Δεύτερη Σειρά Ασκήσεων

Η προθεσμία για την δεύτερη σειρά ασκήσεων είναι την Παρασκευή 12 Ιανουαρίου, 11:55 μ.μ. Παραδώστε Notebooks με τον κώδικα και τις αναφορές σε .ipynb **και** .html μορφή. Για την Ερώτηση 1 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

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.

Υπόδειξη: 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 2

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, καθώς και με μια απλή χρήση του clustering.

Θα χρησιμοποιήσετε τα δεδομένα που χρησιμοποιήσατε και στην πρώτη σειρά ασκήσεων. Θα εξετάσουμε χρήστες οι οποίοι βαθμολογούν κινητά τηλέφωνα. Μπορείτε να κατεβάσετε τις τριάδες reviewerID, asin, overall, από το αρχείο cell_phone_ratings.csv στη σελίδα του μαθήματος.

Βήμα 1: Το πρώτο βήμα της άσκησης είναι η επεξεργασία των δεδομένων. Θα κρατήσουμε μόνο χρήστες που έχουν βαθμολογήσει τουλάχιστον 5 κινητά, και κινητά που έχουν βαθμολογηθεί από τουλάχιστον 5 χρήστες. Υλοποιήσετε ένα *επαναληπτικό* κλάδεμα χρηστών και ταινιών μέχρι να ικανοποιείται αυτή η συνθήκη. Μπορείτε να υλοποιήσετε το κλάδεμα όπως θέλετε (με pandas, ή με απλές δομές). Η υλοποίηση σας θα πρέπει να είναι αποδοτική, άρα θα πρέπει να χρησιμοποιήσετε τις σωστές δομές και μεθόδους. Το αποτέλεσμα είναι 2969

τριάδες με N = 389 χρήστες και M = 177 κινητά. Σας δίνεται το αποτέλεσμα αυτής της επαναληπτικής διαδικασίας στο αρχείο cell_phone_ratings_pruned.csv στη σελίδα του μαθήματος.

Στη συνέχεια θα δημιουργήσετε τα train και test δεδομένα. Από το αποτέλεσμα του προηγούμενου βήματος διαλέξτε τυχαία ένα 10% των τριάδων. Αυτά θα είναι το test set. Τα υπόλοιπα 90% θα είναι το training set. Υλοποιήστε την τυχαία επιλογή όπως προτιμάτε. Για το αποτέλεσμα της επιλογής θα χρησιμοποιήσετε τα έτοιμα αρχεία train.csv και test.csv, ώστε όλοι να δουλεύουμε με τα ίδια δεδομένα.

Στα επόμενα βήματα θα υλοποιήσετε και θα τεστάρετε διαφορετικούς αλγόριθμους συστάσεων. Ο στόχος μας είναι για κάθε ζευγάρι χρήστη-κινητου (u, c) στα test δεδομένα να υπολογίσουμε ένα score που είναι όσο πιο κοντά γίνεται στην πραγματική βαθμολογία του χρήστη για το αντικείμενο. 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 **Item Average (IA)**. Ο πρώτος για κάθε ζευγάρι (u, c) προβλέπει τη μέση βαθμολογία του χρήστη u, ενώ ο δεύτερος τη μέση βαθμολογία του αντικειμένου (κινητού) c. Η μέση βαθμολογία του χρήστη υπολογίζεται μόνο από τα αντικείμενα τα οποία έχει βαθμολογήσει. Αντίστοιχα η μέση βαθμολογία ενός κινητού υπολογίζεται μόνο από τους χρήστες που το έχουν βαθμολογήσει. 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. Υπολογίστε και αναφέρετε το RMSE για τους δύο αλγορίθμους.

