Desenvolvimento de um Agente Jogador de Blackjack

Trabalho de grupo

Inteligência Artificial / Introdução à Inteligência Artificial

Ano Lectivo de 2016/2017

15 de Outubro de 2016

I Observações importantes

1. Este trabalho deverá ser realizado por grupos de 2 a 3 alunos. Em cada módulo Python submetido, inclua um comentário com o nome e número mecanográfico dos autores.

2. Este trabalho deverá ser submetido até ao dia 25 de Novembro de 2016. O trabalho poderá ser submetido para além do prazo, mas será penalizado em 5% por cada dia adicional.

3. Cada grupo deve submeter uma pasta zipada contendo uma apresentação Powerpoint (formato pdf), com no máximo quatro páginas, resumindo a concepção subjacente à implementação desenvolvida, e um ou mais ficheiros contendo todo o código Python relevante. Os programas devem estar preparados para correr em Python 2.7. Se tal não for possível, devem informar o docente. O módulo principal deverá chamar-se `student.py`.

4. Se discutir o trabalho com colegas de outros grupos, inclua um comentário com o nome e número mecanográfico desses colegas. Se recorrer a outras fontes, identifique essas fontes também.

5. Todo o código submetido deverá ser original; embora confiando que a maioria dos grupos fará isso, serão usadas ferramentas de detecção de copianço. Alunos envolvidos em casos de copianço terão os seus trabalhos anulados.

6. Os trabalhos serão avaliados tendo em conta: qualidade da concepção e da implementação; desempenho; e originalidade (incluindo evidência de trabalho autónomo).

II Tema do trabalho

Este trabalho envolve a aplicação de conceitos e técnicas de três capítulos principais da matéria leccionada, nomeadamente: programação em Python; arquitecturas de agentes; e técnicas de pesquisa para resolução de problemas.

No âmbito deste trabalho, deverá desenvolver um agente capaz de jogar de forma inteligente o jogo Blackjack, também conhecido como jogo do Vinte-e-um:

- Deverá ser capaz de jogar um jogo isolado
- Deverá ser capaz de jogar uma sequência de jogos, optimizando o lucro

O Blackjack é um dos jogos de casino mais populares em que o jogador joga contra um representante do casino, designado como *croupier* (*dealer*). O objetivo do jogo é atingir uma pontuação tão próxima quanto possível de 21 pontos, sem ultrapassar esse limite, e pode ser ganho nos seguintes casos:

- O jogador obtém 21 pontos nas primeiras duas cartas (caso chamado *blackjack*), desde que o croupier não tenha também *blackjack*
- O jogador atinge um número de pontos superior ao do croupier sem exceder 21
- Cropiê excede 21 pontos.

III Funcionamento do jogo

Utilizam-se baralhos de cartas típicos, isto é, existem quatro naipes (copas, outros, espadas e paus) e 13 cartas por naipe (2 a 10, dama, valete, rei e ás). Neste trabalho usamos 4 baralhos.

As cartas numeradas (2 a 10) pontuam com o seu próprio valor. O valete, dama e rei contam como 10 e o ás pode contar como 1 ou 11. A pontuação de uma mão é a soma do valor de todas as cartas na mão. O valor dos ases na mão é determinado tal que os pontos sejam o valor mais elevado não superior a 21 (se possível). Uma mão em que um ás seja contado como 11 é uma *soft hand*. Os naipes das cartas não interessam neste jogo.

Um *blackjack* é uma mão inicial (duas cartas) em que uma carta é um ás e a outra é uma carta que vale 10. Um *blackjack* ganha sempre a qualquer outra combinação de 21 cartas.

