Projeto de CTC 17 – Inteligência Artificial
Projeto 2

Integrantes do Grupo:

Davi Grossi Hasuda
Eduardo Henrique Ferreira Silva

Objetivo:

Exercitar e fixar conhecimentos adquiridos sobre algoritmos de Inteligência Artificial que envolvem treinamento com uma base de dados, mais especificamente árvores de decisão. Os códigos deste projetos foram todos feitos em Python 3. Foram tomados alguns cuidados para que seja compatível com Python 2, mas o sistema completo foi testado apenas com Python 3.

Descrição e resultados obtidos:

Descrição dos classificadores

O *dataset* utilizado foi o fornecido pelo professor, não havendo, pois, a necessidade de se decorrer sobre a estruturação dos dados. Quanto ao classificador, implementou-se uma árvore de decisão que baseia sua decisão no tipo de filme (ação, animação, comédia, drama, etc), sexo, idade e ocupação do usuário. Os demais dados foram deixados de fora da classificação dada sua especificidade. Vale lembrar que títulos de filmes raramente se repetem e que o Zip-code dos participantes também era bastante variável, sendo um mesmo Zip-code raramente compartilhado entre usuários e, nesses casos, a quantidade de usuários que o compartilhava era sempre pequena.

O classificador a priori implementado é bastante simples e ele simplesmente calcula a média truncada e a moda do *dataset* fornecido.

Dados e resultados da comparação

Para se realizar a análise comparativa dos classificadore, primeiramente mostra-se a tabela 1 a seguir com avaliações dos alunos e as previsões de cada classificador. Na tabela, os dez primeiros filmes estão incluídos no *dataset* fornecido, contudo os dez filmes restantes não estão. Esse tipo de abordagem é possível pela forma como o algoritmo foi implementado, não utilizando o nome (ou ID) do filme como pergunta classificatória.

Tabela 1. Classificações de diferentes fontes para 20 filmes. Sneod os 10 primeiros incluídos no dataset e os restantes não.

Filme	Classificação alunos	Árvore de decisão	Classificador a priori (média)	Classificador a priori (moda)
Toy Story (1995)	5	4-5	3	4
Dinosaur	4	4	3	4
Free Willy 2: The Adventure Home (1995)	3	1-3	3	4
Swan Princess, The (1994)	5	4	3	4
Flintstones, The (1994)	3	1	3	4
Jungle Book, The	4	3-4	3	4

(1994)				
Next Karate Kid, The (1994)	3	1	3	4
Pinocchio (1940)	5	4	3	4
Space Jam (1996)	2	3	3	4
Hunchback of Notre Dame, The (1996)	4	3	3	4
Baby Driver	5	4	3	4
Black Swan	5	4	3	4
The kissing booth	1	3-5	3	4
Inside out	5	5	3	4
Sing street	4	4	3	4
Annihilation	3	4	3	4
Love, Simon	2	3-4	3	4
Fight Club	4	4	3	4
Moonlight	5	4	3	4
500 Days of Summer	3	3-4	3	4

Nos casos em que há mais uma classificação para a árvore de decisão, as probabilidades calculadas pela a árvore de cada uma das classificações foram as mesmas. Para cálculo da análise dos classificadores foi utilizada a média dos valores obtidos e no caso da taxa de acerto foi considerado um meio acerto (0,5). Todos os filmes que não estavam no *dataset* tiveram os gêneros extraídos do IMDb.

No caso da estatística kappa, foi considerada que a distribuição esperada é uniformemente aleatória. Isto é, a chance de uma avaliação prevista é a mesma independentemente do valor real. Desse modo, a concordância esperada é de 0,2 (20%).

A seguir são mostrados em gráficos todos os resultados obtidos das análise. Os gráficos possuem legendas para auxiliarem o entendimento.

