9. Μετρικές Αποτίμησης (Evaluation Metrics)

***Σύνοψη**: Στο ένατο κεφάλαιο θα περιγράψουμε αναλυτικά τους τρόπους **αποτίμησης της αποτελεσματικότητας** (μετρική **ακρίβειας**, **ανάκλησης** κ.λπ.) των αλγορίθμων συστάσεων. Πιο συγκεκριμένα, στην εισαγωγή θα αναλύσουμε τα χαρακτηριστικά της **off-line** και της **online αξιολόγησης**. Ειδικά για την off-line αξιολόγηση, θα εξειδικεύσουμε τον διαχωρισμό των δεδομένων σε **εκπαίδευσης** και **ελέγχου**. Επιπρόσθετα, θα μελετήσουμε τις μετρικές **Mean Absolute Error** και **Root Mean Absolute Error** για την πρόβλεψη των βαθμολογιών των χρηστών επί των στοιχείων. Επίσης, θα παρουσιάσουμε (με τη χρήση παραδειγμάτων) τις μετρικές **Confusion Matrix**, **Precision**, **Recall**, **Normalized Discounted Cumulative Gain** και **Receiver Operating Characteristic Curve** (**ROC curve**), οι οποίες ειδικεύονται στην αποτίμηση της αποτελεσματικότητας της λίστας προτεινόμενων στοιχείων. Τέλος, θα εξετάσουμε και **μετρικές πέραν της απόλυτης ακριβείας** (**beyond accuracy metrics**), όπως τη μετρική της **επεξηγησιμότητας** (**explainability**) και τη μετρική της **πρωτοτυπίας** (**novelty**) των στοιχείων της λίστας συστάσεων. Με την πρώτη αξιολογούμε κατά πόσο ένα προτεινόμενο στοιχείο μπορεί να **επεξηγηθεί επαρκώς** στον υπό εξέταση χρήστη, ενώ με τη δεύτερη αξιολογούμε το κατά πόσο το στοιχείο που προτείνεται είναι **πρωτότυπο** (**novel**) ως προς τα ενδιαφέροντα του χρήστη βάσει πάντα των πρότερων αλληλεπιδράσεών του με το σύστημα συστάσεων.*

***Προαπαιτούμενη γνώση**: Συστήνουμε στον αναγνώστη να έχει μελετήσει προηγουμένως το δεύτερο και το τέταρτο κεφάλαιο.*

9.1 Εισαγωγή

Τα συστήματα συστάσεων αξιολογούνται με online ή off-line μεθόδους. Αυτό που συμβαίνει στην πράξη είναι ότι οι εταιρείες τρέχουν off-line πειράματα, καταρχάς σε ερευνητικό επίπεδο, προκειμένου να εντοπίσουν εκείνον τον αλγόριθμο που υπερέχει έναντι όλων των άλλων που συμμετέχουν στην πειραματική αξιολόγηση. Στη συνέχεια, αυτός ο αλγόριθμος που προκρίθηκε από

την off-line αξιολόγηση συγκρίνεται -online από real time χρήστες- με τον υπάρχοντα επίσημο αλγόριθμο που υποστηρίζει την υπηρεσία της εταιρείας στο επίπεδο της "βιομηχανικής παραγωγής" προτεινόμενων στοιχείων για τους χρήστες.

Στην online, τώρα, αξιολόγηση, οι αντιδράσεις των χρηστών μετρώνται σε σχέση με τις συστάσεις που τους προβάλλονται. Παραδείγματος χάρη, σε μια διαδικτυακή αξιολόγηση ενός συστήματος σύστασης ταινιών θα μπορούσε να μετρηθεί το ποσοστό των χρηστών που κάνουν κλικ στις ταινίες που τους προτείνονται (*click through rate*). Το *click through rate (CTR)* λοιπόν είναι ο λόγος του αριθμού των κλικ που γίνονται στα προτεινόμενα στοιχεία προς τον αριθμό των προβολών τους. Επίσης, η σύγκριση των δύο αλγορίθμων γίνεται με τη χρήση μιας μεθόδου δοκιμής που είναι γνωστή ως *A/B testing*, το σκεπτικό της οποίας εκτίθεται παρακάτω:

- Αρχικά, διαχωρίζονται οι real time χρήστες σε δύο ομάδες, *A* και *B*.
- Έπειτα, χρησιμοποιείται ο ένας αλγόριθμος για την ομάδα Α και ο άλλος αλγόριθμος για την ομάδα Β για ένα καθορισμένο χρονικό διάστημα.
- Στο τέλος της διαδικασίας συγκρίνεται το *click trough rate (CTR)* των δύο ομάδων.

Ο λόγος για τον οποίο χρησιμοποιείται το *CTR* είναι το ότι σχετίζεται με τα έσοδα που παράγονται από τη διαδικτυακή διαφήμιση (δηλ. *πληρωμή ανά κλικ σε σχέση με τις προβολές*), που είναι και ένας τρόπος να τεκμηριωθεί η απόδοση ενός συστήματος συστάσεων, δηλαδή μέσω των εσόδων που παράγει ή μέσω άλλων στόχων δέσμευσης των χρηστών, όπως τα *κλικ* του χρήστη, τα *like* του χρήστη αλλά και ο *χρόνος παραμονής* του στην υπηρεσία. Για να μετρήσουμε λόγου χάρη τη δέσμευση των χρηστών των μέσων κοινωνικής δικτύωσης (π.χ. *Facebook, Instagram* κ.λπ.) σε μια *ανάρτηση* (*post*) που έκανε μια επιχείρηση, μπορούμε να υπολογίσουμε τον λόγο του αθροίσματος των σχολίων, των *κλικ* και των *like* στην εν λόγω ανάρτηση προς τον αριθμό προβολών της.

Στο κεφάλαιο λοιπόν αυτό θα ασχοληθούμε κυρίως με τη μεθοδολογία αποτίμησης που ακολουθείται στην off-line αξιολόγηση. Σ' α*υτήν την περίπτωση βασικό προαπαιτ*ούμενο είναι να υπάρχει ικανοποιητικός αριθμός αλληλεπιδράσεων ή βαθμολογιών των χρηστών στα αρχεία της υπηρεσίας του συστήματος συστάσεων κατά το παρελθόν. Επομένως, από τον δισδιάστατο πίνακα βαθμολογίας χρήστη-στοιχείου επιλέγουμε ένα υποσύνολο των βαθμολογιών του πίνακα ανά χρήστη για την εκπαίδευση ενός μοντέλου πρόβλεψης, και οι υπόλοιπες βαθμολογίες ανά χρήστη χρησιμοποιούνται για την αξιολόγηση της λίστας προτεινόμενων στοιχείων ως προς την ακρίβειά της.

Τονίζουμε ότι η προσέγγιση είναι παρόμοια με εκείνη που χρησιμοποιείται για την αξιολόγηση *αλγορίθμων κατηγοριοποίησης*. Η κύρια διαφορά εδώ είναι ότι οι *μέθοδοι κατηγοριοποίησης* επιλέγουν ολόκληρες γραμμές που αφορούν σ' ένα στιγμιότυπο δεδομένων (π.χ. όλες τις βαθμολογίες ενός χρήστη), αντί να επιλέγουν μεμονωμένες βαθμολογίες για την εκπαίδευση ενός μοντέλου πρόβλεψης. Η διαφορά αυτή οφείλεται στο γεγονός ότι οι καταχωρίσεις εξαρτώνται πάντα από την *κλάση*

που ανήκουν στην *κατηγοριοποίηση*, ενώ καμμία καταχώριση του πίνακα βαθμολογιών χρήστη-στοιχείου δεν είναι εξαρτώμενη από κάποια μεταβλητή κλάσης. Με απλά λόγια, στα συστήματα συστάσεων δεν εξαιρούμε απο τη δειγματοληψία των *δεδομένων εκπαίδευσης* (*train data*) κανέναν χρήστη. Αυτό που κάνουμε είναι ότι για κάθε χρήστη επιλέγουμε τυχαία έναν αριθμό βαθμολογιών του για την εκπαίδευση του μοντέλου, ενώ οι βαθμολογίες που απομένουν ανά χρήστη είναι αυτές που θα προσπαθήσουμε να προβλέψουμε και θα αποτελέσουν τα *δεδομένα ελέγχου* (*test data*).

Στην προηγούμενη παράγραφο περιγράψαμε τον διαχωρισμό των δεδομένων σε train (εκπαίδευσης) και test (ελέγχου). Παρόλα αυτά, ένα συνηθισμένο σφάλμα που κάνουν οι ερευνητές κατά τη συγκριτική αξιολόγηση των συστημάτων συστάσεων είναι να χρησιμοποιούν τα ίδια δεδομένα τόσο για τη *ρύθμιση και βελτιστοποίηση* των παραμέτρων (*fine tuning*) ενός μοντέλου όσο και για την *εκπαίδευση* και *δοκιμή* του. Για να αποφευχθεί αυτό το ισχυρό ενδεχόμενο, τα δεδομένα συχνά χωρίζονται σε τρία μέρη:

- Δεδομένα *εκπαίδευσης*: Αυτό το τμήμα των δεδομένων χρησιμοποιείται για τη δημιουργία του *μοντέλου εκπαίδευσης*.
- Δεδομένα *επικύρωσης (validation data)*: Αυτό το τμήμα των δεδομένων χρησιμοποιείται για την επιλογή του μοντέλου και για τη βελτιστοποίηση των παραμέτρων του.
- Δεδομένα *δοκιμής*: Αυτό το τμήμα των δεδομένων χρησιμοποιείται για τον έλεγχο της *ακρίβειας* του τελικού μοντέλου που έχει ήδη βελτιστοποιημένες παραμέτρους απο το προηγούμενο στάδιο.

Παρόλο που ο αντικειμενικός στόχος των συστημάτων συστάσεων είναι η παραγωγή μιας *διατεταγμένης λίστας προτεινόμενων στοιχείων*, η αξιολόγηση της αποτελεσματικότητάς τους γίνεται με τα δύο παρακάτω σενάρια:

- Πρόβλεψη της βαθμολογίας που θα βάλει ο υπό εξέταση χρήστης στα στοιχεία (π.χ. στη βαθμολογική κλίμακα από 1 έως 5 αστέρια προβλέπουμε ότι θα βαθμολογήσει ένα στοιχείο με 4 αστέρια). Σ' αυτό το σενάριο μας ενδιαφέρει η διαφορά μεταξύ προβλεπόμενης και πραγματικής τιμής να είναι όσο μικρότερη γίνεται.
- Συστάσεις στοιχείων (π.χ. λίστα με τα *κορυφαία-Ν προτεινόμενα στοιχεία*): όλοι οι αλγόριθμοι μπορούν να εκτελούνται σ' έναν δισδιάστατο πίνακα βαθμολογίας χρήστη-στοιχείου που περιέχει δεδομένα είτε για βαθμολογίες στοιχείων (π.χ. κλίμακα βαθμολογίας με τιμές από 1 έως 5) είτε για αγορές/προβολές προϊόντων (π.χ. αγορασμένο προϊόν ή μη αγορασμένο προϊόν).

Εδώ να σημειώσουμε ότι για το πρώτο σενάριο αξιολόγησης της αποτελεσματικότητας των συστημάτων συστάσεων χρησιμοποιούνται μέτρα αποτίμησης όπως το *Mean Absolute Error (MAE)* και το *Root Mean Square Error (RMSE)*. Από την άλλη μεριά, για το δεύτερο σενάριο αξιολόγησης της αποτελεσματικότητας των συστημάτων συστάσεων χρησιμοποιούνται μέτρα αποτίμησης όπως

η *ακρίβεια*, η *ανάκληση*, το *F1*, και το *nDCG*, καθώς και όλα τα στοιχεία του *πίνακα συγκεχυμένων αποτελεσμάτων πρόβλεψης* (*confusion* ή *error matrix*). Εν συνεχεία, περιγράφουμε την κάθε *μετρική αποτίμησης* αναλυτικά.

9.2 Μετρικές MAE και RMSE

Στο πρώτο σενάριο της πρόβλεψης της βαθμολογίας ενός υποψήφιου προς σύσταση στοιχείου έχουν χρησιμοποιηθεί διάφορες *μετρικές αποτίμησης* για την αξιολόγηση των *αλγορίθμων Συνεργατικού Φιλτραρίσματος (ΣΦ)*. Για παράδειγμα, για τη μέτρηση της ποιότητας της *πρόβλεψης βαθμολογιών των υποψήφιων προς σύσταση στοιχείων* στα ανάλογα συστήματα χρησιμοποιούμε τα *Mean Absolute Error (MAE)* και *Root Mean Squared Error (RMSE)* ως εξής:

$$MAE = \frac{1}{|R_{test}|} \sum_{R_{ui} \in R_{test}} |\hat{R}_{ui} - R_{ui}| \tag{9.1}$$

$$RMSE = \sqrt{\frac{1}{|R_{test}|} \sum_{R_{ui} \in R_{test}} (\hat{R}_{ui} - R_{ui})^2} \tag{9.2}$$

Το *Μέσο Απόλυτο Σφάλμα (MAE)* είναι μία ακόμη μετρική αποτίμησης και μάλιστα ευρέως χρησιμοποιούμενη. Υπολογίζει δε την απόλυτη διαφορά μεταξύ των πραγματικών και των προβλεφθέντων βαθμολογιών των στοιχείων, όπως φαίνεται στην Εξίσωση 9.1. Τονίζεται ότι R είναι ο πίνακας βαθμολογιών χρηστών-στοιχείων, και R_{test} είναι το σύνολο των χρηστών που βρίσκονται στα δεδομένα ελέγχου.

Από την άλλη, όσον αφορά στο *RMSE*, δηλαδή για την *τετραγωνική ρίζα του μέσου τετραγωνικού σφάλματος(Root Mean Squared Error)*, αυτό υπολογίζει τις διαφορές μεταξύ των πραγματικών και των προβλεπόμενων βαθμολογιών με την ιδιαιτερότητα ότι τις υψώνει στο τετράγωνο, ώστε να "τιμωρείται" (*penalized*) μια μεγάλη διαφορά μεταξύ της πραγματικής και της προβλεπόμενης βαθμολογίας ενός στοιχείου έναντι των μικρότερων διαφορών όπως φαίνεται στην Εξίσωση 9.2.

