Collaborative Product Recommendation using Matrix Factorization

Cinthia M. Souza
cinthiasouza@ufmg.br
Universidade Federal de Minas Gerais

ABSTRACT

Este trabalho tem como objetivo a implementação de um recomendador colaborativo de produtos. A solução implementada é uma abordagem *memory-based* que utiliza a técnica de *Matrix Factorization* (MF). A técnica de MF implementada é baseada na solução proposta por Simon Funk[1], que utiliza o método de otimização Stochastic Gradient Descent (SGD). A solução implementada utiliza apenas *feedbacks* explícitos, oriundos de uma base de dados histórica contendo *ratings* de usuários para itens em uma escala da 1-5.

CCS CONCEPTS

• **Computer systems organization → Embedded systems**.

ACM Reference Format:
Cinthia M. Souza. 2021. Collaborative Product Recommendation using Matrix Factorization. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 SOLUÇÃO IMPLEMENTADA

O algoritmo proposto por Simon Funk realiza a fatorização da matriz de user-item matriz em duas matrizes, denominadas aqui como P e Q. A matriz P possui n linhas e k colunas, onde n corresponde ao número de usuários. A matriz Q possui m colunas e k linhas, onde m representa o número de itens. Em ambas as matrizes, o valor k corresponde ao número de fatores latentes. Cada item do sistema está associado a um vetor latente q_i e cada usuário do sistema está associado a um vetor latente p_u. O objetivo é que o produto entre p_u e q_i^T seja uma boa aproximação do *rating* do usuário u para o item i, denominado $\hat{r}_{ui}$.

De acordo com Koren et al. [1], o valor de $\hat{r}_{ui}$, quando calculado como sendo o produto entre p_u e q_i^T, tenta capturar as interações entre usuários e itens que produzem diferentes valores de classificação. Contudo, em muitos casos, essa variação não ocorre somente devido a variação das interações entre usuários e itens, mas, também, a vieses que são independentes de qualquer interação. Neste contexto, a explicabilidade de uma interação obtida apenas utilizando o produto entre p_u e q_i^T não é uma solução ideal. Diante disso, nesta implementação, foi acrescentado os parâmetros b_u e b_i, que indicam os desvios observados do usuário u e do item i e o valor μ. O valor de μ representa a classificação média geral. As predições foram avaliadas com três abordagens para obtenção do μ. Na primeira abordagem, o valor de μ é a média geral de todos os *ratings* de todos os usuários. Na segunda abordagem, o valor de μ é a média dos *ratings* de cada usuário, na terceira abordagem, o valor de μ é a média dos *ratings* de cada item. A terceira estratégia associada a métrica média foi a que obteve os melhores desempenhos e, por isso, é a utilizada na implementação final. Com a adição desses parâmetros, a predição final é dada pela Equação 1, apresenta a seguir.

$$\hat{r}_{ui} = (p_u \cdot q_i^t) + \mu + bu_u + bi_i. \quad (1)$$

O valor de μ é fixo, e os valores de p_u, q_i, b_u e b_i são ajustados utilizando o método SGD a fim de minimizar o erro e_{ui} que é calculado utilizando a Equação 2.

$$e_{ui} = r_{ui} - \hat{r}_{ui}. \quad (2)$$

Os ajustes de p_u, q_i, b_u e b_i são realizados utilizando as Equações 3, 4, 5 e 6. Todas as atualizações são realizadas de modo iterativo e tem como ponto de parada um número de épocas ou o alcance de um valor de erro. Basicamente, o objetivo com essas equações é modificar os parâmetro em uma magnitude proporcional a $alpha$, na direção oposta do gradiente [1]. O parâmetro $lambda$, é o fator de regularização, cujo objetivo é o de reduzir o *overfiting* penalizando de acordo com a magnitude das features.

$$p_u = p_u + \alpha \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u). \quad (3)$$

$$q_i = q_i + \alpha \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i). \quad (4)$$

$$b_u = b_u + \alpha \cdot (e_{ui} \cdot b_i - \lambda \cdot b_u). \quad (5)$$

$$b_i = b_i + \alpha \cdot (e_{ui} \cdot b_u - \lambda \cdot b_i). \quad (6)$$

Nesta implementação, o *rating* final é o valor de $\hat{r}_{ui}$ ajustado para a escala de 1-5. Todas as predições são do tipo float. Foi realizado testes realizando a normalização dos *ratings* com base na média do usuário, contudo, os resultados obtidos forma inferiores aos obtidos sem realizar a normalização. Diante disso, optou-se por não normalizar os ratins. Na Tabela 1 são apresentados os hiperparâmetros utilizando na obtenção do melhor resultado.