Βήμα 3: Ο επόμενος αλγόριθμος που θα δοκιμάσετε είναι ο **Singular Value Decomposition (SVD)**. Για την υλοποίηση του αλγορίθμου θα φορτώσετε το train dataset σε ένα αραιό user-business πίνακα R. Για τον σκοπό αυτό θα χρειαστεί να κάνετε μια αντιστοίχιση των reviewerIDs στα νούμερα από 0 εώς N-1 και αντίστοιχα για τα asins. Στον αλγόριθμο υπολογίζουμε ένα rank-k R_k approximation του πίνακα R αυτό για διάφορες τιμές του k, και για το ζεύγος (u, c) η πρόβλεψη είναι η τιμή $R_k[u, c]$.

Υπολογίστε το Singular Value Decomposition για $K = 20$. Στη συνέχεια για διαφορετικά $k \in [1, K]$ θα υπολογίσετε τις τιμές $R_k[u, c]$ για τα ζευγάρια (u, c) στα test δεδομένα. Δεν θα δημιουργήσετε τον πίνακα R_k ο οποίος είναι πυκνός. Η τιμή $R_k[u, c]$ μπορεί να υπολογιστεί ως

$$R_k[u, c] = U_k[u, :]\Sigma_k V_k^T[:, c]$$

Όπου $U_k[u, :]$ είναι η γραμμή u του πίνακα U_k με τα k πρώτα left singular vectors, Σ_k είναι ο διαγώνιος πίνακας με τα k πρώτα singular values και $V_k^T[:, c]$ είναι η στήλη c του ανάστροφου πίνακα V_k με τα k πρώτα right singular vectors.

Υπολογίστε το RMSE για κάθε $k \in [1, K]$ και κάνετε μια γραφική παράσταση του RMSE ως προς το k. Αναφέρετε την τιμή του k που ο αλγόριθμος πετυχαίνει το ελάχιστο RMSE και ποιο είναι αυτό.

Σημείωση: Σε κανένα σημείο δεν θα πρέπει να δημιουργήσετε ένα πυκνό πίνακα $N \times M$. Αν η βαθμολογία που υπολογίζει ο αλγόριθμος είναι μεγαλύτερη από 5 ή μικρότερη από μηδέν κάνετε "clip" την βαθμολογία σε 5 και 0 αντίστοιχα.

Βήμα 4: Στη συνέχεια θα υλοποιήσετε τον **User-Based Collaborative Filtering (UCF)** αλγόριθμο. Για το σκοπό αυτό θα δημιουργήσετε πάλι τον αραιό user-business πίνακα R. Ο UCF αλγόριθμος έχει μια παράμετρο k, που είναι ο αριθμός των όμοιων χρηστών που κοιτάει. Για να υπολογίσετε την τιμή για ένα ζευγάρι (u, c) υπολογίστε το σύνολο $N_k(u, c)$ με τους k πιο όμοιους χρήστες με τον χρήστη u οι οποίοι έχουν βαθμολογήσει το αντικείμενο c. Στη συνέχεια χρησιμοποιήστε την εξής εξίσωση για την πρόβλεψη σας:

$$p(u, c) = \frac{\sum_{u' \in N_k(u,c)} s(u, u') R[u', c]}{\sum_{u' \in N_k(u,c)} s(u, u')}$$

Στην εξίσωση $s(u, u')$ είναι το cosine similarity μεταξύ των χρηστών u και u'.

Η βασική ιδέα της υλοποίησης είναι η ακόλουθη:

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

1. Βρείτε τους χρήστες που έχουν βαθμολογήσει το αντικείμενο c.
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 αντικείμενο c, χρησιμοποιήστε τους όλους.
3. Χρησιμοποιείστε τα διανύσματα με τις ομοιότητες και τις βαθμολογίες για τους k πιο όμοιους χρήστες, και υπολογίστε την βαθμολογία με την παραπάνω εξίσωση.