Παράδειγμα 9.1 Δίνεται ο παρακάτω δισδιάστατος πίνακας με τις βαθμολογίες 4 χρηστών (γραμμές του πίνακα) για 4 στοιχεία (στήλες του πίνακα):

$$\begin{bmatrix} 4 & 1 & 1 & 4 \\ 1 & 4 & 2 & ? \\ 2 & 1 & 4 & 5 \\ 1 & 4 & 1 & ? \end{bmatrix}$$
$$A_{4\times 4}$$

Ζητείται να προβλέψουμε τις βαθμολογίες των χρηστών επί των στοιχείων. Σημειώνεται ότι, όπου τα κελιά του παραπάνω πίνακα έχουν το σύμβολο "?", αυτό δηλώνει ότι δεν υπάρχει

σχετική βαθμολογία από τον εκάστοτε χρήστη για το συγκεκριμένο στοιχείο. Επίσης, δίνεται το *Root Mean Square Error (RMSE)* - Εξίσωση 9.3:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(a_{ij} - \hat{a}_{ij})^2}{D}}, \quad \forall i \in n \text{ και } j \in m : a_{ij} \neq null \tag{9.3}$$

όπου D ο αριθμός των κελιών του πίνακα A που έχουν μια βαθμολογία.
Ας υποθέσουμε, επίσης, ότι ένας *απλοϊκός αλγόριθμος* προβλέπει τις βαθμολογίες του παρακάτω πίνακα χρηστών-στοιχείων $\hat{A}$:

$$\begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix}$$
$$\hat{A}_{4\times4}$$

Έπειτα, μπορούμε να υπολογίσουμε το *RMSE* για τους πίνακες A και $\hat{A}$ με βάση την Εξίσωση 9.3. Για παράδειγμα, για να υπολογίσουμε το *RMSE* μεταξύ της πρώτης γραμμής των πινάκων A και $\hat{A}$, αφαιρούμε 2 από κάθε στοιχείο της πρώτης γραμμής του A ώστε να λάβουμε 2, -1, -1, 2. Στη συνέχεια τα υψώνουμε στο τετράγωνο και τα αθροίζουμε για να πάρουμε 10. Στη δεύτερη γραμμή του πίνακα A, η τελευταία στήλη είναι "?", οπότε το στοιχείο αυτό παραλείπεται κατά τον υπολογισμό του *RMSE*. Οι διαφορές είναι -1, 2, 0 και το άθροισμα των τετραγώνων τους είναι 5. Για την τρίτη γραμμή, οι διαφορές είναι 0, -1, 2, 3 και το άθροισμα των τετραγώνων τους είναι 14. Η τέταρτη γραμμή του πίνακα έχει μια κενή καταχώριση ("?") στην τέταρτη στήλη, οπότε οι διαφορές είναι -1, 2, -1 και το άθροισμα των τετραγώνων τους είναι 6. Όταν αθροίζουμε τις προαναφερθείσες ποσότητες από καθεμία εκ των τεσσάρων γραμμών των πινάκων, τότε έχουμε $10 + 5 + 14 + 6 = 35$. Τέλος, διαιρούμε με το 14 (τον αριθμό δηλαδή των μη κενών στοιχείων του πίνακα A) και παίρνουμε την τετραγωνική ρίζα. Έτσι, το *RMSE* που προκύπτει είναι το ακόλουθο:

$$\text{RMSE} = \sqrt{\frac{10 + 5 + 14 + 6}{14}} = 1,581$$

Κλείνοντας, σημειώνουμε ότι το *MAE* και το *RMSE* μπορούν να χαρακτηρίσουν ικανοποιητικά την ακρίβεια πρόβλεψης μιας βαθμολογίας για ένα στοιχείο από έναν αλγόριθμο, αλλά δεν μπορούν να είναι αντιπροσωπευτικά για την αποτίμηση της ακρίβειας ολόκληρης της λίστας προτεινόμενων στοιχείων. Έτσι, δεδομένου ότι στα πραγματικά συστήματα συστάσεων η ικανοποίηση ενός χρήστη εξαρτάται κυρίως από την ακρίβεια της λίστας προτεινόμενων στοιχείων που θα του συσταθεί, το *MAE* και το *RMSE* δεν πρέπει να αποτελούν τις προτιμώμενες μετρικές αποτίμησης.

9.3 Πίνακας Συγκεχυμένων Αποτελεσμάτων Πρόβλεψης

Για την αποτίμηση του δεύτερου σεναρίου (*Item Recommendation Task*) -στο οποίο ήδη αναφερθήκαμε στο τέλος της εισαγωγής του παρόντος κεφαλαίου- μπορούμε να χρησιμοποιήσουμε τον *Πίνακα Συγκεχυμένων Αποτελεσμάτων Πρόβλεψης* (Confusion Matrix), σύμφωνα με τον οποίο οποιοδήποτε προτεινόμενο στοιχείο θα εμπίπτει σε μία από τις τέσσερεις πιθανές περιπτώσεις που περιγράφονται στον Πίνακα 9.1:

		Πρόβλεψη	
		TRUE	**FALSE**
Πραγματικότητα	**TRUE**	*True Positive*	*False Negative*
	FALSE	*False Positive*	*True Negative*

Πίνακας 9.1: *Πίνακας Συγκεχυμένων Αποτελεσμάτων Πρόβλεψης (Confusion Matrix).*

Όπως λοιπόν φαίνεται στον παραπάνω Πίνακα 9.1, τα στοιχεία που χαρακτηρίζονται ως *ψευδώς αρνητικά (False Negative)* είναι εκείνα που, ενώ είναι *συναφή (relevant)* με τις προτιμήσεις του υπό εξέταση χρήστη (δηλαδή του αρέσουν), εσφαλμένα δεν προβλέφθηκαν από τον αλγόριθμο ως τέτοια, και, συνεπώς, δεν προτάθηκαν. Από την άλλη, τα στοιχεία που καταμετρώνται ως *αληθώς θετικά (True Positive)* και είναι *συναφή* με τις προτιμήσεις του υπό εξέταση χρήστη αλλά και ο αλγόριθμος ορθώς τα προέβλεψε ως τέτοια. Τα δε στοιχεία που κατατάσσονται στα *ψευδώς θετικά (False Positive)*, προβλέφθηκαν εσφαλμένα από τον αλγόριθμο ως *συναφή*, ενώ στην πραγματικότητα δεν είναι. Τέλος, τα στοιχεία που συμπεριλαμβάνονται στα *αληθώς αρνητικά (True Negative)*, και δεν είναι συναφή με τον υπό εξέταση χρήστη (δηλαδή δεν του αρέσουν) αλλά και ορθώς προβλέφθηκαν ως τέτοια.

Για να αξιολογήσει λοιπόν κάποιος ένα σύστημα συστάσεων, μπορεί απλά να χρησιμοποιήσει τη *μετρική accuracy* (εδώ, *ορθότητα*), η οποία υπολογίζει το ποσοστό των στοιχείων που κατατάχθηκαν στη σωστή κλάση (*συναφή στοιχεία* και *μη-συναφή στοιχεία*). Δηλαδή, ήταν όντως συναφή και προβλέφθηκαν ως τέτοια από τον αλγόριθμο, και ήταν όντως μη συναφή και πάλι προβλέφθηκαν ως τέτοια. Βάσει του Πίνακα 9.1 η *ορθότητα* υπολογίζεται ως εξής:

$$\text{Ορθότητα} = \frac{\#TP + \#TN}{\#TP + \#FP + \#FN + \#TN} \tag{9.4}$$

Παρόλα αυτά, η *μετρική της ορθότητας* δεν αποτελεί κατάλληλο μέτρο για το πρόβλημα των συστημάτων συστάσεων. Το κύριο πρόβλημα είναι ότι τα στιγμιότυπα δεδομένων μεταξύ των δύο κλάσεων δεν είναι *ισορροπημένα (balanced)*. Εάν για παράδειγμα υπάρχει στη βάση δεδομένων ένα πολύ μεγαλύτερο ποσοστό στοιχείων που δεν αρέσουν στον υπό εξέταση χρήστη, τότε ένας αλγόριθμος που έχει ρυθμιστεί για να μεγιστοποιήσει τη *μετρική της ορθότητας* μπορεί να φαίνεται ότι αποδίδει καλά, προβλέποντας απλώς όλα τα στοιχεία ως *μη συναφή (non-relevant)* με τις προτιμήσεις του υπό εξέταση χρήστη. Για τους παραπάνω λόγους, για την αξιολόγηση της λίστας προτεινόμενων στοιχείων προτιμώνται οι γνωστές μετρικές ***precision*** και ***recall***, τις οποίες θα παρουσιάσουμε εκτενώς στην επόμενη ενότητα.

9.4 Precision, Recall, και Μετρική F1

Για την αξιολόγηση της ποιότητας της λίστας προτεινόμενων στοιχείων είναι σημαντικό να γνωρίζουμε εάν ένα προτεινόμενο στοιχείο επιλέχθηκε τελικά ή όχι από τον υπό εξέταση χρήστη. Στα συστήματα συστάσεων λοιπόν γίνονται προτάσεις στοιχείων που βασίζονται τόσο σε *έμμεση* όσο και σε *ρητή, άμεση,* βαθμολογία: *ρητή* είναι αυτή που βάζει ο ίδιος χρήστης σ' ένα στοιχείο, ενώ *έμμεση* είναι όταν εμείς ερμηνεύουμε μια αλληλεπίδραση ενός χρήστη μ' ένα στοιχείο ως ένδειξη ότι αυτό όντως του αρέσει. Για παράδειγμα, μπορεί ένας χρήστης να έκανε απλώς κλικ σ' ένα άρθρο, και εμείς να ερμηνεύουμε αυτήν την ενέργεια ως δείγμα προτίμησης.

Επίσης, είναι σημαντικό να τονίσουμε ότι κατά την αξιολόγηση των *ρητών* βαθμολογιών πρέπει να καθοριστεί ένα "κατώφλι" (π.χ. το θετικό όριο P_τ, με προεπιλεγμένη τιμή 3 σε μια κλίμακα βαθμολογίας 1-5) που να ορίζει ότι από εκείνο το κατώτατο όριο και πέρα θεωρούμε ότι ένα στοιχείο βαθμολογείται θετικώς από τον χρήστη.

Εμβαθύνοντας, οι ακόλουθες εξισώσεις υπολογίζουν την *Ακρίβεια* (*Precision*) και την *Ανάκληση* (*Recall*) βάσει του αριθμού των προτεινόμενων στοιχείων που καταγράφονται σε καθένα από τα κελιά του *πίνακα συγκεχυμένων αποτελεσμάτων πρόβλεψης* (*confusion matrix*) όπως αυτά εμφανίζονται στον Πίνακα 9.1.

$$\text{Ακρίβεια} = \frac{\#TP}{\#TP + \#FP} \qquad \text{Ανάκληση} = \frac{\#TP}{\#TP + \#FN} \tag{9.5}$$

Πιο συγκεκριμένα, η *ακρίβεια (precision)* μετράει τον αριθμό των *συναφών (relevant)* στοιχείων (*True Positive - Αληθώς Θετικά*) διαιρεμένο με τον αριθμό των προτεινόμενων στοιχείων (*Αληθώς Θετικά + Ψευδώς Θετικά*), ενώ η *ανάκληση (recall)* μετράει τον αριθμό των *συναφών* στοιχείων *(Αληθώς Θετικά)* που προτείνονται από τον αλγόριθμο σύστασης διαιρεμένο με τον αριθμό όλων των *συναφών (relevant)* στοιχείων *(Αληθώς Θετικά* και *Ψευδώς Αρνητικά)*. Συνήθως γίνεται μια *αντιστάθμιση (trade-off)* μεταξύ των δύο αυτών ποσοτήτων. Λόγου χάρη, αυξάνοντας το μέγεθος της λίστας προτεινόμενων στοιχείων αναμένουμε να αυξηθεί η *Recall* και να μειωθεί η *Precision*.

Στην εύλογη ερώτηση λοιπόν σχετικά με το ποια μετρική (*precision* ή *recall*) είναι καλύτερη, η απάντηση εξαρτάται από το πεδίο εφαρμογής αυτών των μετρικών. Για παράδειγμα, στον χώρο της υγείας, στα συστήματα συστάσεων φαρμάκων για ασθενείς είναι εξαιρετικά σημαντικό να προσδιορίζονται σωστά όλα εκείνα τα φάρμακα που συνταγογραφήθηκαν από τον γιατρό. Έτσι, η *ανάκληση* μπορεί να θεωρηθεί σημαντικότερη από ό,τι η *ακρίβεια*. Παραδείγματος χάρη, είναι πολύ πιθανό να επιτευχθεί υψηλή *ακρίβεια* από έναν αλγόριθμο που προβλέπει σωστά μόνο ένα φάρμακο για τη θεραπεία του ασθενούς, ενώ στον ασθενή πρέπει να δοθούν πολύ περισσότερα φάρμακα για την κάλυψη των ασθενειών του. Άρα, μπορεί η μετρική της *ακρίβειας* να είναι 100% εὐστοχη, αλλά δεν θα έχουν προβλεφθεί από τον αλγόριθμο όλα τα φάρμακα που χρειάζεται ο ασθενής. Αντιθέτως, η *ανάκληση* θα λάβει υπόψη της τον αριθμό των φαρμάκων που χρειάζεται ο ασθενής και τελικά πόσα από αυτά του προτάθηκαν από τον αλγόριθμο. Αυτό σημαίνει ότι η *ανάκληση* μπορεί να χαρακτηριστεί ως καλύτερη μετρική αποτίμησης συγκριτικά με αυτήν της *ακρίβειας* για το συγκεκριμένο πεδίο εφαρμογής στον τομέα της υγείας.

Τέλος, μια μετρική που συνδυάζει την *ακρίβεια* και την *ανάκληση* είναι η F_1. Πιο συγκεκριμένα, F_1 είναι ο *σταθμισμένος αρμονικός μέσος όρος της* ***ακρίβειας*** *και της* ***ανάκλησης***:

$$F_1 = \frac{2 \cdot \text{Ακρίβεια} \cdot \text{Ανάκληση}}{\text{Ακρίβεια} + \text{Ανάκληση}}$$

Η F_1, επομένως, *αντισταθμίζει* (*trade-off*) τις μετρικές της *ακρίβειας* και της *ανάκλησης* παρέχοντάς μας μια γενική αποτίμηση της ποιότητας της λίστας προτεινόμενων στοιχείων. Ο λόγος που χρησιμοποιείται ο *αρμονικός μέσος* έναντι της *απλής μέσης τιμής* (*mean value* των *precision* και *recall*) είναι ότι η δεύτερη είναι περισσότερο ευαίσθητη στις ακραίες τιμές που μπορεί να εμφανίζονται σε μία από τις δύο μετρικές αποτίμησης.

9.5 Καμπύλη Precision-Recall - Αναφορά στη Μετρική MAP

Ένα σύστημα συστάσεων παρέχει στον υπό εξέταση χρήστη μια *διατεταγμένη λίστα με προτεινόμενα στοιχεία* τα οποία στην ουσία είναι τα *κορυφαία N στοιχεία* (*top-N recommended items*) που προορίζονται για τον υπό εξέταση χρήστη. Συνεπώς μπορούμε να μετρήσουμε την *ακρίβεια* και την *ανάκληση* για διαφορετικές τιμές του N που σχετίζονται με διαφορετικές θέσεις των στοιχείων της λίστας (π.χ. κορυφαίο-1 στοιχείο, κορυφαία-2 στοιχεία, κορυφαία-3 στοιχεία... κορυφαία-N στοιχεία). Για κάθε διαφορετική τιμή των N *κορυφαίων στοιχείων* (top-1, top-2, top-3... top-N) οι τιμές *ακρίβειας* και *ανάκλησης* μπορούν να καταγραφούν ως σημεία στο καρτεσιανό σύστημα αξόνων που, εφόσον ενωθούν με μια γραμμή, μπορούν να δώσουν την *καμπύλη ακρίβειας-ανάκλησης*.

Παράδειγμα 9.2 Έστω ότι έχουμε στην ταινιοθήκη μας 10 ταινίες, και 5 από αυτές αρέσουν στον υπό εξέταση χρήστη. Δύο αλγόριθμοι (*ideal algorithm* και *not so good algorithm*) εφαρμόζονται σ' αυτό το σύνολο δεδομένων προκειμένου να προβλέψουν μια *διατεταγμένη λίστα*

των κορυφαίων-10 ταινιών για τον υπό εξέταση χρήστη. Στη δεύτερη και στην πέμπτη στήλη του Πίνακα 9.2 εμφανίζονται με μονάδα (1) οι θέσεις των 5 ταινιών που πραγματικά αρέσουν στον χρήστη για τους δύο αλγορίθμους. Παρατηρούμε λοιπόν στον Πίνακα 9.2 ότι ο πρώτος αλγόριθμος (*ideal algorithm*) αποτελεί το *ιδανικό μοντέλο πρόβλεψης*, διότι κατατάσσει τις 5 ταινίες που πραγματικά αρέσουν στον υπό εξέταση χρήστη στις 5 πρώτες θέσεις της λίστας των προτεινόμενων ταινιών (θέσεις 1 έως 5). Από την άλλη, ο δεύτερος αλγόριθμος (*not so good algorithm*) έχει κάποιες άστοχες προβλέψεις. Ειδικότερα, στην περίπτωση του *όχι και τόσο καλού αλγορίθμου* (*not so good algorithm*) οι σωστές προβλέψεις αφορούν στις θέσεις 1, 2, 4, 8 και 10 στη *λίστα των κορυφαίων-10 προτεινόμενων ταινιών*.

