Aplicação do algoritmo KNN para a obtenção de informações desejadas sobre os elementos do dataset fornecido

Rodrigo José Vianna[1], **Ulysses de Castro Gonçalves**[2]

[1]Faculdade de Computação – Universidade Federal de Mato Grosso Sul (UFMS)
Caixa Postal 549 – 79.070-900 – Campo Grande – MS – Brazil

Abstract. *This article aims to explain the operation of the code sent together. This code receives a dataset with animes and manipulates it to return desirable data.*

Resumo. *Este artigo tem como objetivo explicar o funcionamento do código enviado em conjunto. Tal código recebe um dataset com animes e o manipula para retornar dados desejáveis.*

1. Introdução

O programa animes.py tem como objetivo manipular o dataset animes.csv para exercer seis diferentes funções. Sua primeira função é listar todos animes de um gênero informado pelo usuário, além de informar a quantidade de animes respectivos àquele genêro. A segunda função lista os cinco animes similares ao informado pelo usuário. Na terceira função, é possivel ver todas as informações de um anime que o usuário deseje saber. Na quarta função, é possivel atualizar a quantidade de episódios de um anime. A quinta função permite que o usuário adicione um novo anime e suas respectivas informações e armazena-os no banco de dados. Finalmente, na sexta função o usuário informa um anime que o mesmo deseja remover do banco de dados e o programa o remove.

Dadas as informações, cujo conteúdo apresentava-se impróprio para a execução do agrupamento, fornecidas pelo dataset original, o grupo fez o uso de códigos auxiliares com possíveis resuluções para tratar tal problemática.

Resolvida a citada problemática, o grupo incorporou o desenvolvimento completo de métodos e uma interface para a interação programa-usuário.

2. Desenvolvimento

2.1. Problemas enfrentados

O dataset inicialmente apresentava inconsistências por falta de dados, dificultando assim, a execução do knn e gerando algumas discrepâncias em relação ao resultado desejado. Esses problemas foram resolvidos após pesquisas no forum do kaggle e a execução do código da Figura 1.

Esse código substitui a informação do número de episódios dos animes, sendo o número igual a ”Unknow”, para o valor médio de episódios por anime, neste caso igual a 2. Analogamente para a informação de rating dos animes cujo valor é vazio. Entretanto o valor da média difere, neste caso igual a 6.57. Além disso, substitui, também, os valores de rating e members de string para float.

```
anime["episodes"] = anime["episodes"].map(lambda x:np.nan if x=="Unknown" else x)
anime["episodes"].fillna(anime["episodes"].median(),inplace = True)
anime["rating"] = anime["rating"].astype(float)
anime["rating"].fillna(anime["rating"].median(),inplace = True)
anime["members"] = anime["members"].astype(float)
```

Figura 1. Parte do código do fórum do kaggle.

Tal código apresentou-se necessário apenas na primeira iteração de execução do programa, resultando, assim, na atualização do dataset utilizado e, portanto, tornando-se desnecessário nas próximas iterações.

Após isso, o grupo decidiu incorporar ao código formatações para toda e qualquer nova entrada. E, para a não necessidade de novos tratamentos sobre a quantidade ”Unknow” de episódios, é feita uma solicitação ao usuário para que informe 0 ou uma quantidade aproximada conhecida pelo mesmo e o programa atualiza essa quantidade de episódios ou apenas a armazena.

Após a resolução das citadas inconsistências, decidiu-se aplicar uma sistemática, para que não houvesse a possibilidade de confusões por parte do usuário ao utilizar o programa. A sistemática refere-se ao reconhecimento da diferença entre letras maiúsculas e minúsculas, ou seja, se o usuário procurar por ”naruto” em vez de ”Naruto” ele encontra o anime. Ainda sobre a mesma sistemática, ao usuário adicionar um anime e informar o tipo do anime de uma forma diferente da padrão solicitada e explicita na interface, o tipo, ainda assim, é identificado de forma correta. O código da Figura 2 representa a solução para tal sistemática.

```
for i in range(0,len(anime)):
    anime.loc[(anime["type"]==anime.ix[i]["type"]),"type"] = str(anime.ix[i]["type"]).title()

for i in range(0,len(anime)):
    anime.loc[(anime["name"]==anime.ix[i]["name"]),"name"] = str(anime.ix[i]["name"]).title()
```

Figura 2. Código da resolução dos problemas de difrenciação na busca.

Esse código formata todas as primeiras letras de cada palavra da string referente ao nome e tipo do anime com letras maiúsculas por meio da aplicação da funcionalidade ”.title()”. Facilitando assim, a manipulação dos dados, tanto para o usuário que não precisa se preocupar com a maneira de digitar, tanto para o programa o qual não irá inserir possíveis dados desnecessários ou informar de forma errada as informações de um dado pertencente ao banco de dados.

Tal funcionalidade é, também, encontrada na interface de interação programa-usuário para a formatação dos dados dos novos elementos a serem inseridos. Entretando, assim como o código da Figura 1, esse código apresentou-se necessário apenas na primeira iteração de execução do programa.

