

TIPE – STRATÉGIES D'ARBITRAGE DANS LES PARIS SPORTIFS

BIER Oscar (22029)

INTRODUCTION



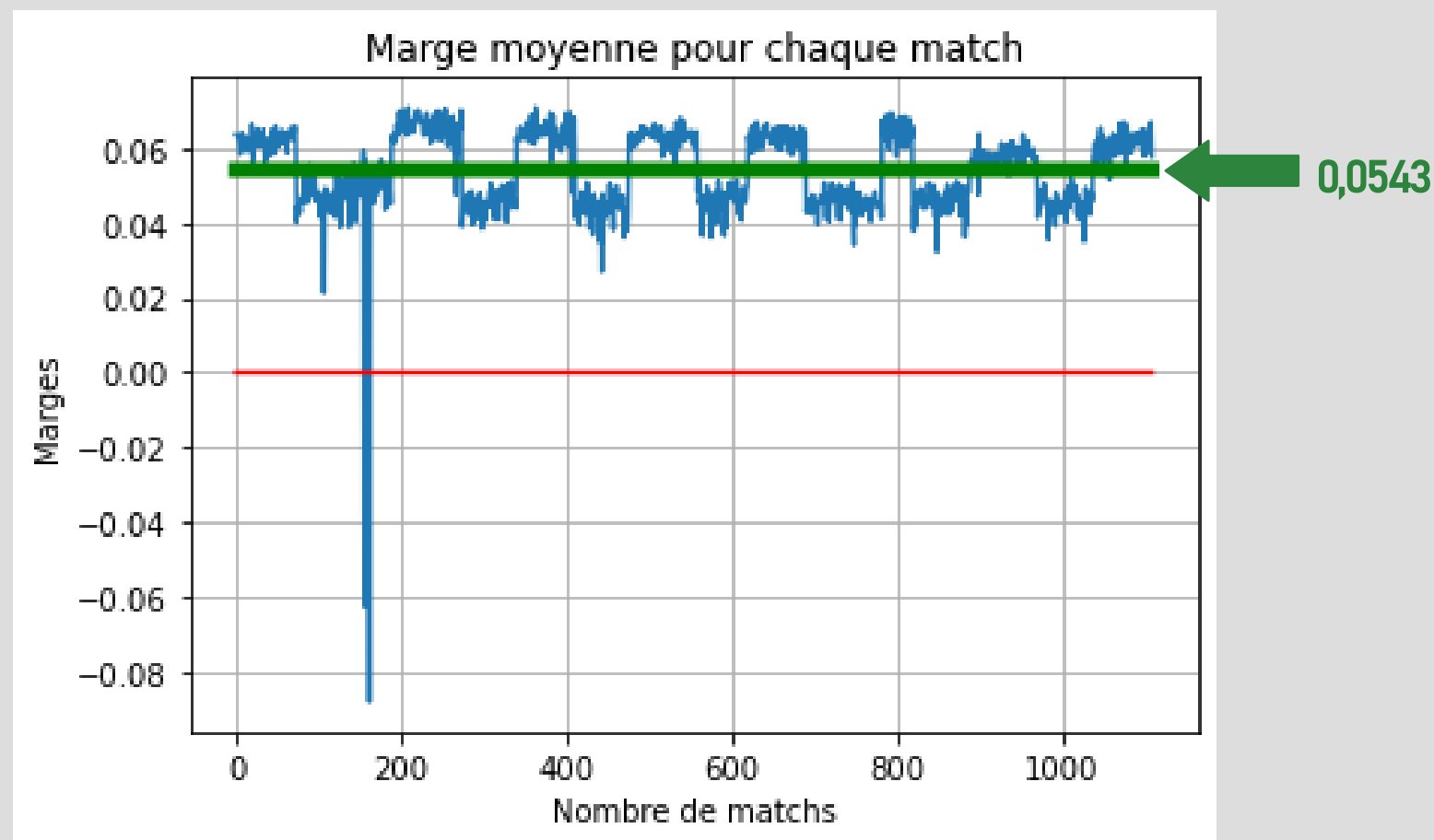
Cote du joueur n°1



Cote du joueur n°2

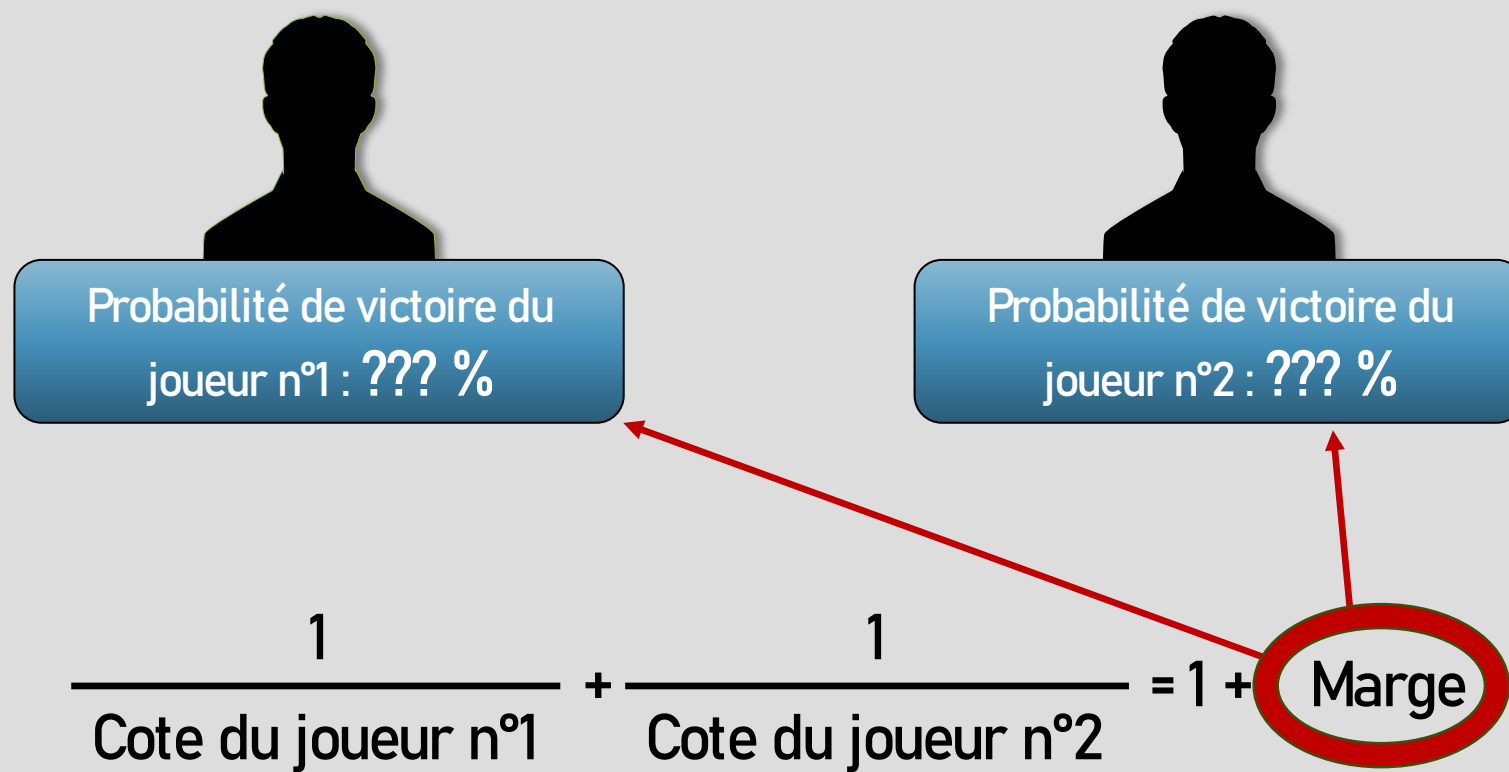
$$\frac{1}{\text{Cote du joueur n°1}} + \frac{1}{\text{Cote du joueur n°2}} = 1 + \text{Marge}$$

INTRODUCTION



Marges établies par un bookmaker pour 1 108 matchs de tennis

INTRODUCTION





INTRODUCTION

Définition : Stratégie d'arbitrage

Une stratégie d'arbitrage est une technique utilisée en finance, qui consiste à tirer profit des **différences de prix pour un même actif**, sur deux marchés différents.



Pour les paris sportifs, l'arbitrage se résume à parier judicieusement sur deux sites de paris sportifs distincts.