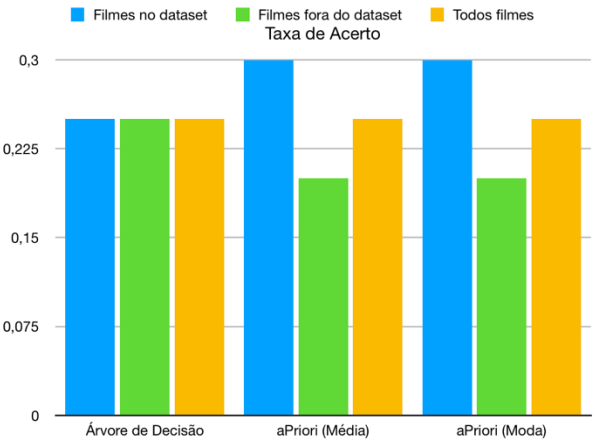

Gráfico 1. Taxa de acerto de cada método de predição com separação entre filmes presentes e não presentes no dataset.

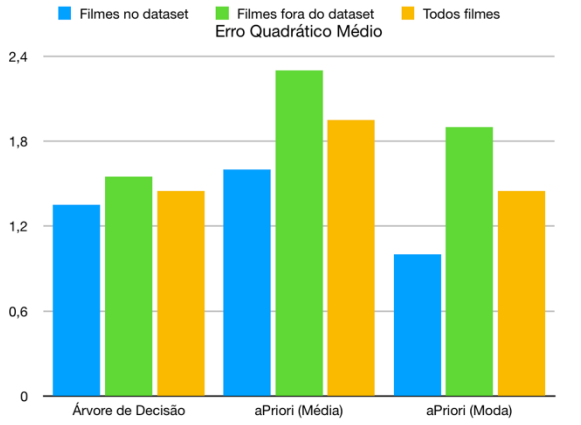

Gráfico 2. Erro quadrático médio de cada método de predição com separação entre filmes presentes e não presentes no dataset.

Árvore de decisão

	Valor previsto					
Valor real		1	2	3	4	5
	1			0,5		0,5
	2			1,5	0,5	
	3	2,5		1,5	1,5	
	4			1,5	3,5	
	5				5,5	1,5

aPriori (média)

	Valor previsto					
Valor real		1	2	3	4	5
	1			1		
	2			2		
	3			5		
	4			5		
	5			7		

aPriori (moda)

	Valor previsto					
Valor real		1	2	3	4	5
	1				1	
	2				2	
	3				5	
	4				5	
	5				7	

Matrizes de confusão. Matrizes de confusão de cada um dos métodos utilizados. Foi utilizado o dado com todos os filmes (aqueles incluídos e não incluídos no dataset). Na matriz de confusão da árvore de decisão há valores não inteiros pois há casos em que a previsão de um "rating" é ambígua, ou seja, o algoritmo acredita que há a mesma probabilidade de duas notas serem a nota real. Assim, nesses casos, para se construir tal matriz, dividiu-se o valor previsto em dois.

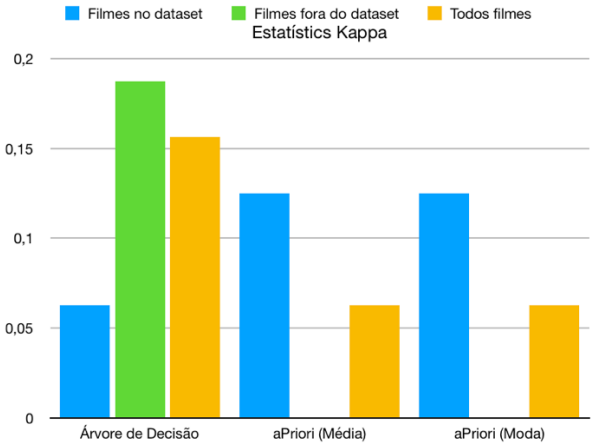

Gráfico 3. Estatística kappa de cada método de predição com separação entre filmes presentes e não presentes no dataset. Há casos em que a estatística kappa é zero, e por isso não aparece a coluna.

A partir da análise dos dados percebe-se que, quando o conjunto dos filmes utilizado na análise do classificador é aquele com filmes que estão e não estão no *dataset*, a árvore de decisão nunca tem um desempenho pior que nenhum classificador a priori utilizado. Vale notar também que diferentes medidas de desempenho podem resultar em resultados totalmente diferentes: enquanto que a taxa de acerto não consegue diferenciar a eficiência entre os classificadores para o caso em que utiliza todos os filmes, o erro quadrático médio dá vantagem à árvore de decisão, assim como a estatística kappa.