Metrics ⇒ Position in top-10 list ⇓	**Ideal algorithm**	Precision	Recall	**Not so Good algorithm**	Precision	Recall
1	1	1.0	0.2	1	1.0	0.2
2	1	1.0	0.4	1	1.0	0.4
3	1	1.0	0.6	0	0.66	0.4
4	1	1.0	0.8	1	0.75	0.6
5	1	1.0	1.0	0	0.6	0.6
6	0	0.83	1.0	0	0.5	0.6
7	0	0.71	1.0	0	0.43	0.6
8	0	0.62	1.0	1	0.5	0.8
9	0	0.55	1.0	0	0.44	0.8
10	0	0.5	1.0	1	0.5	1.0

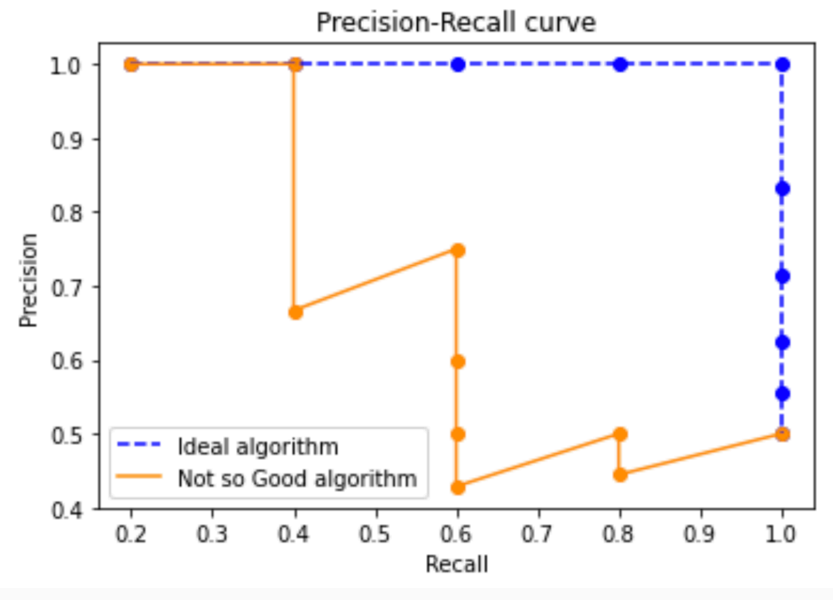
Πίνακας 9.2: Στη δεύτερη και στην πέμπτη στήλη η μονάδα (1) σε μια θέση της *λίστας των κορυφαίων-10 προτεινόμενων στοιχείων* σημαίνει ότι ο αλγόριθμος έχει κάνει σωστή πρόβλεψη (δηλαδή το στοιχείο αρέσει στον χρήστη), ενώ το μηδέν (0) σημαίνει το αντίθετο (δηλαδή το στοιχείο δεν αρέσει στον χρήστη)

.

Βάσει λοιπόν των δεδομένων της τρίτης και τέταρτης στήλης του Πίνακα 9.2 δημιουργούμε την *καμπύλη precision-recall* του *ideal algorithm*, η οποία εμφανίζεται με την μπλε διακεκομμένη γραμμή στο Σχήμα 9.1. Επίσης, εδραζόμενοι στα δεδομένα των τελευταίων δύο στηλών του Πίνακα 9.2 δημιουργούμε την *καμπύλη precision-recall* του *not so good algorithm*, η οποία εμφανίζεται με την πορτοκαλί γραμμή στο Σχήμα 9.1.

Αυτό που έχουμε να σημειώσουμε εδώ, στο Σχήμα 9.1, είναι ότι η *ακρίβεια* του *not so good algorithm* μειώνεται σταθερά όσο αυξάνεται ο αριθμός των *κορυφαίων*-10 *προτεινόμενων στοιχείων*. Αντίθετα, η *ανάκληση* αυξάνεται όσο μεγαλώνει ο αριθμός των *κορυφαίων*-10 *προτεινόμενων στοιχείων*. Άρα, η σχέση μεταξύ των δύο μετρικών είναι αντιστρόφως ανάλογη: δηλαδή

όσο αυξάνουμε το μέγεθος της λίστας προτεινόμενων στοιχείων τόσο μεγαλύτερη πιθανότητα έχουμε να προβλέψουμε τυχαία ένα στοιχείο *συναφές* με τις προτιμήσεις του χρήστη. Αυτό έχει ως αποτέλεσμα να αυξάνεται πάντα η *ανάκληση* καθώς μεγαλώνει το μέγεθος της λίστας προτεινόμενων στοιχείων. Αντιθέτως, η *ακρίβεια* συνήθως μειώνεται με την αύξηση του μεγέθους της λίστας προτεινόμενων στοιχείων, είτε επειδή δεν υπάρχουν τόσα *συναφή* (με τις προτιμήσεις του υπό εξέταση χρήστη) στοιχεία στη βάση δεδομένων (όπως συμβαίνει στην περίπτωση του παραδείγματός μας με τον *ideal algorithm*) είτε επειδή αυξάνεται η πιθανότητα να προβλέψουμε εσφαλμένα ως *συναφές* ένα *μη συναφές* στοιχείο.



Σχήμα 9.1: *Καμπύλη Precision-Recall* για τον *ideal* και για τον *not so good algorithm.*

.

Συνοψίζοντας, μεταβάλλοντας το μέγεθος της λίστας προτεινόμενων στοιχείων μπορούμε να υπολογίσουμε το ποσοστό των *συναφών* (*True Positives*) και των *μη-συναφών* στοιχείων (*True Negatives*) -που ορθώς παραλείπονται- στη λίστα. Εάν, τώρα, το μέγεθος της λίστας προτεινόμενων στοιχείων είναι πολύ μικρό, τότε ο αλγόριθμος θα παραλείψει να συμπεριλάβει στη λίστα πολλά *συναφή* στοιχεία (θα αυξηθούν δηλαδή τα *ψευδώς αρνητικά*). Από την άλλη πλευρά, εάν το μέγεθος της λίστας προτεινόμενων στοιχείων είναι πολύ μεγάλο, αυτό θα οδηγήσει σε πάρα πολλές προβλέψεις *μη συναφών* προς τις προτιμήσεις του υπό εξέταση χρήστη στοιχείων (θα αυξηθούν δηλαδή τα *ψευδώς θετικά*).

Τέλος, με δεδομένο ότι τα συστήματα συστάσεων παρέχουν σ΄ έναν υπό εξέταση χρήστη *u* μια *λίστα με τα κορυφαία N προτεινόμενα στοιχεία*, είναι σημαντικό να μπορούμε να εξετάσουμε τη σειρά και τη θέση κατάταξης των προτεινόμενων στοιχείων όπως αυτά διατάσσονται μέσα σ' αυτήν τη λίστα. Τονίζουμε ότι είναι προτιμότερο να έχουμε μια σωστή πρόβλεψη στις πρώτες θέσεις της

λίστας συστάσεων, από το να έχουμε μια σωστή πρόβλεψη στις τελευταίες θέσεις της λίστας προτεινόμενων στοιχείων. Ως εκ τούτου, χρησιμοποιούμε τη *Μέση Αντιπροσωπευτική Ακρίβεια* (*Mean Average Precision - MAP*) για να δώσουμε έμφαση στην κατάταξη των συναφών στοιχείων σε κορυφαίες θέσεις της ανάλογης λίστας. Η *MAP* ορίζεται με την Εξίσωση 9.6:

$$MAP = \frac{1}{|U|}\sum_{u=1}^{|U|}\frac{1}{r_u}\sum_{N=1}^{r_u}\text{Ακρίβεια}_u@N \tag{9.6}$$

όπου U ο αριθμός των χρηστών στο σύνολο των *δεδομένων ελέγχου (test set)*, r_u ο αριθμός των συναφών στοιχείων γι' έναν χρήστη u, και $\text{Ακρίβεια}_u@N$ η τιμή *ακρίβειας* στη N-οστή θέση της λίστας προτεινόμενων στοιχείων για τον χρήστη u. Σημειώστε ότι η *μετρική MAP* λαμβάνει υπόψη τόσο την *ακρίβεια* όσο και την *ανάκληση*, ενώ γεωμετρικώς προσδιορίζεται ως η περιοχή κάτω από την *καμπύλη Precision-Recall*.

9.6 Καμπύλη ROC, και Μετρική AUC

Μία σημαντική μετρική αποτίμησης της ποιότητας της λίστας προτεινόμενων στοιχείων είναι και η *καμπύλη ROC* (*Receiver Operating Characteristics*), η οποία εκφράζει το *ποσοστό των αληθώς θετικών* προβλέψεων (*True Positive Rate*) έναντι του *ποσοστού των ψευδώς θετικών* προβλέψεων (*False Positive Rate*). Αναλυτικότερα, το *ποσοστό των αληθώς θετικών* προβλέψεων δίνεται από την εξίσωση 9.7:

$$\text{True Positive Rate} = \frac{\#TP}{\#TP + \#FN} \tag{9.7}$$

Τονίζουμε στο σημείο αυτό ότι το *True Positive Rate* είναι ισοδύναμο της *ανάκλησης* (*recall*) που ορίσαμε με την Εξίσωση 9.5. Το δε *ποσοστό των ψευδώς θετικών* προβλέψεων δίνεται από την εξίσωση 9.8:

$$\text{False Positive Rate} = \frac{\#FP}{\#FP + \#TN} \tag{9.8}$$

Παράδειγμα 9.3 Έστω ότι έχουμε στην ταινιοθήκη μας 10 ταινίες, και 5 από αυτές αρέσουν στον υπό εξέταση χρήστη. Δύο αλγόριθμοι (*ideal algorithm* και *not so good algorithm*) εφαρμόζονται σ΄ αυτό το σύνολο δεδομένων, προκειμένου να προβλέψουν μια *διατεταγμένη λίστα των κορυφαίων-10 ταινιών* για τον υπό εξέταση χρήστη. Έτσι, οι τιμές του *ποσοστού των αληθώς θετικών* προβλέψεων (*True Positive Rate*) και του *ποσοστού των ψευδώς θετικών* προβλέψεων (*False Positive Rate*) μπορούν να παραχθούν από τις θέσεις όπου βρίσκονται οι 5 ταινίες που πραγματικά αρέσουν στον χρήστη μέσα στη *διατεταγμένη λίστα των προτεινόμενων ταινιών* που προβλέπει ο κάθε αλγόριθμος. Στον Πίνακα 9.3 εμφανίζονται με μονάδα (1) οι θέσεις των 5 ταινιών που πραγματικά αρέσουν στον χρήστη σε σχέση με τους δύο αλγορίθμους. Πα-

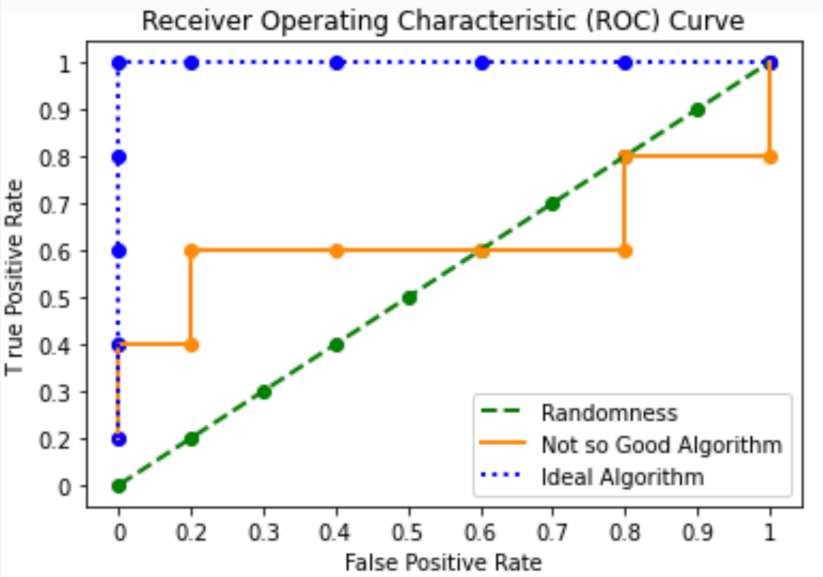
ρατηρούμε λοιπόν στον Πίνακα 9.3 ότι ο πρώτος αλγόριθμος (*ideal algorithm*) αποτελεί το ιδανικό μοντέλο πρόβλεψης μια και κατατάσσει τις 5 ταινίες που πραγματικά αρέσουν στον υπό εξέταση χρήστη στις 5 πρώτες θέσεις της λίστας των προτεινόμενων ταινιών (θέσεις 1 έως 5). Από την άλλη, ο δεύτερος αλγόριθμος (*not so good algorithm*) έχει κάποιες άστοχες προβλέψεις, με τις σωστές να αφορούν στις θέσεις 1, 2, 4, 8 και 10 στη *λίστα των κορυφαίων-10 προτεινόμενων ταινιών*.

Algorithm ⇒ Position in top-10 list ⇓	Ideal	Not so Good
1	1	1
2	1	1
3	1	0
4	1	1
5	1	0
6	0	0
7	0	0
8	0	1
9	0	0
10	0	1

Πίνακας 9.3: Η μονάδα (1) σε μια θέση της λίστας των *top-10* προτεινόμενων στοιχείων σημαίνει ότι ο αλγόριθμος έχει κάνει σωστή πρόβλεψη (δηλαδή προέβλεψε ένα στοιχείο που αρέσει στον χρήστη), ενώ το μηδέν (0) σημαίνει το αντίθετο (δηλαδή προέβλεψε ένα στοιχείο που δεν αρέσει στον χρήστη).

.

Οι *καμπύλες ROC* που αντιστοιχούν στα δεδομένα του Πίνακα 9.3 εμφανίζονται στο Σχήμα 9.2. Μια *καμπύλη ROC* πηγαίνει πάντα από κάτω αριστερά προς τα πάνω δεξιά στο γράφημα. Για ένα καλό σύστημα συστάσεων (ο ιδανικός αλγόριθμος στην περίπτωση του παραδείγματός μας), η γραφική παράσταση ανεβαίνει απότομα στην αριστερή πλευρά. Σημειώστε ότι οι *καμπύλες ROC* αυξάνονται πάντα μονοτονικά. Ωστόσο, δεν είναι όλα τα τμήματα της *καμπύλης ROC* εξίσου σημαντικά, επειδή συνήθως υπάρχουν πρακτικά όρια στο μέγεθος της λίστας προτεινόμενων στοιχείων, και άρα μας ενδιαφέρει περισσότερο το πρώτο τμήμα της *καμπύλης ROC*, διότι αφορά στις πρώτες θέσεις της *λίστας των κορυφαίων προτεινόμενων στοιχείων*.



Σχήμα 9.2: *Καμπύλη ROC* για τον *ιδανικό αλγόριθμο* (*ideal algorithm*) και για έναν *όχι και τόσο καλό αλγόριθμο* (*not so good algorithm*).

Σε πολλές δε περιπτώσεις χρειαζόμαστε μια μοναδική τιμή που να αποτυπώνει συνολικά την ποιότητα της λίστας προτεινόμενων στοιχείων. Ένα τέτοιο συνολικό μέτρο είναι το *AUC* (*Area Under Curve*) της *καμπύλης ROC*, το οποίο γεωμετρικά είναι ισοδύναμο με την περιοχή που βρίσκεται κάτω από την *καμπύλη ROC*. Με το *AUC* μπορούμε να ποσοτικοποιήσουμε την *ακρίβεια* των αλγορίθμων πρόβλεψης και να ελέγξουμε πόσο αποτελεσματικότεροι είναι από την "καθαρή τύχη" (*true randomness*). Παρατηρήστε σχετικώς στο Σχήμα 9.2 ότι ο *not so good algorithm* υπερέχει έναντι της "καθαρής τύχης" στο πρώτο τμήμα της *καμπύλης ROC*.

