<u>תרגיל בית 2 - אחזור מידע ומערכות המלצה</u>
<u>מגישים</u>: 203016217, 201314796, 300339785

- **Simple Mean**

בהינתן יוזר i ואייטם j הפונקציה תחזיר את ממוצע דירוגי יוזר i ללא תלות באייטם j.
מבחינת סיבוכיות הפונקציה מומשה בצורה שרצה על כל השורות בנתוני האימון בכדי לאסוף את נתוני הדירוגים של כל יוזר (מילון שמפתח הוא היוזר והערך הוא רשימה של הדירוגים שלו) ואז רצה על כל היוזרים בשביל לחשב להם את ממוצע הדירוגים שלהם (מילון שמפתח הוא היוזר והערך הוא ממוצע הדירוגים שלו).
ולבסוף כמובן אנו רצים על כל שורה בנתוני הוולידציה ומחשבים את המרחק מהחיזוי.
שגיאה - 1.049310719086559

- **Linear Regression**

במודל זה החיזוי הוא הממוצע רייטינג הכללי עם התחשבות ב-biases של כל יוזר ואייטם, לשם כך בדקנו כמה יוזרים ואייטמים יש לנו בנתוני האימון ויצרנו שני וקטורים של מערכי numpy באורכים של מספר היוזרים ומספר האייטמים בהתאמה.
עדכון ה-biases נעשה על ידי 10 איטרציות, בכל איטרציה אנו עוברים שורה שורה על נתוני האימון ומעדכנים על פי הנוסחה בדף ההנחיות עם התחשבות במקדם הלמידה ובפרמטר gamma.
במידה והיוזר או האייטם עליהם בוצע החיזוי לא הופיעו בנתוני האימון, החיזוי שהוחזר הוא הממוצע הכללי.
הממוצע הכללי והוקטורים המעודכנים של היוזרים והאייטמים נכתבו לקובץ pickle בשביל שימוש במודל Baseline KNN.

- **KNN**

הפונקציה מומשה בצורה שרצה על כל נתוני האימון ויוצרת מילון דירוגים לכל אייטם (מילון שמפתח הוא אייטם והערך הוא מילון שהמפתח הוא יוזר והערך הוא הדירוג).
כעת הפונקציה למעשה מחשבת את הקורלציות בין כל האייטמים בצורה שרצה עם 2 לולאות על כל האייטמים (עבור על איטרציה של אייטם מסוים נרוץ על כל האייטמים האחרים ונחשב ככה מילון קורלציות בין אייטמים (יש לשים לב כי אם כבר קיים קובץ קורלציות אז הפונקציה מדלגת על חלק זה בקוד) וכעת נכתוב לקובץ אקסל את הקורלציות בין האייטמים שלנו בצורה שרצה עבור כל צמד אייטמים וכותבת את הקורלציה שלהם לתוך האקסל (צמד יופיע פעם אחת ולא פעמיים).
אם הקובץ אקסל כבר קיים אז נרוץ עליו בצורה שרצה על כל צמד אייטמים ונכתוב לתוך מילון קורלציות בין האייטמים שלנו את הקורלציות.

לבסוף נרוץ על כל שורה בקובץ הוולידציה שלנו וניתן תחזית ליוזר ואייטמים המסויימים שמופיעים בשורה מסויימת כדי לחשב את מרחק התחזית מהדירוג האמיתי שבקובץ.
חישוב התחזית עבור יוזר i ואייטם j מתבצע בצורה שרצה על כל האייטמים שהיא בנתוני האימון ובודקת האם הם רלוונטיים מבחינת זה שהם דורגו על ידי יוזר i ושיש להם איזושהי קורלציה לאייטם j לאחר שהכנסנו לתוך רשימה כל כל האייטמים הרלוונטיים ניקח רק את ה k אייטמים עם הקורלציה הכי גבוהה ואיתם נבצע את החישוב של הפרדיקציה (נשים לב שאם אנו נדרשים לתת חיזוי על יוזר או אייטם שלא מופיעים לנו בנתוני האימון אנחנו נותנים חיזוי של ממוצע כלל הדירוגים מנתוני האימון).
שגיאה - 1.0014306961089237

- **Baseline KNN**

במודל זה השתמשנו בוקטורי ה-biases שחושבו במודל Linear Regression, נתונים אלו נשמרו במערכי numpy, ובקורלציות שחושבו במודל KNN, נתונים אלה נשמרו במילון (מילון שמפתח הוא אייטם והערך הוא מילון שהמפתח הוא אייטם שני והערך הוא הקורלציה בין שני האייטמים).
לאחר שקראנו את הפרמטרים אנו מייצרים מילון דירוגים באופן דומה למה שמתואר במודל KNN.
לאחר מכן אנו מבצעים חיזוי שעושה שימוש גם ב-biases וגם בקורלציות לפי הנוסחה בקובץ ההנחיות.
במידה ויוזר או אייטם לא נמצאים בנתוני האימון החיזוי המוחזר הוא הממוצע הכללי, במידה ואין אייטמים

שדומים לאייטם עליו מבוצע החיזוי אני חוזים עם ה-baises בלבד ללא הקורלציות, בדומה לחיזוי במודל Linear Regression.

