Relatório 2º projeto IA 2020/2021

Grupo: tg001

Alunos: João Miguel Pipa Ferreira Caldeira (93729)

João Tomás Cardoso (93730)

Descrição do Problema e da Solução

O objetivo deste projeto é estudar o comportamento de árvores de decisão, aprendendo a construir e otimizar uma árvore de decisão dado um conjunto de características e suas classificações num conjunto de treino.

A solução que este projeto apresenta, utiliza o algoritmo DTL (aprendido em aula) para aprender e induzir uma árvore de decisão com base num conjunto de treino dado. A ideia principal deste algoritmo é descobrir e escolher **recursivamente** o atributo mais significativo para a raiz da árvore e para possíveis subárvores "filhas" da raiz principal, através da recursão, finalizando com apenas 0 (False) e 1 (True) nas folhas.

O algoritmo começa por calcular a incerteza inicial total do conjunto de treino a estudar no momento, em seguida irá calcular o ganho de informação das características disponíveis, usando o cálculo da entropia e incerteza inicial, e escolhe a característica com maior ganho de informação para a raiz da árvore. Escolhida a raiz o algoritmo prossegue para avaliar o que deve ser feito para as folhas da árvore a ser criada. Para realizar esta tarefa, a árvore tem de agora considerar um novo "cenário filho" para cada folha.

No projeto é determinado que a folha da esquerda corresponde ao comportamento da característica selecionada quando a mesma é 0 e a folha da direita corresponde ao comportamento quando a característica selecionada é 1. Assim sendo a árvore para a folha da esquerda vai apenas considerar o conjunto de treino onde a característica selecionada é 0 e para a folha da direita considera apenas o conjunto de treino onde a característica selecionada é 1. Estando formados os novos conjuntos de treino "filhos" do conjunto de treino inicial, recomeça todo o procedimento recursivamente chamando a função com o novo conjunto de treino, (novo D, novo Y). A solução apresentada, antes de chamar recursivamente outra árvore, verifica se a característica selecionada para raiz consegue classificar com exatidão o conjunto a ser estudado, ou seja se a característica selecionada tem incerteza = 0 quando é falsa ou quando é verdadeira, e se isso se verificar, coloca na árvore a classificação determinada por essa característica, significando o fim da recursão para esse ramo da árvore. A solução termina a sua execução quando todas as recursões são finalizadas, ou seja, já foram utilizadas todas as características do conjunto de treino e, portanto, não pode haver mais subárvores (classificando a folha de acordo com o "caso default" ou seja utiliza-se a classificação moda do conjunto a ser estudado) ou então conseguiu-se classificar com incerteza = 0 os casos dados, retornando as subárvores obtidas e constrói-se a árvore final. Note-se que em ambos os casos, as folhas são 0 ou 1 obrigatoriamente. Esta foi a implementação inicial para inferir árvores de decisão. Esta implementação tem alguns problemas, nomeadamente o

problema do ***Overfitting*** e o problema de lidar com o **ruído**. Apesar de inferir uma árvore de decisão correta, esta árvore poderá ser demasiado extensa inferindo regularidades sem sentido devido à consistência do conjunto de hipóteses com os exemplos. Isto significa que a árvore irá ser "demasiado precisa" moldando-se demasiado ao conjunto de treino dado e, portanto, perde uma generalização necessária para poder classificar outros conjuntos. Isto é o ***Overfitting*** e é um problema do algoritmo DTL e como tal um problema da implementação inicial. Este encontra também problemas quando existem dois ou mais exemplos com os **mesmos valores de características**, mas, no entanto, **com classificações distintas**, impossibilitando assim a criação de uma árvore consistente pelo algoritmo. Por outro lado, o algoritmo não consegue identificar **atributos irrelevantes** e como tal irá tentar sempre encontrar uma árvore consistente com todos os atributos o que pode levar a árvores erradas através de distinções erradas. Isto é o que se chama de **ruído.** Possíveis soluções para estes problemas são: *decision tree pruning* e *cross validation*. Devido à existência destes problemas foi necessário adaptar a solução para lidar com eles de forma a obter uma solução correta aquando o acontecimento destes casos. Assim sendo era necessário: obter árvores mais pequenas, mantendo-se corretas e obtendo uma generalização maior (resolver *Overfitting*) e retirar os atributos irrelevantes (resolver o ruído).

Para diminuir o tamanho das árvores de decisão inferidas, foram identificados 2 casos onde se poderia simplificar a árvore: quando existem duas folhas iguais e quando existem duas subárvores com a mesma característica selecionada, nas folhas de uma mesma raiz. Quando existem duas folhas iguais a solução é simples e óbvia, a raiz dessas folhas é eliminada e trocada por uma das folhas que origina (relembrando que são iguais). Reduz-se assim um nível de profundidade num ramo, por cada caso destes. No segundo caso, quando ambas as folhas de uma raiz são subárvores e ambas têm a mesma característica, existe a possibilidade de existir uma árvore mais curta com raiz na característica selecionada para as folhas (essa possibilidade seria de 100% se as folhas fossem exatamente iguais, mas isso seria um exemplo do primeiro caso). Assim sendo a solução apresentada calcula uma segunda árvore hipotética substituindo a característica anteriormente selecionada pela característica utilizada nas subárvores das folhas, comparando no fim de execução o tamanho de ambas as árvores, escolhendo-se a mais pequena. Ambos estes métodos ajudam a generalizar e obter uma árvore mais curta.

Para lidar com o ruído, foi utilizada uma técnica denominada de *early stopping*. A solução utilizada determina que um dado é irrelevante se o seu ganho de informação for abaixo de 5% (valor normalmente utilizado para um teste estatístico de significância). E, portanto, se a característica retirada do conjunto de treino com o maior ganho de informação tiver um ganho de informação inferior a 5% então chega-se a uma folha cujo valor é a moda das classificações dos exemplos do conjunto de treino a analisar no momento. Esta técnica tem problemas pois apesar de um dado poder ser irrelevante, uma combinação de dados usando esse dado pode ser relevante e o *early stopping* irá ignorar esses pois não considera a combinação de dados. O *early stopping* é um tipo de *decision tree prunning* que combina x^2 *prunning* e o ganho de informação.

Testes sem Ruído

Tempo (s)	Nº Teste
0.158	1
0.161	16
0.165	20
0.674	22

Testes com Ruído

Tempo (s)	Nº Teste
0.741	1
1.332	2
1.357	3
1.335	4