Δημιουργήστε μια συνάρτηση η οποία για ένα ζευγάρι (u, b), για μια λίστα από τις τιμές για το k υπολογίζει τις βαθμολογίες για το (u, b) για όλες τις τιμές του k. Για αρχή, μπορείτε να υπολογίσετε μόνο για ένα συγκεκριμένο k, και μετά να την επεκτείνετε για μια λίστα με τιμές για το k. Η σωστή υλοποίηση της συνάρτησης αυτής είναι ένα μεγάλο κομμάτι αυτής της υποερώτησης. Για να πάρετε όλους τους βαθμούς της άσκησης, η συνάρτηση αυτή πρέπει να υλοποιηθεί χρησιμοποιώντας μόνο μεθόδους διαχείρισης πινάκων και διανυσμάτων της numpy ή scipy. Σε κανένα σημείο του προγράμματος δεν πρέπει να υπολογίσετε κάποιο πυκνό πίνακα $N \times M$, και σε κανένα σημείο δεν πρέπει να χρησιμοποιήσετε for loop για να διατρέξετε τους χρήστες ή τα businesses στα training data.

Τρέξτε τον αλγόριθμο για k από 1 έως 20 και φτιάξετε μια γραφική παράσταση με το RMSE ως συνάρτηση του k. Δώστε την τιμή η οποία δίνει τα καλύτερα αποτελέσματα, και σχολιάστε τα αποτελέσαμτα. Μπορείτε να δοκιμάσετε περισσότερες τιμές αν θέλετε. Αν έχετε υλοποιήσει τη μέθοδο που υπολογίζει score για όλες τις τιμές του k, ένα τρέξιμο του αλγορίθμου θα πάρει 2-3 λεπτά. Είναι πιο αργό αν καλείτε τη μέθοδο για διαφορετικό k κάθε φορά. Λάβετε υπόψη σας το υπολογιστικό κόστος του αλγορίθμου στην υλοποίηση σας. Αν θέλετε να κάνετε πολλά εξερευνητικά πειράματα χρησιμοποιήστε κάποιο sample των test δεδομένων. Τα αποτελέσματα που θα αναφέρετε θα πρέπει να είναι στο σύνολο των δεδομένων.

Βήμα 5: Υλοποιήστε τον αλγόριθμο **Item-Based Collaborative Filtering (ICF)**. Ο αλγόριθμος είναι πολύ παρόμοιος με αυτόν στο Βήμα 4, απλά δουλεύετε με στήλες αντί για γραμμές.

Θα χρησιμοποιήσετε τον πίνακα R όπως στο Βήμα 4. Ο αλγόριθμος έχει μια παράμετρο k , που είναι ο αριθμός των όμοιων αντικειμένων που θα κοιτάξει. Για να υπολογίσετε την τιμή ενός κελιού (u, c) υπολογίστε το σύνολο $N_k(c, u)$ με τα k πιο όμοια αντικείμενα ως προς το c από αυτά που έχει βαθμολογήσει ο χρήστης u. Στη συνέχεια χρησιμοποιήστε την εξής εξίσωση για την πρόβλεψη σας:

$$p(u, c) = \frac{\sum_{c' \in N_k(c,u)} s(c, c') R[u, c']}{\sum_{c' \in N_k(c,u)} s(c, c')}$$

Στην εξίσωση $s(c, c')$ είναι το cosine similarity μεταξύ των αντικειμένων c και c'.

Η βασική ιδέα της υλοποίησης είναι η ακόλουθη:

Για κάθε ζευγάρι (u, c) στα test:

1. Βρείτε τα αντικείμενα που έχει βαθμολογήσει ο χρήστης u.
2. Υπολογίστε την ομοιότητα αυτών των αντικειμένων με το αντικείμενο c και κρατήστε τα k πιο όμοια. Αν υπάρχουν λιγότερα από k αντικείμενα που έχει βαθμολογήσει ο u, χρησιμοποιήστε τα όλα.
3. Χρησιμοποιείστε τα διανύσματα με τις ομοιότητες και τις βαθμολογίες για τα k πιο όμοια αντικείμενα, και υπολογίστε την βαθμολογία με την παραπάνω εξίσωση.

