Πανεπιστήμιο Ιωαννίνων,
Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Εξόρυξη Δεδομένων – ΜΥΕ012/ΠΛΕ 059
Χειμερινό Εξάμηνο, 2021

Δεύτερη Σειρά Ασκήσεων

Η προθεσμία για την δεύτερη σειρά ασκήσεων είναι την Κυριακή 9 Ιανουαρίου 11:55 μ.μ. Παραδώστε Notebooks με τον κώδικα και τις αναφορές. Για τις Ερωτήσεις 1 και 2, μπορείτε να παραδώσετε και pdf με την απόδειξη, ή φωτογραφίες από χειρόγραφα. Για καθυστερημένες υποβολές ισχύει η πολιτική στην σελίδα του μαθήματος. Η παράδοση θα γίνει μέσω του ecourse. Λεπτομέρειες στη σελίδα Ασκήσεις του μαθήματος. Η άσκηση είναι **ατομική**.

Ερώτηση 1

Έστω ένας $n \times m$ πίνακας A, και έστω $(a_1^T, a_2^T, \dots, a_n^T)$ τα διανύσματα-γραμμές του πίνακα A. Έστω v το πρώτο δεξί singular vector του πίνακα A. Στην τάξη αναφέραμε ότι το διάνυσμα v έχει την ιδιότητα ότι ελαχιστοποιεί το άθροισμα των τετραγώνων των αποστάσεων των διανυσμάτων $(a_1^T, a_2^T, \dots, a_n^T)$ από τις προβολές τους πάνω στο διάνυσμα v. Αποδείξτε αυτή την ιδιότητα.

Ερώτηση 2

Α. Μία power-law κατανομή ορίζεται ως $P(X = x) = (a - 1)x^{-a}$, όπου a είναι ο εκθέτης της κατανομής. Σας δίνεται ένα σύνολο από παρατηρήσεις $X = \{x_1, \dots, x_n\}$ που έχουν παραχθεί από μία power-law κατανομή. Χρησιμοποιήστε την Maximum Likelihood Estimation τεχνική που περιγράψαμε στην τάξη για να βρείτε τον εκθέτη της power-law κατανομής που ταιριάζει (fits) τα δεδομένα των παρατηρήσεων.

Β. Υποθέστε ότι οι παρατηρήσεις $X = \{x_1, \dots, x_n\}$ έχουν παραχθεί από ένα μείγμα δύο power-law κατανομών, L_1, L_2, με παραμέτρους a_1, a_2, και πιθανότητες μίξης (mixture probabilities) π_1, π_2. Θα χρησιμοποιήσουμε τον ΕΜ αλγόριθμο για να υπολογίσουμε τις παραμέτρους $\Theta = (a_1, a_2, \pi_1, \pi_2)$ του mixture μοντέλου, όπως κάναμε και για την περίπτωση της μίξης από Gaussian κατανομές. Στο Μ βήμα, υποθέτουμε ότι έχουμε τις πιθανότητες ανάθεσης $P(L_k|x_i)$, για $k = 1,2$ και $i = 1, \dots, n$, και θέλουμε να υπολογίσουμε τις παραμέτρους Θ. Δώστε τις εξισώσεις για τα a_1, a_2, π_1, π_2 και τους υπολογισμούς με τους οποίους τις παρήγατε.

Σημείωση: Θα σας βοηθήσει να διαβάσετε τις σημειώσεις του Άρη Αναγνωστόπουλου που είναι στην σελίδα του μαθήματος για την περίπτωση της μίξης των Gaussians, τις οποίες παρουσιάσαμε στο μάθημα.

Ερώτηση 3

Ο στόχος αυτής της άσκησης είναι να πειραματιστείτε με αλγορίθμους για συστήματα συστάσεων και να εξασκηθείτε στην διαχείριση πινάκων μέσα από τις βιβλιοθήκες numpy και scipy που έχουμε μάθει.

Θα χρησιμοποιήσετε το jokes dataset που μπορείτε να κατεβάσετε από εδώ. Θα χρησιμοποιήσουμε μόνο τα train δεδομένα. Στο train.csv έχουμε 1,092,059 βαθμολογίες από N = 40863 χρήστες σε M=139 αστεία. Στο jokes.csv έχουμε το κείμενο γι αυτά τα αστεία.

