσθέτει στη λίστα του πίνακα. Αυτό το βήμα φιλτράρει τις αξιολογήσεις με τιμή “0”.

```
array = []
# get all the books(isbn) for a user into a list
for hit in rs['hits']['hits']:
    # keep rated by user books
    userRating = int(hit['_source']['rating'])
    if userRating:
        isbn_rating = (hit['_source']['isbn'], userRating)
        array.append(isbn_rating)
```

Με δομή επανάληψης επαναλαμβάνεται ο πίνακας των πλειάδων και για κάθε πλειάδα εκτελεί ένα άλλο ερώτημα αναζήτησης με όνομα “books” για την ανάκτηση πληροφοριών του βιβλίου με βάση την τιμή isbn.

Ελέγχει αν τα αποτελέσματα της αναζήτησης δείχνουν ότι το βιβλίο υπάρχει (rs['hits']['total']['value'] μεγαλύτερο του 0). Εάν το βιβλίο υπάρχει, εξάγει τις τιμές book\_title, summary και isbn από κάθε εύρημα και τις προσθέτει στις αντίστοιχες λίστες (book\_titles, summaries, book\_isbn). Προσθέτει επίσης την αντίστοιχη βαθμολογία από τον πίνακα στη λίστα βαθμολογιών.

```
for t in array:
    rs = cn.search(index='books', query={"match": {"isbn": t[0]}})
    # if specific book exists in index 'books'
    if rs['hits']['total']['value']:
        # get the rating
        ratings.append(int(t[1]))
        for hit in rs['hits']['hits']:
            book_titles += [hit['_source']['book_title']]
            summaries += [hit['_source']['summary']]
            book_isbn += [hit['_source']['isbn']]
```

Τέλος, δημιουργείται ένα λεξικό d με τις συλλεγμένες λίστες δεδομένων (book\_titles, ratings, summaries, book\_isbn) ως τιμές και τα αντίστοιχα ονόματα στηλών (isbn, book\_title, summary, ratings) ως κλειδιά. Χρησιμοποιείται το λεξικό για τη δημιουργία πλαισίου δεδομένων pandas DataFrame και το επιστρέφει.

```
d = {'isbn': book_isbn, 'book title': book_titles, 'summary': summaries, 'rating': ratings}
```

## teachData.py

Μέσω της συνάρτησης `teach_data(userId, zero_books)` του ορισμένου αρχείου εκπαιδεύεται ένα μοντέλο μηχανικής μάθησης χρησιμοποιώντας δεδομένα που αφορούν συγκεκριμένους χρήστες και προβλέπει βαθμολογίες για ένα σύνολο βιβλίων με μηδενική βαθμολογία. Το αποτέλεσμα της κλήσης ανατίθεται στη μεταβλητή `user_nz_data` (περιέχονται δεδομένα βιβλίων για κάθε χρήστη που αποκλείουν τα βιβλία με βαθμολογία 0).

Έπειτα ελέγχεται αν ο αριθμός των ειδικών για τον χρήστη βιβλίων είναι μικρότερος ή ίσος με 1. Εάν ναι, τότε εμφανίζεται μια εξαίρεση που υποδεικνύει ότι τα δεδομένα εκπαίδευσης του χρήστη δεν είναι επαρκή.

```
if len(user_nz_data) <= 1:  
    raise Exception("User's train data are not enough")
```

Συνδέονται οι στήλες `book_title` και `summary` του `user_nz_data` σε μια νέα στήλη με όνομα `title_summary`.

```
user_nz_data['title_summary'] = user_nz_data['book_title'] + ' ' + user_nz_data['summary']
```

Διαχωρίζονται τα δεδομένα που αφορούν συγκεκριμένο χρήστη σε σύνολα εκπαίδευσης και δοκιμής χρησιμοποιώντας τη συνάρτηση `train_test_split`. Επιλέγονται οι στήλες `title_summary` και `rating` για διαχωρισμό, με το 70% των δεδομένων να χρησιμοποιείται για εκπαίδευση (training) και το 30% για δοκιμή (testing). Η παράμετρος `random_state` ορίζεται σε 42 λόγους αναπαραγωγικότητας. Εξάγονται οι τιμές `title_summary` και `rating` από τα σύνολα εκπαίδευσης και δοκιμής και τις αποθηκεύει σε ξεχωριστές λίστες.

```
train, test = train_test_split(user_nz_data.loc[:, ['title_summary', 'rating']], test_size=0.3,  
                               random_state=42)
```

Για τη μετατροπή των δεδομένων κειμένου στα `train_x` και `test_x` σε αριθμητικά διανύσματα χαρακτηριστικών δημιουργείται αντικείμενο `CountVectorizer`. Προσαρμόζεται ο διανυσματοποιητής στα δεδομένα εκπαίδευσης χρησιμοποιώντας την `fit_transform` και μετασχηματίζει τα δεδομένα δοκιμής χρησιμοποιώντας την `transform`.

```
vectorizer = CountVectorizer()  
train_x_vectors = vectorizer.fit_transform(train_x)  
test_x_vectors = vectorizer.transform(test_x)
```

Έπειτα, για την κλιμάκωση των διανυσμάτων χαρακτηριστικών δημιουργείται ένα αντικείμενο `MaxAbsScaler`. Προσαρμόζεται ο `scaler` στα δεδομένα εκπαίδευσης με τη χρήση `fit_transform` και μετασχηματίζεται στα δεδομένα δοκιμής με τη χρήση `transform`.

```
scaler = MaxAbsScaler()  
train_x_vectors = scaler.fit_transform(train_x_vectors)  
test_x_vectors = scaler.transform(test_x_vectors)
```

Αναφορικά με την ταξινόμηση, αρχικοποιούνται τέσσερις ταξινομητές (`RandomForestClassifier`, `LogisticRegression`, `DecisionTreeClassifier`, `SVC`) και εκπαιδεύονται χρησιμοποιώντας κλιμακωτά δεδομένα εκπαίδευσης (`train_x_vectors`, `train_y`).

```
clf_rfc = RandomForestClassifier()  
clf_log = LogisticRegression()  
clf_dec = DecisionTreeClassifier()  
clf_svm = svm.SVC(kernel='linear', C=4)  
clf_rfc.fit(train_x_vectors, train_y)
```

Τέλος, εκτυπώνονται οι μέσες ακρίβειες των εκπαιδευμένων ταξινομημένων στα δεδομένα δοκιμής. Συνδέονται οι στήλες `book_title` και `summary` της `zero_books` σε μια νέα λίστα `zero_books_x`. Μετασχηματίζονται τα `zero_books_x` μέσω του προσαρμοσμένου διανυσματικού

μετασχηματιστή έτσι ώστε να ληφθούν τα διανύσματα χαρακτηριστικών για τα βιβλία με μηδενική διαβάθμιση (*zero\_books\_x\_vectors*). Με βάση την επιλογή του χρήστη επιστρέφονται οι προβλεπόμενες βαθμολογίες για τα βιβλία με μηδενική βαθμολογία χρησιμοποιώντας τον αντίστοιχο ταξινομητή.

## Περιγραφή Γραφικού Περιβάλλοντος

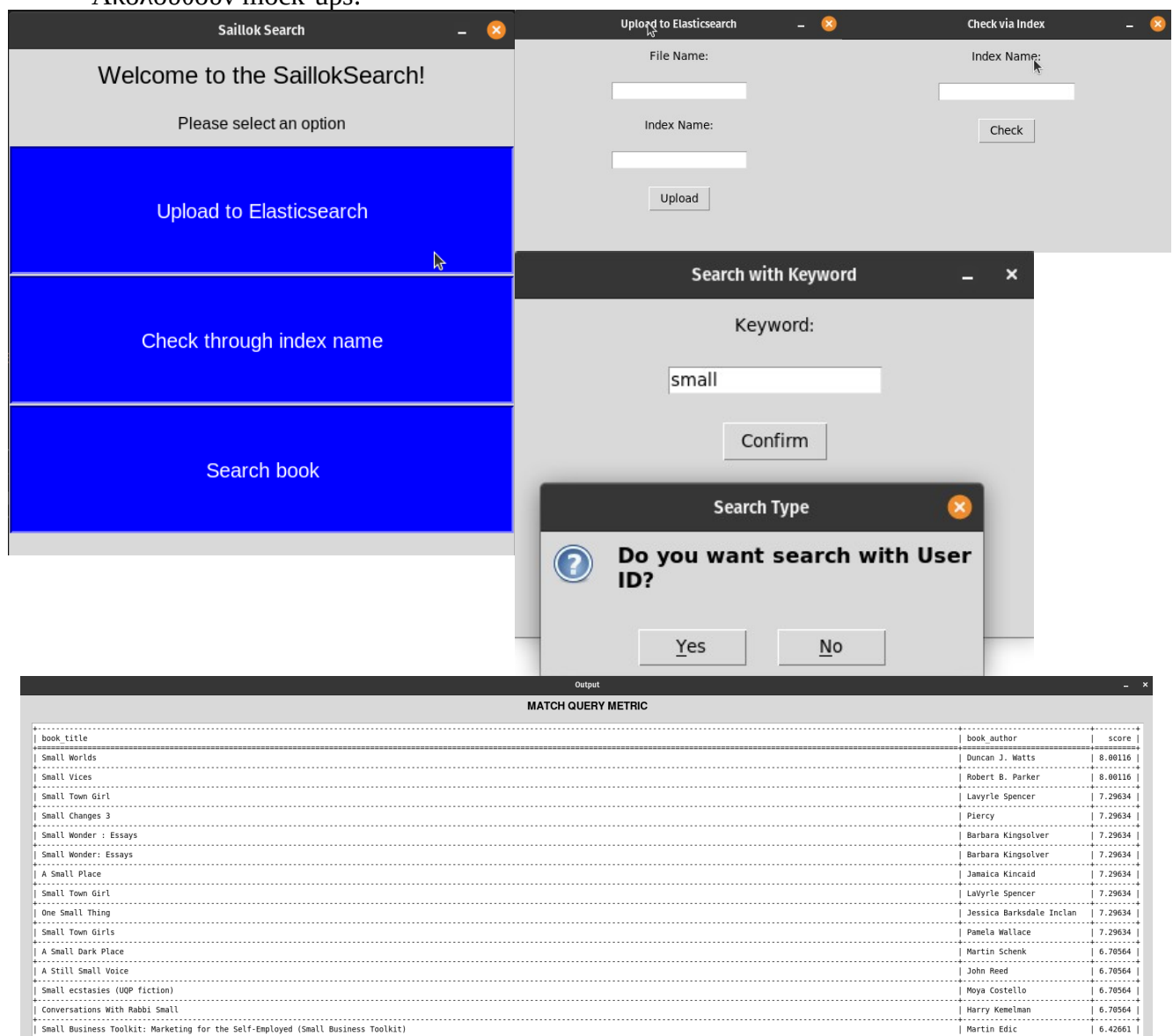
Όπως προαναφέρθηκε υλοποιήθηκε και GUI (Graphical User Interface) για διάφορες υλοποιήσεις του συγκεκριμένου project.

Συγκεκριμένα, ανέβασμα ενός csv αρχείου, έλεγχος επιτυχούς ανεβάσματος μέσω του index του αρχείου

Όπως προαναφέρθηκε στην ενότητα [περιγραφή περιβάλλοντος υλοποίησης](#) υλοποιήθηκε και GUI (Graphical User Interface) για διάφορες υλοποιήσεις του συγκεκριμένου project.

Συγκεκριμένα, ανέβασμα ενός csv αρχείου, έλεγχος επιτυχούς ανεβάσματος μέσω index που περιγράφει το αρχείο και ανάκτηση δεδομένων με εισαγωγή λέξης κλειδί, ήταν αυτές που διεκπεραιώθηκαν.

Ακολουθούν mock-ups:



## Περιγραφή Αποτελεσμάτων Υλοποίησης

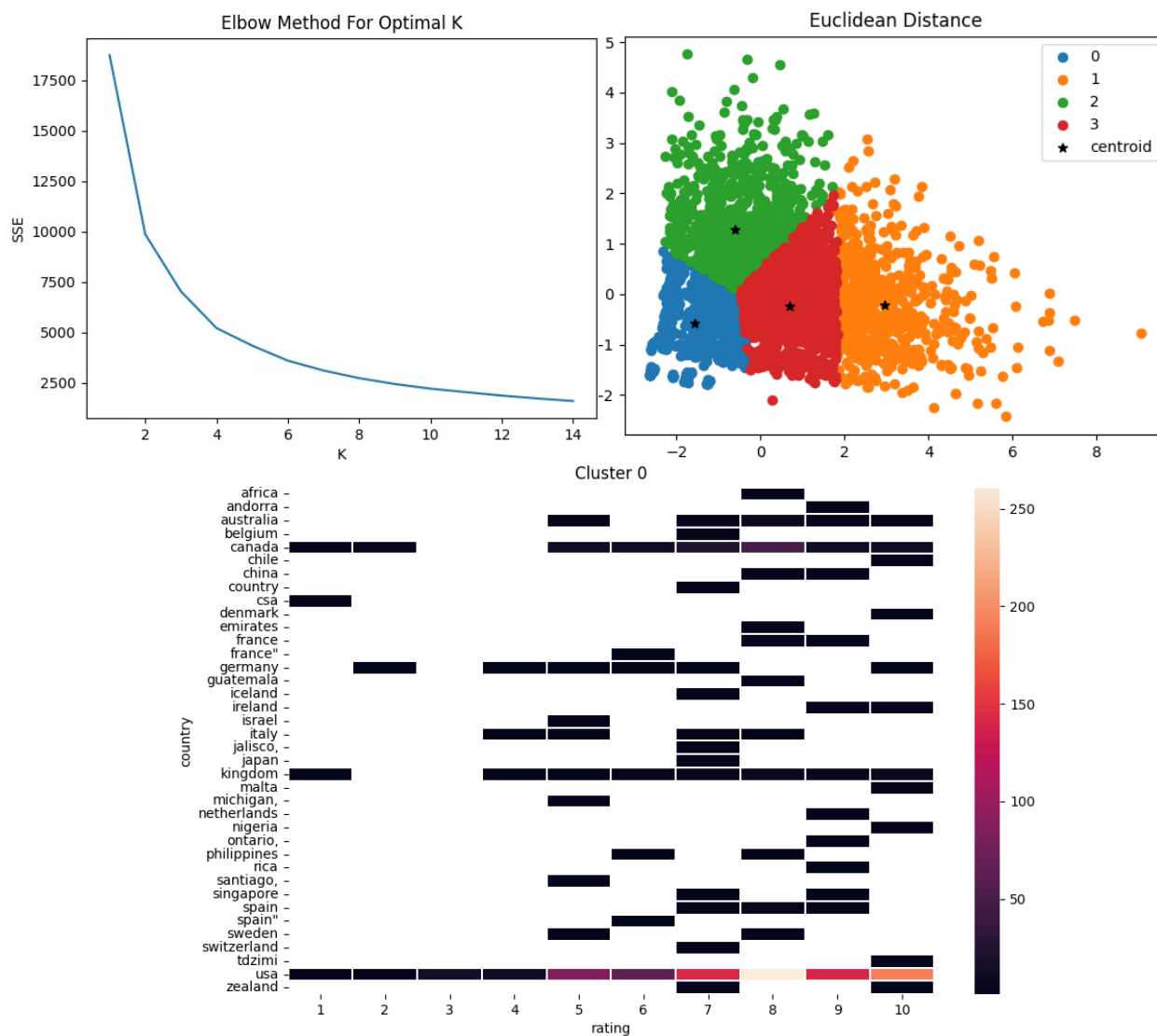
Ως καλύτερες τιμές testing χρησιμοποιήθηκε ένα script που να βρίσκει στο csv αρχείο των Ratings το uid με τις περισσότερες κριτικές (common.py).

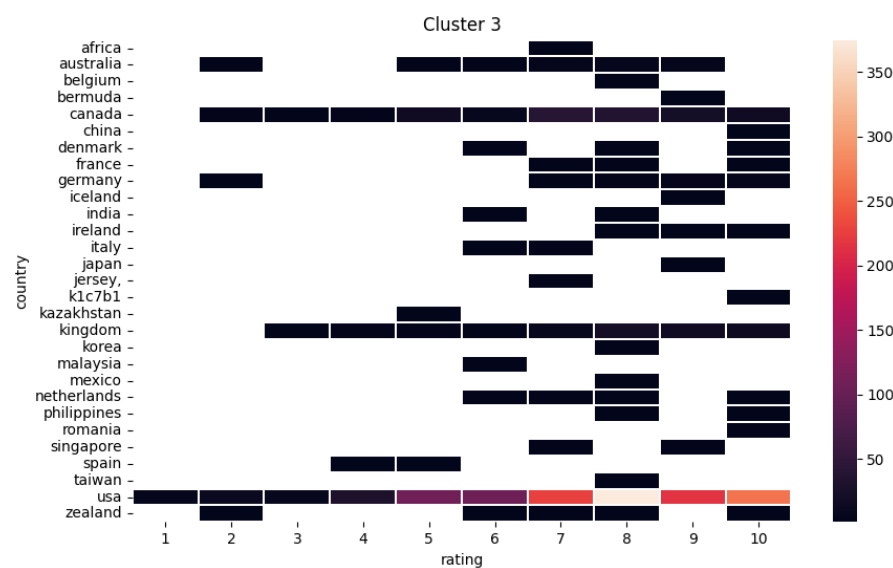
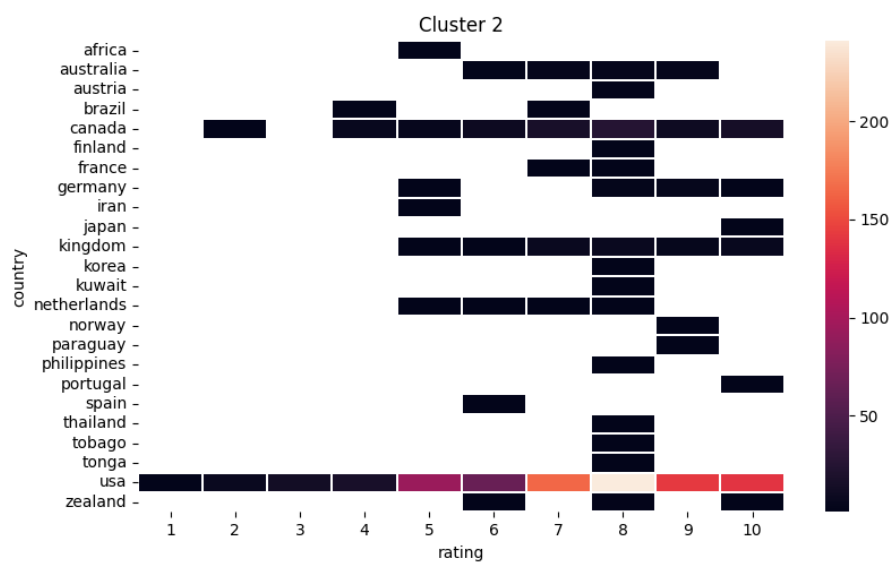
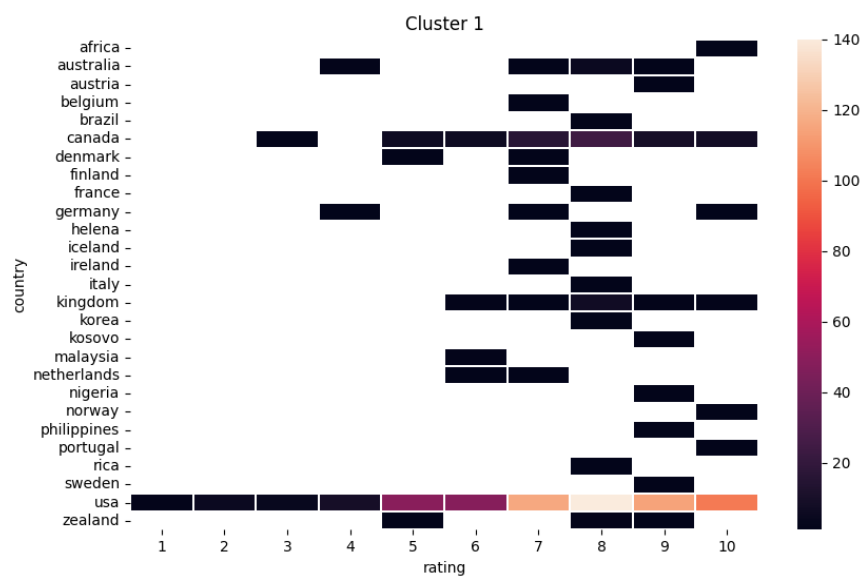
### Ερώτημα Πρώτο

```
Menu:
1. Upload a File to ElasticSearch
2. Check via index
3. Retrieve Data from ElasticSearch
4. Data Cluster
5. Quit
Enter your choice: 3
Give the keyword: small
1. Search with simply metric
2. Search with uid metric
Choice: 2
Give user id: 11676
Use neural network? [Y/n]: n
```

```
=====MATCH QUERY METRIC THROUGH USER ID=====
book_title book_author score
5 Small Wonder: Essays Barbara Kingsolver 8.098175
0 Small Worlds Duncan J. Watts 8.000751
4 Small Wonder : Essays Barbara Kingsolver 7.846465
2 Small Town Girl Lavyrle Spencer 7.720205
3 Small Changes 3 Piercy 7.542619
16 The God of Small Things Arundhati Roy 7.291484
17 Notes from a Small Island Bill Bryson 6.584225
10 A Small Dark Place Martin Schenk 6.441999
33 Don't Sweat the Small Stuff and It's All Small... Richard Carlson 5.533744
1 Small Vices Robert B. Parker 5.200751
15 The God of Small Things Arundhati Roy 4.800266
9 Small Town Girls Pamela Wallace 4.742619
7 Small Town Girl LaVyrle Spencer 4.742619
8 One Small Thing Jessica Barksdale Inclan 4.742619
6 A Small Place Jamaica Kincaid 4.742619
11 A Still Small Voice John Reed 4.358666
13 Conversations With Rabbi Small Harry Kemelman 4.358666
12 Small ecstasies (UQP fiction) Moya Costello 4.358666
25 Don't Sweat the Small Stuff in Love Richard Carlson 4.290258
14 Small Business Toolkit: Marketing for the Self... Martin Edic 4.177297
19 A Small Book of Fairies Eugene Stiles 4.032225
20 All Creatures Great and Small James Herriot 4.032225
21 Small Gods (Discworld Novels (Paperback)) Terry Pratchett 4.032225
22 Small Hours of the Morning Yorke 4.032225
23 The Small Rain : A Novel Madeleine L'Engle 4.032225
18 Notes from a Small Island Bill Bryson 4.032225
24 First Test (Protector of the Small) Tamora Pierce 3.751274
26 Small Pig (I Can Read Book 2) Arnold Lobel 3.506924
27 Mr. Small (Mr. Men and Little Miss) Roger Hargreaves 3.506924
28 Any Small Goodness: A Novel of the Barrio Tony Johnston 3.292460
30 Friday the Rabbi Slept Late (Rabbi Small Mystery) Harry Kemelman 3.292460
31 The Large, the Small and the Human Mind Roger Penrose 3.292460
32 Any Small Thing Can Save You: A Bestiary Christina Adam 3.292460
29 Sunday the Rabbi Stayed Home (Rabbi Small Myst... Harry Kemelman 3.292460
34 Runamok: A Novel About the Realities of Small ... Tom Park 3.102716
35 Girl Named Zippy: Growing Up Small in Moorelan... Haven Kimmel 3.102716
36 Idyll Banter : Weekly Excursions to a Very Sma... CHRIS BOHJALIAN 3.102716
37 Small Sacrifices: A True Story of Passion and ... Ann Rule 3.102716
38 Hope's Edge: The Next Diet for a Small Planet Frances Moore Lappe 3.102716
