Τμήμα Μηχανικών Η/Υ & Πληροφορικής
Πανεπιστήμιο Πατρών

Ανάκτηση Πληροφορίας

Εργαστηριακή Άσκηση
Χειμερινό Εξάμηνο 2020
Διδάσκων: Χ. Μακρής
Επικουρικό: Α. Μπομπότας, Γ. Ρόμπολας

Εργασία: Δασούλας Ιωάννης – 1053711 – 5ο Έτος
Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών

Εισαγωγή

Η εργασία αφορά την υλοποίηση μιας μηχανής αναζήτησης κινηματογραφικών ταινιών η οποία βασίζεται στην Elasticsearch και αποφασίζει τη σειρά παρουσίασης των αποτελεσμάτων χρησιμοποιώντας τεχνικές μηχανικής μάθησης.

Για τα 4 ερωτήματα της εργασίας δημιουργήθηκαν 2, 1, 2 και 2 προγράμματα, αντίστοιχα γραμμένα στην γλώσσα Python. Αναλυτικά σχόλια για τον κώδικα και την λειτουργία τους υπάρχουν στα αρχεία. Στην παρούσα αναφορά αναλύεται η φιλοσοφία υλοποίησης που ακολουθήθηκε, τα βήματα υλοποίησης και τα αποτελέσματα, ενώ γίνεται αναφορά και στις βιβλιοθήκες της Python που χρησιμοποιήθηκαν και στην διαδικασία εγκατάστασής τους. Τα προγράμματα δημιουργήθηκαν στο περιβάλλον Spyder. Τα αρχεία “new_ratings3.csv” και “new_ratings4.csv” είναι τα αρχεία που δημιουργούνται στα προγράμματα task3.1, task4.1 αντίστοιχα και διατίθενται στο παραδοτέο για να μπορούν να χρησιμοποιηθούν απευθείας στις μετρικές. Εναλλακτικά, μπορούν να δημιουργηθούν από την αρχή, με την εκτέλεση των αντίστοιχων προγραμμάτων.

Ερώτημα 1

1.1 - 1ο Πρόγραμμα ερωτήματος: Εισαγωγή ταινιών στην Elasticsearch (task1.1.py)

Αρχικά, για την υλοποίηση της άσκησης χρειάζεται να γίνει η εγκατάσταση της Elasticsearch. Η Elasticsearch είναι μηχανή αναζήτησης και προβολής στατιστικών στοιχείων με γρήγορο τρόπο για δεδομένα που αποθηκεύει σε αυτή ο χρήστης. Η λήψη της έγινε από τη διεύθυνση https://www.elastic.co/downloads/elasticsearch. Μετά την λήψη και την αποσυμπίεση του φακέλου, ο χρήστης πρέπει να τρέξει την εντολή 'bin/elasticsearch' (ή 'bin\elasticsearch.bat' στα Windows) για να την ενεργοποιήσει. Εναλλακτικά μπορεί να τρέξει την εντολή 'curl http://localhost:9200/' ή 'Invoke-RestMethod http://localhost:9200' στο PowerShell των Windows.

Επόμενο βήμα ήταν η δημιουργία του προγράμματος. Για το πρόγραμμα χρησιμοποιήθηκε η βιβλιοθήκη pandas για ευκολότερη αποθήκευση των δεδομένων, η sys για έξοδο σε περίπτωση αδυναμίας σύνδεσης με την Elasticsearch ή μη εύρεσης του αρχείου των ταινιών και η elasticsearch μέσω της οποίας γίνεται η σύνδεση στην βάση.

Οδηγίες πριν την εκτέλεση:

- pip install elasticsearch
- pip install pandas

Αρχικά, στο πρόγραμμα επιχειρείται η σύνδεση με την elasticsearch, που γίνεται επιτυχώς αν τρέχει το elasticsearch.bat αρχείο με τον τρόπο που εξηγήθηκε παραπάνω. Έπειτα, φορτώνονται σε ένα pandas dataframe οι ταινίες από το αρχείο movies.csv και με μία επαναληπτική διαδικασία αποθηκεύονται μία-μία στην βάση δεδομένων 'movies' χρησιμοποιώντας ως id το movieId της κάθε μίας.

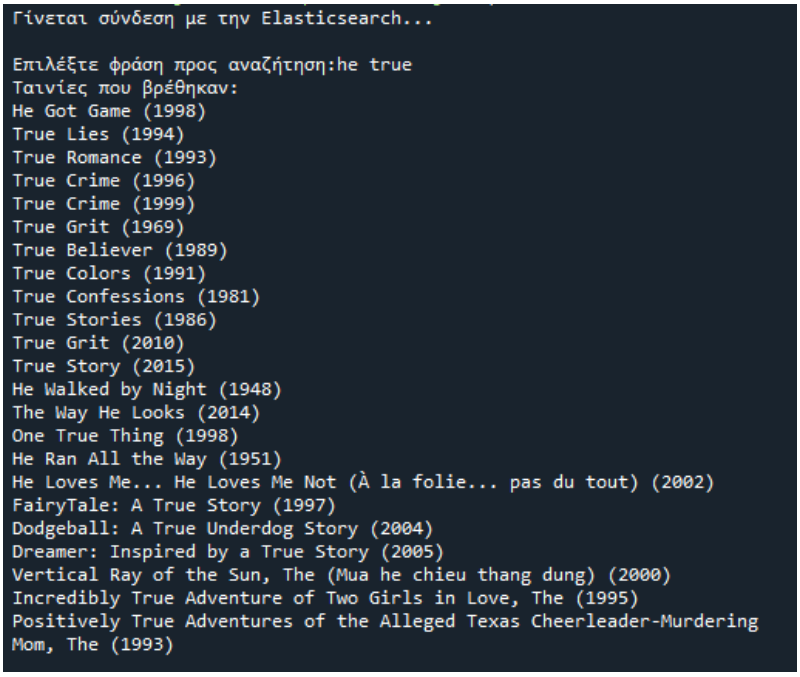
1.2 - 2ο Πρόγραμμα ερωτήματος: Αναζήτηση ταινιών με τη μετρική ομοιότητας της Elasticsearch (task1.2.py)