Βήμα 1: Το πρώτο βήμα της άσκησης είναι η επεξεργασία των δεδομένων. Φορτώστε τα δεδομένα με τις βαθμολογίες σε ένα dataframe. Τροποποιείστε τα user_id και joke_id αφαιρώντας 1, ώστε η αρίθμηση τους να ξεκινάει από το μηδέν και να μπορείτε να τα χρησιμοποιήσετε κατευθείαν ως index σε πίνακα. Πάρετε μια τυχαία αναδιάταξη των γραμμών του dataframe χρησιμοποιώντας την εντολή shuffle από τη βιβλιοθήκη sklearn.utilities, με παράμετρο random_state = 2021. Από το αναδιατεταγμένο dataframe, κρατήστε τις 10,000 πρώτες εγγραφές ως τα test δεδομένα, και τις υπόλοιπες ως τα train δεδομένα. Παρακαλώ ακολουθείστε πιστά τις οδηγίες σε αυτό το κομμάτι ώστε να έχουμε όλοι τα ίδια δεδομένα.

Ο στόχος μας είναι για κάθε ζευγάρι χρήστη-αστείου (u, j) στα test δεδομένα να υπολογίσουμε ένα score που είναι όσο πιο κοντά γίνεται στην πραγματική βαθμολογία του χρήστη για το αστείο. Για την αξιολόγηση θα χρησιμοποιήσετε το RMSE (Root Mean Square Error). Αν $r_1, r_2, \dots, r_n$ είναι τα ratings που θέλουμε να προβλέψουμε, και $p_1, p_2, \dots, p_n$είναι οι προβλέψεις ενός αλγορίθμου, το RMSE του αλγορίθμου ορίζεται ως

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_i - p_i)^2}$$

Βήμα 2: Οι δύο πρώτοι αλγόριθμοι που θα δοκιμάσουμε είναι ο **User Agerage (UA)** και ο **Joke Average (JA)**. Ο πρώτος για κάθε ζευγάρι (u, j) προβλέπει τη μέση βαθμολογία του χρήστη u, ενώ ο δεύτερος τη μέση βαθμολογία του αστείου j. Η μέση βαθμολογία του χρήστη υπολογίζεται μόνο από τα αστεία στα οποία έχει δώσει βαθμολογία, αλλά περιλαμβάνει και τα αστεία στα οποία έδωσε μηδενική βαθμολογία. Αντίστοιχα η μέση βαθμολογία ενός αστείου υπολογίζεται μόνο από τους χρήστες που το έχουν βαθμολογήσει, αλλά περιλαμβάνει και αυτούς που έχουν δώσει μηδενική βαθμολογία. Για το βήμα αυτό σας προτείνεται να κρατήσετε τα δεδομένα σε dataframes. Για να πάρετε όλους τους βαθμούς της άσκησης η υλοποίηση σας θα πρέπει να γίνει χρησιμοποιώντας εντολές της βιβλιοθήκης pandas, χωρίς να διατρέξετε τις γραμμές του dataframe με for loop. Εκτυπώστε το RMSE για τους δύο αλγορίθμους και κάνετε ένα σύντομο σχόλιο για το τι βλέπετε.

Βήμα 3: Ο επόμενος αλγόριθμος που θα εξετάσουμε είναι ο **User-Based Collaborative Filtering (UCF)**. Για την υλοποίηση του αλγορίθμου θα φορτώσετε το train dataset σε ένα αραιό πίνακα R. Για να πάρετε όλους τους βαθμούς της άσκησης, η υλοποίηση σας θα πρέπει να γινει με πράξεις πινάκων της numpy και scipy, χωρίς να διατρέξετε τις γραμμές ή τις στήλες του πίνακα με for loop, και κρατώντας τον πίνακα αραιό.

Ο UCF αλγόριθμος έχει μια παράμετρο k, που είναι ο αριθμός των όμοιων χρηστών που κοιτάει. Για να υπολογίσετε την τιμή για το ζευγάρι (u, j) υπολογίστε το σύνολο $N_k(u, j)$ με τους k πιο όμοιους χρήστες με τον χρήστη u οι οποίοι έχουν βαθμολογήσει το αστείο j. Στη συνέχεια χρησιμοποιήστε την εξής εξίσωση για την πρόβλεψη σας:

$$p(u, j) = \frac{\sum_{u' \in N_k(u,j)} s(u, u') r(u', j)}{\sum_{u' \in N_k(u,j)} s(u, u')}$$

Στην εξίσωση $s(u, u')$ είναι η ομοιότητα μεταξύ των χρηστών u και u'. Για την υλοποίηση σας θα χρησιμοποιήσετε το cosine similarity.