- **Matrix Factorization**

הפונקציה מומשה ע"י שמירת 2 מטריצות (מערכים רב מימדיים), P ו-Q כאשר כל שורה בהן מייצגת משתמש ו-אייטם בהתאמה.
כל שורה הינה בעלת K מימדים לטנטיים (בדוגמא אצלנו K=24). בנוסף, מימשנו שני מערכים, ל-Biases של היוזרים ושל האייטמים. נאתחל את שני מערכי הbiases ואת הערכים במימדים הלטנטיים באמצעות התפלגות נורמלית מסביב ל-0 עם סטיית תקן של 0.01. בנוסף, חישבנו את הממוצע של כלל הדירוגים(mu).
כעת, נרוץ מספר קבוע של אפוקים (10 בדוגמא שלנו) כשבכל אפוק נרוץ על הרשומות, נחשב את פרדיקציית הרייטינג ליוזר והאייטם הספציפי ונחשב את ההפרש של התחזית לעומת הערך האמיתי (נסמן בתור ה"טעות"): $e_{ui} = r_{ui} - q_i^T p_u$
לאחר מכן, נעדכן את הBiases :

$$b_u = b_u + lr(e_{ui} - \lambda b_u)$$
$$b_i = b_i + lr(e_{ui} - \lambda b_i)$$

נשים לב שגודל ה"צעד" שמבוצע נקבע ע"י ה-Learning rate, כאשר במקרה שלנו נקבע עם ערך של 0.01.
לאחר מכן, שוב בשיטת SGD, נעדכן את הערכים (הלטנטים) בעבור כל יוזר ואייטם ע"י עדכונים הערכים בהתאם לכיוון הטעות שקיבלנו- ע"י שימוש בנוסחאות:

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$
$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

כאשר ניתן לראות כי כיוון הטעות (חיובי או שלילי) ישפיע על כיוון ה"תיקון" שנעשה בבכל איטרציה.
בסיום כל אפוק נחשב את ערך פונקציית המטרה, וניעזר בפונקציית calc_regularization לחישוב הרגולריזציה. בנוסף, נדפיס את ערך הפונקציה ואת השגיאה הריבועית.
בסיום ריצת כל האפוקים, נריץ את המודל על ה-Validation set שלנו. נקבל שגיאה ריבועית ממוצעת של 0.913 לעומת הדירוג האמיתי.

```
train_mse: 0.8379734941063659
| epoch   # 00 :  target function (train) : 7.202e+05  train_MSE : 0.838
train_mse: 0.8192761523466328
| epoch   # 01 :  target function (train) : 7.041e+05  train_MSE : 0.8193
train_mse: 0.8133212242190345
| epoch   # 02 :  target function (train) : 6.99e+05  train_MSE : 0.8133
train_mse: 0.8073658988978257
| epoch   # 03 :  target function (train) : 6.938e+05  train_MSE : 0.8074
train_mse: 0.7915763859956254
| epoch   # 04 :  target function (train) : 6.803e+05  train_MSE : 0.7916
train_mse: 0.7682155636049423
| epoch   # 05 :  target function (train) : 6.602e+05  train_MSE : 0.7682
train_mse: 0.7415395886981536
| epoch   # 06 :  target function (train) : 6.373e+05  train_MSE : 0.7415
train_mse: 0.7083664437336579
| epoch   # 07 :  target function (train) : 6.088e+05  train_MSE : 0.7084
train_mse: 0.673524244426522
| epoch   # 08 :  target function (train) : 5.788e+05  train_MSE : 0.6735
train_mse: 0.6410915478241029
| epoch   # 09 :  target function (train) : 5.51e+05  train_MSE : 0.6411
validation_rmse:
0.9125530585477556
```

- **סיכום:**

Model	**RMSE Validation**
Simple Mean	1.049310719086559
Linear Regression	0.9678148131789798
KNN	1.0014306961089237
Baseline KNN	0.935445150526125
Matrix Factorization	0.9125530585477556