Για το πρόγραμμα χρησιμοποιήθηκε η sys για έξοδο σε περίπτωση αδυναμίας σύνδεσης με την Elasticsearch και η elasticsearch μέσω της οποίας γίνεται η σύνδεση στην βάση.

Οδηγίες πριν την εκτέλεση:

- pip install elasticsearch

Όπως και προηγουμένως, πρώτη ενέργεια είναι η προσπάθεια σύνδεσης με την elasticsearch, τρέχοντας πρώτα το αρχείο elasticsearch.bat. Έπειτα, το πρόγραμμα ζητάει από τον χρήστη να πληκτρολογήσει την λέξη/φράση που θέλει να αναζητήσει. Η φράση που πληκτρολογεί ο χρήστης, στην συνέχεια τροφοδοτεί το σώμα αναζήτησης 'body', το οποίο εισάγεται στην συνάρτηση search της elasticsearch ως παράμετρος, όπως και το όνομα της βάσης 'movies' που δημιουργήθηκε προηγουμένως. Τα αποτελέσματα φιλτράρονται για να κρατηθούν οι τίτλοι των ταινιών, καθώς η συνάρτηση search επιστρέφει ένα dictionairy αποτελεσμάτων. Έπειτα τα αποτελέσματα παρουσιάζονται στην κονσόλα του χρήστη. Ενδεικτικά, για την αναζήτηση της φράσης "he true" τα αποτελέσματα είναι τα εξής:



Παρατηρείται ότι μεγαλύτερο ‘σκορ’ συγκεντρώνουν οι ταινίες που έχουν τίτλο ο οποίος αποτελείται σε μεγάλο ποσοστό από κάποια από τις λέξεις που δίνονται για αυτό και παρουσιάζονται πρώτες, στο πάνω μέρος. Αντίθετα, ο τελευταίος τίτλος που μόνο ένα πολύ μικρό κομμάτι του αποτελείται από τη λέξη “true” που αναζητήθηκε, παρουσιάζεται τελευταίος στην κατάταξη.

Επίσης, παρά τον μεγάλο αριθμό ταινιών της βάσης, αξίζει να σημειωθεί η ταχύτητα με την οποία η elasticsearch δίνει τα αποτελέσματα, χαρακτηριστικό που αποτελεί βασικό πλεονέκτημα της, παρέχοντας πολύ γρήγορα, όχι μόνο τους τίτλους, αλλά όλα τα δεδομένα που αφορούν τις ταινίες.

Ερώτημα 2

2.1 - Πρόγραμμα ερωτήματος: Τροποποίηση μετρικής αναζήτησης βάσει βαθμολογιών χρήστη και μέσους όρους βαθμολογιών (task2.py)

Για το πρόγραμμα χρησιμοποιήθηκε η βιβλιοθήκη pandas για ευκολότερη αποθήκευση των δεδομένων, η sys για έξοδο σε περίπτωση αδυναμίας σύνδεσης με την Elasticsearch ή μη εύρεσης του αρχείου των βαθμολογιών και η elasticsearch μέσω της οποίας γίνεται η σύνδεση στην βάση.

Οδηγίες πριν την εκτέλεση:

- pip install elasticsearch
- pip install pandas

Όπως και προηγουμένως, πρώτη ενέργεια είναι η προσπάθεια σύνδεσης με την elasticsearch, τρέχοντας πρώτα το αρχείο elasticsearch.bat. Έπειτα, ζητείται από τον χρήστη να πληκτρολογήσει ένα user Id και μία φράση προς αναζήτηση. Με ίδιο τρόπο, όπως στο προηγούμενο ερώτημα, επιστρέφονται τα στοιχεία των ταινιών που μοιάζουν με τη δοσμένη φράση, αλλά δεν τυπώνονται ακόμα.

Στην συνέχεια, αποθηκεύεται το score της elasticsearch και τα movie ids των ταινιών που επιστράφηκαν, για την αναζήτηση των υπολοίπων βαθμολογιών. Φορτώνεται το αρχείο των βαθμολογιών και διατηρούνται οι βαθμολογίες των ταινιών που η elasticsearch επέστρεψε. Από αυτές βρίσκεται ο μέσος όρος των βαθμολογιών για κάθε ταινία, αν έχει βαθμολογηθεί από έναν τουλάχιστον χρήστη, και το σκορ του χρήστη για κάθε ταινία, αν έχει βαθμολογηθεί από τον χρήστη με id αυτό που δηλώθηκε στην αρχή.

Αφού, έχουν υπολογιστεί τα 3 αυτά σκορ (σκορ της elasticsearch, μέσος βαθμός και βαθμός χρήστη) εφαρμόζεται σε αυτά μια μετρική για την δημιουργία του τελικού σκορ. Σε κάθε σκορ δίνεται ένα βάρος (πολλαπλασιαστής) σημαντικότητας. Συγκεκριμένα, δίνεται βάρος 1 στο σκορ της elasticsearch και στον μέσο βαθμό και βάρος 2 στο σκορ του χρήστη για να δοθεί μεγαλύτερη σημασία σε αυτό. Κάθε σκορ πολλαπλασιάζεται με το βάρος του και έπειτα προστίθενται μεταξύ τους. Στο τέλος, το σκορ διαιρείται με το 3, αν υπήρχαν και τα 3 σκορ ή αλλιώς με 2 αν υπήρχε μόνο σκορ της elasticsearch και μέσο σκορ. Αν δεν υπάρχει ούτε μέσο σκορ, ούτε σκορ χρήστη, το σκορ δεν διαιρείται. Η διαίρεση γίνεται για να βρεθεί στο τέλος ένα μέσο σκορ μεταξύ

τους και να αναδειχτούν οι ταινίες με καλή βαθμολογία. Αν δεν γινόταν η διαίρεση, μια ταινία με χαμηλή μέση βαθμολογία ή βαθμολογία χρήστη θα είχε μεγαλύτερο σκορ από μία ταινία χωρίς βαθμολογία, με την υπόθεση ότι έχουν ίδιο σκορ στην elasticsearch. Επομένως, η διαίρεση έχει σκοπό την ανάδειξη των καλών βαθμολογιών και την και την πτώση στην κατάταξη των ταινιών με κακές βαθμολογίες. Επίσης, η μετρική δίνει μεγάλη βαρύτητα στο σκορ της elasticsearch, παρόλο που έχει βάρος 1, διότι για μεγάλη ομοιότητα μεταξύ φράσης εισόδου και τίτλου, η βαθμολογία της elasticsearch δίνει σκορ πάνω από 5 στον τίτλο, ενώ οι βαθμολογίες περιορίζονται μέχρι το 5. Δηλαδή, ακόμα κι αν μια ταινία έχει μέτρια βαθμολογία, αν η ομοιότητα με την φράση αναζήτησης είναι πολύ μεγάλη, η ταινία θα έχει μεγάλο τελικό σκορ.

Αφού δημιουργηθεί το τελικό σκορ, ο πίνακας με τα movie ids που είχε υπολογιστεί ταξινομείται με φθίνουσα σειρά τελικού σκορ. Έπειτα, για κάθε movie id του πίνακα καλείται η συνάρτηση get της elasticsearch, η οποία παίρνει ως όρισμα το όνομα της βάσης δεδομένων (‘movies’) και το id της ταινίας, κι έτσι επιστρέφει τους τίτλους των ταινιών με φθίνουσα σειρά τελικού σκορ. Τέλος, οι τίτλοι τυπώνονται στην κονσόλα. Ενδεικτικά, για τον χρήστη 37 και την φράση “king”, τα αποτελέσματα είναι τα εξής:

```
Γίνεται σύνδεση με την Elasticsearch...

Πληκτρολογήστε το αναγνωριστικό σας:37

Επιλέξτε φράση προς αναζήτηση:king

Εύρεση σκορ χρήστη και μέσου σκορ...
Οι ταινίες που βρέθηκαν:
Lion King, The (1994)
King Kong (1933)
King Is Alive, The (2000)
King Rat (1965)
King Solomon's Mines (1950)
King Kong (2005)
King of Hearts (1966)
King Solomon's Mines (1937)
King of Kong, The (2007)
Fisher King, The (1991)
King of Comedy, The (1983)
King Kong (1976)
King of New York (1990)
King Solomon's Mines (1985)
Last King of Scotland, The (2006)
King of Marvin Gardens, The (1972)
Madness of King George, The (1994)
King Arthur (2004)
King of Kings (1961)
Anna and the King (1999)
King and I, The (1956)
Man Who Would Be King, The (1975)
King Kong Lives (1986)
Librarian: Return to King Solomon's Mines, The (2006)
King in New York, A (1957)
King of the Hill (1993)
```

Παρατηρείται πως ενώ ο τίτλος “King Kong” είναι πιο όμοιος με την λέξη ‘king’ από τον τίτλο “Lion King, The”, ο δεύτερος είναι πιο ψηλά στην βαθμολογία. Με μία σύντομη αναζήτηση στις βαθμολογίες του χρήστη 37, μπορεί να φανεί ότι αυτό γίνεται διότι έχει βαθμολογήσει την ταινία “Lion King, The” με 5.0 που είναι το μέγιστο σκορ χρήστη. Επίσης, ενώ και ο τίτλος “King Arthur” είναι κατά πολύ όμοιος με τη λέξη ‘king’, παρατηρείται ότι και αυτός είναι χαμηλά στην κατάταξη. Με μία σύντομη ματιά στις βαθμολογίες αποδεικνύεται ότι ο χρήστης δεν έχει βαθμολογήσει αυτή την ταινία και η ταινία αυτή έχει μικρό μέσο όρο βαθμολογίας (περίπου 2.5). Οπότε ενώ είχε μεγάλο σκορ στην αναζήτηση της elasticsearch, οι κακές επιδόσεις της στις βαθμολογίες την έριξαν στην κατάταξη. Τα αποτελέσματα επιβεβαιώνουν την ορθή μεταφορά της φιλοσοφίας της μετρικής στην κατάταξη.