Η βασική ιδέα της υλοποίησης είναι η ακόλουθη:

Για κάθε ζευγάρι (u, j) στα test δεδομένα:

1. Βρείτε τους χρήστες που έχουν βαθμολογήσει το j
2. Υπολογίστε την ομοιότητα αυτών των χρηστών με τον χρήστη u και κρατήστε τους k πιο όμοιους χρήστες. Αν υπάρχουν λιγότεροι από k χρήστες που έχουν βαθμολογήσει το j, χρησιμοποιήστε τους όλους.
3. Πάρετε τις ομοιότητες και τις βαθμολογίες για τους k πιο όμοιους χρήστες, και υπολογίστε την βαθμολογία με την παραπάνω εξίσωση.

Όλοι οι παραπάνω υπολογισμοί μπορούν να γίνουν χρησιμοποιώντας μόνο μεθόδους διαχείρισης πινάκων και διανυσμάτων της numpy ή scipy. Για ευκολία ποθέστε ότι το αστείο j έχει μη μηδενικές βαθμολογίες.

Τρέξτε τον αλγόριθμο για $k = 10,25,50,75,100$, και φτιάξετε μια γραφική παράσταση με το RMSE ως συνάρτηση του k. Σχολιάστε το αποτέλεσμα και την τιμή η οποία δίνει τα καλύτερα αποτελέσματα. Μπορείτε να δοκιμάσετε περισσότερες τιμές αν θέλετε. Να έχετε υπόψη σας ότι λόγω του μεγάλου όγκου δεδομένων, μια εκτέλεση του αλγορίθμου παίρνει περίπου 5 λεπτά, οπότε αν θέλετε να κάνετε πολλά εξερευνητικά πειράματα χρησιμοποιήστε κάποιο sample των test δεδομένων. Τα αποτελέσματα σας θα πρέπει να είναι στο σύνολο των δεδομένων.

Βήμα 4: Υλοποιήστε τον αλγόριθμο **Item-Based Collaborative Filtering (ICF)**. Ο αλγόριθμος είναι ουσιαστικά ο ίδιος με αυτόν στο Βήμα 3, απλά δουλεύετε με τον ανάστροφο πίνακα και ανταλλάσσετε χρήστες και αστεία και ανάποδα. Παρακάτω είναι η περιγραφή του για πληρότητα:

Ο αλγόριθμος έχει μια παράμετρο k , που είναι ο αριθμός των όμοιων αστείων που θα κοιτάξει. Για να υπολογίσετε την τιμή ενός κελιού (u, j) υπολογίστε το σύνολο $N_k(j, u)$ με τα k πιο όμοια αστεία ως προς το j από αυτά που έχει βαθμολογήσει ο χρήστης u. Στη συνέχεια χρησιμοποιήστε την εξής εξίσωση για την πρόβλεψη σας:

$$p(u, j) = \frac{\sum_{j' \in N_k(j,u)} s(j, j') r(u, j')}{\sum_{j' \in N_k(j,u)} s(j, j')}$$

Στην εξίσωση $s(j, j')$ είναι η ομοιότητα μεταξύ των αστείων j και j'. Για την υλοποίηση σας θα χρησιμοποιήσετε το cosine similarity.

Η βασική ιδέα της υλοποίησης είναι η ακόλουθη:

Για κάθε ζευγάρι (u, j) στα test:

1. Βρείτε τα αστεία που έχει βαθμολογήσει ο χρήστης u
2. Υπολογίστε την ομοιότητα αυτών των αστείων με το αστείο j και κρατήστε τα k πιο όμοια. Αν υπάρχουν λιγότερα από k αστεία που έχει βαθμολογήσει ο u, χρησιμοποιήστε τα όλα.
3. Πάρετε τις ομοιότητες και τις βαθμολογίες για τα k πιο όμοια αστεία, και υπολογίστε την βαθμολογία με την παραπάνω εξίσωση.

Όλοι οι παραπάνω υπολογισμοί μπορούν να γίνουν χρησιμοποιώντας μόνο μεθόδους διαχείρισης πινάκων και διανυσμάτων της numpy ή scipy. Για ευκολία, υποθέστε ότι ο χρήστης u έχει δώσει μη μηδενικές βαθμολογίες.

