УДК 004.021:004.85

ДОСЛІДЖЕННЯ РОБОТИ АЛГОРИТМУ ML.NET ДЛЯ ОБРОБКИ КОРИСТУВАЦЬКИХ РЕКОМЕНДАЦІЙ

Верепа Д. С.
email: denys.verepa@nure.ua
Науковий керівник – к.т.н., ас. Кобилін І. О.
Харківський національний університет радіоелектроніки, каф. ІНФ,
м. Харків, Україна

This work explores the ML.NET algorithm for processing user recommendations. It addresses the challenge of large-scale data and diverse user preferences. The research employs matrix factorization and collaborative filtering techniques. The MatrixFactorizationTrainer is central in learning latent factors for users and items. An Alternating Least Squares (ALS) method is used to optimize predictions. Key hyperparameters such as latent dimensionality, iteration count, and regularization are tuned. The study demonstrates ML.NET's capacity to build scalable, effective recommendation systems.

Цифровий простір стрімко розвивається, і персоналізовані рекомендаційні системи стали невід'ємною складовою багатьох сервісів. Однак побудова ефективних рекомендаційних систем є складним завданням через великі обсяги даних та різноманітність уподобань користувачів. Необхідна автоматизована обробка даних, яка здатна виявити приховані шаблони в поведінці користувачів і надати релевантні поради в режимі реального часу. Традиційні підходи (наприклад, ручне налаштування правил або фільтрація за вмістом) часто не справляються з цією складністю, тому все більшої ваги набувають машинно-навчені алгоритми рекомендацій.

Рішенням цієї проблеми є використання спеціалізованих алгоритмів у рамках ML.NET – відкритої платформи машинного навчання від Microsoft для .NET. ML.NET реалізує колаборативну фільтрацію на основі методу матричної факторизації, що дозволяє автоматично виявляти зв'язки між користувачами та продуктами за історією взаємодій.

Зокрема, бібліотека надає клас MatrixFactorizationTrainer, який є основним інструментом для навчання моделі рекомендацій. Цей клас будує модель на основі наявних даних про вподобання: входом служать три колонки – ідентифікатор користувача, ідентифікатор елемента (товару, фільму тощо) та числова оцінка або індикатор взаємодії. Внутрішньо ці ідентифікатори конвертуються у ключі (цілі індекси) для формування координат у матриці користувач-елемент.

Результатом навчання є модель (MatrixFactorizationModelParameters), що містить латентні фактори для кожного користувача та елемента і здатна передбачати рейтинг або ступінь цікавості до нового елемента.

Для налаштування алгоритму використовується клас MatrixFactorizationTrainer.Options, який дозволяє визначити ключові гіперпараметри: розмірність простору латентних факторів (параметр ApproximationRank), кількість ітерацій тренування (NumberOfIterations), коефіцієнти регуляризації та тип функції втрат.

Зокрема, опцією LossFunction можна обрати стандартну квадратичну втрату для завдання регресії або режим “one-class” (SquareLossOneClass) для роботи з неявними вподобаннями, коли явні рейтинги відсутні і всі наявні взаємодії трактуються як позитивні впливи.

Сутністю математичного алгоритму є матрична факторизація, що лежить в основі рекомендаційного алгоритму ML.NET, намагається наблизити велику розріджену матрицю взаємодій R розмірності $m \times n$ (де m – кількість користувачів, n – кількість елементів) добутком двох матриць нижчого рангу:

$$R = U \times V^T \tag{1}$$

де U має розмір $m \times f$, а $V - n \times f$ (f – число латентних факторів). Рядок матриці U відповідає вектору характеристик (профілю) певного користувача, а стовпець матриці V^T (або рядок V) – вектору характеристик відповідного елемента. Таким чином, кожен користувач і кожен продукт представлено у спільному f-вимірному латентному просторі.

Прогнозована оцінка взаємодії користувача u з елементом i обчислюється як скалярний добуток їхніх векторів:

$$\hat{r}_{ui} = u_u \times v_i \tag{2}$$

де u_u – f-вимірний вектор користувача u з матриці U, а v_i – f-вимірний вектор елемента i з матриці V.

Для знаходження матриць U і V алгоритм мінімізує різницю між передбаченими значеннями $\hat{r}_{ui}$ та відомими значеннями r_{ui} з навчальних даних. Функція помилки, що оптимізується під час навчання, включає суму квадратів похибок та член регуляризації для запобігання переобученню:

$$\sum_{(u,i)\in K} \left(r_{ui} - u_u \times v_i\right)^2 + \lambda \times \left(\sum_{u=1}^{m} \left\|u_u\right\|^2 + \sum_{i=1}^{n} \left\|v_i\right\|^2\right) \tag{3}$$

де K – множина індексів відомих оцінок у матриці (спостережувані взаємодії), а λ – коефіцієнт регуляризації, що штрафує занадто великі значення компонент векторів u та v. Мінімізація цього виразу є задачою методу найменших квадратів.

Алгоритм, який використовується в ML.NET для розв’язання цієї оптимізаційної задачі, – це метод Alternating Least Squares (ALS), або чергування найменших квадратів. ALS ітеративно покращує розв’язок: на кожному кроці він фіксує одна з матриць (U або V) і обчислює іншу шляхом розв’язання систем рівнянь найменших квадратів, потім навпаки. При

фіксованій матриці V оптимальні значення векторів U знаходяться розв'язанням так званих нормальних рівнянь для кожного користувача u:

$$u_u = \left(\sum_{i:(u,i)\in K} v_i \times v_i^T + \lambda \times I \right)^{-1} \times \sum_{i:(u,i)\in K} v_i \times r_{ui} \quad (4)$$

де I – одинична матриця розмірності f, а сума береться по всіх елементах i, для яких відомий рейтинг від користувача u. Аналогічно оновлюються вектори v_i для кожного елемента при фіксованих $\mathrm{u_u}$ Чергуючи ці кроки, алгоритм поступово зменшує середньоквадратичну похибку і сходиться до розв'язку – наближення вихідної матриці R матрицями U та V.

Метод ALS є стійким і ефективним для великомасштабних даних, оскільки кожен крок зводиться до розв'язання незалежних менших задач, що добре піддаються паралелізації.

Вбудований в ML.NET алгоритм рекомендацій на базі матричної факторизації дозволяє автоматизувати обробку великих масивів користувацьких даних і формувати персоналізовані рекомендації високої якості. За допомогою латентно-факторних моделей вдається врахувати приховані вподобання користувачів і схожість між елементами, яку важко виявити традиційними методами. Математично обґрунтований підхід (ALS) забезпечує збалансовану точність прогнозів та продуктивність алгоритму.

Таким чином, поєднання можливостей платформи ML.NET із сучасними методами колаборативної фільтрації створює потужний інструмент для побудови рекомендативних систем, спроможних адаптивно підлаштовуватися під індивідуальні потреби кожного користувача.

Список використаних джерел:

1. Microsoft Tutorial: Create, evaluate, and score a recommendation system. URL: https://learn.microsoft.com/en-us/fabric/data-science/retail-recommend-model

2. Nikola M. Zivkovic. Machine Learning with ML.NET – Recommendation Systems. 2021. URL: https://ar5iv.labs.arxiv.org/html/1905.05715

3. Guide to Product Recommendation Using ML.NET. URL: https://analyticsindiamag.com/deep-tech/guide-to-product-recommendation-using-ml-net/

4. Koren, Y., Bell, R., Volinsky, C. Matrix Factorization Techniques for Recommender Systems. 2009. URL: https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf