Documentação Research Challenge 1 – Sistemas de Recomendação

Francisco Teixeira Rocha Aragão - 2021031726
franciscoaragao@protonmail.com
Departamento de Ciência da Computação - UFMG
Belo Horizonte, Minas Gerais, Brasil

Abstract

Sistemas de recomendação são fundamentais na atualidade para entregar uma melhor experiência do usuário nas plataformas, além de contribuir para maiores lucros das organizações. Esse trabalho implementa um sistema de recomendação baseado em filtragem colaborativa para atribuir notas a itens para diferentes usuários.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

filtragem, colaborativa, recomendação

ACM Reference Format:
Francisco Teixeira Rocha Aragão - 2021031726. 2018. Documentação Research Challenge 1 – Sistemas de Recomendação. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Sistemas de Recomendação - RC1).* ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introdução

O primeiro trabalho prático proposto na disciplina de sistemas de recomendação implementa os conceitos iniciais da disciplina sobre recomendações para novos usuário e itens, recomendações com base nos vizinhos (filtragem colaborativa) e similaridade entre elementos do sistema. Dessa forma, a seguir será explicado mais sobre a abordagem utilizada, testes realizados e estratégias para realização da atividade.

2 Metodologia

A abordagem utilizada foi a de filtragem colaborativa, mais especificamente focada em itens, com toda a recomendação focada em: o quanto o usuário U gosta de itens similares ao item I? Essa estratégia foi pensada tendo como base a ideia das informações e cálculos de similaridade entre itens serem mais 'estáveis' em comparação a abordagem tendo em vista os usuários. Assim, com essa ideia inicial o código foi organizado com a seguinte estrutura:

- Etapa 1 - Leitura dos parâmetros via linha de comando e manipulação dos dados de entrada pela biblioteca Pandas
- Etapa 2 - Normalização dos valores de entrada. Tal ação foi necessária para assim remover o viés inerente em cada usuário durante as suas votações.
- Etapa 3 - Realização da recomendação com base na similaridade de cosseno tendo em vista diferentes cenários:
  - Usuário e item não são frequentes na base de dados
  - Usuário não é frequente
  - Item não é frequente
  - Usuário e item frequentes
- Etapa 4 - Resultados salvos e impressos na tela (saída padrão).

Vale mencionar algumas decisões e detalhes de implementação adotados durante o trabalho. Como exemplo, durante a etapa de normalização, são salvos em dicionários as informações de cada item: usuários que avaliaram e os respectivos ratins, além dos valores das médias dos votos de cada usuário. Tal fato é importante devido a utilidade dessas informações, que serão usados durante o cálculo das recomendações, necessitando de diversas visitas a base de dados para a obtenção desses valores.

Com isso, durante a normalização dos dados de entrada, como todo o arquivo 'ratings' é lido, essa ação é aproveitada para então serem calculados os valores mencionados acima. Também vale destacar que tais informações são salvas em arquivos .json, para que em futuras execuções do programa (para a mesma base de dados de entrada), não seja necessário recalcular esses valores.

Sobre a recomendação, o processo foi feito pensando em diferentes cenários como mencionado acima. Primeiramente, foi-se utilizado um parâmetro destinado a 'frequência' dos itens e dos usuários para a recomendação ser realizada. Tal fato é motivado tendo em vista que a estratégia de filtragem colaborativa sofre com o problema de 'cold start', quando novos itens ou usuários precisam receber recomendações. Além disso, como o cálculo da similaridade leva em consideração elementos em comum envolvendo os vizinhos, não seria interessante computar a similaridade para usuários ou itens pouco presentes na base de dados. Desse modo, na versão final enviada o valor 7 foi utilizado para filtrar a frequência de ratings que cada item deveria ter e que cada usuário deveria ter feito para a estratégia da recomendação colaborativa ser utilizada.

Deste modo, caso tanto o item quanto o usuário forem novos na base de dados, uma recomendação genérica é executada, em que é retornado o valor médio de todos os itens da base de dados. Caso somente o usuário seja novo, é retornado a média de ratings do item em questão. Caso o item seja novo, é retornado a média dos votos do usuário alvo. Por fim, caso ambos usuário e item sejam frequentes, é calculada a similaridade entre o item alvo e os itens avaliados pelo usuário, para no final ser feita e recomendação.

Vale destacar também que o cálculo da similaridade entre o item alvo e os itens avaliados pelo usuário somente é levado em

Sistemas de Recomendação - RC1, Novembro 11, 2024, Belo Horizonte, MG

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

consideração caso seja maior que um threshold = 0.3. Assim, caso a similaridade seja menor que esse valor, esse item avaliado pelo usuário é desconsiderado para a recomendação do item alvo. Esses valores de similaridade também são salvos em um dicionário para facilitar a futura verificação.

3 Testes realizados

Testes realizados em uma máquina com Debian 12, kernel 6.1.0, CPU Intel i5 - 11º geração e 16 GB de ram.

Ambos os valores relacionados a 'frequência' e 'threshold' foram obtidos com base em experimentação, com diferentes valores sendo testados e submetidos no kaggle. Outro fato importante diz respeito a utilização dos dicionários, agrupando informações dos usuários itens. Sem essa otimização, a execução do trabalho ocorria em mais de 10 minutos utilizando a biblioteca Pandas para manipulação e filtragem dos dados. Com a utilização dos dicionários, a execução demora menos de 2 minutos. Assim, com tais otimizações de performance e com os valores de frequencia = 7 e threshold = 0.3, foi possível obter o erro de: 1.21977 no kaggle.

4 Complexidade

Sendo M = número de itens Sendo N = número de usuários Considerando leitura e escrita de arquivos com tempo O(1).

- Etapa 1

O(1) Apenas leitura dos valores iniciais

- Etapa 2

O(N * M) no pior caso se a matriz de avaliações seja completa em que todo usuário avalia todos os itens.

Na prática, a matriz de avaliações não é completa e os usuários não avaliam todos os itens.

- Etapa 3

O (N * M) * O (M) no pior caso, preciso calcular a similaridade entre todos os N usuários em comum dos itens, para cada um dos M possíveis vizinhos, para cada item M.

Na prática isso não ocorre, por conta das checagens de cada possível caso de recomendação, além das similaridades calculadas ficarem armazenadas em um dicionário.

- Etapa 4

O(1) escrita na tela do resultado final

- Total

O(N * M * M) no pior caso.

5 Execução

Para executar o programa, basta seguir:

* Importante – Utilizar python 3.9.13

```
python3 -m venv <nome do ambiente>
source <caminho até o ambiente>/bin/activate
pip install -r requirements
python3 main.py <ratings.csv> <targets.csv>
```

Assim, o código irá imprimir na tela o resultado final com o arquivo csv com as recomendações. Caso necessário salvar o resultado em um arquivo csv com o nome 'output_file.csv", basta executar o código com o comando:

```
python3 main.py <ratings> <targets> --storeOutput Y
```

6 Conclusão

Com o presente trabalho foi possível exercitar os conceitos de filtragem colaborativa, recomendação baseada em item, cálculo de similaridade além de estratégias para otimização da execução e diminuição dos erros. Vale destacar que outras abordagens poderiam ser utilizadas, como SVD para redução de dimensionalidade, além de uma análise mais cuidadosa dos dados para obtenção de melhores parâmetros de 'frequência' e do 'threshold', ajustando assim a abordagem a diferentes tipos de cenários.