Τρέξτε τον αλγόριθμο για $k = 1,2,5,7,10$, και φτιάξετε μια γραφική παράσταση με το RMSE ως συνάρτηση του k. Σχολιάστε το αποτέλεσμα και την τιμή η οποία δίνει τα καλύτερα αποτελέσματα. Μπορείτε να δοκιμάσετε περισσότερες τιμές αν θέλετε. Να έχετε υπόψη σας ότι λόγω του μεγάλου όγκου δεδομένων, μια εκτέλεση του αλγορίθμου παίρνει περίπου 5 λεπτά, οπότε αν θέλετε να κάνετε πολλά εξερευνητικά πειράματα χρησιμοποιήστε κάποιο sample των test δεδομένων. Τα αποτελέσματα σας θα πρέπει να είναι στο σύνολο των δεδομένων.

Βήμα 5: Εφαρμόστε το **Singular Value Decomposition (SVD)** στον πίνακα R, και κρατήστε τα k μεγαλύτερα singular vectors για να πάρετε ένα rank-k πίνακα R_k. Στη συνέχεια χρησιμοποιήστε την τιμή $p(u,j) = R_k(u,j)$ για την πρόβλεψη σας.

Δοκιμάστε όλες τις τιμές για το k από το 2 εώς 20 και φτιάξετε μια γραφική παράσταση με το RMSE ως συνάρτηση του k. Σχολιάστε το αποτέλεσμα και την τιμή η οποία δίνει τα καλύτερα αποτελέσματα.

Βήμα 6: Αφού ολοκληρώσετε τα Βήματα 1-5, κάνετε μια συγκριτική αξιολόγηση των αλγορίθμων. Φτιάξετε ένα πίνακα που να έχει όλους τους αλγορίθμους μαζί, και το καλύτερο error που επιτυγχάνει ο κάθε αλγόριθμος, και σχολιάστε τα αποτελέσματα. Μπορείτε στο σχολιασμό σας να λάβετε υπόψη και του χρόνους εκτέλεσης (θα πρέπει να τους αναφέρετε).

Επίσης, βαθμολογήστε μόνοι σας τουλάχιστον 10 αστεία τα οποία θα χρησιμοποιήσετε για να δοκιμάσετε τα μοντέλα σας, και τουλάχιστον 1 το οποίο θα χρησιμοποιήσετε για τεστ. Αναφέρετε τα αποτελέσματα. Εφόσον έχετε πλήρη έλεγχο του πειράματος μπορείτε να δώσετε και μια ποιοτική ανάλυση για τα αποτελέσματα.

Παραδώσετε ένα notebook με τον κώδικα σας και την αναφορά με τις παρατηρήσεις σας για τα αποτελέσματα. Στο notebook το κάθε βήμα θα πρέπει να έχει δική του επικεφαλίδα.

Bonus: Υλοποιήσετε και τεστάρετε και την παραλλαγή του UCF που προβλέπει τις αποκλίσεις από την μέση τιμή. Στην περίπτωση αυτή , θα χρησιμοποιήστε την εξής εξίσωση για την πρόβλεψη σας:

$$p(u,j) = \overline{r(u)} + \frac{\sum_{u' \in N_k(u,j)} s(u,u')(r(u',j) - \overline{r(u')})}{\sum_{u' \in N_k(u,j)} s(u,u')}$$

Για την ομοιότητα χρησιμοποιήστε το correlation coefficient (το cosine similarity, μετά την αφαίρεση της μέσης τιμής από την κάθε γραμμή). Για να αφαιρέσετε την μέση τιμή, θα σας είναι πιο βολικό να δουλέψετε με το dataframe αντί για τον πίνακα.

Υποδείξεις

Θα σας βολέψει τους UCF και ICF αλγορίθμους να τους ορίσετε σαν συναρτήσεις που παίρνουν σαν όρισμα τον πίνακα, το k, και το ζευγάρι u, j.

Οι παρακάτω συναρτήσεις μπορεί να σας φανούν χρήσιμες:

- Οι πράξεις μεταξύ πινάκων και διανυσμάτων με τη βιβλιοθήκη numpy μερικές φορές επιστρέφουν διανύσματα που μπορεί να έχουν διαφορετική μορφή απ ότι θα θέλατε οπότε θα πρέπει να είσαστε προσεκτικοί. Οι μέθοδοι reshape και flatten μπορεί να σας βοηθήσουν.
- Αν θέλετε να πάρετε τον αριθμό των τιμών σε μια γραμμή ή στήλη ενός αραιού πίνακα, μπορείτε να χρησιμοποιήσετε την getnnz. Επίσης η εντολή argwhere σας βοηθάει να βρείτε τις θέσεις που εμφανίζεται μια τιμή. Η μέθοδος nonzero σας δίνει τις μη μηδενικές τιμές (αλλά αγνοεί τις περιπτώσεις που υπάρχει μια τιμή αλλά είναι μηδενική).
- Για το RMSE μπορείτε να χρησιμοποιήσετε τη μέθοδο sklearn.mean_squared_error.

