Roteiro de Atividade Prática

Nome: Turma: .

**Título da atividade: Recomendação baseada em similaridade de gênero**

**Objetivo**

Implementar a prática dos códigos vistos em aula.

**Lista de materiais**

* Um computador com Browser internet;
* Caderno para anotações;
* 1 caneta.

**Procedimento experimental**

ALURA. Machine Learning: intro a sistemas de recomendação em Phyton. 02 Recomendação baseada em similaridade de gênero. Disponível em: <https://cursos.alura.com.br/course/introducao-a-sistemas-de-recomendacao-com-python/task/49602>. Acesso em: 30 maio 2024.

|  |
| --- |
| 1. Transcrição do vídeo: Vamos explorar outra abordagem de recomendação, desta vez considerando os filmes que o usuário já assistiu. Utilizando como exemplo, foi compilada uma lista de filmes "eu\_assisti": 1, 21, 19, 10, 11, 7 e 2. 2. Vamos criar um *array* com esses filmes, atribuindo-o à variável "eu\_assisti". Em seguida, filtraremos esses filmes em nosso conjunto de dados usando filmes.loc[eu\_assisti]. A lista resultante incluirá os filmes na mesma ordem: Toy Story, Get Shorty, Ace Ventura, GoldenEye, The American President, Sabrina e Jumanji. 3. Agora, podemos desenvolver um sistema de recomendação que ignora as informações colaborativas, baseando-se apenas nas informações disponíveis sobre o usuário – no caso, os filmes que assisti e seus respectivos gêneros. 4. Por exemplo, suponha que o último filme que assisti tenha sido o Jumanji original, que pertence aos gêneros aventura, infantil e fantasia. Vamos utilizar esses três gêneros na função filmes.query("generos=='Aventure|Children|Fantasy'") para buscar outros filmes em nosso conjunto de dados que compartilham exatamente esses três gêneros. 5. Ao fazer isso, obteremos diversos filmes desse conjunto de gêneros. Agora, filtraremos esses filmes e os ordenaremos, por exemplo, pelas notas médias. No entanto, é importante considerar que muitos desses filmes possuem um total de votos muito baixo, já que estamos trabalhando com todo o conjunto de dados de filmes. A questão é: devemos considerar filmes com um total de votos baixo? Acredito que não. 6. Portanto, ajustaremos nossa *query*() para basear-se em filmes\_com\_mais\_de\_50\_votos, atribuindo o resultado a uma variável chamada "aventura\_infantil\_e\_fantasia". Em seguida, os ordenaremos da maior média para a menor e exibiremos os 10 primeiros. 7. Nossa lista de recomendações para alguém que assistiu a Jumanji inclui filmes como Harry Potter e a Pedra Filosofal, A História sem Fim (um clássico da minha infância), Crônicas de Nárnia, A Bússola Dourada, o próprio Jumanji e A Chave Mágica (um filme desconhecido para mim). 8. Neste ponto, é interessante remover Jumanji da lista. Antes de utilizar o sort\_values(), realizaremos um *drop*(), passando como parâmetro a variável "eu\_assisti". No entanto, nosso código pode falhar, pois nem todos os filmes listados em eu\_assisti estão presentes em aventura\_infantil\_e\_fantasia. 9. A função *drop*() do Pandas possui um parâmetro adicional chamado "erros", que, por padrão, retorna um erro quando não encontra um elemento. Para evitar esse problema, configuraremos esse parâmetro como errors='ignore'. Desta vez, conseguiremos remover Jumanji da lista com sucesso, resultando em 5 filmes que, com base em nosso conjunto de dados, seriam recomendados para alguém que acabou de assistir a Jumanji – independentemente de sabermos se a pessoa gostou ou não de Jumanji. 10. Observa-se que, nessa abordagem, inicialmente não utilizamos o total\_de\_votos ou a nota\_media. No entanto, ao refinar a recomendação, voltamos a empregar esses critérios: primeiro, para excluir filmes de nicho e, posteriormente, para introduzir alguma preferência na recomendação. |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |