Yilun JIANG

Sokem AYIGAH

IMA 6

Université Grenoble Alpes

Projet final - Jeu du Blackjack

Sommaire

1 – Introduction

A – Le jeu du Blackjack

Le Blackjack est un jeu d'argent qui vit le jour en France, entre le 17^{e} et le 18^{e} siècle. A l'origine, le valet noir octroyait aux joueurs un bonus, d'où le nom Blackjack qui a été maintenu, bien que le bonus ait été modifié. Rapidement, le jeu fut très apprécié des amateurs de casino, notamment grâce à ses règles, qui se trouvent être plutôt simples.

B – Règles du jeu

Il existe de nombreuses variantes du jeu mais nous nous intéresserons ici à une version simplifiée des règles.
Chaque joueur joue individuellement contre la banque, qui est incarnée par le croupier. Le but est d'obtenir un score plus élevé que le croupier, sans jamais dépasser les 21 points. On joue avec des paquets de 52 cartes (autant de paquets que de joueurs) qui ont les valeurs suivantes :

- Les cartes numérotées de 2 à 10 valent autant de points que leur numéro;
- Les têtes (valet, dame et roi) valent 10 points;
- L'as vaut 1 ou 11 points au choix;

Le score d'un joueur est égal à la somme de la valeur des cartes qu'il possède.

Au début du tour, chaque joueur mise la somme qu'il souhaite. Le croupier distribue ensuite deux cartes face visible à chacun d'entre eux, et une à lui-même. Si un joueur reçoit un as et une carte de valeur 10, il obtient alors un ‘‘blackjack’’, et remporte immédiatement son duel face au croupier, et remporte 2,5 fois sa mise de départ. Les joueurs choisissent un à un de s'arrêter ou de piocher une carte supplémentaire, action qui peut être répétée autant de fois que voulu, mais attention, si un joueur dépasse les 21 points, il est éliminé.
Une fois que tous les joueurs ont fini de piocher, c'est au tour du croupier de jouer. Il pioche une carte, puis joue avec les mêmes restrictions. Après cela, tous les joueurs ayant un score supérieur à celui du croupier récupèrent leurs mises respectives x2, et les autres la cèdent à la banque. En cas d'égalité, le joueur récupère sa mise une fois seulement.

C – Objectifs

Ce projet consiste à développer en python une version fonctionnelle du Blackjack. Pour ce faire, nous suivrons l'énoncé en quatre parties qui nous a été proposé, tout en laissant libre cours à notre imagination, pour obtenir la version qui nous convient le mieux.

Les principaux objectifs du projet :

- Utiliser les différentes connaissances et notions acquises tout au long de l'UE pour créer un jeu.

- Comprendre et traduire en algorithme les règles du black jack.
- Écrire un code efficace, lisible et compréhensible, grâce aux annotations.
- Apprendre et découvrir de nouvelles fonctions du langage Python, pour améliorer notre code.

2 – Réalisation du Blackjack

A – Initialisations et gestion

Comme indiqué dans le poly, nous avons commencé par créer toutes les fonctions nécessaires au bon fonctionnement du programme. Chacune d'entre elles sont commentées dans le programme, avec ses arguments, ses sorties et son rôle. On peut trier nos fonctions en différentes catégories que voici :

Les fonctions de bases (communes à toutes les version du jeux) :

- paquet
- valeurCarte
- initPioche
- piocheCarte,
- initJoueurs
- initOrdi
- initScors
- premierTour
- gagnant
- joueur_continuer
- tourJoueur
- tourComplet

Toutes ces fonctions sont décrites dans le code. Avec elles, il est envisageable de coder une première version du jeu, ou les joueurs jouent les uns contre les autres, sans mises, ni croupier. Après cela, nous avons modifié le code pour permettre aux joueurs de miser.

Les fonctions d'historique (pour créer une base de données et l'exploiter) :

- history_save_to_txt
- read_database
- read_history

Ces fonctions nous permettent de créer un historique des parties, qui nous sera utile pour la réalisation de nos intelligences artificielles. Si l'historique n'est pas assez grand, les IA utiliseront la base de données, réalisée au préalable avec 250 000 essais.

Les fonctions d'IA (utilisées pour les différentes difficultés du croupier) :

- croupier_easy
- croupier_normal
- croupier_hard

- croupier_prendre_carte
- bot_decision_multitask
- bot_decision

Les fonctions d'IA seront décrites plus en détail dans la partie 3.

3 – Intelligences artificielles

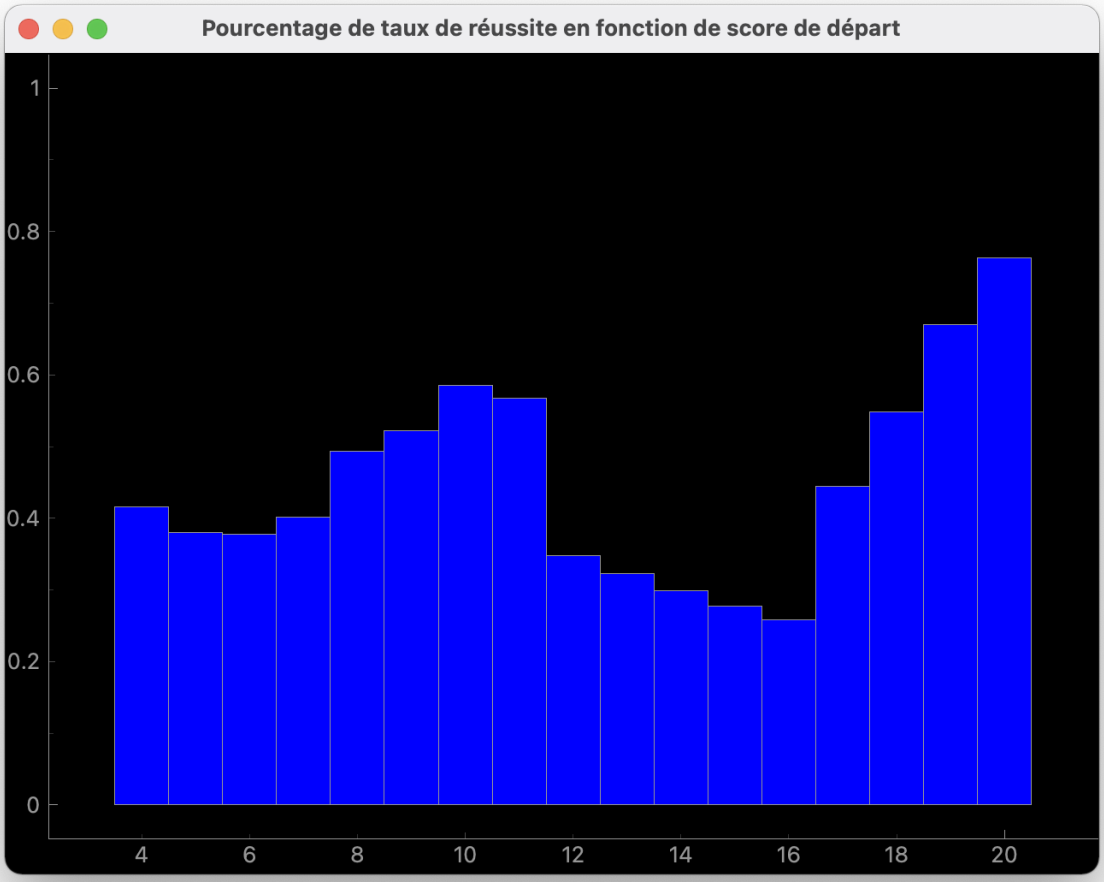
Pour obtenir une version finale du jeu, on ajoute des IA (Intelligences artificielles). Pour les coder, nous avons fait des recherches et nous nous sommes appuyés sur différents ouvrages, dont une grande partie faisait référence à l'utilisation des méthodes de Monte Carlo, et de la formule de Kelly.

A – Taux de réussite

Grâce à nos recherches et nos tests, nous avons pu identifier certains facteurs qui influent sur le pourcentage de taux de réussite (chances de gagner) du joueur.

Le joueur joue avant le croupier. Il peut donc être éliminé immédiatement en dépassant les 21 points, et céder sa mise à la banque, sans que le croupier n'ait eu besoin de jouer. Cette règle est particulièrement dangereuse pour le joueur, lorsque son score de départ est compris entre 12 et 17 points, car la prochaine carte déterminera alors s'il est éliminé ou non. Le graphique ci-dessous nous montre clairement que le joueur a un pourcentage de réussite bien plus faible dans ce cas précis.

Diagramme du pourcentage de taux de réussite du joueur en fonction de score de départ, basé sur l''historique des parties



Nous avons constaté que le simple fait de s'arrêter de piocher dès que l'on n'est plus sûr de ne pas pouvoir perdre (à partir d'un score de 11), permet d'augmenter grandement les chances du joueur de battre le croupier, car cette stratégie contraint le croupier à jouer et à faire mieux que le joueur, avec à son tour le risque de dépasser les 21 points.

B – IA des joueurs

Les facteurs listés précédemment nous on servi de base pour coder l'IA du joueur. Son rôle est de déterminer quand piocher ou non (dans le code, l'IA des joueurs est définie par les fonctions bot_decision et bot_decision_multitask). Pour cela, elle doit calculer la probabilité d'élimination, et la comparer avec celle d'obtenir un score élevé. Par exemple, si le bot obtient au premier tour un score de 16, il calculera alors la probabilité de tirer une carte qui lui permettrait d'obtenir un score compris entre 17 et 22. Si le résultat obtenu est supérieur à une certaine valeur, le bot piochera, sinon il se retirera.

Ces probabilités sont calculées grâce à la base de données et à l'historique de partie que nous avons réalisé au préalable. On peut coder le bot de telle manière à ce qu'il prenne plus ou moins de risque, en modifiant la valeur du pourcentage de probabilité à partir de laquelle le bot pioche. Ici, nous l'avons fixé à 0,2 (dans la fonction

bot_decision). Notre bot est donc plutôt prudent.

C – IA du croupier

Pour le croupier, nous avons donc codé 3 IA, de difficultés différentes. Chacun d'entre eux utilise sa propre stratégie :

Le bot facile (fonction croupier_easy dans le code)

Dans cette version, le croupier va continuer à piocher (sans réfléchir), jusqu'à l'élimination. La seule façon pour ce bot de gagner, est donc de laisser le joueur perdre (le joueur peut très simplement gagner à tous les coups, en ne piochant jamais).

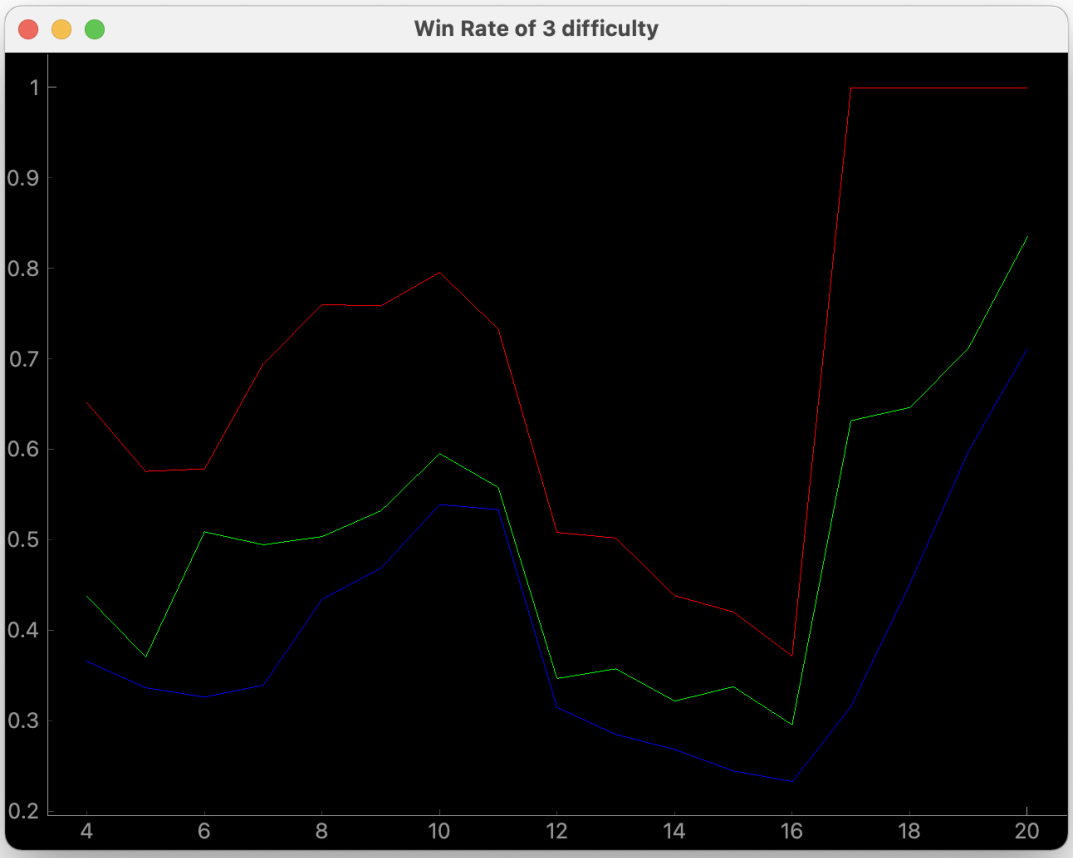
Le bot normal (fonction croupier_normal)

Ici le croupier pioche jusqu'à obtenir 17 points, puis il choisit au hasard de tirer une carte ou de se coucher, sans tenir compte des scores des joueurs.

Le bot difficile (fonction croupier_hard)

Comme le bot normal, ce croupier pioche jusqu'à obtenir 17 points. Une fois ce score atteint, il prend en compte son score actuel, et le score de chaque joueur, ainsi que leurs mises, pour calculer s' il doit piocher ou non.
Contrairement à l'IA des joueurs, le bot difficile n'a pas besoin d'utiliser la base de donnée ou l'historique, car il sait exactement quelles cartes sont encore présentes dans le paquet, ce qui lui permet de calculer la probabilité d'obtenir chacune d'entre elles.
Par exemple, on suppose que dans une partie à trois joueurs, le J1 mise une grosse somme, et les deux autres joueurs, une somme bien plus raisonnable. Le croupier ne sera peut être pas en mesure de battre le J2 et le J3, mais s' il bat le J1, il sera quand même rentable.

Graphique du pourcentage de réussite du joueur en fonction de son score, pour chacune des version du croupier



Les courbes rouge, vert et bleu correspondent respectivement aux bots facile, normal et difficile. Ce graphique nous permet de comparer très clairement les stratégies. En effet, on peut par exemple voir que le bot facile perd toujours à partir de 17 points, car il pioche jusqu'à l'élimination.

4 – Conclusion

A – Améliorations envisageables

Ce projet est une première mise en pratique des notions d'informatique que nous avons apprises durant ce premier semestre, par conséquent, nos connaissances en langage python et en algorithmie sont donc limitées.
Toutefois, il est tout à fait possible d'envisager des améliorations de notre programme.
Nous pourrions approfondir les méthodes de Monte Carlo afin de trouver la stratégie optimale à adopter pour créer une meilleure intelligence artificielle.

En faisant des recherches, nous avons également découvert la méthode de l'apprentissage profond (ou deep learning en anglais) pour l'intelligence artificielle, que nous pourrions utiliser à la place de notre base de données, mais celle-ci nécessiterait encore quelques années d'études pour être mise en œuvre.

Nous pourrions également créer une interface graphique, pour obtenir un véritable jeu interactif, chose que nous n'avons pas pu faire, faute de temps.

B – Bilan

Lors de ce projet, nous avons réussi à obtenir une version fonctionnelle du jeu du BlackJack, tout en respectant les indications du sujet. Les recherches pour comprendre et concevoir les différentes intelligences artificielles ont été particulièrement compliquées, mais tout le processus de réalisation du projet a été très instructif. Cela nous a permis d'en apprendre davantage sur l'algorithmie, tout en exploitant les différentes fonctions du langage python.