Ερώτημα 3

3.1 – 1ο Πρόγραμμα ερωτήματος: Κατηγοριοποίηση των χρηστών σε συστάδες και συμπλήρωση για κάθε χρήστη των βαθμολογιών που του λείπουν χρησιμοποιώντας τον μέσο όρο της ταινίας στην συστάδα στην οποία εκείνος ανήκει (task3.1.py)

Το ερώτημα 3 ασχολείται με το πρόβλημα που προκύπτει στην προηγούμενη μετρική, το γεγονός ότι οι χρήστες συνήθως βαθμολογούν μόνο ένα υποσύνολο των ταινιών. Σε αυτό το ερώτημα οι χρήστες συσταδοποιούνται βάσει των βαθμολογιών που έχουν δώσει σε κάθε κατηγορία ταινίας και στο τέλος συμπληρώνονται οι βαθμολογίες που τους λείπουν βάσει των βαθμολογιών της συστάδας στην οποία ανήκουν.

Μιας και το ερώτημα αναφέρει την συμπλήρωση για κάθε χρήστη των βαθμολογιών, που του λείπουν χρησιμοποιώντας τον μέσο όρο της ταινίας στην συστάδα στην οποία εκείνος ανήκει, αποφασίστηκε να δημιουργηθούν δύο προγράμματα για το ερώτημα. Το ένα δημιουργεί το νέο σύνολο δεδομένων που περιέχει μια βαθμολογία για κάθε ταινία για κάθε χρήστη, με τον τρόπο και την στοίχιση που έχει το αρχείο ‘ratings’, ώστε να μην είναι δύσκολο στην ανάγνωσή του, ενώ το δεύτερο εφαρμόζει τη νέα μετρική, φορτώνοντας το νέο αρχείο ‘new_ratings3.csv’ που δημιουργείται. Με αυτή τη μέθοδο οι νέες βαθμολογίες υπολογίζονται μία μόνο φορά, με το μειονέκτημα όμως ότι το πρόγραμμα υπολογισμού απαιτεί αρκετό χρόνο για τη συμπλήρωση όλων των βαθμολογιών για τους 671 χρήστες, αλλά δημιουργήθηκε με τη φιλοσοφία ενός νέου αρχείου βαθμολογιών διαθέσιμο για ανάγνωση και χρήση στις μετρικές με μεγάλη ταχύτητα. Το αρχείο “new_ratings3.csv” διατίθεται και έτοιμο στην αναφορά για να μπορεί να χρησιμοποιηθεί κατευθείαν στις νέες μετρικές.

Για το πρόγραμμα χρησιμοποιήθηκε η βιβλιοθήκη pandas για ευκολότερη αποθήκευση των δεδομένων, η sys για έξοδο σε περίπτωση μη εύρεσης των αρχείου των ταινιών και των βαθμολογιών, η numpy για την διαχείριση των NaN τιμών και η sklearn για την αντικατάσταση NaN τιμών με τον μέσο όρο των πεδίων που έχουν βαθμό (συνάρτηση SimpleImputer()) και για την εκτέλεση του Kmeans αλγορίθμου (συνάρτηση Kmeans()).

Οδηγίες πριν την εκτέλεση:

- pip install numpy
- pip install pandas
- pip install sklearn

Το πρόγραμμα ξεκινάει με την φόρτωση των αρχείων των ταινιών και βαθμολογιών και στην συνέχεια ανιχνεύει από το αρχείο των ταινιών όλες τις κατηγορίες ταινιών που υπάρχουν. Επόμενο βήμα είναι η εύρεση μέσης βαθμολογίας κάθε κατηγορίας για κάθε χρήστη. Για τους χρήστες που δεν έχουν βαθμολογήσει ταινία κάποιας κατηγορίας, η βαθμολογία συμπληρώνεται με τον μέσο όρο βαθμολογιών της κατηγορίας με τη συνάρτηση SimpleImputer() της sklearn. Ακολουθεί η συσταδοποίηση των χρηστών. Επιλέχτηκε ο αριθμός των 8 συστάδων για τους 671 χρήστες και τις 20 κατηγορίες ταινιών ως ένας αντιπροσωπευτικός αριθμός για το πρόβλημα. Φυσικά, το πρόγραμμα είναι παραμετροποιήσιμο, και με μία αλλαγή μπορεί να δουλέψει για οποιοδήποτε αριθμό συστάδων. Ακολουθεί η εύρεση μέσης βαθμολογίας κάθε συστάδας για κάθε ταινία. Για τις ταινίες που δεν έχουν βαθμολογηθεί από κανέναν χρήστη της συστάδας, η βαθμολογία συμπληρώνεται με τον μέσο όρο όλων των χρηστών. Στο τέλος γίνεται η δημιουργία του νέου συνόλου δεδομένου, με βαθμολογία σε κάθε ταινία για κάθε χρήστη. Στο καινούργιο σύνολο δεδομένων συμπληρώνεται για κάθε χρήστη οι βαθμολογίες που είχε προηγουμένως και οι μέσοι βαθμοί της συστάδας του για τις ταινίες που δεν είχε βαθμολογήσει. Το αρχείο αποθηκεύεται με το όνομα “new_ratings3.csv” και μπορεί να χρησιμοποιηθεί κανονικά μετά από την αποσυμπίεσή του.

```
Υπολογισμός υπόλοιπων βαθμολογιών για κάθε χρήστη βάσει συσταδοποίησης...
Φόρτωση ταινιών και βαθμολογιών...
Εύρεση όλων των κατηγοριών ταινίας που υπάρχουν...
#Εύρεση μέσης βαθμολογίας κάθε κατηγορίας για κάθε χρήστη ...
Συσταδοποίηση χρηστών βάσει μέσης βαθμολογίας σε κάθε κατηγορία ...
Εύρεση μέσης βαθμολογίας κάθε συστάδας για κάθε ταινία ...
Δημιουργία νέου συνόλου δεδομένου, με βαθμολογία σε κάθε ταινία για κάθε χρήστη ...
Υπολογισμός νέων βαθμολογιών για τον χρήστη 1 ...
Ο υπολογισμός για τον χρήστη 1 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 2 ...
Ο υπολογισμός για τον χρήστη 2 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 3 ...
Ο υπολογισμός για τον χρήστη 3 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 4 ...
Ο υπολογισμός για τον χρήστη 4 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 5 ...
Ο υπολογισμός για τον χρήστη 5 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 6 ...
Ο υπολογισμός για τον χρήστη 6 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 7 ...
Ο υπολογισμός για τον χρήστη 7 ολοκληρώθηκε!

Υπολογισμός νέων βαθμολογιών για τον χρήστη 8 ...
Ο υπολογισμός για τον χρήστη 8 ολοκληρώθηκε!
```