2.2. O Programa

Para o correto funcionamento do programa as importações das bibliotecas de extensão pandas e sys, assim como sklearn.neighbors e sklearn com suas respectivas funcionalidades NearestNeighbors e preprocessing, apresentaram-se necessárias.

O programa tem por objetivo listar as informações de um elemento do banco de dados escolhido pelo usuário, listar os elementos classificados como similares a um elemento do banco de dados escolhido pelo usuário, listar a quantidade de elementos pertencentes a um gênero do banco de dados escolhido pelo usuário, atualizar a quantidade de episódios de um elemento do banco de dados escolhido pelo usuário, inserir um novo elemento fornecido pelo usuário ao banco de dados e remover um elemento do banco de dados escolhido pelo usuário.

Os métodos definidos estão devidamente comentados para explicitar suas funcionalidades. O programa detém uma interface para a interação programa-usuário na qual todas as possíveis funcionalidades do programa são explicitadas e disponibilizadas ao usuário de maneira simples e direta. O programa, também, detém impressões na tela informando o início e o encerramento da manipulação dos dados para a realização do agrupamento já que este demanda um tempo de processamento considerável, resultante do tamanho do dataset utilizado.

O algoritmo utilizado para a realização do agrupamento e classificação dos elementos como similares é o algoritmo dos k-vizinhos mais próximos, ou seja, knn (visto em aula), e, neste caso, utilizando k=6. Para que tal possua precisão melhorada e tempo de processamento reduzido, a formatação dos dados e a aplicação de normalização para limitar os valores a serem analisados em valores no intervalo [0,1], apresentaram-se necessários. Esses estão identificados por comentários no código animes.py. Ainda assim, o tempo de processamento é considerável, atingindo até 26 segundos nas diferentes máquinas em que o programa foi testado.

O algoritmo demonstrou ter diferentes tempos de processamento para diferentes k's escolhidos durante o desenvolvimento. Os k's implicam na quantidade de elementos que são classificados como similares. Abaixo encontra-se uma tabela com os diferentes valores de k e seus respectivos tempos de processamento:

K	**Tempo de Processamento**
K = 3	t = 20s
K = 6	t = 20s
K = 10	t = 20s
K = 50	t = 20s
K = 100	t = 21s
K = 200	t = 21s
K = 400	t = 22s
K = 800	t = 24s
K = 1200	t = 26s
K = 2400	t = 30s
K = 3600	t = 34,5s
K = 4800	t = 36s
K = 6000	t = 37s
K = 8000	t = 39,5s

Tabela 1. Resultados testados por Rodrigo em sua máquina

K	Tempo de Processamento
K = 3	t = 26s
K = 6	t = 26s
K = 10	t = 28s
K = 50	t = 30s
K = 100	t = 32s
K = 200	t = 34s
K = 400	t = 35s
K = 800	t = 38s
K = 1200	t = 40s
K = 2400	t = 43s
K = 3600	t = 48s
K = 4800	t = 55s
K = 6000	t = 67s
K = 8000	t = 97s

Tabela 2. Resultados testados por Ulysses em sua máquina

2.3. Comentários adicionais

O dataset utilizado está disponível em 3, assim como, as informações das bibliotecas utilizadas, sklearn em 1 e pandas em 2. Entretanto, é importante ressaltar que o dataset utilizado para a versão final do programa desenvolvido é uma atualização feita pelo grupo do dataset original.

O código foi desenvolvido tanto na versão 2.7 quanto na versão 3 da linguagem de programação python, embora por escolha do grupo o escolhido para compartilhamento está na versão 2.7.

Para a execução do código, apenas caminhe até o diretório em que o mesmo está e execute em linha de comando:

python2.7 animes.py anime.csv

, ou apenas

python animes.py anime.csv

caso a versão 2.7 do python seja a versão padrão da máquina a ser utilizada para teste do programa. E, siga as instruções da interface de interação programa-usuário.

3. Conclusão

O desenvolvimento do programa teve por objetivo dos integrantes do grupo avaliar o grau de compreensão e entendimento do conteúdo abordado em aula sobre agrupamento e classificação de indivíduos utilizando o algoritmo dos k-vizinhos mais próximos, e sua aplicação prática.

Ao final do desenvolvimento, o grupo conclui seu sucesso para com ambos os objetivos citados, ressaltando as seguintes características da execução do programa: o tempo para a realização do agrupamento e classificação dos elementos é considerável, chegando a 26 segundos de processamento nas diferentes máquinas em que foi testado;

tal tempo de processamento é resultado do tamanho do dataset utilizado e da escolha de k=6.

Além disso, conlui-se, também, a gratificação pelo empenho para a resolução dos problemas encontrados durante o processo de desenvolvimento do programa e seu sucesso.

4. Referências

1. Informações da biblioteca sklearn disponíveis em:
   http://scikit-learn.org
2. Informações da biblioteca pandas disponíveis em:
   https://pandas.pydata.org/
3. Download do dataset disponíveis em:
   https://www.kaggle.com/CooperUnion/anime-recommendations-database
4. Informações sobre a funcionalidade .fillna() disponíveis em:
   https://pandas.pydata.org/pandas-docs/stable/generated/pandas.Series.map.html
5. Informações sobre a funcionalidade .map() disponíveis em:
   https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.fillna.html