Existem diversas variantes deste jogo. No presente trabalho, o funcionamento do jogo segue os seguintes passos:

- Cada jogador define a sua aposta inicial.
- Após este momento, cada jogador recebe duas cartas com a face voltada para cima. Por sua vez, o croupier recebe uma carta com a face virada para cima e outra com a face virada para baixo (apenas o croupier sabe o valor desta carta).
- Se o croupier tiver blackjack, então ganha todas as apostas feitas, excepto se um dos jogadores tenha também blackjack. Caso tal aconteça (*push*) o jogador pode receber de volta a sua aposta.
- Se um jogador tiver blackjack (e o croupier não) então o jogador ganha imediatamente. O jogador recebe a sua aposta de volta e um prémio de 100%.
- A cada jogador é dada rotativamente a oportunidade de receber cartas adicionais e de tomar uma das seguintes decisões:

  - `Double-down` – (após a mão inicial, e antes de receber qualquer outra carta) o jogador pode duplicar a sua aposta e recebe uma única carta extra.
  - `Hit` – pede uma carta extra
  - `Stand` – o jogador termina a sua jogada (não quer receber mais cartas)
  - `Surrender` - o jogador pode desistir do jogo perdendo parte da sua aposta (conforme as regras do jogo)

  Um jogador pode receber quantas cartas entender, mas se exceder 21 pontos perde imediatamente a sua aposta.

- O croupier é o último jogador em cada ronda. O croupier pode fazer `hit` ou `stand`. Se estoirar (ultrapassar 21) então todos os jogadores da mesa ganham as suas apostas (tirando os que já estoiraram).
- Quando todos os jogadores fizerem `stand`, as apostas são resolvidas.
- Se o croupier tiver uma mão mais pontuada que o jogador então o jogador perde a aposta. Se as pontuações forem iguais há um *push*, e a aposta é devolvida. Se a mão do jogador for superior à do croupier, o jogador ganha a sua aposta.

IV Código de apoio

Um motor de jogo Blackjack escrito em Python encontra-se disponível em `https://code.ua.pt/projects/iia-ia-blackjack`.

Todas as entidades do jogo são representadas por classes. Uma `Hand` tem uma lista de `Card`s. Um `Game` tem uma `Hand` do `Dealer` e uma lista de `Hand`s de `Player`s.

A cada novo `Game`, todas as cartas são baralhadas de volta. Cada grupo desenvolve um agente na forma de uma classe derivada de `Player` (como é o caso de `RandomPlayer`) e apenas pode fazer modificações ao ficheiro `casino.py` (com propósito exclusivo de importar e utilizar o seu agente).

O agente desenvoldido deverá ser entregue num módulo `student.py` e deverá ter o nome `StudentPlayer`.

Por herança pode implementar todos os métodos da class Player. É essencial a implementação dos métodos: `play` e `bet`, mas pode também implementar `payback`.

- `bet` - sempre que o jogador é chamado a fazer uma aposta recebe o estado do `dealer` e de todos os jogadores na mesa.
- `play` - a cada turno o jogador é chamado a decidir a sua acção através de uma string: h, s ou d (hit, stand ou double down.
- `payback` - no final do jogo é sinalizado ao agente se ganhou (recebe o valor da sua aposta), perdeu (recebe a sua aposta em negativo) ou empate (recebe 0)
- `want_to_play` - no ínicio de cada jogo este metodo é invocado afim de apurar se o jogador deseja ir a jogo, ou não. O método passa ainda as regras do jogo (aposta minima, aposta máxima, multiplicador de aposta, número de baralhos).

Nos métodos `play` e `bet` o estado do `dealer` não inclui a carta escondida.

Um canal de apoio existirá em `https://detiuaveiro.slack.com/messages/ia/` onde poderão colocar dúvidas e receber notificações de alterações a este documento.

Estejam atentos a modificações no código disponibilizado (`git pull`) e a notificações no *Slack* e *e-learning*.

V Conselhos

Começe por desenvolver um agente que lide com as regras mais simples apenas (apenas soft-hands e sem double-down).

Se o seu agente se comportar pior que o agente aleatório (`RandomPlayer`), algo de errado se passa com o seu agente!

Técnicas de casino como contagem de cartas não funcionam, uma vez que o baralho é sempre novo para cada jogo.

VI Esclarecimento de dúvidas

Esclarecimentos sobre as principais dúvidas que venham a surgir durante a realização do trabalho serão colocados aqui.

1. Fazer `git log` para se manter informado de pequenas alterações que foram ou venham a ser feitas.

2. **Questão**: Como serão avaliados os agentes?

   Resposta: Os agentes serão avaliados pelo seu desempenho (número de jogos ganhos, lucro obtido). As condições de avaliação poderão variar (limites das apostas, número de baralhos, jogadores na mesa, número de jogos e dinheiro disponível)