2 RESULTADOS

Nessa seção são apresentados os resultados obtidos com os experimentos. A Tabela 1 sumariza os resultados dos experimentos submetidos na plataforma Kaggle e apresenta as mudanças realizadas que geraram impactos no valor da métrica utilizada para avaliação de desempenho. Na primeira coluna, é apresentado o

[1]https://sifter.org/ simon/journal/20061211.html

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Resultados obtidos com os experimentos utilizando hiperparâmetros e abordagens de cálculo de média (Resultado com entrada normalizada)**

Submissão	Média	Inicialização	Saída	Fatores	Alpha	Épocas	Tempo	RMSE
1	Média Geral	Uniforme 0 -1	Inteiro	20	0.1	5	NC	1.58594
3	Média do Usuário	Uniforme 0 -1	Inteiro	20	0.01	5	NC	1.37328
22	Média do Usuário	Uniforme 0 -1	Float	20	0.01	5	NC	1.32818
35	Média do Item	Uniforme 0 -1	Float	20	0.01	5	NC	1.31032
36	Média do Item	Normal 0 a 0.1	Float	50	.001	5	7 - 10 min	1.30822
** 56	Média do Item	Normal 0 a 0.2	Float	40	0.001	5	6 - 8 min	1.46807
59	Média do Item	Normal 0 a 0.3	Float	40	0.01	3	3.61 min	1.31150
60	Média do Item	Normal 0 a 0.1	Float	40	0.01	3	3.54 min	1.30879
61	Média do Item	Normal 0 a 0.1	Float	40	0.001	3	3.46 min	1.30930
62	Média do Item	Normal 0 a 0.1	Float	100	0.001	3	6.61 min	1.30763
63	Média do Item	Normal 0 a 0.1	Float	70	0.001	3	4.83 min	1.30856
64	Média do Item	Normal 0 a 0.1	Float	200	0.001	2	8.98 min	1.31087
65	Média do Item	Normal 0 a 0.1	Float	50	.001	5	5.72 min	1.30852
66	Média do Item	Normal 0 a 0.1	Float	30	0.01	8	5.92 min	1.30807
68	Média do Item	Normal 0 a 0.1	Float	40	0.001	5	4.92 min	1.30800
69	**Média do Item**	**Normal 0 a 0.1**	**Float**	**40**	**0.001**	**5**	**1.86 min**	**1.30753**

número da submissão. Na Segunda coluna, é apresentado o tipo de média utilizado. Os valores dessa coluna são os utilizado como μ utilizado na Equação 1. Nas sete últimas colunas, são apresentadas as formas de inicialização, o tipo do valor predito, a quantidade de fatores latentes gerados, o valor da taxa de aprendizagem, o número de épocas, o tempo de predição e o o valor da métrica *Root Mean Squared Error* (RMSE) obtido com a submissão das predições no Kaggle. Nos primeiros testes não foi computado o tempo total para geração das predições, nestes casos, a célula da tabela foi preenchida com a sigla NC (Não computado).

Com base nos dados apresentados na Tabela 1, verifica-se que, uma dos fatores que geraram maiores ganhos no resultado final foi a mudança da abordagem utilizada para calcular média. Posteriormente, a mudança na inicialização e no formato da saída também ocasionaram impactos significativos. A submissão 56, destacada com **, foi a única que utilizou os dados de entrada normalizados pela média dos *ratings* dos usuários. Como podemos ver pelo valor de RMSE, essa estratégia não alcançou bons resultados. Os demais experimentos mostram o impacto da mudança do número de fatores latentes, taxa de aprendizagem e número de épocas no treinamento do modelo. A ideia neste experimentos é encontrar um bom *trade-off* entre tempo de execução e RMSE. Com o experimento 62, por exemplo, podemos ver que aumentar o número de fatores latentes diminuiu o valor do RMSE, contudo, o tempo de execução do algoritmo também aumento.

Vale ressaltar que, nos primeiros experimentos, cuja coluna de tempo está marcada com NC, não havia uma preocupação em obter resultados dentro do tempo especificado no trabalho. O objetivo inicial, era produzir um solução com resultados coerentes. Embora, a implementação sido realizada tentando utilizar estratégias mais eficientes. A partir do experimento 31, onde foi obtido um valor de RMSE inferior ao do *baseline* de melhor desempenho, todas as variações de parâmetros foi realizada com intuito de obter um resultado que continue tendo um resultado superior ao *baseline* melhor desempenho em termos de RMSE e que o desempenho tenha um bom *trade-off* em relação ao erro e tempo de execução. Para isso, foram necessárias ajustes no código. Dentre elas, tem-se a utilização de dicionário para armazenar as agregações de média, por exemplo. A indexação da matrizes utilizando índices inteiros e sequencias. A substituição de iterações utilizando 'pd.loc' para utilização da função 'pd.groupby' e utilização de 'pd.merge' para, por exemplo, gerar a matriz normalizada. Após essas modificações obteve-se um ganho de desempenho significativo. Conforme apresentado na Tabela 1, o melhor resultado foi obtido com o experimento 69, que obteve um RMSE de 1.30753 com um tempo de execução de 1.86 minutos.

REFERENCES

[1] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.