44 Don't Sweat the Small Stuff at Work : Simple W... Richard Carlson 3.097188
39 Lasagna Gardening for Small Spaces : A Layerin... Patricia Lanza 3.096239
40 A Still, Small Voice: A Practical Guide on Rep... Benedict J., Fr. Groeschel 2.933650
42 Working in a Very Small Place: The Making of a... Mark L. Shelton 2.782056
41 The City Gardener's Handbook: The Definitive G... Linda Yang 2.782056
43 Small Patchwork Projects: With Step-By-Step In... Barbara. Brondolo 2.645359
45 Moving To A Small Town : A Guidebook To Moving... Wanda Urbanska 2.408660
46 Don't Sweat the Small Stuff in Love : Simple W... Richard Carlson 2.352223
47 A Girl Named Zippy: Growing Up Small in Moorel... HAVEN KIMMEL 2.305515
48 Ozarks: The Hills Are Alive With Small Towns a... Hannah Alexander 2.210840
49 Grassroots Organizations: A Resource Book for ... Robert L. Clifton 2.210840
50 A Death in Texas: A Story of Race, Murder, and... Dina Temple-Raston 2.210840
51 BLOW: How a Small-Town Boy Made $100 Million w... Bruce Porter 2.043048
52 The INVENTION THAT CHANGED THE WORLD: HOW A SM... Robert Buderl 1.773802
=====
```

```
Menu:
1. Upload a File to Elasticsearch
2. Check via index
3. Retrieve Data from Elasticsearch
4. Data Cluster
5. Quit
Enter your choice: 4
1. Euclidean Distance
2. Cosine Similarity
Choice: 1
Books to fetch: 5000
Plot Elbow Graph? (Y/N) --> y
```