Ερώτηση 4

Στην ερώτηση αυτή θα χρησιμοποιήσετε τον αλγόριθμο k-means για το πρόβλημα στην Ερώτηση 3. Η ερώτηση έχει δύο μέρη:

Α. Στο μέρος αυτό θα εφαρμόσετε τον k-means αλγόριθμο πάνω στον user-joke πίνακα R. Δοκιμάστε τιμές του k από 2 εώς 10 και δημιουργείστε το συνδυασμένο διάγραμμα με το SSE και το Silhouette Coefficient για να αποφασίσετε ποιος είναι ο "σωστός" αριθμός από clusters. Σχολιάστε το γράφημα και την απόφαση σας.

Για το k που θα διαλέξετε θα εξετάσετε δύο παραλλαγές των αλγορίθμων συστάσεων που υλοποιήσατε στην Ερώτηση 3.

1. **Cluster-Based JA (CB-JA):** Για κάθε cluster c_i, και για κάθε αστείο j, υπολογίστε την μέση τιμή της βαθμολογίας των χρηστών στο cluster c_i για το αστείο j. Για ένα ζευγάρι (u, j) στα test δεδομένα, βρείτε το cluster που ανήκει ο χρήστης u, και προβλέψτε τη μέση τιμή του cluster για το αστείο j. Αν κανείς δεν έχει βαθμολογήσει το αστείο j στο cluster c_i επιστρέψετε την γενική μέση βαθμολογία του αστείου j.
2. **Cluster-Based UCF (CB-UCF):** Για κάθε cluster c_i δημιουργείστε ένα αραιό υποπίνακα R_i με τις γραμμές που ανήκουν στο cluster c_i. Για ένα ζευγάρι (u, j) στα test δεδομένα, βρείτε το cluster που ανήκει ο χρήστης u, και τρέξτε τον UCF αλγόριθμο γι αυτό το cluster. Αν κανείς δεν έχει βαθμολογήσει το αστείο j στο cluster c_i επιστρέψετε την γενική μέση βαθμολογία του αστείου j

Τρέξτε τους αλγορίθμους στα test δεδομένα από την Ερώτηση 3 και αναφέρετε τα αποτελέσματα σας. Συγκρίνετε με τα αποτελέσματα στην Ερώτηση 3.

Β. Σε αυτό το μέρος θα κάνουμε cluster τα αστεία χρησιμοποιώντας το κείμενο τους. Φορτώστε το κείμενο από το αρχείο jokes.csv. Χρησιμοποιήστε ένα tf-idf vectorizer για να πάρετε διανύσματα για τα αστεία (αφαιρέστε τα stop-words) και εφαρμόστε τον k-means αλγόριθμο. Όπως και στο Α, χρησιμοποιήστε το συνδυασμένο γράφημα με το SSE και το Silhouette coefficient για να αποφασίσετε την τιμή του k.

Για το k που θα διαλέξετε θα εξετάσετε την παρακάτω παραλλαγή του UA αλγορίθμου που υλοποιήσατε στην Ερώτηση 3:

Cluster-Based UA (CB-UA): Για κάθε cluster c_i, και για κάθε χρήστη u, υπολογίστε την μέση τιμή της βαθμολογίας του χρήστη u για τα αστεία στο cluster c_i. Για ένα ζευγάρι (u, j) στα test δεδομένα, βρείτε το cluster που ανήκει το αστείο j, και προβλέψτε τη μέση τιμή του χρήστη u για το cluster. Αν ο u δεν έχει βαθμολογήσει κανένα αστείο στο cluster c_i επιστρέψετε την γενική μέση βαθμολογία του χρήστη.

Τρέξτε τον αλγόριθμο στα test δεδομένα από την Ερώτηση 3 και αναφέρετε τα αποτελέσματα σας. Συγκρίνετε με τα αποτελέσματα στην Ερώτηση 3.

Παραδώσετε ένα notebook με τον κώδικα σας και την αναφορά με τον σχολιασμό και ανάλυση των αποτελεσμάτων σας.