Recomendação de séries de TV utilizando Lógica Fuzzy

Arthur Teixeira Jardim e Marcelo Marchioro Cordeiro

24 de Novembro de 2019

1 Introdução

Após a popularização dos sistemas de *streaming* de séries e filmes, os algoritmos de recomendação estão cada vez mais sendo utilizados e requisitados. O processo de escolha por parte do usuário torna-se demorado e em diversos casos ocasionam a não escolha de algum conteúdo. Um algoritmo de recomendação possui uma grande relevância para as empresas que procuram, através de dados, entender o perfil do consumidor e traçar os assuntos de relevância destes, com o objetivo final de obter um lucro maior. Os sistemas de recomendação para os clientes representa um "conselheiro" que possui a missão de conquistar a confiança ao longo do tempo, consoante a qualidade das recomendações apresentadas [3]. Para o usuário, as recomendações também tornam-se interessantes devido aos grandes catálogos disponíveis e a difícil tomada de decisão ao escolher algo de agrado. Por exemplo a Netflix, popular serviço de *streaming*, força o sistema de recomendações após o usuário criar uma nova conta, assim fazendo perguntas em relação a séries que tenha interesse. Também utilizam-se de dados como o horário que o usuário assiste, aparelho que está utilizando e tempo médio assistido, assim gerando todo o catálogo da página inicial para o usuário [1]. O uso da lógica fuzzy a partir de dados sobre o usuário ajuda a modelar a imprecisão de uma maneira lógica, assim como regras heurísticas para a construção de um sistema de recomendação [5].

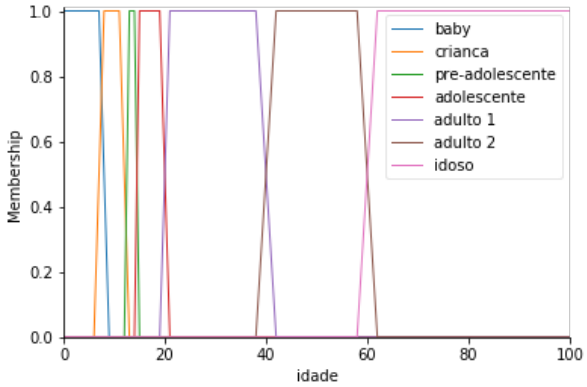
2 Objetivos e Metodologia

O propósito do trabalho é definir as séries que possuam o maior grau de recomendação com base nos dados dos usuário, assim tomando a decisão em condições de incerteza, utilizando-se da lógica fuzzy para a construção do sistema. Foi utilizado a plataforma *Anaconda Distribution* [2], desenvolvendo em *Jupyter* com a linguagem de programação *Python*. Para a construção dos conjuntos fuzzy foi utilizada a biblioteca *skfuzzy* [4] onde foram realizadas as etapas de fuzzyficação, inferência e defuzzyficação.

3 Desenvolvimento

De uma maneira inicial foram utilizadas e analisadas somente três variáveis e duas séries, buscando simular um ambiente de recomendação de séries. O objetivo final é incluir diversos fatores para obter um ambiente mais completo e preciso quanto a recomendação das séries. As variáveis utilizadas foram a idade, com o propósito de classificar as séries por popularidade em cada faixa etária e a duração de episódio, tentando simular um possível tempo médio de episódio visto pelo usuário. A variável de idade (Figura 1) foi dividida em sete categorias diferentes, diferenciando os públicos em cada categoria, são elas: *baby*, criança, pré-adolescente, adulto 1, adulto 2 e idoso. A partição de adulto deve-se ao fato de ser a maior fase da vida de uma pessoa, logo uma pessoa de 25 anos já não possui os mesmos costumes do que uma de 45 anos.

Figura 1 – Gráfico da variável da idade.



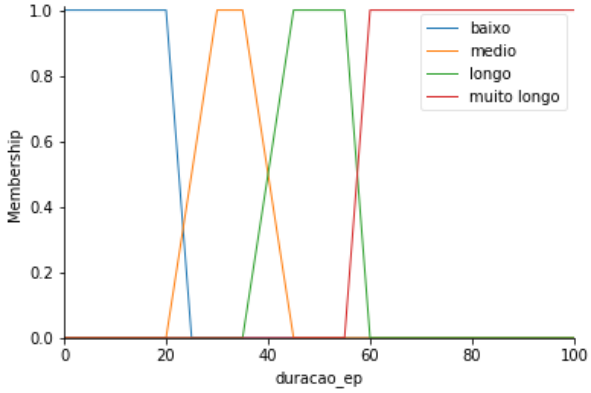
Fonte: autores, 2019.