Ενδεικτικά, το αρχείο “new_ratings3.csv” έχει αυτή τη μορφή, στα πρότυπα του αρχείου “ratings.csv”:

	A	B	C
1	userId,movieId,rating		
2	1,1,2.71429		
3	1,2,2.3		
4	1,3,1.5		
5	1,4,2.38462		
6	1,5,2.75		
7	1,6,3.5		
8	1,7,1.25		
9	1,8,3.8		
10	1,9,3.15		
11	1,10,2.66667		
12	1,11,2.5		
13	1,12,2.86111		
14	1,13,2.5		
15	1,14,2.5		
16	1,15,3.0		
17	1,16,3.66667		
18	1,17,3.0		
19	1,18,2.5		
20	1,19,1.9		

3.2 – 2ο Πρόγραμμα ερωτήματος: Εφαρμογή νέας μετρικής χρησιμοποιώντας τις μέσες βαθμολογίες των συστάδων (task3.2.py)

Για το πρόγραμμα χρησιμοποιήθηκε η βιβλιοθήκη pandas για ευκολότερη αποθήκευση των δεδομένων, η sys για έξοδο σε περίπτωση αδυναμίας σύνδεσης με την Elasticsearch ή μη εύρεσης του αρχείου των βαθμολογιών και η elasticsearch μέσω της οποίας γίνεται η σύνδεση στην βάση.

Οδηγίες πριν την εκτέλεση:

- pip install elasticsearch
- pip install pandas

Η μετρική ακολουθεί το προηγούμενο μοντέλο με την προσθήκη της βαθμολογίας συστάδων από το αρχείο “new_ratings3.csv”. Τα βάρη των σκορ της elasticsearch, του χρήστη, της μέσης βαθμολογίας και της βαθμολογίας συστάδας είναι 1, 2, 1 και 1.5 αντίστοιχα. Αν μία ταινία δεν έχει βαθμολογηθεί από κανέναν χρήστη, λαμβάνεται υπόψη μόνο το σκορ της elasticsearch. Αν η ταινία έχει βαθμολογηθεί, αλλά όχι από τον χρήστη, λαμβάνεται υπόψη το σκορ της elasticsearch, η μέση βαθμολογία και η βαθμολογία συστάδας. Αν ταινία έχει βαθμολογηθεί από τον χρήστη, λαμβάνεται υπόψη το σκορ της elasticsearch, του χρήστη και το μέσο σκορ. Στο τέλος το σκορ διαιρείται για τους λόγους που εξηγήθηκαν στο δεύτερο ερώτημα. Η φόρτωση του αρχείου “new_ratings3.csv” και η απουσία υπολογισμών έχει ως αποτέλεσμα τη γρήγορη παρουσίαση των αποτελεσμάτων στην οθόνη. Η μετρική πετυχαίνει τον στόχο της σύμφωνα με τη φιλοσοφία που αναλύθηκε στο ερώτημα 2. Ενδεικτικά, για τον χρήστη 12 και την λέξη “power” τα αποτελέσματα είναι τα εξής:

```
Γίνεται σύνδεση με την Elasticsearch...

Πληκτρολογήστε το αναγνωριστικό σας:12

Επιλέξτε φράση προς αναζήτηση:power

Εύρεση σκορ χρήστη, σκορ συστάδας και μέσου σκορ...
Οι ταινίες που βρέθηκαν:
Absolute Power (1997)
Power of One, The (1992)
Power of Nightmares, The: The Rise of the Politics of Fear (2004)
Pete Seeger: The Power of Song (2007)
Dragon Ball: The Path to Power (Doragon bôru: Saikyô e no michi) (1996)
Fragile Trust: Plagiarism, Power, and Jayson Blair at the New York Times, A (2013)
Mighty Morphin Power Rangers: The Movie (1995)
Turbo: A Power Rangers Movie (1997)
```

Ερώτημα 4

4.1 – 1ο Πρόγραμμα ερωτήματος: Συμπλήρωση βαθμολογιών με την χρήση Word Embeddings (task4.1.py)

Στο ερώτημα 4 ακολουθήθηκε η ίδια φιλοσοφία με το ερώτημα 3, δηλαδή ο υπολογισμός πρώτα του συγκεντρωτικού αρχείου βαθμολογιών με την προσθήκη των προβλέψεων, και στην συνέχεια η εφαρμογή της ανανεωμένης μετρικής.