## Ερώτημα Τρίτο

```
Menu:
1. Upload a File to ElasticSearch
2. Check via index
3. Retrieve Data from ElasticSearch
4. Data Cluster
5. Quit
Enter your choice: 3
Give the keyword: small
1. Search with simply metric
2. Search with uid metric
Choice: 2
Give user id: 11676
Use neural network? [Y/n]: y
564 243
Mean Accuracies:
1. RandomForest: 0.2345679012345679
2. LogisticRegression: 0.18106995884773663
3. DecisionTree: 0.18930041152263374
4. SVM: 0.1934156378600823
Classifier: 3
```

```
=====MATCH QUERY METRIC THROUGH USER ID=====
5          book title          book author          score
0          Small Wonder: Essays Barbara Kingsolver 8.098175
4          Small Worlds        Duncan J. Watts    8.000751
2          Small Wonder : Essays Barbara Kingsolver 7.846465
3          Small Town Girl     Lavyrle Spencer   7.720205
16         Small Changes 3     Piercy            7.542619
8          The God of Small Things Arundhati Roy      7.291484
1          One Small Thing     Jessica Barksdale Inclan 6.992619
13         Small Vices        Robert B. Parker   6.950751
15         Conversations With Rabbi Small Harry Kemelman    6.858666
6          The God of Small Things Arundhati Roy      6.800266
9          A Small Place       Jamaica Kincaid    6.742619
17         Small Town Girls    Pamela Wallace     6.742619
7          Notes from a Small Island Bill Bryson        6.584225
10         Small Town Girl     LaVyrle Spencer   6.492619
11         A Small Dark Place   Martin Schenk      6.441999
25         A Still Small Voice John Reed          6.358666
21         Don't Sweat the Small Stuff in Love Richard Carlson    6.290258
20         Small Gods (Discworld Novels (Paperback)) Terry Pratchett    6.282225
18         All Creatures Great and Small James Herriot      6.282225
19         Notes from a Small Island Bill Bryson        6.032225
26         A Small Book of Fairies Eugene Stiles      6.032225
12         Small Pig (I Can Read Book 2) Arnold Lobel       6.006924
30         Small ecstasies (UQP fiction) Moya Costello     5.858666
22         Friday the Rabbi Slept Late (Rabbi Small Mystery) Harry Kemelman    5.792460
23         Small Hours of the Morning Yorke              5.782225
24         The Small Rain : A Novel Madeleine L'Engle 5.782225
34         First Test (Protector of the Small) Tamora Pierce     5.751274
32         Hope's Edge: The Next Diet for a Small Planet Frances Moore Lappe 5.602716
33         Any Small Thing Can Save You: A Bestiary Christina Adam     5.542460
14         Don't Sweat the Small Stuff and It's All Small... Richard Carlson    5.533744
35         Small Business Toolkit: Marketing for the Self... Martin Edic        5.427297
28         Girl Named Zippy: Growing Up Small in Moorelan... Haven Kimmel       5.352716
29         Any Small Goodness: A Novel of the Barrio Tony Johnston      5.292460
31         Sunday the Rabbi Stayed Home (Rabbi Small Myst... Harry Kemelman    5.292460
37         The Large, the Small and the Human Mind Roger Penrose      5.292460
39         Small Sacrifices: A True Story of Passion and ... Ann Rule          5.102716
42         Lasagna Gardening for Small Spaces : A Layerin... Patricia Lanza     5.096239
44         Working in a Very Small Place: The Making of a... Mark L. Shelton    5.032056
46         Don't Sweat the Small Stuff at Work : Simple W... Richard Carlson    4.847188
27         Mr. Small (Mr. Men and Little Miss) Roger Hargreaves  4.756924
34         Runamok: A Novel About the Realities of Small ... Tom Park          4.602716
47         A Girl Named Zippy: Growing Up Small in Moorel... HAVEN KIMMEL      4.555515
40         A Still, Small Voice: A Practical Guide on Rep... Benedict J., Fr. Groeschel 4.433650
36         Idyll Banter : Weekly Excursions to a Very Sma... CHRIS BOHJALIAN   4.352716
41         The City Gardener's Handbook: The Definitive G... Linda Yang        4.282056
48         Ozarks: The Hills Are Alive With Small Towns a... Hannah Alexander  4.210840
49         Grassroots Organizations: A Resource Book for ... Robert L. Clifton 4.210840
46         Don't Sweat the Small Stuff in Love : Simple W... Richard Carlson    4.102223
45         Moving To A Small Town : A Guidebook To Moving... Wanda Urbanska    3.908660
43         Small Patchwork Projects: With Step-By-Step In... Barbara. Brondolo 3.895359
51         BLOW: How a Small-Town Boy Made $100 Million w... Bruce Porter      3.793048
50         A Death in Texas: A Story of Race, Murder, and... Dina Temple-Raston 3.710840
52         The INVENTION THAT CHANGED THE WORLD: HOW A SM... Robert Buderl     3.523802
=====
```

Σας Ευχαριστώ