A variável tempo (Figura 2) de duração teve sua divisão feita em quatro parcelas sendo elas: baixo, médio, longo e muito longo. É importante avaliar esse aspecto pois existem pessoas que manifestam preferências por séries com episódios curtos para assistir em seu momento de descanso, já outros dão preferência para as que possuem episódios com durações maiores.

Cada série inserida no sistema possui uma variável de resultado como mostra a Figura 3. A variável refere-se ao resultado obtido, ou seja, o quanto a série é indicada com base no perfil do usuário. Ela se subdivide em cinco partes, sendo elas: nada, pouco, média, muito e muito alto. Essa variável vai definir se a série deve ou não ser indicada ao usuário em questão.

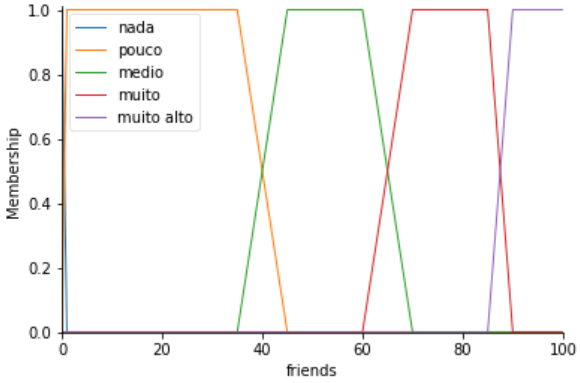
As duas séries analisadas foram *Friends* e *Vikings*, com base nos dados relevantes das séries foram criadas as regras para cada uma, buscando atender as associações feitas com as variáveis idade e duração de episódio. Nota-se que o resultados das regras são armazenados pelas variáveis de cada uma das séries.

Figura 2 – Gráfico da variável tempo de duração.



Fonte: autores, 2019.

Figura 3 – Gráfico da variável resultado de todas séries.



Fonte: autores, 2019.

Na série *Friends* foi levado em conta que o tempo médio de episódio é 22 min e que a série possui um público adolescente e adulto 1 de acordo com a nossa categorização. Na Figura 4 mostra as regras destinadas a série *Friends*.

Já na série Vikings (Figura 5) o tempo médio de episódio é de 47 min e o público adolescente, adulto 1 e adulto 2.

Figura 4 – Regras da série *Friends*.

```
friends1 = ctrl.Rule(idade['baby'] & duracao_ep['baixo'], friends['nada'])
friends2 = ctrl.Rule(idade['baby'] & duracao_ep['medio'], friends['nada'])
friends3 = ctrl.Rule(idade['baby'] & duracao_ep['longo'], friends['nada'])
friends4 = ctrl.Rule(idade['baby'] & duracao_ep['muito longo'], friends['nada'])
friends5 = ctrl.Rule(idade['crianca'] & duracao_ep['baixo'], friends['pouco'])
friends6 = ctrl.Rule(idade['crianca'] & duracao_ep['medio'], friends['pouco'])
friends7 = ctrl.Rule(idade['crianca'] & duracao_ep['longo'], friends['pouco'])
friends8 = ctrl.Rule(idade['crianca'] & duracao_ep['muito longo'], friends['pouco'])
friends9 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['baixo'], friends['muito'])
friends10 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['medio'], friends['medio'])
friends11 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['longo'], friends['pouco'])
friends12 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['muito longo'], friends['nada'])
friends13 = ctrl.Rule(idade['adolescente'] & duracao_ep['baixo'], friends['muito alto'])
friends14 = ctrl.Rule(idade['adolescente'] & duracao_ep['medio'], friends['muito'])
friends15 = ctrl.Rule(idade['adolescente'] & duracao_ep['longo'], friends['medio'])
friends16 = ctrl.Rule(idade['adolescente'] & duracao_ep['muito longo'], friends['pouco'])
friends17 = ctrl.Rule(idade['adulto 1'] & duracao_ep['baixo'], friends['muito alto'])
friends18 = ctrl.Rule(idade['adulto 1'] & duracao_ep['medio'], friends['muito'])
friends19 = ctrl.Rule(idade['adulto 1'] & duracao_ep['longo'], friends['medio'])
friends20 = ctrl.Rule(idade['adulto 1'] & duracao_ep['muito longo'], friends['pouco'])
friends21 = ctrl.Rule(idade['adulto 2'] & duracao_ep['baixo'], friends['muito'])
friends22 = ctrl.Rule(idade['adulto 2'] & duracao_ep['medio'], friends['medio'])
friends23 = ctrl.Rule(idade['adulto 2'] & duracao_ep['longo'], friends['pouco'])
friends24 = ctrl.Rule(idade['adulto 2'] & duracao_ep['muito longo'], friends['nada'])
friends25 = ctrl.Rule(idade['idoso'] & duracao_ep['baixo'], friends['medio'])
friends26 = ctrl.Rule(idade['idoso'] & duracao_ep['medio'], friends['pouco'])
friends27 = ctrl.Rule(idade['idoso'] & duracao_ep['longo'], friends['nada'])
friends28 = ctrl.Rule(idade['idoso'] & duracao_ep['muito longo'], friends['nada'])
```

Fonte: autores, 2019.

Figura 5 – Regras da série *Vikings*.

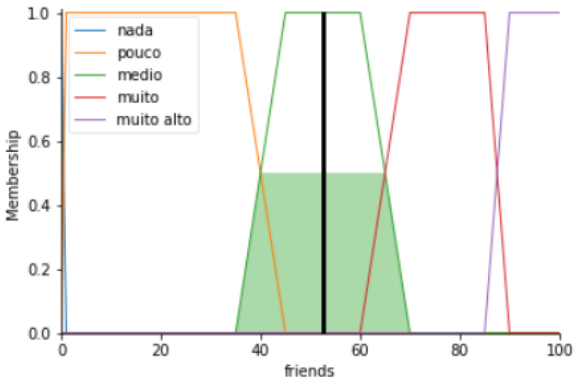
```
vikings1 = ctrl.Rule(idade['baby'] & duracao_ep['baixo'], vikings['nada'])
vikings2 = ctrl.Rule(idade['baby'] & duracao_ep['medio'], vikings['nada'])
vikings3 = ctrl.Rule(idade['baby'] & duracao_ep['longo'], vikings['nada'])
vikings4 = ctrl.Rule(idade['baby'] & duracao_ep['muito longo'], vikings['nada'])
vikings5 = ctrl.Rule(idade['crianca'] & duracao_ep['baixo'], vikings['nada'])
vikings6 = ctrl.Rule(idade['crianca'] & duracao_ep['medio'], vikings['nada'])
vikings7 = ctrl.Rule(idade['crianca'] & duracao_ep['longo'], vikings['pouco'])
vikings8 = ctrl.Rule(idade['crianca'] & duracao_ep['muito longo'], vikings['pouco'])
vikings9= ctrl.Rule(idade['pre-adolescente'] & duracao_ep['baixo'], vikings['pouco'])
vikings10 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['medio'], vikings['medio'])
vikings11 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['longo'], vikings['muito'])
vikings12 = ctrl.Rule(idade['pre-adolescente'] & duracao_ep['muito longo'], vikings['medio'])
vikings13 = ctrl.Rule(idade['adolescente'] & duracao_ep['baixo'], vikings['pouco'])
vikings14 = ctrl.Rule(idade['adolescente'] & duracao_ep['medio'], vikings['muito'])
vikings15 = ctrl.Rule(idade['adolescente'] & duracao_ep['longo'], vikings['muito alto'])
vikings16 = ctrl.Rule(idade['adolescente'] & duracao_ep['muito longo'], vikings['muito'])
vikings17 = ctrl.Rule(idade['adulto 1'] & duracao_ep['baixo'], vikings['pouco'])
vikings18 = ctrl.Rule(idade['adulto 1'] & duracao_ep['medio'], vikings['muito'])
vikings19 = ctrl.Rule(idade['adulto 1'] & duracao_ep['longo'], vikings['muito alto'])
vikings20 = ctrl.Rule(idade['adulto 1'] & duracao_ep['muito longo'], vikings['muito'])
vikings21 = ctrl.Rule(idade['adulto 2'] & duracao_ep['baixo'], vikings['pouco'])
vikings22 = ctrl.Rule(idade['adulto 2'] & duracao_ep['medio'], vikings['muito'])
vikings23 = ctrl.Rule(idade['adulto 2'] & duracao_ep['longo'], vikings['muito alto'])
vikings24 = ctrl.Rule(idade['adulto 2'] & duracao_ep['muito longo'], vikings['medio'])
vikings25 = ctrl.Rule(idade['idoso'] & duracao_ep['baixo'], vikings['pouco'])
vikings26 = ctrl.Rule(idade['idoso'] & duracao_ep['medio'], vikings['medio'])
vikings27 = ctrl.Rule(idade['idoso'] & duracao_ep['longo'], vikings['muito'])
vikings28 = ctrl.Rule(idade['idoso'] & duracao_ep['muito longo'], vikings['medio'])
```