Στο πρώτο πρόγραμμα γίνεται η πρόβλεψη των υπόλοιπων βαθμολογιών για όλους τους χρήστες, στην ίδια μορφή με τα αρχεία “ratings.csv” και “new_ratings3.csv”, και αποθηκεύονται στο αρχείο “new_ratings4.csv”. Για το πρόγραμμα χρησιμοποιήθηκε η βιβλιοθήκη pandas για ευκολότερη αποθήκευση των δεδομένων, η sys για έξοδο σε περίπτωση μη εύρεσης των αρχείου των ταινιών και των βαθμολογιών, η numpy για την μετατροπή των λιστών σε numpy arrays και χρησιμοποίησή τους στο νευρωνικό δίκτυο, η keras για την υλοποίηση του νευρωνικού δικτύο και η gensim για την υλοποίηση του μοντέλου των Word Embeddings.

Οδηγίες πριν την εκτέλεση:

- pip install numpy
- pip install pandas
- pip install keras
- pip install genism

Αφού φορτώνονται τα αρχεία ταινιών και βαθμολογιών, για κάθε τίτλο εφαρμόζεται ένα φιλτράρισμα με σκοπό την διατήρηση των χαρακτηριστικών λέξεων του τίτλου. Συγκεκριμένα, τα κεφαλαία γράμματα αντικαθίστανται με μικρά, αφαιρείται η ημερομηνία, τα σύμβολα: ! ( ) - [ ] { } ; : ' " \ , < > . / ? @ # $ % ^ & * _ ~ και οι συνηθισμένες λέξεις (stop words): 'the', 'a', 'and', 'is', 'be', 'will', 'can', 'could', 'would', 'was', 'to' και 'of'. Στο περιεχόμενο που απομένει γίνεται tokenization, δηλαδή η πρόταση γίνεται λίστα λέξεων. Οι λίστες αυτές στην συνέχεια προστίθενται σε μία λίστα και τροφοδοτούν το μοντέλο Doc2Vec με την ομώνυμη συνάρτηση της gensim.

Το μοντέλο Doc2Vec επιλέχτηκε διότι η εργασία αφορά τη σχέση μεταξύ των τίτλων και όχι μεταξύ λέξεων των τίτλων. Το Doc2Vec είναι ένα μοντέλο που αναπαριστά κάθε πρόταση ή σύνολο προτάσεων ως ένα διάνυσμα και αποτελεί την εξέλιξη του Word2Vec μοντέλου. Σε αντίθεση με το Word2Vec μοντέλο που παράγει ένα διάνυσμα

για κάθε λέξη, το Doc2Vec παρέχει και τη δυνατότητα παραγωγής διανύσματος για κάθε πρόταση/παράγραφο, καθιστώντας το κατάλληλο για το συγκεκριμένο πρόβλημα. Τα διανύσματα παράγονται με νευρωνικό δίκτυο της βιβλιοθήκης και παρέχουν δείγμα της σχέσης μεταξύ των προτάσεων σε μορφή αριθμών, ανιχνεύοντας συνώνυμες λέξεις και προβλέποντας πιθανές πρόσθετες λέξεις για κάθε πρόταση. Τα διανύσματα επιλέγονται προσεκτικά, έτσι ώστε μια απλή μαθηματική συνάρτηση (η συνημιτονική ομοιότητα μεταξύ των διανυσμάτων) να δείχνει το επίπεδο εννοιολογικής ομοιότητας μεταξύ των προτάσεων που τα διανύσματα αναπαριστούν. Το μήκος του διανύσματος τίθεται ως παράμετρος στην συνάρτηση Doc2Vec, με μελέτες να έχουν δείξει πως όσο μεγαλύτερο, τόσο καλύτερο για την ακρίβεια των αποτελεσμάτων, κάτι που εξαρτάται και από το πλήθος των προτάσεων. Για τις 9125 προτάσεις του ερωτήματος, τέθηκε ίσο με 100.

Αφού δημιουργηθούν τα διανύσματα αυτά για κάθε ταινία, προστίθενται στο καθένα το διάνυσμα κατηγοριών της ταινίας με μορφή one-hot encoding. Για τις 20 κατηγορίες που ανιχνεύονται, δημιουργείται ένα διάνυσμα 20 διαστάσεων με τον αριθμό 0 να αντιστοιχεί στην απουσία της κατηγορίας και τον αριθμό 1 στην παρουσία της κατηγορίας για κάθε ταινία. Το διάνυσμα αυτό προστίθεται έτσι ώστε ακόμα κι αν δεν υπάρχει καμία λεκτική ομοιότητα μιας ταινίας με μία άλλη, να μπορεί να γίνεται κάποιου είδους σύγκριση μεταξύ των ταινιών βάσει των κατηγοριών στις οποίες ανήκουν.

Έπειτα, ακολουθεί η πρόβλεψη της βαθμολογίας των μη βαθμολογημένων ταινιών για κάθε χρήστη, η οποία γίνεται με την κατασκευή νευρωνικού δικτύου. Για κάθε χρήστη πρώτα δημιουργούνται τα δεδομένα εκπαίδευσης, οι επιθυμητές έξοδοι και τα δεδομένα αξιολόγησης/πρόβλεψης. Τα δεδομένα εκπαίδευσης κάθε χρήστη αποτελούν τα ενωμένα διανύσματα (embeddings + κατηγορίες) των ταινιών που έχει βαθμολογήσει. Οι επιθυμητές έξοδοι είναι οι βαθμολογίες των ταινιών σε μορφή one-hot encoding ώστε να μπορεί να γίνει κατηγοριοποίηση μεταξύ τους. Το διάνυσμα εξόδου αποτελείται από 10 θέσεις, μία για κάθε βαθμό από το 0.5 έως το 5.0 με βήμα 0.5. Έχει τον αριθμό 1 στην αντίστοιχη θέση της βαθμολογίας και τον αριθμό 0 σε όλες τις άλλες θέσεις. Τα δεδομένα αξιολόγησης αποτελούν όλα τα υπόλοιπα ενωμένα διανύσματα, αυτά δηλαδή των ταινιών που ο χρήστης δεν έχει βαθμολογήσει και η βαθμολογία τους είναι προς πρόβλεψη.

