Hide and Seek - Análise de Dados

Agentes e Inteligência Artificial Distribuída 2019

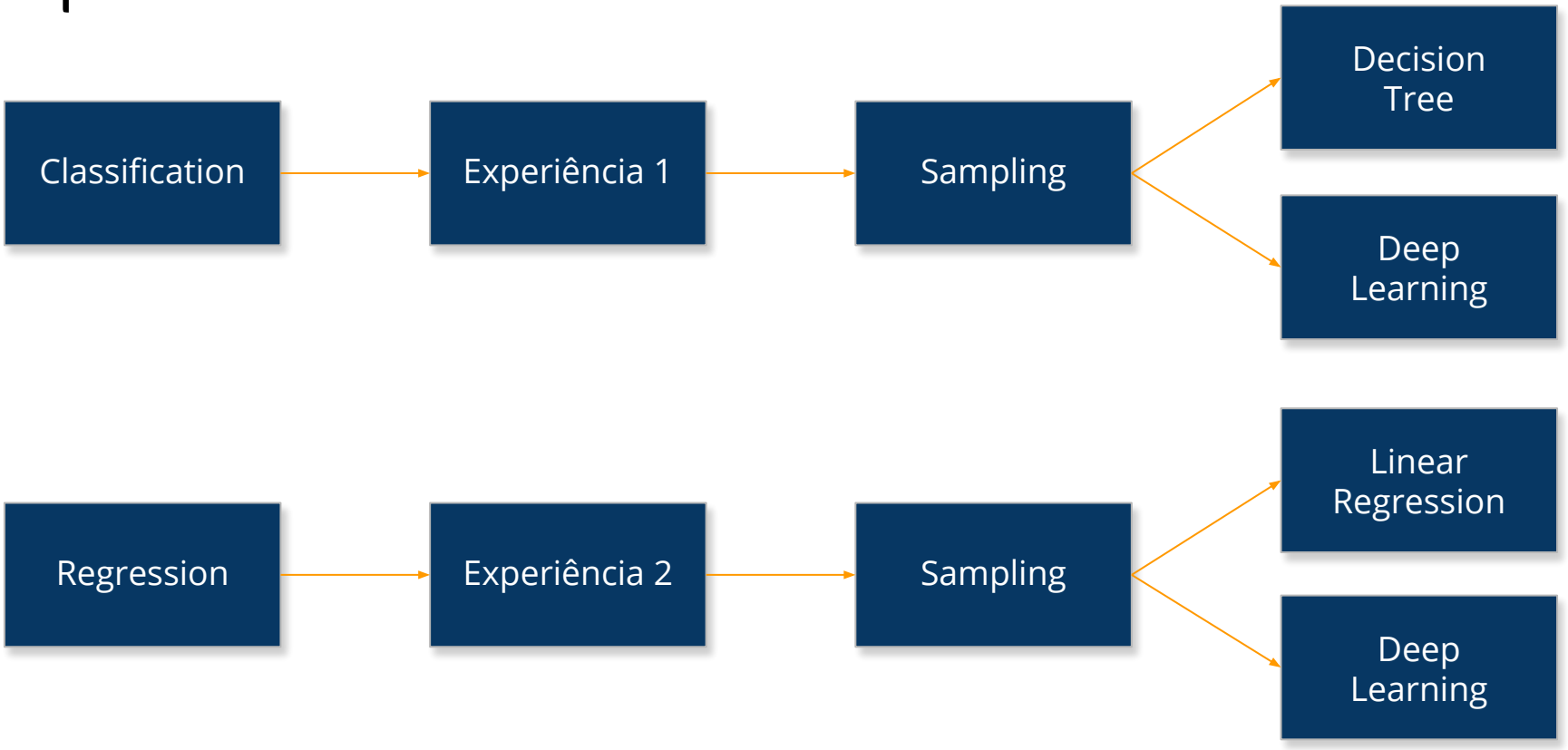
Descrição do problema

Este projeto é uma extensão do trabalho realizado na primeira parte. De forma a permitir uma análise de dados mais interessante, foi implementada uma alteração: os agentes têm o seu movimento melhorado de forma a não ficarem presos.

Existem então 2 problemas:

- Regressão - prever o número de rondas jogadas numa determinada partida
- Classificação - prever a duração duma determinada partida (Curta, Média ou Longa)

Experiências realizadas



Experiências realizadas

Classificação

Objetivo

Prever o nº de rondas jogadas numa determinada partida.

Variáveis Dependentes

Nº de rondas jogadas

Regressão

Objetivo

Prever a duração duma determinada partida (Curta, Média, Longa)

Variável Dependente

Duração da partida

Variáveis Independentes

Nº de *hiders*, Nº de *seekers*, Nº de células, Nº de obstáculos, Nº máximo de rondas

Estatísticas sobre os dados recolhidos

De forma a conseguir gerar dados em grande volume foi desenvolvido um pequeno programa que gera um ficheiro de dados. Este programa recebe como argumento um número de linhas, e gera de forma aleatório um ficheiro em que cada linha representa os dados de entrada para uma execução do programa (mapa e nº máximo de rondas).

Este ficheiro é utilizado como entrada ao programa HideNSeek. O programa lê cada linha do ficheiro e executa um jogo, sendo que no final escreve num ficheiro de saída os resultados (nº de rondas jogadas e duração da partida).