Όπως και πριν θα υλοποιήσετε μια μέθοδο που για ένα ζευγάρι θα υπολογίζει όλες τις βαθμολογίες για όλες τις τιμές του k. Η υλοποίηση σας θα γίνει χρησιμοποιώντας μόνο μεθόδους διαχείρισης πινάκων και διανυσμάτων της numpy ή scipy. Δεν πρέπει ποτέ να διατρέξετε τα αντικείμενα ή τους χρήστες στα training δεδομένα, και δεν πρέπει να υλοποιήσετε ένα πυκνό πίνακα $N \times M$.

Τρέξτε τον αλγόριθμο για k από 1 έως 20 και φτιάξετε μια γραφική παράσταση με το RMSE ως συνάρτηση του k. Αναφέρετε το αποτέλεσμα και την τιμή η οποία δίνει τα καλύτερα αποτελέσματα. Μπορείτε να δοκιμάσετε περισσότερες τιμές αν θέλετε.

Βήμα 6: Σε αυτό το βήμα θα χρησιμοποιήσετε clustering για τον υπολογισμό των ratings και θα υλοποιήσετε ένα αλγόριθμο που ονομάζουμε **Cluster-Average (CA).** Εφαρμόσετε τον k-means αλγόριθμο πάνω στον πίνακα R. Δοκιμάστε τιμές του k από 2 έως 50 και δημιουργείστε το συνδυασμένο διάγραμμα με το SSE και το Silhouette Coefficient όπως είδαμε στο φροντιστήριο για να αποφασίσετε ποιος είναι ο "σωστός" αριθμός από clusters. Σχολιάστε το γράφημα και την απόφαση σας.

Στη συνέχεια, για ένα γκρουπ (cluster) g από χρήστες, για κάθε αντικείμενο c θα υπολογίσετε την μέση τιμή $R(g, c)$ των βαθμολογιών των χρηστών στο g για το αντικείμενο c, από αυτούς που έχουν βαθμολογήσει το αντικείμενο c, εφόσον υπάρχουν τέτοιοι χρήστες. Επίσης θα υπολογίσετε την μέση τιμή $R(g)$ των ratings των χρηστών στο g. Εξετάστε τον αλγόριθμο όπου για το ζεύγος (u, c) βρίσκει το cluster g στο οποίο ανήκει ο χρήστης u και επιστρέφει την τιμή $R(g, c)$ αν είναι μη μηδενική, αλλιώς την τιμή $R(g)$.

Αναφέρετε τα αποτελέσματα του αλγόριθμου σας. Δοκιμάστε και άλλες τιμές του k για να δείτε αν μπορείτε να βελτιώσετε την απόδοση του αλγορίθμου, και κάνετε μια γραφική παράσταση με τα αποτελέσματα σας. Όλοι οι υπολογισμοί θα πρέπει να γίνουν με χρήση Pandas ή NumPy/SciPy.

Bonus: Εφαρμόστε τον k-means αλγόριθμο για μεγάλο αριθμό από clusters (μεταξύ 50 και 100) και δημιουργείστε τον cluster-item πίνακα, με τις τιμές $R(g, c)$ όπως περιγράφουμε παραπάνω. Στη συνέχεια εφαρμόστε τον UCF αλγόριθμο, με την διαφορά ότι πλέον για ένα ζευγάρι (u, c) θα υπολογίζετε ομοιότητα μεταξύ του χρήστη u και των clusters στα δεδομένα..

Βήμα 7: Αφού ολοκληρώσετε τα Βήματα 1-6, κάνετε μια συγκριτική αξιολόγηση των αλγορίθμων. Φτιάξετε ένα πίνακα που να έχει όλους τους αλγορίθμους, και το καλύτερο error που επιτυγχάνει ο κάθε αλγόριθμος, και σχολιάστε 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.

Υποδείξεις

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.
- Η μέθοδος nonzero σας δίνει τις μη μηδενικές τιμές σε ένα numpy διάνυσμα ή πίνακα.
- Η εντολή argsort της numpy σας δίνει την σειρά των indices ενός διανύσμτος όπως προκύπτει μετά την ταξινόμηση των τιμών του.
- Για το RMSE μπορείτε να χρησιμοποιήσετε τη μέθοδο sklearn.mean_squared_error.