Fonte: autores, 2019.

4 Resultados e Discussões

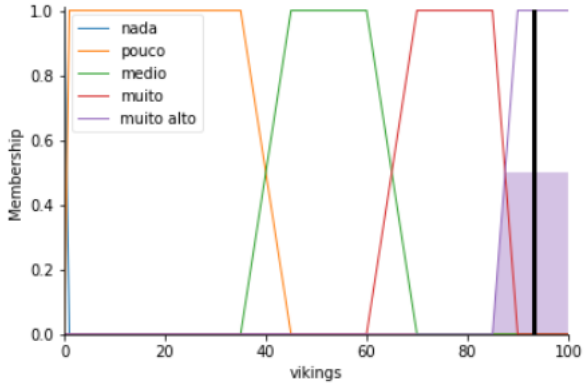
Para um exemplo comparativo foi definido que a informação do usuário referente a idade é de 20 anos e o tempo de episódio é de 45 min. Para a série *Friends* a recomendação da série em porcentagem (Figura 6) foi de 52,50%, enquanto que na série *Vikings* (Figura 7) a recomendação foi de 93,10%. Os resultados mostraram-se coerentes com as regras aplicadas e informações de ambas as séries em um escopo pequeno como o abordado. Em um sistema de recomendação mais robusto algumas outras informações seriam levadas em conta, por exemplo na construção de uma plataforma seria possível analisar o comportamento do usuário e por consequência otimizar o sistema, obtendo resultados mais concretos em relação aos seus costumes. A abordagem utilizada com uma variável de resultado para cada série foi a maneira mais eficiente encontrada para que somente um conjunto fuzzy fosse calculado, caso contrário teríamos que criar um conjunto para cada série. Um conjunto para cada série ocasionaria em um custo computacional muito elevado, impedindo a expansão do sistema que a partir de determinado momento passaria a sofrer com as consequências dessa escolha. Com base nos dados informados conseguimos testar o nosso sistema de recomendação, determinando a série mais recomendada em condições de incerteza utilizando a lógica fuzzy.

Figura 6 – Resultado *Friends*.



Fonte: autores, 2019.

Figura 7 – Resultado *Vikings*.



Fonte: autores, 2019.

Referências

[1] Como funciona o sistema de recomendações da netflix. https://help.netflix.com/pt/node/100639. Acesso em: 15 nov. 2019.

[2] Anaconda Inc. Anaconda versão 2019.10. https://www.anaconda.com/. Acesso em: 15 nov. 2019.

[3] J. Ramos. Algoritmos colaborativos para sistemas de recomendação. https://repositorio-aberto.up.pt/bitstream/10216/58817/1/000145087.pdf, Jul 2010. Acesso em: 15 nov. 2019.

[4] Scikit-Fuzzy Development Team. Skfuzzy versão 0.2. https://pythonhosted.org/scikit-fuzzy/. Acesso em: 20 nov. 2019.

[5] R.R. Yager. *Fuzzy logic methods in recommender systems*, pages 133–149. Fuzzy Sets Syst, 2003.