Αφού δημιουργηθούν τα δεδομένα, κατασκευάζεται το νευρωνικό δίκτυο με τη βιβλιοθήκη keras. Το δίκτυο αποτελείται από ένα επίπεδο εισόδου με αριθμό νευρώνων εισόδου ίσο με το μέγεθος του ενωμένου διανύσματος, 2 κρυφά επίπεδα με 64 νευρώνες το καθένα, κι ένα επίπεδο εξόδου με 10 νευρώνες, όσες και οι πιθανές βαθμολογίες χρησιμοποιώντας τη συνάρτηση ενεργοποίησης softmax που χρησιμοποιείται συχνά για κατηγοριοποίηση σε πάνω από 2 ομάδες (10 στην προκειμένη περίπτωση). Γίνεται χρήση του adam optimizer και της συνάρτησης

κόστους categorical crossentropy γιατί γίνεται κατηγοριοποίηση μεταξύ πολλών κατηγοριών. Στην συνέχεια, εκπαιδεύεται το δίκτυο με τα δεδομένα εκπαίδευσης που υπολογίστηκαν για 100 εποχές, για κάθε χρήστη και υπολογίζεται η ακρίβεια της εκπαίδευσης. Η ακρίβεια κυμαίνεται στο 70% με 95% συνήθως. Πολύ καλύτερες επιδόσεις έχει αν δεν αφαιρεθούν οι ημερομηνίες και τα stop words από τους τίτλους, αλλά αφαιρούνται ώστε να συγκρίνονται μεταξύ τους μόνο οι πιο σημαντικές εννοιολογικά λέξεις. Έπειτα, γίνεται ο υπολογισμός των προβλέψεων βαθμολογιών για τα δεδομένα αξιολόγησης που υπολογίστηκαν. Οι προβλέψεις είναι σε μορφή διανυσμάτων. Κάθε θέση αντιστοιχεί σε μία πιθανότητα η ταινία που αξιολογείται να έχει μία από τις 10 βαθμολογίες, κάτι που γίνεται λόγω της συνάρτησης ενεργοποίησης softmax στο τελευταίο επίπεδο. Το άθροισμα των 10 πιθανοτήτων ισούται με το 1. Επιλέγεται για κάθε ταινία η μεγαλύτερη πιθανότητα, αντιστοιχίζεται στο ανάλογο βαθμό και αυτή είναι η πρόβλεψη του δικτύου που προκύπτει.

Αυτή η διαδικασία εκτελείται για κάθε χρήστη και δημιουργείται το νέο σύνολο δεδομένων “new_ratings4.csv” το οποίο περιέχει τις πραγματικές βαθμολογίες των χρηστών για τις ταινίες που είχαν βαθμολογήσει και τις προβλέψεις του δικτύου για όλες τις υπόλοιπες.

```
Using TensorFlow backend.
Πρόγραμμα πρόβλεψης βαθμολογιών με την χρήση των Word Embedings.
Φόρτωση απαραίτητων αρχείων...
Υπολογισμός Word Embeddings ταινιών...

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 1 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 1: 89.999998% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 1.

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 2 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 2: 82.894737% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 2.

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 3 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 3: 84.313726% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 3.

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 4 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 4: 76.960784% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 4.

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 5 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 5: 72.000003% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 5.

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 6 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 6: 68.181819% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 6.

Υπολογισμός υπόλοιπων βαθμολογιών για τον χρήστη: 7 ...
Ακρίβεια εκπαίδευσης για βαθμολογίες του χρήστη 7: 85.227275% (Εποχές: 100)
Οι υπόλοιπες βαθμολογίες υπολογίστηκαν για τον χρήστη 7.
```

Ενδεικτικά, το αρχείο “new_ratings4.csv” έχει αυτή τη μορφή, στα πρότυπα του αρχείου “ratings.csv”:

	A	B	C
1	userId,movieId,rating		
2	1,1,3.0		
3	1,2,2.0		
4	1,3,3.0		
5	1,4,3.0		
6	1,5,2.0		
7	1,6,2.0		
8	1,7,4.0		
9	1,8,3.0		
10	1,9,3.0		
11	1,10,2.0		
12	1,11,2.0		
13	1,12,3.0		
14	1,13,3.0		
15	1,14,1.0		
16	1,15,2.5		
17	1,16,2.0		
18	1,17,2.0		
19	1,18,2.0		
20	1,19,3.0		

Με αναζητήσεις στις βαθμολογίες, τους τίτλους, τις κατηγορίες και τις προβλέψεις φαίνεται ότι οι προβλέψεις ήταν επιτυχημένες σε μεγάλο βαθμό. Για παράδειγμα, ο χρήστης 1 είχε βαθμολογήσει την ταινία με id 31 και τίτλο “Dangerous Minds” με τον βαθμό 2.5. Το δίκτυο προέβλεψε για την ταινία “Hearts and Minds (1996)” με id 1423 βαθμολογία ίση με 3.0 για τον χρήστη 1, για την ταινία “Hearts and Minds (1974) με id 26325, βαθμολογία ίση με 3.5 που όμως ανήκει σε άλλες κατηγορίες από τις 2 προηγούμενες, αλλά και περιέχουν τις διαφορετικές λέξεις “dangerous” και “heart” που πιθανώς έχουν φέρει την διαφοροποίηση. Οι κατηγορίες παίζουν μεγάλο ρόλο στην πρόβλεψη γιατί εκπροσωπούνται από τον αριθμό 1, ενώ οι ομοιότητες με μικρούς δεκαδικούς αριθμούς στο διάνυσμα εισόδου. Παρόμοιες συγκρίσεις αποδεικνύουν τις όμοιες βαθμολογίες μεταξύ ταινιών με όμοιους τίτλους ή κατηγορίες.