Ως εκ τούτου, συνάγεται ότι το *AUC* μετρά την πιθανότητα να προταθεί ένα στοιχείο με το οποίο έχει αλληλεπιδράσει ένας χρήστης ή το οποίο του αρέσει έναντι ενός στοιχείου για το οποίο αμφότερα δεν ισχύουν. Ορίζουμε λοιπόν το *AUC* με την εξίσωση 9.9:

$$AUC = \frac{n' + 0.5 \times n''}{n} \tag{9.9}$$

όπου n ο αριθμός των *ανεξάρτητων ζευγαριών συγκρίσεων*, n' ο αριθμός των περιπτώσεων όπου ένα τυχαία επιλεγμένο στοιχείο έχει χαμηλότερη θέση στη *διατεταγμένη λίστα προτεινόμενων στοιχείων* έναντι ενός στοιχείου με το οποίο έχει πραγματικά αλληλεπιδράσει ο χρήστης, και n'' ο αριθμός των περιπτώσεων όπου ένα τυχαία επιλεγμένο στοιχείο βρίσκεται σε υψηλότερη θέση στην παραπάνω λίστα. Εάν όλες οι προβλέψεις παράγονται από μια ανεξάρτητη και πανομοιότυπη κατανομή, η *ακρίβεια* θα πρέπει να είναι περίπου 0.5. Επομένως, ο βαθμός στον οποίο η *ακρίβεια* υπερβαίνει το 0.5 δείχνει το πόσο καλύτερα αποδίδει ο αλγόριθμος έναντι της (απλής) "τύχης" (*randomness*).

Έναλλακτικά το *AUC* ορίζεται ως ακολούθως [Rendle et al., 2009]:

$$\text{AUC} = \frac{1}{|U|} \sum_{u} \frac{1}{|E(u)|} \sum_{(i,j) \in E(u)} \delta\left(\hat{x}_{ui} > \hat{x}_{uj}\right)$$

όπου τα ζεύγη αξιολόγησης ανά χρήστη u είναι:

$$E(u) := \{(i,j) \mid (u,i) \in S_{\text{test}} \wedge (u,j) \notin (S_{\text{test}} \cup S_{\text{train}})\}$$

Τονίζεται ότι το σύνολο των δεδομένων είναι διαχωρισμένο σε δύο υποσύνολα: το σύνολο εκπαίδευσης S_{train} και το σύνολο δοκιμής $S_{\text{test.}}$ Επιπρόσθετα, τα μοντέλα πρόβλεψης εκπαιδεύονται στο S_{train} και η προβλεπόμενη εξατομικευμένη κατάταξή τους αξιολογείται στο σύνολο ελέγχου S_{test} με το *AUC*. Ακόμη, όπου U το σύνολο των χρηστών στο $S_{\text{test.}}$, $E(u)$ το σύνολο όλων των δυνατών ζευγαριών μεταξύ ενός στοιχείου με το οποίο ο χρήστης αλληλεπίδρασε και ενός στοιχείου με το οποίο ο χρήστης δεν έχει αλληλεπιδράσει στο παρελθόν, ενώ $\hat{x}_{ui}$ η προβλεπόμενη βαθμολογία του χρήστη u για το στοιχείο i με το οποίο έχει αλληλεπιδράσει στο παρελθόν - και θα πρέπει να το προβλέπουμε με υψηλότερη βαθμολογία έναντι του στοιχείου j με το οποίο ο χρήστης δεν έχει αλληλεπιδράσει στο παρελθόν. Μια υψηλότερη τιμή *AUC* υποδηλώνει την καλύτερη ποιότητα της λίστας προτεινόμενων στοιχείων. Η τιμή *AUC* για μια μέθοδο τυχαίων προβλέψεων είναι 0.5, ενώ η μέγιστη τιμή *AUC* είναι 1.

9.7 Μετρική nDCG

Με τις μετρικές *precision* και *recall*, που έχουμε ήδη μελετήσει, δεν λαμβάνεται υπόψη η θέση κατάταξης των στοιχείων μέσα στη λίστα συστάσεων. Παρόλα αυτά, έχει μεγάλη διαφορά εάν ένας αλγόριθμος προβλέπει σωστά το στοιχείο της πρώτης θέσης της λίστας συγκριτικά μ' έναν άλλο αλγόριθμο που προβλέπει σωστά το τελευταίο στοιχείο της λίστας. Όσο πιο χαμηλά στην κατάταξη εμφανίζεται ένα στοιχείο, τόσο λιγότερο πιθανό είναι να το κοιτάξει ο χρήστης, εξαιτίας του ότι θα απαιτηθεί πολύ περισσότερος χρόνος για να φτάσει ο χρήστης να εξετάσει τα στοιχεία που βρίσκονται στο τέλος της διατεταγμένης λίστας συστάσεων.

Μια μετρική που μπορεί να διαχειριστεί το παραπάνω πρόβλημα και επιτρέπει έτσι τη δικαιότερη αποτίμηση της ποιότητας μιας διατεταγμένης λίστας προτεινόμενων στοιχείων είναι το *normalized Discounted Cumulative Gain (nDCG)*, το οποίο είναι κανονικοποιημένο, επειδή είναι ο λόγος του *Discounted Cumulative Gain* (DCG) προς το *Ideal Discounted Cumulative Gain (IDCG)* - τα οποία και θα οριστούν στη συνέχεια.

Ειδικότερα, το DCG εξετάζει τη σχετική θέση των προτεινόμενων στοιχείων μέσα στη λίστα συστάσεων L. Για κάθε στοιχείο στη L υπάρχει ένα "κέρδος", το οποίο όμως αθροιστικώς συνυπολογίζουμε απομειούμενο βάσει της θέσης κατάταξης του υπό εξέταση στοιχείου. Όσο πιο χαμηλά, δηλαδή, στη λίστα συστάσεων βρίσκεται ένα στοιχείο, τόσο μικρότερο θα είναι και το "κέρδος".

Συνεπώς, το συνολικό "κέρδος" που συγκεντρώνεται μέχρι μια συγκεκριμένη θέση κατάταξης *pos* ορίζεται στην Εξίσωση 9.10 ως εξής:

$$DGC_{pos} = rel_1 + \sum_{i=2}^{pos} \frac{rel_i}{\log_2 i} \tag{9.10}$$

όπου rel_i ο βαθμός *συνάφειας* του στοιχείου στη θέση i της λίστας με τις προτιμήσεις του υπό εξέταση χρήστη.

Είναι, βεβαίως, προφανές ότι με την Εξίσωση 9.10 "τιμωρούνται" τα στοιχεία που προτείνονται σε χαμηλότερη θέση μέσα στη λίστα συστάσεων L, εφόσον ο παρονομαστής του κλάσματος στην προαναφερθείσα Εξίσωση 9.10 αυξάνεται σταθερά με την αύξηση της θέσης κατάταξης i. Αυτό έχει ως αποτέλεσμα να επιβάλλεται ένας λογαριθμικός παράγοντας απομείωσης του βαθμού συνάφειας του στοιχείου στη θέση i της λίστας συστάσεων, διότι βρίσκεται στον αριθμητή του κλάσματος στην Εξίσωση 9.10 και άρα διαιρείται σταθερά με αυξανόμενη ποσότητα.

Στη συνέχεια, κανονικοποιούμε το DCG έναντι του "ιδανικού" *DCG*. Στην περίπτωσή μας, το "ιδανικό κέρδος" προκύπτει από το γεγονός ότι τα στοιχεία στη L κατατάσσονται σε φθίνουσα σειρά βάσει του βαθμού *συνάφειάς* τους. Έτσι, η *ιδανική (ideal) DCG* υπολογίζεται ως εξής:

$$IDGC_{pos} = rel_1 + \sum_{i=2}^{|N|-1} \frac{rel_i}{\log_2 i} \tag{9.11}$$

όπου N ο αριθμός των προτεινόμενων στοιχείων στη λίστα.

Τέλος, το *normalized* DCG είναι ο λόγος μεταξύ του DCG και του *IDCG*:

$$nDCG_{pos} = \frac{DCG}{IDCG} \tag{9.12}$$

Τονίζουμε στο σημείο αυτό ότι το $nDCG_{pos}$ παίρνει τιμές στο διάστημα μεταξύ [0...1].

Παράδειγμα 9.4 Έστω, λοιπόν, ότι ένα σύστημα συστάσεων προτείνει μια διατεταγμένη λίστα με τις κορυφαίες-5 ταινίες, εκ των οποίων μόνο οι 3 (θέση 2, 3 και 4 στη λίστα) αρέσουν στον υπό εξέταση χρήστη όπως φαίνεται στον Πίνακα 9.4. Άρα, προκειμένου να υπολογίσουμε το DCG_5, εφαρμόζουμε καταρχάς στα δεδομένα του Πίνακα 9.4 την Εξίσωση 9.10, και παίρνουμε το παρακάτω αποτέλεσμα:

$$DCG_5 = \frac{1}{\log_2 2} + \frac{1}{\log_2 3} + \frac{1}{\log_2 4} = 2.13$$

Προσέξτε ότι στον παρονομαστή υπολογίζουμε τον λογάριθμο με βάση το 2 μόνο για τη 2η, 3η, και 4η θέση της λίστας με τις κορυφαίες-5 προτεινόμενες ταινίες, διότι σ' αυτές τις θέσεις βρέθηκαν οι σωστές προβλέψεις (*true positives*).

Εν συνεχεία, για να υπολογίσουμε την επίδοση που θα είχε το ιδανικό μοντέλο συστάσεων, εφαρμόζουμε στα δεδομένα του Πίνακα 9.4 την Εξίσωση 9.11 όπως φαίνεται παρακάτω:

$$IDCG_5 = 1 + \frac{1}{\log_2 2} + \frac{1}{\log_2 3} = 2,63$$

Παρατηρήστε ότι για το ιδανικό μοντέλο στον παρονομαστή υπολογίζουμε τον λογάριθμο με βάση το 2 μόνο για τη 2η και 3η θέση της *λίστας με τις κορυφαίες-5 προτεινόμενες ταινίες*, διότι στο ιδανικό μοντέλο οι σωστές προβλέψεις (*true positives*) βρίσκονται στις τρεις πρώτες θέσεις. Τονίζουμε ότι για την πρώτη θέση δεν υπάρχει καμμία λογαριθμική απομείωση και γι' αυτό εκεί έχουμε μονάδα (1). Τέλος, το *normalized DCG* επί των δεδομένων του παραδείγματός μας υπολογίζεται με τη βοήθεια της Εξίσωσης 9.12, και το τελικό αποτέλεσμα είναι το παρακάτω:

$$nDCG_5 = \frac{DCG_5}{IDCG_5} \approx 0,81$$

Position in top-5 Recommendation list	Is it True Positive?
1	0
2	1
3	1
4	1
5	0

Πίνακας 9.4: Η μονάδα (1) σε μια θέση της *λίστας των κορυφαίων-5 προτεινόμενων στοιχείων* σημαίνει ότι ο αλγόριθμος έχει κάνει σωστή πρόβλεψη (δηλαδή, προέβλεψε ένα στοιχείο που αρέσει στον χρήστη), ενώ το μηδέν (0) σημαίνει το αντίθετο (δηλαδή προέβλεψε ένα στοιχείο που δεν αρέσει στον χρήστη).

.

9.8 Μετρικές πέραν της Απόλυτης Ακρίβειας (Beyond Accuracy Metrics)

Τα τελευταία χρόνια έχουν προταθεί μετρικές που αποτιμούν και τη *χρησιμότητα* των συστάσεων πέρα από την *αποτελεσματικότητα* και την *ακρίβειά* τους, όπως για παράδειγμα, την ανάγκη για *πρωτότυπες (novel)* και *ποικιλόμορφες συστάσεις (diversified recommendations)*, την ύπαρξη *συστάσεων που μπορούν να επεξηγηθούν επαρκώς* στον χρήστη (*explainable recommendations*), τη μέτρηση της *συνολικής ικανοποίησης του χρήστη (user satisfaction)* ως προς την αλληλεπίδρασή του με το σύστημα συστάσεων, και άλλες μετρικές οι οποίες θα αναπτυχθούν στη συνέχεια.

9.8.1 Πρωτότυπες Συστάσεις (Novel Recommendations)

Ένας από τους στόχους των συστημάτων συστάσεων είναι να προτείνουν στους χρήστες "μη δημοφιλή" στοιχεία, δηλαδή *πρωτότυπα*, που να ταιριάζουν όμως με τα ενδιαφέροντά τους. Μ' αυτόν τον τρόπο οι επιχειρήσεις μπορούν να αυξήσουν τα κέρδη τους, καθώς αυτά τα νέα προτεινόμενα στοιχεία ή προϊόντα συνήθως έχουν υψηλότερα περιθώρια κέρδους. Επιπλέον, κατ' αυτόν τον

τρόπο οι χρήστες ενθαρρύνονται να χρησιμοποιούν την υπηρεσία συστάσεων και δεν αποτρέπονται, όπως θα συνέβαινε εάν λάμβαναν βαρετές και τετριμμένες συστάσεις δημοφιλών προϊόντων. Στην υποενότητα αυτή, λοιπόν, θα ορίσουμε τη *μετρική αποτίμησης της πρωτοτυπίας* ενός προτεινόμενου στοιχείου.

Καθορίζοντας Πόσο Πρωτότυπο είναι ένα Προτεινόμενο Στοιχείο

Ο απλούστερος ορισμός της *πρωτοτυπίας ενός στοιχείου* (*item novelty*) βασίζεται κυρίως στη *δημοφιλία των στοιχείων* (*item popularity*) επί του συνόλου των χρηστών του συστήματος συστάσεων [Castells et al., 2015]. Ο ορισμός αυτός εστιάζει στην εύρεση σχετικά άγνωστων στοιχείων από τη βάση δεδομένων με τα οποία δεν έχουν αλληλεπιδράσει πολλοί χρήστες, και είναι γνωστός ως *global long-tail item novelty*.

Ωστόσο, χρησιμοποιώντας την *πρωτοτυπία* με βάση τη *δημοτικότητα* των στοιχείων, δεν μπορούμε να αποτυπώσουμε το κατά πόσο ένα προτεινόμενο στοιχείο είναι καινούργιο για έναν συγκεκριμένο χρήστη (βλ. *εξατομικευμένη πρωτοτυπία στοιχείου - personalized item novelty*). Γι' αυτόν τον λόγο, θα χρησιμοποιήσουμε μια μετρική αποτίμησης της *πρωτοτυπίας* στοιχείου με βάση την *απόσταση* ([Castells et al., 2015], [Vargas and Castells, 2011]), την επονομαζόμενη *unexpectedness* (*μέτρηση του απροσδόκητου*). Συνεπώς, η *πρωτοτυπία* ενός στοιχείου ορίζεται από μια *συνάρτηση απόστασης* μεταξύ του υπό εξέταση στοιχείου i και του συνόλου των στοιχείων I_u με τα οποία έχει ήδη αλληλεπιδράσει ένας χρήστης (πρόκειται δηλαδή για την προηγούμενη συνολική εμπειρία του χρήστη με το σύστημα συστάσεων).

Με την Εξίσωση 9.13 λοιπόν ορίζουμε πόσο *πρωτότυπο* είναι γι' έναν χρήστη u ένα στοιχείο i:

$$N_{u,i} = \frac{\sum_{\forall j \in I_u} d(i,j)}{|I_u|} \tag{9.13}$$

Σημειώστε ότι η *απόσταση* μεταξύ δύο στοιχείων μπορεί να βασιστεί στα κοινά ή μη χαρακτηριστικά τους (π.χ. εάν πρόκειται για ταινίες, τα κοινά είδη ταινιών στα οποία μπορεί να ανήκουν οι δύο ταινίες ή όχι).