Vale notar também que a árvore de decisão só tem desempenho pior comparado a pelo menos um dos classificadores a priori quando se utilizam apenas filmes presentes no *dataset*. Isso ocorre em todos os indicadores. Em contrapartida, o desempenho dessa nunca está atrás dos outros classificadores nas demais estatísticas.

Como formas de melhoria do algoritmo da árvore de decisão, além de se utilizar um *dataset* maior poderia-se medir a impureza de um conjunto de instâncias de forma mais detalhada. Isto é, considerar que notas com valores próximos carregam menos impureza quando comparadas com notas muito discrepantes entre si. Isso melhoraria a eficiência na separação de instâncias com *ratings* visivelmente diferentes. Vale lembrar que uma nota 4 está muito mais próxima de uma nota 5 do que de uma nota 1 e essa informação não é considerada pelo algoritmo atualmente.

Conclusões:

A implementação da árvore de decisão foi de complexidade média, e um dos motivos foi a disponibilidade de várias fontes na internet e do material utilizado em sala de aula. Contudo, o que mais demandou tempo do projeto foi a demora no treinamento da árvore, uma vez que ele foi feito diversas vezes durante o desenvolvimento do projeto (por identifição de erros ou possíveis melhoras ao longo de seu desenvolvimento). Ainda assim, o trabalho ajudou na compreensão do conceito, bem como familiaridade com algumas bibliotecas do Python ainda não familiares aos estudantes (mais especificamente a biblioteca *pickle*).

Descrição da Implementação:

O código foi todo desenvolvido em Python 3, sendo que foi utilizado apenas editores de texto e nenhuma IDE. O dataset utilizado foi o disponibilizado pelo professor. Para rodar o projeto, coloque os códigos *classifier.py, TrainedDecisionTree.py, question.py, leaf.py* e *node.py* no mesmo diretório, junto com os dados fornecidos. Com isso, rode inicialmente o programa *classifier.py* utilizando Python 3. Esse é o código que treina a árvore de decisão, e leva alguns minutos para gerar a árvore.

Tal árvore é salva no arquivo *myobject*, no mesmo diretório dos demais arquivos. Esse arquivo é então lido na execução de *TrainedDecisionTree.py*. Ao se rodar este arquivo, no terminal serão pedidos dados do filme (gênero), e do usuário (gênero, idade e ocupação). Tais dados devem ser digitados no terminal da mesma forma como são salvos no *dataset*. O programa irá, então imprimir o resultado fornecido pela árvore de decisão. Veja-se o exemplo a seguir:

```
davigrossihasuda@MacBook-Pro-de-Davi:~/Documents/ITA/Paulo André/Projetos/Projeto 2/code % python3 TrainedDecisionTree.py
What genre?Animation
M or F?M
Age?18
Occupation?12
{5: 8}
```

No exemplo, o filme em questão é do tipo *“Animation”*, e o usuário é do sexo masculino, com idade entre 18 e 24 anos e trabalha como programador.

O resultado, “{5: 8}”, significa que há 8 filmes no set de treinamento que receberam 5 estrelas para esses dados inseridos.

No segundo exemplo a seguir, há 18 filmes com classificação 4 no set de treinamento, o que faz com que a provável classificação para esse caso testado seja 4 estrelas. Vale notar que há 16 avaliações no set de treinamento com esses dados que deram nota 5. Observando do ponto de vista probabilístico, há uma chance de aproximadamente 38% de ser avaliada com nota 4 e 34%, com nota 5.

```
davigrossihasuda@MacBook-Pro-de-Davi:~/Documents/ITA/Paulo André/Projetos/Projeto 2/code % python3 TrainedDecisionTree.py
What genre?Drama
M or F?F
Age?50
Occupation?0
{5: 16, 4: 18, 2: 6, 3: 6, 1: 1}
```

Foi utilizado como bibliografia não apenas os slides fornecidos pelo professor, mas também o link https://www.youtube.com/watch?v=LDRbO9a6XPU.

A implementação do classificador a priori está no arquivo *aPriori.py* e, para que funcione, precisa estar na mesma pasta do dataset. O que ele faz é simplesmente mostrar a média das classificações e o número de vezes que cada classificação aparece, como já foi explicado.