4.2 – 2ο Πρόγραμμα ερωτήματος: Εφαρμογή νέας μετρικής χρησιμοποιώντας τις μέσες βαθμολογίες των συστάδων και τις προβλέψεις που προέκυψαν (task4.2.py)

Για το πρόγραμμα χρησιμοποιήθηκε η βιβλιοθήκη pandas για ευκολότερη αποθήκευση των δεδομένων, η sys για έξοδο σε περίπτωση αδυναμίας σύνδεσης με την Elasticsearch ή μη εύρεσης του αρχείου των βαθμολογιών και η elasticsearch μέσω της οποίας γίνεται η σύνδεση στην βάση.

Οδηγίες πριν την εκτέλεση:

- pip install elasticsearch
- pip install pandas

Η μετρική ακολουθεί το προηγούμενο μοντέλο με την προσθήκη της βαθμολογίας συστάδων από το αρχείο “new_ratings3.csv” και των προβλέψεων βαθμολογιών από το αρχείο “new_ratings4.csv” ταυτόχρονα. Τα βάρη των σκορ της elasticsearch, του χρήστη, της μέσης βαθμολογίας, της βαθμολογίας συστάδας και της πρόβλεψης βαθμολογίας είναι 1, 2, 1, 1.5 και 1.5 αντίστοιχα. Αν μία ταινία δεν έχει βαθμολογηθεί από κανέναν χρήστη, λαμβάνεται υπόψη μόνο το σκορ της elasticsearch. Αν η ταινία έχει βαθμολογηθεί, αλλά όχι από τον χρήστη, λαμβάνεται υπόψη το σκορ της elasticsearch, η μέση βαθμολογία, η βαθμολογία συστάδας και η πρόβλεψη βαθμολογίας. Αν ταινία έχει βαθμολογηθεί από τον χρήστη, λαμβάνεται υπόψη το σκορ της elasticsearch, του χρήστη και το μέσο σκορ. Στο τέλος το σκορ διαιρείται για τους λόγους που εξηγήθηκαν στο δεύτερο ερώτημα. Η φόρτωση των αρχείων “new_ratings3.csv”, “new_ratings4.csv” και η απουσία υπολογισμών έχει ως αποτέλεσμα τη γρήγορη παρουσίαση των αποτελεσμάτων στην οθόνη. Η μετρική πετυχαίνει τον στόχο της σύμφωνα με τη φιλοσοφία που αναλύθηκε στο ερώτημα 2. Ενδεικτικά, για τον χρήστη 45 και την λέξη “need” τα αποτελέσματα είναι τα εξής:

```
Γίνεται σύνδεση με την Elasticsearch...

Πληκτρολογήστε το αναγνωριστικό σας:45

Επιλέξτε φράση προς αναζήτηση:need

Εύρεση σκορ χρήστη, σκορ συστάδας, σκορ πρόβλεψης και μέσου σκορ...
Οι ταινίες που βρέθηκαν:
We Need to Talk About Kevin (2011)
Need for Speed (2014)
Mike & Dave Need Wedding Dates (2016)
```

Σημείωση: Για τα ερωτήματα 3, 4 έχουν δημιουργηθεί και εκδόσεις που στο ίδιο πρόγραμμα υπολογίζονται οι βαθμολογίες συστάδων και οι προβλέψεις του χρήστη, αντίστοιχα και εφαρμόζεται η μετρική. Προτιμήθηκε να σταλεί η υλοποίηση με τα αρχεία “new_ratings3.csv” και “new_ratings4.csv” για να ικανοποιείται η φράση του ερωτήματος 3: «στο αρχικό σύνολο δεδομένων θα συμπληρώσετε για κάθε χρήστη τις βαθμολογίες που του λείπουν». Οι υλοποιήσεις είναι ίδιας λογικής χωρίς καμία σχεδόν διαφορά.

Πηγές:

- Word2vec - Wikipedia
- Introduction to Word Embedding and Word2Vec | by Dhruvil Karani | Towards Data Science
- Free and Open Search: The Creators of Elasticsearch, ELK & Kibana | Elastic
- Creating Word Embeddings: Coding the Word2Vec Algorithm in Python using Deep Learning | by Eligijus Bujokas | Towards Data Science
- How to Develop Word Embeddings in Python with Gensim (machinelearningmastery.com)
- How to Use Word Embedding Layers for Deep Learning with Keras (machinelearningmastery.com)
- pandas - Python Data Analysis Library (pydata.org)
- Keras: the Python deep learning API
- Python Elasticsearch Client — Elasticsearch 7.10.1 documentation (elasticsearch-py.readthedocs.io)
- A gentle introduction to Doc2Vec. TL;DR | by Gidi Shperber | Wisio | Medium
- DOC2VEC gensim tutorial. Today I am going to demonstrate a… | by Deepak Mishra | Medium
- models.doc2vec – Doc2vec paragraph embeddings — gensim (radimrehurek.com)