Παράδειγμα 9.5 Έστω, λοιπόν, ότι είμαστε οι υπεύθυνοι ενός συστήματος συστάσεων που αφορά σε ειδησεογραφικά άρθρα. Έστω, ακόμη, ότι γνωρίζουμε την κατηγορία στην οποία ανήκει ένα άρθρο (π.χ. πολιτική, αθλητικά κ.λπ.). Έτσι, όταν προτείνουμε ένα άρθρο για σπορ σ' έναν χρήστη που έχει ήδη διαβάσει πολλά αντίστοιχου περιεχομένου, αυτή η σύσταση δεν μπορεί να θεωρηθεί το ίδιο νέα και φρέσκια γι' αυτόν σε σχέση με την περίπτωση να παρέχουμε το ίδιο άρθρο σ' ένα άτομο που δεν έχει διαβάσει ποτέ κάτι ανάλογο. Για να αποτυπώσουμε το πόσο νέα (εδώ, *πρωτότυπη*) είναι μια θεματική κατηγορία γι' έναν χρήστη, μπορούμε να αξιοποιήσουμε την Εξίσωση 9.14, η οποία βασίζεται στη γνωστή *μετρική Topic*

Coverage (ή αλλιώς *S-recall*) [Castells et al., 2015], και η οποία είναι προσαρμοσμένη στην περίπτωσή μας:

$$N_{u,c} = \frac{1}{|\{i \in I_u : i \text{ ανήκει στην κατηγορία c}\}|} \tag{9.14}$$

όπου i ένα στοιχείο και C το σύνολο όλων των θεματικών κατηγοριών. Συνοψίζοντας, όταν ένας χρήστης u έχει αλληλεπιδράσει με πολλά στοιχεία που ανήκουν στην ίδια θεματική κατηγορία c, τότε αυτή η κατηγορία εύλογα δεν θα είναι τόσο νέα, δηλαδή *πρωτότυπη*, γι' αυτόν.

Αξιολόγηση της Πρωτοτυπίας μιας Λίστας Συστάσεων

Στην υποενότητα αυτή θα ορίσουμε τον τρόπο μέτρησης της *πρωτοτυπίας* μιας λίστας προτεινόμενων στοιχείων, ο οποίος θα μπορεί να χρησιμοποιηθεί είτε για την *πρωτοτυπία στοιχείου βάσει δημοτικότητας* είτε για την *εξατομικευμένη πρωτοτυπία στοιχείων*. Γι' έναν, λοιπόν, χρήστη u στον οποίο συστήνεται μια *λίστα των κορυφαίων-N προτεινόμενων στοιχείων* υιοθετούμε τον ακόλουθο ορισμό της *πρωτοτυπίας* της προαναφερθείσας λίστας L_u ([Castells et al., 2015], [Vargas and Castells, 2011]):

$$N_{L_u} = \frac{1}{N} \sum_{\forall i \in L_u} N_{u,i} \tag{9.15}$$

όπου $N_{u,i}$ η *πρωτοτυπία* του στοιχείου i για τον υπό εξέταση χρήστη u, και η οποία μπορεί να είναι οποιοδήποτε από τα μοντέλα που εκφράζουν την *πρωτοτυπία ενός στοιχείου* (*item novelty*) και που παρουσιάζονται στη στήλη 3 του Πίνακα 9.5. Σημειώστε ότι η τελευταία στήλη του Πίνακα 9.5 δείχνει το *πλαίσιο (context)* της εμπειρίας και αλληλεπίδρασης των χρηστών με το σύστημα συστάσεων. Για το μοντέλο που βασίζεται στη *δημοτικότητα ενός στοιχείου (item popularity)*, το *πλαίσιο (context)* της εμπειρίας αποτελείται από όλους τους χρήστες που αλληλεπίδρασαν με το στοιχείο i (δηλ. U_i), ενώ για το μοντέλο που βασίζεται στην *απόσταση* (*unexpectedness*, δηλαδή *μέτρηση του απροσδόκητου*), το *πλαίσιο (context)* της εμπειρίας αποτελείται από τα στοιχεία με τα οποία μόνο ο χρήστης u έχει αλληλεπιδράσει στο παρελθόν (δηλαδή το σύνολο I_u).

Metric	Used in	Item Novelty ($N_{u,i}$)	Ideal Item Novelty (N_{max})	Context
Popularity-based (Global Long-Tail)	[Vargas and Castells, 2011]	$-log\frac{\lvert U_i\rvert}{\lvert U\rvert}$	$-log\frac{1}{\lvert U\rvert}$	U_i
Distance-based (Unexpectedness)	[Ge et al., 2010]	$\frac{\sum_{\forall j \in I_u} d(i,j)}{\lvert I_u\rvert}$	d_{max}	I_u
Topic-Coverage (S-Recall)	[Castells et al., 2015]	$\frac{1}{\lvert\{i \in I_u : i \text{ belongs to category c}\}\rvert}$	1	C_u

Πίνακας 9.5: Διαφορετικοί ορισμοί *πρωτοτυπίας* ενός στοιχείου.

Ένα μειονέκτημα της Εξίσωσης 9.15, η οποία αποτυπώνει την *πρωτοτυπία* μιας λίστας συστάσεων L_u, είναι ότι δεν μπορεί να επιβάλλει κάποια ποινή επί της περίπτωσης όπου ένα λιγότερο *πρωτότυπο* (ακριβέστερα, με λιγότερα *πρωτότυπα* σημεία) στοιχείο κατατάσσεται σε καλύτερη θέση στη L_u συγκριτικά μ' ένα άλλο περισσότερο *πρωτότυπο*. Για να επιλυθεί το πρόβλημα αυτό, έχει προταθεί [Coba et al., 2019] η *μετρική* $N\text{-}DCG_u$, η οποία λαμβάνει υπόψη τη σχετική θέση των προτεινόμενων *πρωτότυπων (novel)* στοιχείων μέσα στη λίστα L_u. Πιο συγκεκριμένα, με τη χρήση της εξίσωσης 9.16 υπολογίζουμε *το σωρευτικό κέρδος* $N\text{-}DCG_u$ για κάθε στοιχείο l στη λίστα L_u, το οποίο "κέρδος" αποτελείται από την *πρωτοτυπία ενός στοιχείου* ($N_{u,l}$) όπως αυτή ορίζεται στην τρίτη στήλη του Πίνακα 9.5.

$$N\text{-}DCG_u = N_{u,l_1} + \sum_{i=2}^{N} \frac{N_{u,l_i}}{log_2 i} \tag{9.16}$$

Με την Εξίσωση 9.16, λοιπόν, επιβάλλουμε "ποινή" στα *πρωτότυπα στοιχεία* που προτείνονται χαμηλότερα μέσα στη λίστα L_u. Εν συνεχεία, κανονικοποιούμε το $N\text{-}DCG_u$ έναντι ενός "ιδανικού" μοντέλου πρόβλεψης *πρωτότυπων στοιχείων*. Στην περίπτωσή μας, το "ιδανικό" μοντέλο θεωρεί ότι κάθε στοιχείο στη λίστα L_u έχει μέγιστη *πρωτοτυπία*, N_{max}, η οποία είναι διαφορετική για κάθε μοντέλο *πρωτοτυπίας στοιχείου* όπως φαίνεται στην τέταρτη στήλη του Πίνακα 9.5. Προσέχουμε εδώ ότι η μεταβλητή d_{max}, η οποία παρουσιάζεται στην τέταρτη στήλη του Πίνακα 9.5, εξαρτάται από το εύρος της χρησιμοποιούμενης *συνάρτησης απόστασης*. Έτσι, η "ιδανική" *(ideal) N-IDCG* υπολογίζεται ως εξής:

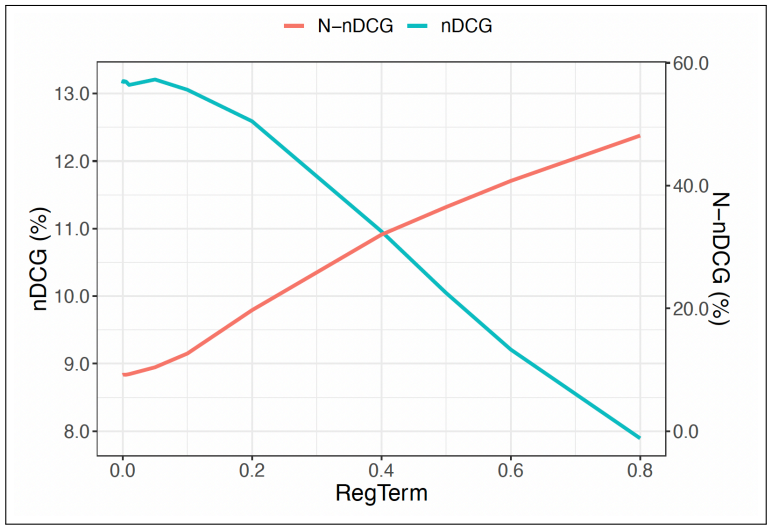
$$N\text{-}IDCG = N_{max} + \sum_{i=2}^{N} \frac{N_{max}}{log_2 i} \tag{9.17}$$

Τέλος, το $N\text{-}nDCG_u$ είναι ο λόγος μεταξύ του $N\text{-}DCG_u$ και του *N-IDCG*:

$$N\text{-}nDCG_u = \frac{N\text{-}DCG_u}{N\text{-}IDCG} \tag{9.18}$$

Ακόμη, σημειώνουμε ότι ο Coba και οι λοιποί συν-συγγραφείς [Coba et al., 2019] απέδειξαν πειραματικά ότι υπάρχει μια *αντιστάθμιση* (*trade-off*) μεταξύ της *ακρίβειας* και της *πρωτοτυπίας* των προτεινόμενων στοιχείων της λίστας , όπως φαίνεται και στο Σχήμα 9.3 που αναφέρεται στο σύνολο δεδομένων *Movielens 1M*.

Από το προαναφερθέν, λοιπόν, Σχήμα 9.3 γίνεται εύκολα κατανοητό ότι θα πρέπει να "θυσιάσει" κανείς την *ακρίβεια (nDCG)*, για να πετύχει υψηλότερο βαθμό *πρωτοτυπίας (N-nDCG)* των προτεινόμενων στοιχείων της λίστας συστάσεων ώστε το σύστημα που τις παράγει να γίνει περισσότερο ενδιαφέρον για τους χρήστες, κάτι που ενδεχομένως θα αυξήσει σε βάθος χρόνου την ικανοποίησή τους. Στο σημείο τομής των δύο καμπυλών στο Σχήμα 9.3 συνήθως πετυχαίνουμε την καλύτερη *αντιστάθμιση (trade-off)* μεταξύ *αποτελεσματικότητας* και *πρωτοτυπίας* των προτεινόμενων στοιχείων της λίστας συστάσεων.



Σχήμα 9.3: *Καμπύλη ακρίβειας* και *πρωτοτυπίας* συστάσεων για το σύνολο δεδομένων *Movielens 1M*.

Συνοψίζοντας, στην υποενότητα αυτή επικεντρωθήκαμε στη μελέτη της *πρωτοτυπίας* μιας λίστας προτεινόμενων στοιχείων. Παρόλα αυτά, δεν έχουμε εξετάσει ακόμη την *ποικιλομορφία (item diversity)* μιας τέτοιας λίστας. Η μετρική της *ποικιλομορφίας* μιας λίστας συστάσεων, λοιπόν, μελετά το κατά πόσο διαφοροποιούνται μεταξύ τους τα στοιχεία της λίστας με στόχο να συμπεριληφθούν σ' αυτήν στοιχεία που ανήκουν σε διαφορετικούς τομείς ενδιαφέροντος του υπό εξέταση χρήστη. Στην επόμενη υποενότητα θα αναλύσουμε την προαναφερθείσα μετρική (*item diversity*).

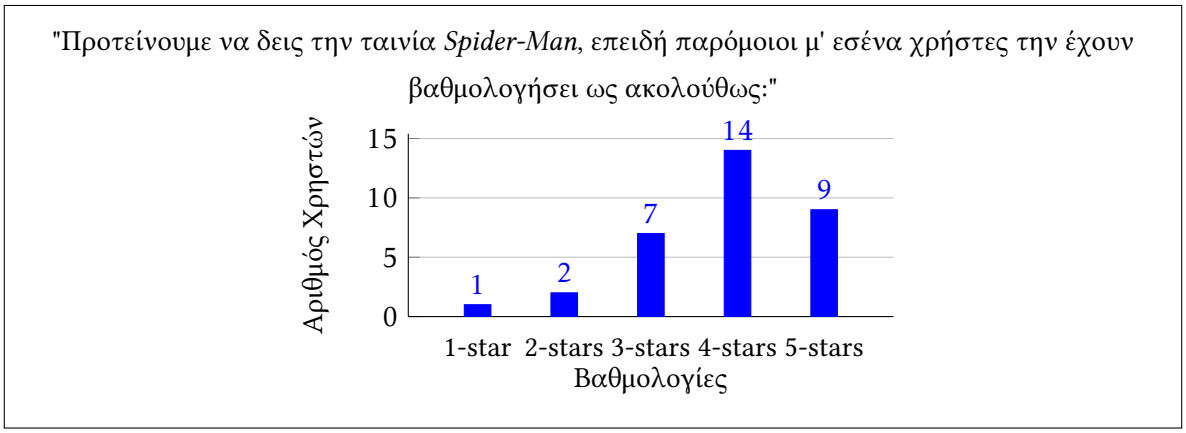
9.8.2 Ποικιλόμορφες Συστάσεις (Diversified Recommendations)

Με τον όρο *ποικιλομορφία συστάσεων* εννοούμε την ύπαρξη "διαφορετικότητας" εντός της λίστας προτεινόμενων στοιχείων. Μ' αλλά λόγια, τα στοιχεία που συμπεριλαμβάνονται σ' αυτή δεν είναι παρόμοια, αλλά αφορούν σε διαφορετικούς τομείς ενδιαφέροντος του υπό εξέταση χρήστη.

Ένας απλός τρόπος για να διακρίνουμε, επομένως, τη *μετρική item novelty* από τη *μετρική item diversity* είναι ότι για να μετρήσουμε την *πρωτοτυπία* ενός στοιχείου χρειαζόμαστε μόνο ένα προτεινόμενο στοιχείο (δηλαδή, μπορούμε απλώς να μετρήσουμε εάν ένα στοιχείο είναι *novel* γι' έναν χρήστη), ενώ για τη μέτρηση της *ποικιλομορφίας* μιας λίστας συστάσεων χρειαζόμαστε τουλάχιστον 2 προτεινόμενα στοιχεία (π.χ. για να μπορέσουμε να εξετάσουμε πόσο διαφοροποιούνται μεταξύ τους). Ακόμη, στη μετρική του *item diversity* δεν χρειάζεται να γνωρίζουμε το προφίλ του υπό εξέταση χρήστη.

Προκειμένου, τώρα, να ορίσουμε την *item diversity*, υποθέτουμε αρχικά την ύπαρξη ενός μέτρου *ομοιότητας* μεταξύ των στοιχείων. Στη συνέχεια, ορίζουμε την *ποικιλομορφία των στοιχείων της λίστας συστάσεων (intra-list diversity)* μετρώντας τη μέση *απόσταση* μεταξύ όλων των πιθανών ζευγαριών στοιχείων ανά δύο που εμπεριέχονται σ' αυτή με την παρακάτω Εξίσωση:

$$ILD = \frac{1}{|R|(|R|-1)} \sum_{i \in R} \sum_{j \in R} d(i,j)$$



Σχήμα 9.4: Με τη χρήση ενός ιστογράμματος βαθμολογιών παρομοίων χρηστών αιτιολογείται η σύσταση της ταινίας *Spider-Man*.

όπου R η λίστα των προτεινόμενων στοιχείων και $d(i, j)$ ένα *μέτρο απόστασης* μεταξύ των στοιχείων i και j. Αυτό το μέτρο μπορεί να βασίζεται στα χαρακτηριστικά των στοιχείων (π.χ. είδος ταινίας, κατηγορία άρθρου κ.λπ.).