Todos os ficheiros estão no formato .csv (*Comma Separated Values*), de modo a facilmente passar a informação necessária para o RapidMiner poder analisar os dados.

Análise dos dados com RapidMiner

Problema de Classificação - Decision Tree

	true Short	true Medium	true Long	class precision
pred. Short	290	17	1	94.16%
pred. Medium	15	120	55	63.16%
pred. Long	2	35	215	85.32%
class recall	94.46%	69.77%	79.34%	

Accuracy: 83,33%

Analisando os dados da tabela é possível verificar que este algoritmo possui algumas dificuldades na previsão da classe “Medium”. Quando o valor verdadeiro é “Medium” a taxa de acerto é apenas de 70%, por outro lado, quando o algoritmo prevê “Medium”, apenas acerta em 63% das vezes.

A razão pela qual isto acontece é pela presença de um menor número de dados iniciais em que o valor final é “Medium”. Analogamente, é possível verificar que o elevado número de dados iniciais contendo valores “Short” resulta numa maior taxa de acerto por parte do algoritmo Decision Tree.

Análise dos dados com RapidMiner

Problema de Classificação - Deep Learning

	true Short	true Medium	true Long	class precision
pred. Short	300	0	0	100.00%
pred. Medium	7	170	2	94.97%
pred. Long	0	2	269	99.26%
class recall	97.72%	98.84%	99.26%	

Accuracy: 98,53%

Em relação ao algoritmo Deep Learning, os resultados da aprendizagem são excelentes. Em todas as classes a previsão foi bastante elevada o que mostra que a aprendizagem foi bem conseguida. Estes resultados não são, no entanto, nada surpreendentes, dada a baixa complexidade do problema a resolver e o elevado número de dados iniciais.

Análise dos dados com RapidMiner

Problema de Regressão	Linear Regression	Deep Learning
Correlation	0.957	0.990
Root mean squared error	2.933 +/- 0.0	1.509 +/- 0.0
Root relative squared error	0.290	0.144

- Os valores de correlação de ambos os modelos são bastante elevados o que demonstra que o nº de rondas previsto está altamente dependente das variáveis independentes.
- Quanto aos valores de *root mean squared error*, estes são relativamente reduzidos, indicando que os valores previstos não se afastam em grande escala dos valores reais
- Em ambos os modelos o *root relative squared error* é mais uma vez bastante baixo. Por este motivo se considera que os modelos utilizados são bastante mais úteis quando comparados à previsão trivial (média dos valores).

Conclusões

Infelizmente, na primeira parte do projeto, colocamo-nos em desvantagem ao fazer um trabalho “unidimensional”, na medida em que agora se tornou difícil encontrar problemas variados para tratar, pelo que no futuro seria interessante implementar alterações como por exemplo personalidades nos agentes.

Em relação à análise de dados, os resultados foram bastantes satisfatórios para ambos os problemas. Ainda assim, devido a restrições de tempo, não foi possível testar com um número maior de dados, pelo que seria algo a ter em consideração de forma a reforçar a confiança nos resultados gerados.

Informação Adicional

Processos RapidMiner

Como ilustrado no **slide 3**, no total foram efetuados **4 processos RapidMiner** para analisar os dados gerados pelo programa.
O operador **Split Validation** está presente em todos os processos desenvolvidos. Este é responsável por:

- Distribuir os dados de treino e teste. A distribuição efetuada em todos os processos foi a distribuição *default*:
  - **TESTE:** 30%
  - **TREINO:** 70%

- Agrupar os dados de treino e de teste. Existem várias formas de agrupar, sendo que neste projeto apenas foram consideradas:
  - **Stratified Sampling:** constrói subconjuntos aleatório e garante que a distribuição de classes nos subconjuntos é a mesma que na amostra
  - **Automatic:** usa **stratified sampling** por defeito, mas substitui por **shuffled sampling** caso a amostra não tenha uma etiqueta nominal,

Processos RapidMiner

O que diferencia cada experiência são nomeadamente os operadores utilizados. Neste projeto recorreu-se aos seguintes operadores:

- **DEEP LEARNING:** Executa o algoritmo de aprendizagem profunda usando H2O 3.8.2.6.
- **DECISION TREE:** Gera um modelo de árvore de decisão, que pode ser usado tanto para problemas de classificação como regressão.
- **LINEAR REGRESSION:** Calcula um modelo de regressão linear através da amostra fornecida.

Outras observações

De forma a permitir uma análise de dados mais interessante nesta segunda parte do projeto, tentamos introduzir alterações no programa:

- Melhorar o movimento dos agentes - corrigir a possibilidade dos agentes ficarem presos
- Introduzir fator de mentira nos agentes - de acordo com uma certa probabilidade, os agentes poderiam dar informações falsas aos colegas de equipa

A primeira alteração foi concluída com sucesso, mas a segunda foi descartada: devido ao fator aleatório, não é possível fazer uma análise correta pelo RapidMiner (duas execuções com as mesmas variáveis podem dar resultados diferentes).

Grupo 30

- Francisco Filipe - up201604601@fe.up.pt
- João Miguel - up201604241@fe.up.pt
- Pedro Fernandes - up201603846@fe.up.pt