INTRODUCTION

	 Joueur n°1	 Joueur n°2
Site de paris sportifs n°1	1,62	2,23
Site de paris sportifs n°2	1,64	2,17

INTRODUCTION

Stratégie d'arbitrage

	 Joueur n°1	 Joueur n°2
Site de paris sportifs n°1	1,62	2,23
Site de paris sportifs n°2	1,64	2,17

INTRODUCTION

Avantages

INTRODUCTION

Avantages

- Aucun risque
- Bénéfice quel que soit l'issue de l'événement

INTRODUCTION

Avantages

- Aucun risque
- Bénéfice quel que soit l'issue de l'événement

Inconvénients

INTRODUCTION

Avantages

- Aucun risque
- Bénéfice quel que soit l'issue de l'événement

Inconvénients

- Rarement utilisable
- Difficile à mettre en place à cause de la fluctuation quasi-permanence des cotes
- Surveillance des bookmakers qui peuvent geler les comptes du parieur

PROBLÉMATIQUE

Quelle stratégie un parieur doit-il adopter afin de maximiser ses gains ?

SOMMAIRE

I – Estimation de probabilités

- A) Récolte de données
- B) Mise en place d'un système de points-score
- C) Fiabilité de l'algorithme

II – Stratégies pour parier

- A) Différentes stratégies possibles
- B) Implémentation en Python
- C) Résultats graphiques
- D) Interprétation

Annexes

SOMMAIRE

I – Estimation de probabilités

- A) Récolte de données
- B) Mise en place d'un système de points-score
- C) Fiabilité de l'algorithme

II – Stratégies pour parier

- A) Différentes stratégies possibles
- B) Implémentation en Python
- C) Résultats graphiques
- D) Interprétation

Annexes

I – ESTIMATION DE PROBABILITÉS

A) RÉCOLTE DE DONNÉES

```
"Djokovic N.": {  
  "Rank (YTD)": 1,  
  "Titles (Career)": 98,  
  "Wins (YTD)": 12,  
  "Loses (YTD)": 5,  
  "Wins (Career)": 1099,  
  "Loses (Career)": 218,  
  "Preferred Surface": "Grass",  
  "Dominant Hand": "Right-Handed",  
  "1st Serve Points Won": 74,  
  "1st Serve Return Points Won": 34,  
  "2nd Serve Points Won": 56,  
  "2nd Serve Return Points Won": 55,  
  "Break Points Converted": 44,  
  "Break Points Saved": 65,  
  "Clay Index (Career)": 0.801,  
  "Grass Index (Career)": 0.858,  
  "Hard Index (Career)": 0.847,  
  "VS Right-Handers Index (Career)": 0.845,  
  "VS Left-Handers Index (Career)": 0.768  
},
```

Données d'un joueur

```
"2024C-MiHa0": {  
  "Date": 2024,  
  "Surface": "Clay",  
  "Player 1": "Misolic F.",  
  "Player 2": "Halys Q.",  
  "Odds Player 1": 2.24,  
  "Odds Player 2": 1.62,  
  "Score Player 1": 2,  
  "Score Player 2": 1  
},
```

Données d'un match

I – ESTIMATION DE PROBABILITÉS

A) RÉCOLTE DE DONNÉES

Exemples de données d'un joueur

Critère	Valeur
Rang	27
Surface favorite	“Terre battue”
Nombre de victoires/défaites	12 – 9
Proportion de 1 ^{er} service gagnant	72 %

I – ESTIMATION DE PROBABILITÉS

A) RÉCOLTE DE DONNÉES

Exemples de données d'un joueur

Critère	Valeur
Rang	27
Surface favorite	"Terre battue"
Nombre de victoires/défaites	12 - 9
Proportion de 1 ^{er} service gagnant	72 %



Les données ne se présentent pas sous la même forme

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Système de points-score



Points-score du joueur n°1



Points-score du joueur n°2

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Système de points-score



Points-score du joueur n°1



Points-score du joueur n°2

$$\frac{\text{Points-score du joueur}}{\text{Points-score du joueur} + \text{Points-score de l'adversaire}} = \text{Probabilité de victoire du joueur}$$

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Quantification en points-score

Critère	Valeur	Formule de calcul des points-score
Rang	27	
Surface favorite	“Terre battue”	
Nombre de victoires/défaites	12 - 9	
Proportion de 1 ^{er} service gagnant	72 %	

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Quantification en points-score

Critère	Valeur	Formule de calcul des points-score
Rang	27	Rang du joueur – Rang de l'adversaire
Surface favorite	"Terre battue"	
Nombre de victoires/défaites	12 – 9	
Proportion de 1 ^{er} service gagnant	72 %	