9.8.3 Αιτιολογημένες Συστάσεις (Explainable Recommendations)

Υπάρχει ένα πλήθος διαφορετικών τρόπων αιτιολόγησης μιας σύστασης. Για παράδειγμα, έχουν προταθεί 21 διαφορετικές *διεπαφές χρήστη (user interfaces)* για την επεξήγηση των συστάσεων που βασίζονται στο *συνεργατικό φιλτράρισμα (CF)* [HKR]. Επίσης, έχει αποδειχθεί ότι η *επεξήγηση σε στυλ χρήστη* (*user style of explanation*), είναι ένας πολύ πειστικός τρόπος αιτιολόγησης μιας σύστασης [HKR].

Στο παρόν θα περιγράψουμε την *επεξήγηση σε στυλ χρήστη* [BM0], η οποία αιτιολογεί μια σύσταση με βάση το τί στοιχεία έχουν αξιολογήσει άλλοι χρήστες, παρόμοιοι μ' εμάς, δεδομένου ότι έχουν ίδια συμπεριφορά (αγοράζουν ίδια προϊόντα, βαθμολογούν με τον ίδιο τρόπο τις υπηρεσίες κ.λπ.).

Η δε αξία αυτής της προσέγγισης αποδεικνύεται επίσης από το γεγονός ότι αρκετές μεγάλες εταιρείες λιανικής πώλησης προϊόντων, όπως η *Amazon*, η *Zalando* κ.λπ., υιοθέτησαν την *επεξήγηση σε στυλ χρήστη*, η οποία συνήθως δηλώνεται στις διάφορες ιστοσελίδες ηλεκτρονικού εμπορίου με την ακόλουθη σύσταση προς τον υποψήφιο χρήστη-αγοραστή: "Οι πελάτες που αγόρασαν το είδος X αγόρασαν επίσης τα είδη $Y, Z \ldots$". Μπορούμε να δούμε την *επεξήγηση σε στυλ χρήστη* στο Σχήμα 9.4, το οποίο εμφανίζει ένα ιστόγραμμα των βαθμολογιών (οριζόντιες μπλε μπάρες) των *κοντινότερων γειτόνων του υπό εξέταση χρήστη* για ένα προτεινόμενο στοιχείο.

Καθορισμός του Πόσο Επεξηγήσιμο είναι ένα Προτεινόμενο Στοιχείο

Για να υπολογίσουμε την *επεξηγηματική δύναμη/ισχύ (explainability power)* ενός στοιχείου i γι' έναν χρήστη u με βάση τις προτιμήσεις των γειτόνων του, πρέπει πρώτα να βρούμε τους παρόμοιους γείτονες του υπό εξέταση χρήστη u από τον αρχικό πίνακα βαθμολογιών χρήστη-στοιχείου. Πιο συγκεκριμένα, μπορούμε να χρησιμοποιήσουμε τη *συσχέτιση Pearson* (Εξίσωση 9.19), η οποία μετράει την *ομοιότητα* μεταξύ δύο χρηστών u και v:

$$\text{sim}(u,v) = \frac{\sum_{\forall i \in S}(r_{u,i} - \overline{r}_u)(r_{v,i} - \overline{r}_v)}{\sqrt{\sum_{\forall i \in S}(r_{u,i} - \overline{r}_u)^2}\sqrt{\sum_{\forall i \in S}(r_{v,i} - \overline{r}_v)^2}}, \quad S = I_u \cap I_v. \tag{9.19}$$

όπου $r_{u,i}$, $r_{v,i}$ οι βαθμολογίες στο στοιχείο i του χρήστη u και του χρήστη v αντίστοιχα, I_u και I_v το σύνολο των στοιχείων που έχουν βαθμολογηθεί από τους χρήστες u και v αντίστοιχα, και $\overline{r}_u$, $\overline{r}_v$ οι μέσοι όροι των βαθμολογιών των u και v επί των στοιχείων που έχουν βαθμολογήσει από κοινού. Η Εξίσωση 9.19 λαμβάνει υπόψη μόνο το σύνολο των στοιχείων S, τα οποία έχουν βαθμολογηθεί και από τους δύο χρήστες.

Κατόπιν, ορίζουμε έναν αριθμό k *πλησιέστερων γειτόνων* για τον υπό εξέταση χρήστη u, και μέσα σ' αυτήν τη γειτονιά μπορούμε να ορίσουμε $NN^k(u)_{i,r}$ ως τον αριθμό των k *πλησιέστερων γειτόνων* του u που έχουν δώσει βαθμολογία r στο στοιχείο i. Σημειώστε ότι το $r \in [1..\text{R}]$ είναι μια βαθμολογία εντός του διαστήματος τιμών της καθορισμένης κλίμακας βαθμολογιών.

Έπειτα, υπολογίζουμε για έναν χρήστη u στον οποίο προτείνεται ένα στοιχείο i το πόσο επεξηγήσιμο είναι αυτό για τον προαναφερθέντα, μετρώντας στη γειτονιά του (δηλαδή στο προφίλ των k πιο παρόμοιων χρηστών του) πόσες φορές το στοιχείο i έχει υψηλή βαθμολογία. Μ' άλλα λόγια, κατασκευάζουμε έναν *πίνακα επεξηγηματικότητας χρήστη-στοιχείου* E που περιέχει την "επεξηγηματική δύναμη" ενός στοιχείου i για έναν χρήστη u όπως φαίνεται στην Εξίσωση 9.20:

$$E_{u,i} = \sum_{\substack{\forall r \in R \\ r \geq P_\tau}} r * |NN^k(u)_{i,r}| \tag{9.20}$$

όπου $NN^k(u)$ το σύνολο των k *πλησιέστερων γειτόνων* ενός χρήστη και R το σύνολο όλων των διαφορετικών βαθμολογιών στην κλίμακα βαθμολογίας. Σημειώνουμε, επίσης, ότι $NN^k(u)_{i,r}$ είναι το σύνολο των *πλησιέστερων γειτόνων του υπό εξέταση χρήστη* u, ο οποίος έδωσε μια "θετική" βαθμολογία r (πάνω από το όριο P_τ) στο στοιχείο i κατά το παρελθόν. Όπως είναι αναμενόμενο, οι τιμές του πίνακα E θα πρέπει να είναι υψηλότερες για στοιχεία που έχουν μεγάλη βαθμολογία από πολλούς χρήστες και χαμηλότερες στην αντίθετη περίπτωση.

Παράδειγμα 9.6 Με βάση την Εξίσωση 9.20 μπορούμε να μετρήσουμε πόσο μπορεί να *αιτιολογηθεί* η σύσταση ενός στοιχείου (*επεξηγησιμότητα*) και για το παράδειγμα του Σχήματος 9.4. Όσον αφορά στο τελευταίο (Σχήμα 9.4), η *επεξηγηματική ισχύς* για την *αιτιολόγηση της σύστασης* της ταινίας *Spider-Man* είναι 127 ($1 \times 1 + 2 \times 2 + 3 \times 7 + 4 \times 14 + 5 \times 9 = 127$).

Αξιολόγηση της Επεξηγησιμότητας μιας Λίστας Συστάσεων

Η Abdollahi και η Nasraoui [Abdollahi and Nasraoui, 2017] μετρούν την *επεξηγησιμότητα* μιας λίστας σύστασης προτεινόμενων στοιχείων L_u, η οποία δηλαδή παρέχεται στον χρήστη u, ως τον λόγο του αριθμού των *επεξηγήσιμων στοιχείων* μέσα στη L_u προς το μέγεθος του αριθμού των *κορυφαίων-N προτεινόμενων στοιχείων (Mean Explainability Precision - MEP)*, όπως φαίνεται στην Εξίσωση 9.21:

$$MEP = \frac{1}{|U|} \times \sum_{u \in U} \frac{|\{i : i \in L_u, E_{u,i} > 0\}|}{N} \tag{9.21}$$

όπου U το σύνολο των χρηστών.

Το κύριο μειονέκτημα του *MEP* είναι το γεγονός ότι δεν μπορεί να διακρίνει ένα περισσότερο *επεξηγήσιμο* στοιχείο από ένα λιγότερο [Coba et al., 2019]. Επιπλέον, είναι προφανές ότι για μικρές τιμές του *κατωφλιού επεξηγησιμότητας* P_T το *MEP* είναι σχεδόν πάντα ίσο με 1, ακόμη και εάν το L_u αποτελείται από στοιχεία με μικρή ισχύ *επεξηγησιμότητας*. Έτσι, για να αντιμετωπίσουμε τα προαναφερθέντα προβλήματα, υιοθετούμε τη μετρική $E\text{-}nDCG_u$, η οποία λαμβάνει επίσης υπόψη τη σχετική θέση των προτεινόμενων στοιχείων μέσα στη λίστα συστάσεων L_u.

Το πρώτο λοιπόν βήμα στον υπολογισμό του $E\text{-}DCG_u$ είναι η σωρευτική άθροιση του *κέρδους* που προκύπτει από κάθε προτεινόμενο στοιχείο. Στην περίπτωσή μας, το *κέρδος* για κάθε στοιχείο l στη L_u χαρακτηρίζει την ισχύ της *επεξηγησιμότητας* του στοιχείου ($E_{u,l}$), όπως αυτή ορίζεται στην Εξίσωση 9.20 για την *επεξήγηση σε στυλ χρήστη* (*user style of explanation*).

Το δεύτερο βήμα στον υπολογισμό του $E\text{-}DCG_u$ εφαρμόζει το *Discounted Cumulative Gain* στο προαναφερθέν *σωρευτικό κέρδος* όπως φαίνεται στην Εξίσωση 9.22:

$$E\text{-}DCG_u = E_{u,l_1} + \sum_{i=2}^{N} \frac{E_{u,l_i}}{log_2 i} \tag{9.22}$$

Με βάση αυτήν την Εξίσωση (9.22) υπολογίζουμε το *κέρδος* για κάθε θέση στοιχείου μέσα στη L_u, για να επιβάλουμε μια "ποινή" στα στοιχεία που προτείνονται χαμηλότερα στη διατεταγμένη λίστα, γεγονός που συνεπάγεται την πρόσθετη απαιτούμενη προσπάθεια από την πλευρά του χρήστη για να τα εξετάσει, προκειμένου να λάβει την αντίστοιχη αιτιολόγηση της σύστασής τους [Charles et al., 2008].

Το τρίτο βήμα είναι η κανονικοποίηση του $E\text{-}DCG_u$ έναντι του "ιδανικού" μοντέλου. Στην περίπτωσή μας, το "ιδανικό" μοντέλο θεωρεί ότι όλα τα προτεινόμενα στοιχεία στη L_u έχουν τη μέγι-

στη *επεξηγησιμότητα* E_{max}. Επομένως, λαμβάνεται ως δεδομένο ότι όλα τα προτεινόμενα στοιχεία στη λίστα συστάσεων L_u έχουν βαθμολογηθεί με τη μέγιστη βαθμολογία *max* (π.χ. με 5 στην κλίμακα βαθμολογίας 1 έως 5) από όλους τους γείτονες του υπό εξέταση χρήστη. Έτσι, το "ιδανικό" *(idealized) E-IDCG* υπολογίζεται ως εξής:

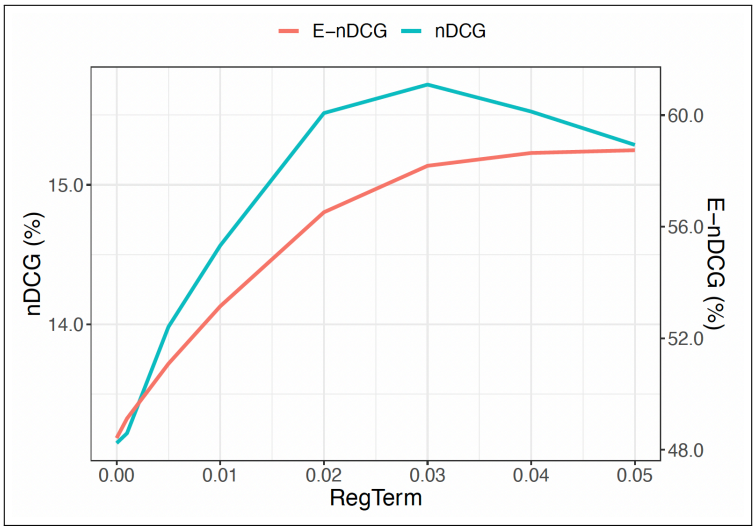
$$E\text{-}IDCG = E_{max} + \sum_{i=2}^{N} \frac{E_{max}}{log_2 i}, \tag{9.23}$$

$$E_{max} = max \times |NN| \tag{9.24}$$

όπου $|NN|$ το μέγεθος της γειτονιάς που χρησιμοποιείται για την *επεξήγηση μιας σύστασης*. Επιπρόσθετα, $E\text{-}nDCG_u$ είναι ο λόγος του *E-DCG* προς το *E-IDCG*, όπως φαίνεται στην Εξίσωση 9.25:

$$E\text{-}nDCG_u = \frac{E\text{-}DCG_u}{E\text{-}IDCG} \tag{9.25}$$

Τέλος, σημειώνουμε ότι ο Coba και οι λοιποί συν-συγγραφείς [Coba et al., 2019] απέδειξαν πειραματικά ότι υπάρχει μια *θετική συσχέτιση* (*positive correlation*) μεταξύ της *ακρίβειας* και της *επεξηγησιμότητας* των στοιχείων της λίστας συστάσεων, όπως φαίνεται στο Σχήμα 9.5 για το σύνολο δεδομένων *Movielens 1M*. Η *θετική συσχέτιση* οφείλεται στο γεγονός ότι, συνήθως όταν προτείνονται στοιχεία που μπορούν να αιτιολογηθούν καλύτερα, αυτά τα στοιχεία είναι και δημοφιλή, δεδομένου ότι έχουν αλληλεπιδράσει μαζί τους πάρα πολλοί χρήστες, και γι' αυτόν τον λόγο αυξάνεται και η *ακρίβεια* της *αποτελεσματικότητας* της λίστας συστάσεων, όπως φαίνεται στο Σχήμα 9.5.



Σχήμα 9.5: *Καμπύλη αποτελεσματικότητας και επεξηγησιμότητας* συστάσεων για το σύνολο δεδομένων *Movielens 1M*.

9.8.4 Αιτιολόγηση Συστάσεων βάσει Στοιχείου (Item Style of Explanation)

Στην προηγούμενη υποενότητα μελετήσαμε αποκλειστικά την *αιτιολόγηση συστάσεων βάσει των δεδομένων που αφορούν στον χρήστη (user style of explanation).* Στην παρούσα θα αναλύσουμε την *αιτιολόγηση συστάσεων βάσει δεδομένων που αφορούν στο στοιχείο (item style of explanation).*

Με τη χρήση λοιπόν των *κανόνων συσχέτισης (association rules)* ένα προτεινόμενο στον υπό εξέταση χρήστη στοιχείο μπορεί επίσης να θεωρηθεί ως στοιχείο που μπορεί να αιτιολογηθεί γι' αυτόν, εάν αυτός έχει αλληλεπιδράσει στο παρελθόν μ' ένα ή περισσότερα στοιχεία του συστήματος συστάσεων τα οποία (στοιχεία) έχουν υψηλό *support (υποστήριξη)* και *confidence (εμπιστοσύνη)* με το προτεινόμενο στοιχείο [Peake and Wang, 2018].

