TP1 - Sistemas de Recomendação com SVD

THAÍS FERREIRA DA SILVA - 2021092571

O trabalho prático proposto envolve a predição de avaliações dadas por alguns usuários para certos itens utilizando algum sistema de recomendação colaborativo personalizado. Por isso, apresentarei neste documento as escolhas que tomei para obter um bom sistema de recomendação.

Additional Key Words and Phrases: Recomendação, SVD, SGD

ACM Reference Format:
Thaís Ferreira da Silva - 2021092571. 2023. TP1 - Sistemas de Recomendação com SVD. 1, 1 (September 2023), 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUÇÃO

Para esse trabalho prático, foi proposto a construção de um sistema de recomendações colaborativo, onde apenas implementações de recomendação personalizadas são aceitáveis. Por esse motivo, escolhi realizar com Matrix Factorization e Stochastic gradient descent (SGD) responsável por um resultado superior a modelos baseados em usuário e itens.

2 METODOLOGIA

Como foi comentado anteriormente a metodologia escolhida engloba a fatoração de matrizes com o algoritmo simples de SGD para recalcular os valores, desta forma o funcionamento do algoritmo depende da escolha de 4 parâmetros, k (tamanho da matriz de valores singulares de tamanho k * k), epochs (tempo de treinamento do sistema de recomendação), alpha e lambda (valores de correção da matriz).

Inicialmente foi necessário ler os arquivos ratings.csv e target.csv e realizar um breve tratamento nos dados para separa-los em 3 colunas User, Itens e Ratings onde todos os ratings de target.csv foram inicializados como 0.

Após esse processo, dois dicionários foram criados para relacionar os nomes presentes nos arquivos com uma sequencia de index de 0 até o número de usuários ou itens. Esse dicionário é o que permite a criação das matrizes P e Q necessárias para o SVD, já que por padrão todos os usuários e itens estão anotados por padrão utilizando hexadecimal de 10 dígitos.

Author's address: Thaís Ferreira da Silva - 2021092571, thaisfds@gmail.com.

Depois foi necessário iterar 3 vezes seguidas onde:

- O primeiro loop é realizado em relação ao tempo de treinamento, ou melhoria das matrizes P (usuários*k) e Q (k*itens).
- O segundo loop é responsável por calcular o erro referente a nota dada pelos usuários e a nota calculada com a reconstrução da fatoração da matriz. Esse erro é um dos parâmetros utilizados no próximo loop para recalcular valores melhorias para P e Q.
- O terceiro loop temos a atualização de P e Q onde:
  - Pnovo = alpha * (erro * Q - lambda * P)
  - Qnovo = alpha * (erro * P - lambda * Q)

Para verificação do algoritmo a métrica utilizada foi o RMSE (Root Mean Squared Error) que alcançou no melhor dos testes a nota de 0.057 com 10 épocas.

3 COMPLEXIDADE

A maior complexidade do programa esta presente no algoritmo SGD. Neste método é necessário inicializar 2 matrizes de tamanho (usuários * k), e (k * itens), onde k é o tamanho da matriz singular utilizada de tamanho (k * k), e após este processo 3 loops alinhados são realizados. Por esse motivo o algoritmo de SGD possui complexidade O(m*n*k) onde m é o número de épocas, n é são todas as relações existentes entre usuário x item, e k é a reconstrução das matriz P e Q.

Todas as demais operações realizadas ao longo do código possuem complexidade O(1) ou O(n), onde o n é em relação a quantidade de linhas presentes nos arquivos .csv de entrada.

4 TESTES

Para alcançar a pontuação 1.29608 na competição, 13 submissões e diversos testes prévios foram realizado. Inicialmente foi necessário escolher a melhor implementação para o projeto, onde o SVD aparentemente seria responsável pelo menor tempo de execução e número de linhas de código. Depois tentei implementar o algoritmo de duas formas diferentes, uma manipulando diretamente os valores das matrizes, e a outra utilizando as ferramentas da biblioteca numpy cujo resultados foram melhor em relação ao tempo de execução.

Por fim, os parâmetros k, alpha, lambda e epochs foram testados em diversas varições, aumentar o número de épocas e o tamanho da matriz de valores singulares não foi muito efetivo, o incremento ocasionou no aumento do tempo de execução entre 30 e 60 segundos ao longo dos testes. Com isso, os parâmetros escolhidos foram k=18, alpha=0.01, lamb=0.02, epochs=10 ocasionando no melhor resultado possível com um tempo de execução de aproximadamente 3:30min.

5 CONCLUSÃO

Podemos concluir através deste projeto, a superioridade dos sistemas der recomendação personalizados em detrimento de recomendações aleatórias ou baseadas em usuário e item, já que os resultados obtidos são melhores com um pequeno aumento no tempo de execução e memória necessária. Além disso, diversas dificuldades foram enfrentadas e ainda estão sem uma resposta exata, como por exemplo: Como teríamos que modificar o algoritmo para torná-lo ainda mais eficiente alterando os valores de alpha e lambida em cada época.

Received 25 Setembro 2023