Συνεπώς, μπορούμε να εξάγουμε *κανόνες συσχέτισης* από σύνολα στοιχείων τα οποία έχουν βρεθεί να είναι συχνά στον πίνακα βαθμολογίας χρήστη-στοιχείου με βάση τα καθορισμένα *μέτρα υποστήριξης και εμπιστοσύνης* (*support* και *confidence*). Ένας *κανόνας συσχέτισης* της μορφής $\{X \Rightarrow Y\}$ είναι ισχυρός με υποστήριξη $supp$, εάν ένα ποσοστό $supp\%$ των αλληλεπιδράσεων χρήστη-στοιχείων περιέχει την ένωση των συνόλων στοιχείων $supp(X \cup Y)$. Η *υποστήριξη* (*support*) είναι ένα μέτρο που αποτυπώνει πόσο συχνά παρατηρείται ένα *μοτίβο (pattern).* Ο *κανόνας συσχέτισης* ισχύει με *εμπιστοσύνη* $conf$, εάν ένα ποσοστό $conf\%$ της διάδρασης χρήστη-στοιχείων που περιέχει το σύνολο στοιχείων *X*, περιέχει επίσης το σύνολο στοιχείων *Y* όπως ορίζεται από το κλάσμα $supp(\mathrm{X} \cup \mathrm{Y})/supp(\mathrm{X})$.

Επομένως, θέτοντας ένα *κατώφλι t* που ορίζεται από τον ερευνητή για τα $supp$ και $conf$, εντοπίζουμε όλους τους *κανόνες συσχέτισης* μεγέθους 2 (δηλ. $\{X \Rightarrow Y\}$) από τον πίνακα βαθμολογίας χρήστη-στοιχείου. Με βάση αυτούς λοιπόν είμαστε σε θέση να κατασκευάσουμε έναν *πίνακα επεξηγησιμότητας στοιχείου-στοιχείου* $E_{x,y}$, ο οποίος εμπεριέχει την πληροφορία για το πόσο *επεξηγήσιμο* είναι το y με βάση το x: άρα, για έναν χρήστη u, στον οποίο προτείνεται ένα στοιχείο y, υπολογίζουμε πόσο *επεξηγήσιμο* είναι το στοιχείο y με βάση τα στοιχεία x_1,x_2 κ.λπ. που του έχουν αρέσει στο παρελθόν και έχουν τιμές πάνω από ένα *κατώφλι t* στον *πίνακα επεξηγησιμότητας στοιχείου* (δηλαδή $E_{x,y}$ >t) όπως φαίνεται στην Εξίσωση 9.26:

$$Explainability_{u,y} = \Pr\left(E_{x,y} > t | x \in I_u\right) = \frac{\left|I_u \cap S_y\right|}{\left|S_y\right|} \tag{9.26}$$

όπου I_u το σύνολο των στοιχείων που άρεσαν στον χρήστη u στο παρελθόν, και S_y το σύνολο των στοιχείων που εξηγούν το αντικείμενο y από τον *πίνακα επεξηγησιμότητας στοιχείου* $E_{x,y}$. Όπως είναι αναμενόμενο, οι τιμές του πίνακα E θα πρέπει να είναι υψηλότερες για στοιχεία που έχουν υψηλή βαθμολογία από πολλούς χρήστες και χαμηλότερες στην αντίθετη περίπτωση.

Τέλος, ένα παράδειγμα για ένα *επεξηγήσιμο στοιχείο βάσει δεδομένων που αφορούν στο στοιχείο* (*item style of explanation*) είναι το ακόλουθο: "Επειδή βαθμολογήσατε την ταινία *Terminator*, σας προτείνουμε την ταινία *Avatar*". Άρα, η ταινία *Avatar* είναι ένα *επεξηγήσιμο* στοιχείο για τον υπό εξέταση χρήση, επειδή του άρεσε στο παρελθόν η ταινία *Terminator*. Ο *κανόνας συσχέτισης* της

μορφής $\{Επεξήγηση\} \Rightarrow \{Recommendation\}$ (π.χ. $\{Terminator \Rightarrow Avatar\}$) είναι γνωστός ως *επεξηγηματικό στυλ βάσει στοιχείου* ([Bilgic and Mooney, 2005], [Herlocker et al., 2000]) και αποτυπώνει το στοιχείο -από το προφίλ του χρήστη- με τη μεγαλύτερη επιρροή το οποίο είναι υπεύθυνο για ένα προτεινόμενο στοιχείο. Ένα άλλο παράδειγμα θα μπορούσε να ήταν: "Επειδή δείξατε ενδιαφέρον για άλλες 3 ταινίες δράσης, *Avengers, Rambo, Indiana Jones,* σας προτείνουμε την ταινία *Terminator*."

9.9 Ερωτήσεις Κεφαλαίου

1. Ένα σύστημα συστάσεων προτείνει σ' έναν χρήστη 20 ταινίες εκ των οποίων οι 8 του αρέσουν. Συνολικά όμως του αρέσουν 40 ταινίες από αυτές που υπάρχουν στον κατάλογο ταινιών του συστήματος συστάσεων. Η *ακρίβεια (precision)* τότε θα είναι:

   Α) 8/40

   Β) 20/40

   Γ) 8/20

   Δ) 20/8

2. Ένα σύστημα συστάσεων προτείνει σ' έναν χρήστη 40 ταινίες εκ των οποίων οι 16 του αρέσουν. Συνολικά όμως του αρέσουν 60 ταινίες από αυτές που υπάρχουν στον κατάλογο ταινιών του συστήματος συστάσεων. Η *ανάκληση (recall)* τότε θα είναι:

   Α) 40/60

   Β) 16/60

   Γ) 16/40

   Δ) 40/16

3. Στον παρονομαστή της *μετρικής precision* και *recall* προσμετρώνται πάντα (μεταξύ άλλων) τα:

   Α) TP (true positives)

   Β) FN (false negatives)

   Γ) FP (false positives)

   Δ) TN (true negatives)

4. Στο καρτεσιανό σύστημα συντεταγμένων η *καμπύλη ROC* υπολογίζει το *True Positive Rate (TPR)* στον άξονα y έναντι του *False Positive Rate (FPR)* στον άξονα x, ενώ οι συντεταγμένες ενός σημείου πρόβλεψης είναι *TPR* και *FPR*. Ένα μοντέλο είναι "ιδανικό", όταν οι συντεταγμένες του σημείου πρόβλεψης είναι:

Α) (0,0)

Β) (1,1)

Γ) (1,0)

Δ) (0,1)

5. Ποιο από τα παρακάτω στάδια δεν υλοποιείται από τη μέθοδο *User-based Collaborative Filtering (συνεργατικό φιλτράρισμα βάσει χρηστών)*:

   Α) Δημιουργία *πίνακα ομοιοτήτων χρηστών (users).*

   Β) Δημιουργία *πίνακα ομοιοτήτων αντικειμένων (items).*

   Γ) Εύρεση *κοντινότερων γειτόνων.*

   Δ) Πρόβλεψη βαθμολογίας χρηστών για τα *αντικείμενα.*

6. Ποια η διαφορά μεταξύ *μετρικής Precision* και *nDCG* στα συστήματα συστάσεων;

   Α) Η *μετρική Precision* λαμβάνει υπόψη τα *True Positives* της λίστας προτεινόμενων στοιχείων.

   Β) Η *nDCG* λαμβάνει υπόψη τα *True Positives* της λίστας προτεινόμενων στοιχείων.

   Γ) Η *nDCG* λαμβάνει υπόψη τη σειρά κατάταξης των στοιχείων που προτείνονται.

   Δ) Η *μετρική Precision* λαμβάνει υπόψη τη σειρά κατάταξης των στοιχείων που προτείνονται.

7. Ένα *μοντέλο πρόβλεψης* έχει τις ακόλουθες προβλέψεις: *True Positives*= 100, *False Positives*= 50, *True Negatives*= 50 και *False Negatives*= 5. Το *False Positive Rate (FPR)* (*ποσοστό ψευδώς θετικών προβλέψεων*) της *καμπύλης ROC* είναι ίσο με:

   Α) 50/(50+50)

   Β) 5/(50+50)

   Γ) 100/(50+50)

   Δ) 50/(50+100)

8. Στο *πρόβλημα πρόβλεψης/συμπλήρωσης βαθμολογιών* ενός πίνακα χρηστών–στοιχείων (*Rating prediction* ή *matrix completion problem*) ορίζεται μια *αντικειμενική συνάρτηση* με την οποία επιδιώκουμε:

   Α) Να μεγιστοποιήσουμε τη διαφορά μεταξύ πραγματικής και προβλεπόμενης βαθμολογίας των στοιχείων από τους χρήστες.

   Β) Να μεγιστοποιήσουμε τη διαφορά ως προς την κατάταξη των στοιχείων μεταξύ προβλεπόμενης και πραγματικής λίστας στοιχείων που προτιμούν οι χρήστες.

Γ) Να ελαχιστοποιήσουμε τη διαφορά μεταξύ πραγματικής και προβλεπόμενης βαθμολογίας των αντικειμένων από τους χρήστες.

Δ) Να ελαχιστοποιήσουμε τη διαφορά ως προς την κατάταξη των αντικειμένων μεταξύ προβλεπόμενης και πραγματικής λίστας αντικειμένων που προτιμούν οι χρήστες.

9.10 Ασκήσεις Προγραμματισμού

Άσκηση 1: Δημιουργία ενός περιβάλλοντος αξιολόγησης αλγορίθμων συστάσεων
Μας ζητείται να αξιολογήσουμε την απόδοση ενός αλγορίθμου (*user-based CF*) στα δεδομένα της προγραμματιστικής άσκησης του 2ου κεφαλαίου, τα οποία αφορούν σε 25 είδη ταινιών που αρέσουν ή όχι σε 6 θεατές:
(α) Αρχικά, θα πρέπει να υλοποιήσουμε έναν κατάλληλο διαχωρισμό των δεδομένων σε *train* και *test* με τέτοιον τρόπο ώστε για κάθε θεατή κάποια από τα είδη ταινιών που του αρέσουν να παραμένουν στο *σύνολο train* και τα υπόλοιπα να τοποθετούνται στο *σύνολο δεδομένων test.*

(β) Στη συνέχεια, για καθέναν από τους 6 θεατές θα πρέπει προτείνουμε με τη χρήση του *αλγορίθμου user-based CF* τα είδη ταινιών (Drama, Romance κ.λπ.) που ενδιαφέρουν τους παρόμοιους θεατές με τον υπό εξέταση κάθε φορά θεατή.

(γ) Τέλος, θα πρέπει να αξιολογήσουμε την απόδοση του αλγορίθμου ως προς την αποτελεσματικότητα της λίστας των προτεινόμενων ειδών ταινιών με τη χρήση της *καμπύλης Precision-Recall* και της *καμπύλης ROC.*

Μας δίνεται λοιπόν μια *δομή δεδομένων dictionary* με τα ενδιαφέροντα 6 θεατών ως προς τα είδη ταινιών:

```
import pandas as pd
import numpy as np
import math
from typing import Dict, List, Tuple
Vector = List[float]
from sklearn.metrics.pairwise import cosine_similarity

viewer_genres = [
(0, "Action"), (0, "Adventure"), (0, "Animation"), (0, "
```

```
    Biography"), (0, "Black comedy"), (0, "Comedy"), (0, "
    Concert"),
(1, "Crime"), (1, "Documentary"), (1, "Concert"), (1, "
    Animation"), (1, "Drama"),
(2, "Educational"), (2, "Family"), (2, "Fantasy"), (2, "Film
    Noir"), (2, "Game-show"), (2, "History"),
(3, "Horror"), (3, "Educational"), (3, "Music"), (3, "Musical")
    , (3, "Mystery"),
(4, "Mystery"), (4, "Musical"), (4, "Reality-TV"), (4, "Romance
    "),
(5, "Educational"), (5, "Horror"), (5, "Biography"), (5, "
    Romantic comedy"), (5, "Sci-Fi"), (5, "Short Film")
]
```

Για να διαχειριστούμε τα ενδιαφέροντα των θεατών πιο αποδοτικά, μπορούμε να καταργήσουμε τα ids τους και να δημιουργήσουμε μια *δομή δεδομένων λίστας μέσα σε λίστα* (*list within list*) όπου θα παραμείνουν μόνο τα ενδιαφέροντα των θεατών. Τονίζεται ότι κάθε επιμέρους υπο-λίστα αυτής της λίστας εμπεριέχει τις προτιμήσεις ενός θεατή. Επίσης, ένα είδος ταινίας μπορεί να εμφανίζεται σε περισσότερες της μίας υπο-λίστες. Τη *δομή δεδομένων list within list* που δημιουργήσαμε την ονομάζουμε *movie_genres*.

```
movie_genres = [
 ["Action", "Adventure", "Animation", "Biography", "Black
   comedy", "Comedy", "Concert"],
 ["Crime", "Documentary", "Concert", "Animation", "Drama"],
 ["Educational", "Family", "Fantasy", "Film Noir", "Game-show",
    "History"],
 ["Horror", "Educational", "Music", "Musical", "Mystery"],
 ["Mystery", "Musical", "Reality-TV", "Romance"],
 ["Educational", "Horror", "Biography", "Romantic comedy", "Sci
   -Fi", "Short Film"]]
```

Κατόπιν, είναι αναγκαίο να αφαιρέσουμε τις διπλότυπες εμφανίσεις ενός είδους ταινίας στη λίστα *movie_genres* με τον παρακάτω κώδικα:

```
def unique(list1):
    unique_list = []
```

```
    for i in range(len(list1)):
        for j in range(len(list1[i])):
                    if list1[i][j] not in unique_list:
                    unique_list.append(list1[i][j])
    return unique_list

print("Τα είδη ταινιών που εμφανίζονται μόνο μία φορά:")

unique(movie_genres)
```

Έτσι, προκύπτει μια λίστα με τα 25 είδη ταινιών που εμφανίζονται μόνο μία φορά:

Τα είδη ταινίας που εμφανίζονται μόνο μια φορά:
['Action', 'Adventure', 'Animation', 'Biography', 'Black comedy', 'Comedy', 'Concert', 'Crime', 'Documentary', 'Drama', 'Educational', 'Family', 'Fantasy', 'Film Noir', 'Game-show', 'History', 'Horror', 'Music', 'Musical', 'Mystery', 'Reality-TV', 'Romance', 'Romantic comedy', 'Sci-Fi', 'Short Film']

Έπειτα, χρειάζεται να δημιουργήσουμε ένα διάνυσμα που θα εμπεριέχει τις προτιμήσεις (με 1 ή 0) του κάθε θεατή ως προς τα 25 διαφορετικά είδη ταινιών. Πρέπει, άρα, απλώς να επαναδημιουργήσουμε τη λίστα, όπου το κάθε είδος ταινιών θα εμφανίζεται μόνο μία φορά, και στη συνέχεια θα τοποθετούμε το 1 εάν ο θεατής δείχνει ενδιαφέρον για ένα είδος ταινιών (π.χ. πολεμικές), και το 0 εάν δεν ενδιαφέρεται για κάποιο είδος ταινιών (π.χ. ρομαντικές), με τη χρήση του παρακάτω κώδικα:

```
viewer_genre_vectors = []
for viewer_genres in movie_genres:
    viewer_genre_vectors.append ([0] * 25)
for viewer_genres in movie_genres:
    for genre in unique_genres:
        if genre in viewer_genres:
            viewer_genre_vectors[movie_genres.index(
   viewer_genres)][unique_genres.index(genre)] = 1

print("Τα διανύσματα που εκφράζουν τα προφίλ των 6 θεατών είναι τα
   παρακάτω:")

viewer_genre_vectors
```

Τα διανύσματα που εκφράζουν τα προφίλ των 6 θεατών είναι τα παρακάτω:

[[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0],

[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1]]

Ακολούθως, θα υλοποιήσουμε με τον παρακάτω κώδικα τον διαχωρισμό των δεδομένων σε 50% *train* και 50% *test* με τέτοιον τρόπο ώστε για κάθε θεατή τα μισά από τα είδη ταινιών που του αρέσουν να παραμένουν στο *σύνολο train*, και τα υπόλοιπα να τοποθετούνται στο *σύνολο test*.

```
1
2  from random import sample
3  import pandas as pd
4  import numpy as np
5  import math
6  from collections import defaultdict
7
8  def train_test_split(viewer_genre_df, split):
9      train = pd.DataFrame(0, index=range(viewer_genre_df.shape
       [0]), columns=range(viewer_genre_df.shape[1]))
10     test = pd.DataFrame(0, index=range(viewer_genre_df.shape
       [0]), columns=range(viewer_genre_df.shape[1]))
11
12     for viewer in viewer_genre_df.index:
13         genres = viewer_genre_df.iloc[viewer, :] == 1
14         genres = np.where(genres)[0].tolist()
15         n_genre = len(genres)
16         n_train = math.floor(n_genre *split)
```

```
        train_genre = sample(genres, n_train)
        test_genre = [movie for movie in genres if movie not in
    train_genre]

        train.iloc[viewer, train_genre] = 1
        test.iloc[viewer, test_genre] = 1

    return train, test
viewer_genre_df = pd.DataFrame(viewer_genre_vectors)
print(viewer_genre_df)
train, test = train_test_split(viewer_genre_df, 0.5)
print(train)
print(test)
```

ALL DATA

0 1 2 3 4 5 6 7 8 9 ... 15 16 17 18 19 20 21 22 23 24

0 1 1 1 1 1 1 0 0 0 ... 0 0 0 0 0 0 0 0 0 0

1 0 0 1 0 0 0 1 1 1 ... 0 0 0 0 0 0 0 0 0 0

2 0 0 0 0 0 0 0 0 0 0 ... 1 0 0 0 0 0 0 0 0 0

3 0 0 0 0 0 0 0 0 0 0 ... 0 1 1 1 1 0 0 0 0 0

4 0 0 0 0 0 0 0 0 0 0 ... 0 0 0 1 1 1 1 0 0 0

5 0 0 0 1 0 0 0 0 0 0 ... 0 1 0 0 0 0 0 1 1 1 [6 rows x 25 columns]

TRAIN DATA

0 1 2 3 4 5 6 7 8 9 ... 15 16 17 18 19 20 21 22 23 24

0 0 0 1 1 0 1 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0

1 0 0 1 0 0 0 1 0 0 0 ... 0 0 0 0 0 0 0 0 0 0

2 0 0 0 0 0 0 0 0 0 0 ... 1 0 0 0 0 0 0 0 0 0

3 0 0 0 0 0 0 0 0 0 0 ... 0 0 1 0 0 0 0 0 0 0

4 0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 1 1 0 0 0 0

5 0 0 0 0 0 0 0 0 0 0 ... 0 1 0 0 0 0 0 1 0 0 [6 rows x 25 columns]

TEST DATA

0 1 2 3 4 5 6 7 8 9 ... 15 16 17 18 19 20 21 22 23 24

0 1 1 0 0 1 0 1 0 0 0 ... 0 0 0 0 0 0 0 0 0 0

1 0 0 0 0 0 0 0 1 1 1 ... 0 0 0 0 0 0 0 0 0 0

```
2  0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0
3  0 0 0 0 0 0 0 0 0 0 ... 0 1 0 1 1 0 0 0 0 0
4  0 0 0 0 0 0 0 0 0 0 ... 0 0 0 1 0 0 1 0 0 0
5  0 0 0 1 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 1 1 [6 rows x 25 columns]
```

Εν συνεχεία, ακολουθεί ο κώδικας που υπολογίζει τα *predicted_ratings* βάσει του *user-based collaborative filtering* όπως αυτός περιγράφηκε στο τέλος της προγραμματιστικής άσκησης του δευτέρου κεφαλαίου.

```
from scipy.linalg import fractional_matrix_power
from numpy import inf
import math

def ubcf_rank_movies(train, threshold=1):
    R = train.copy()
    P = np.diag(np.sum(R, axis=1))
    P = fractional_matrix_power(P, -0.5)
    PR =  np.matmul(P, R)

    #Δημιουργία του πίνακα ομοιότητας μεταξύ των θεατών
    Svv=np.matmul(PR,np.transpose(PR));
    Svv=np.matmul(Svv,R)
    # υπολογισμός του πίνακα με τις προβλεπόμενες τιμές
    result = Svv
    # μηδένισε τις προβλεπόμενες βαθμολογίες για τα είδη ταινιών που
    βρίσκονται ήδη στα δεδομένα εκπαίδευσης, προκειμένου να μην προτείνουμε
    είδη ταινιών που ήδη γνωρίζουμε ότι αρέσουν στον χρήστη

    msk = train >= threshold
    result[msk] = 0
    print(result)
    return result

def top_n_recommended_genres(result, n):
     return result[:,-n:]
```

Τα αποτελέσματα του παραπάνω κώδικα είναι οι ακόλουθες προβλεπόμενες βαθμολογίες για τον *viewer-genre interaction matrix* στο διάστημα τιμών 0-1.

```
0 1 2 3 4 5 6 7 8 9 ... 15
0   0.0 0.0 0.0 0.0 0.0 0.0 0.408 0.0 0.0 0.0 ... 0.0
1   0.0 0.0 0.0 0.408 0.0 0.408 0.0 0.0 0.0 0.0 ... 0.0
2   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0
3   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0
4   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0
5   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0


16 17 18 19 20 21 22 23 24
0   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
2   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
3   0.408 0.0 0.0 0.0 0.0 0.0 0.408 0.0 0.0
4   0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
5   0.0 0.408 0.0 0.0 0.0 0.0 0.0 0.0 0.0 [6 rows x 25 columns]
```

Τέλος, ο κώδικας που ακολουθεί αξιολογεί τις προβλέψεις του *user-based Collaborative Filtering* με την επιλογή των 10 κορυφαίων προτεινόμενων *genres* και ποσοστό διαχωρισμού *train-test δεδομένων* ίσο με 0,5 με τη χρήση της *συνάρτησης evaluation*(10, 0.5).
Πιο συγκεκριμένα, με τον παρακάτω κώδικα δημιουργείται για κάθε θεατή μια λίστα με τα προτεινόμενα *genres*. Επίσης, παράγονται 2 γραφήματα (*precision-recall curve* και *ROC curve*). Τονίζεται ότι τα γραφήματα λόγω του μικρού μεγέθους των δεδομένων και της *αραιότητας* του συνόλου δεδομένων κάποιες φορές ενδεχομένως να μην εμφανίζονται όπως κανείς θα ανέμενε. Τέλος, παράγεται και η *ακρίβεια* και η *ανάκληση* όταν προτείνονται 10 *genres* σε κάθε θεατή.

```
from statistics import mean
from matplotlib import pyplot as plt

def evaluation(n:int, split:float):

########## Διαχωρισμός δεδομένων εκπαίδευσης και ελέγχου ##########
    train_viewer_genre_vector = train.values.tolist()
    test_viewer_genre_vector = test.values.tolist()

    # Υπολόγισε πόσα συναφή είδη ταινιών υπάρχουν στα δεδομένα ελέγχου
   ανά θεατή
```

```
    list_total_relevants_per_viewer = [sum(test_viewer_genre_vector[i])
                for i in range(len(test_viewer_genre_vector))]

########## Αλγόριθμος UBCF ##########
    ubcf_precisions = []
    ubcf_recalls = []
    ubcf_fpr = []

    rating_threshold=1
    ratings = ubcf_rank_movies(train, threshold=rating_threshold)
    predicted_ratings = np.argsort(ratings,axis=1)
    my_predicted_ratings = np.array(predicted_ratings)

    for i in range(1, n+1):
        recommended_genres = top_n_recommended_genres(my_predicted_ratings, i)
    ubcf_suggestions = np.fliplr(recommended_genres)

    for viewer in range(6):
        prec = []
        rec = []
        fpr = []
        tpr = []
        true_positives = 0
        false_positives = 0
        total_relevants = list_total_relevants_per_viewer[viewer]

        print()
        print('Viewer ' + str(viewer))
        print()

        for i in (range(0,n)):
            sugg_id = ubcf_suggestions[viewer][i]
```

```
43 print(unique_genres[sugg_id] + ": " + str(ratings[sugg_id][
   viewer]))
44
45         if test_viewer_genre_vector[viewer][sugg_id] == 1:
46               true_positives += 1
47         elif test_viewer_genre_vector[viewer][sugg_id] == 0:
48               false_positives += 1
49
50           # precision@i = tp / (i+1)
51           temp_prec = true_positives/(i+1)
52           prec.append(temp_prec)
53
54           #recall@i = tp / (tp+fn)
55           temp_rec = true_positives/total_relevants
56           rec.append(temp_rec)
57
58           # total_relevants = true_positives+false_negatives
59           true_negatives = len(test_viewer_genre_vector[
   viewer]) - total_relevants - false_positives
60           #false positive rate =
61           #false_positive/(false_positive+true_negative)
62           temp_fpr = false_positives/(false_positives+
   true_negatives)
63           fpr.append(temp_fpr)
64
65       ubcf_precisions.append(prec)
66       ubcf_recalls.append(rec)
67       ubcf_fpr.append(fpr)
68
69       # μέση τιμή της precision και recall
70
71       ubcf_avg_precision = [mean(elem) for elem in zip(*
   ubcf_precisions)]
72       ubcf_avg_recall = [mean(elem) for elem in zip(*
   ubcf_recalls)]
73       ubcf_avg_fpr = [mean(elem) for elem in zip(*ubcf_fpr)]
```

```
########## Διαγράμματα και Πίνακας Αποτελεσμάτων #########

    plt.plot(ubcf_avg_recall, ubcf_avg_precision, color = "blue", linestyle="--")
    plt.scatter(ubcf_avg_recall, ubcf_avg_precision, color = "blue")
    plt.legend(["User-based CF"])
    plt.xlabel("Recall")
    plt.ylabel("Precision")
    plt.title("Precision-Recall curve")
    plt.show()
    plt.plot(ubcf_avg_fpr, ubcf_avg_recall, color = "blue")
    plt.scatter(ubcf_avg_fpr, ubcf_avg_recall, color = "blue")
    plt.legend(["User-based CF"])
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title("ROC evaluation")
    plt.show()
    headers = ["Algorithm", "Precision @ {0}".format(n), "Recall @ {0}".format(n)]
    table = [["UBCF", ubcf_avg_precision[-1], ubcf_avg_recall[-1]]]
    print(table, headers)
evaluation(10, 0.5)
```

Το αποτέλεσμα του προηγηθέντος κώδικα είναι:

Προτεινόμενα είδη ταινιών (*movie genres*) ανά θεατή:

Viewer 0

Concert: 0.408

...

Documentary: 0.0

Viewer 1

Biography: 0.408

Comedy: 0.408

...

Documentary: 0.0

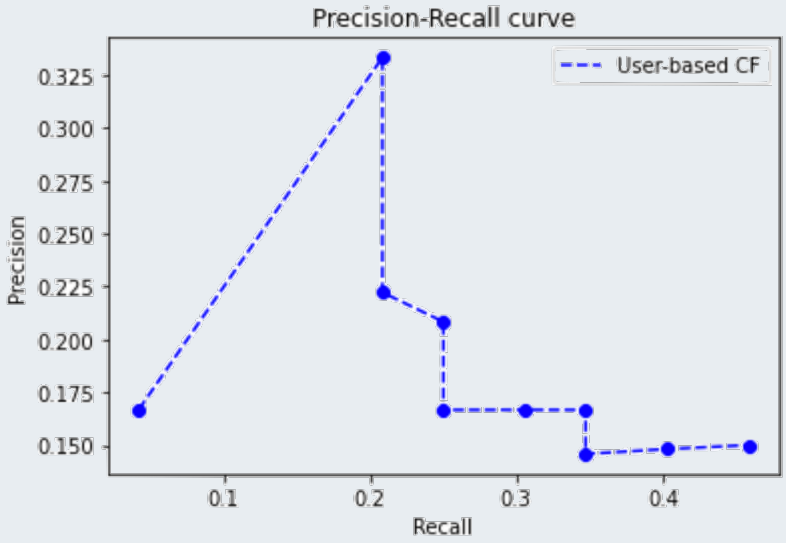
...

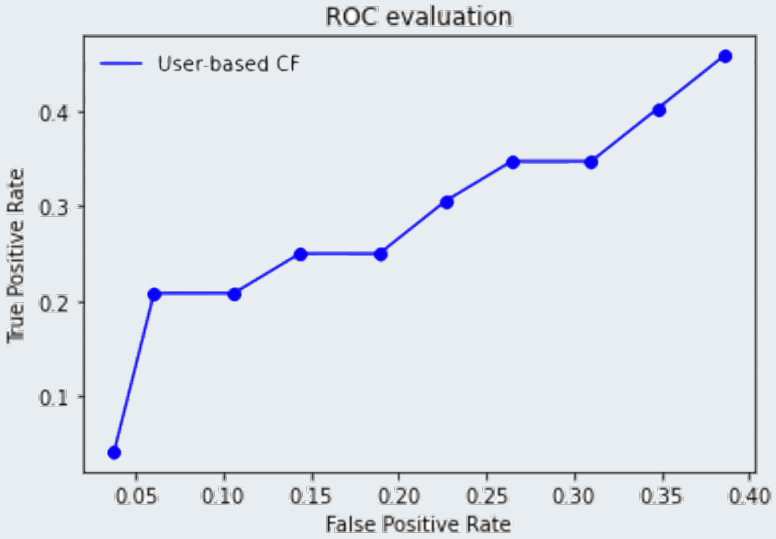
Viewer 5

Music: 0.408

...

Crime: 0.0





Τέλος, εκτυπώνεται η *ακρίβεια* και η *ανάκληση* όταν προτείνονται τα *κορυφαία-10 είδη ταινιών* ανά θεατή:

['Algorithm', 'Precision @ 10', 'Recall @ 10']

['UBCF', 0.15, 0.46]

Βιβλιογραφία

B. Abdollahi and O. Nasraoui. Using explainability for constrained matrix factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, pages 79--83, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4652-8. 10.1145/3109859.3109913. URL http://doi.acm.org/10.1145/3109859.3109913.

M. Bilgic and R. J. Mooney. Explaining Recommendations: Satisfaction vs. Promotion. *Proceedings of Beyond Personalization 2005: A Workshop on the Next Stage of Recommender Systems Research at The 2005 International Conference on Intelligent User Interfaces*, pages 13--18, 2005. 10.1145/1040830.1040839. URL http://www.cs.utexas.edu/~ml/papers/submit.pdfhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.64.4814&rep=rep1&type=pdf%5Cnhttp://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.64.4814.

P. Castells, N. J. Hurley, and S. Vargas. Novelty and diversity in recommender systems. In F. Ricci, L. Rokach, and B. Shapira, editors, *Recommender Systems Handbook, 2nd edition*, pages 881--918, 2015. ISBN 978-1-4899-7637-6.

C. Charles, M. Kolla, G. Cormack, O. Vechtomova, A. Ashkan, S. Buttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *SIGIR Conference*, SIGIR '08, pages 659--666, 2008. ISBN 978-1-60558-164-4.

L. Coba, P. Symeonidis, and M. Zanker. Personalised novel and explainable matrix factorisation. *Data & Knowledge Engineering*, 122:142--158, 2019.

M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 257--260, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-906-0. 10.1145/1864708.1864761. URL http://doi.acm.org/10.1145/1864708.1864761.

J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work - CSCW '00*, pages 241--250, 2000. ISBN 1581132220. 10.1145/358916.358995.

G. Peake and J. Wang. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2060--2069. ACM, 2018.

S. Rendle, C. Freudenthaler, Z. Gantner, and S.-T. Lars. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial*

Intelligence, UAI '09, pages 452--461, Arlington, Virginia, United States, 2009. AUAI Press. ISBN 978-0-9749039-5-8. URL http://portal.acm.org/citation.cfm?id=1795114.1795167.

S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 109--116, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0683-6. 10.1145/2043932.2043955. URL http://doi.acm.org/10.1145/2043932.2043955.