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Quantification en points-score

Critère	Valeur	Formule de calcul des points-score
Rang	27	Rang du joueur – Rang de l'adversaire
Surface favorite	"Terre battue"	Index de victoire associé à la surface du match
Nombre de victoires/défaites	12 – 9	
Proportion de 1 ^{er} service gagnant	72 %	

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Quantification en points-score

Critère	Valeur	Formule de calcul des points-score
Rang	27	Rang du joueur – Rang de l'adversaire
Surface favorite	"Terre battue"	Index de victoire associé à la surface du match
Nombre de victoires/défaites	12 – 9	Nombre de victoires – Nombre de défaites
Proportion de 1 ^{er} service gagnant	72 %	

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Quantification en points-score

Critère	Valeur	Formule de calcul des points-score
Rang	27	Rang du joueur – Rang de l'adversaire
Surface favorite	"Terre battue"	Index de victoire associé à la surface du match
Nombre de victoires/défaites	12 – 9	Nombre de victoires – Nombre de défaites
Proportion de 1 ^{er} service gagnant	72 %	/

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Critère	Valeur	Points-score
Rang	27	19
Surface favorite	“Terre battue”	0,81
Nombre de victoires/défaites	12 - 9	3
Proportion de 1 ^{er} service gagnant	72 %	0,72

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Critère	Valeur	Points-score
Rang	27	19
Surface favorite	“Terre battue”	0,81
Nombre de victoires/défaites	12 - 9	3
Proportion de 1 ^{er} service gagnant	72 %	0,72

- Les points-score sont très différents selon le critère
- Tous les critères n'admettent pas la même importance

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Critère	Valeur	Points-score
Rang	27	19
Surface favorite	“Terre battue”	0,81
Nombre de victoires/défaites	12 - 9	3
Proportion de 1 ^{er} service gagnant	72 %	0,72

- Les points-score sont très différents selon le critère
- Tous les critères n'admettent pas la même importance



Nécessité d'un ajustement

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Ajustement

$$\sum_i (\text{points-score})_i$$

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Ajustement

$$\sum_i (\text{points-score})_i$$



$$\underbrace{\sum_i c_i (\text{points-score})_i}_{\text{Points-score total du joueur}}$$

Points-score total du joueur

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Choix de pondérations

Critère	Valeur	Points-score	Coefficient c_i
Rang	27	19	0,015
Surface favorite	"Terre battue"	0,81	3
Nombre de victoires/défaites	12 - 9	3	0,1
Proportion de 1 ^{er} service gagnant	72 %	0,72	2

I – ESTIMATION DE PROBABILITÉS

B) MISE EN PLACE D'UN SYSTÈME DE POINTS-SCORE

Système de points-score



Points-score du joueur n°1



Points-score du joueur n°2

$$\frac{\text{Points-score du joueur}}{\text{Points-score du joueur} + \text{Points-score de l'adversaire}} = \text{Probabilité de victoire du joueur}$$

I – ESTIMATION DE PROBABILITÉS

C) FIABILITÉ DE L'ALGORITHME

Systeme de points-score



Points-score du joueur n°1



Points-score du joueur n°2

$$\frac{\text{Points-score du joueur}}{\text{Points-score du joueur} + \text{Points-score de l'adversaire}} = \text{Probabilité de victoire du joueur}$$

I – ESTIMATION DE PROBABILITÉS

C) FIABILITÉ DE L'ALGORITHME

Systeme de points-score



Points-score du joueur n°1



Points-score du joueur n°2

$$\frac{\text{Points-score du joueur}}{\text{Points-score du joueur} + \text{Points-score de l'adversaire}} = \text{Probabilité de victoire du joueur}$$



Algorithme fiable à 75 %

I – ESTIMATION DE PROBABILITÉS

C) FIABILITÉ DE L'ALGORITHME

Et les 25 % restants ?

I – ESTIMATION DE PROBABILITÉS

C) FIABILITÉ DE L'ALGORITHME

Et les 25 % restants ?

- Utilisation de données statistiques, provenant des matchs précédents, uniquement
- Existence d'autres critères (forme physique, météo, jeu à domicile ou non, ...)
- Choix des pondérations c_i non optimal

SOMMAIRE

I – Estimation de probabilités

- A) Récolte de données
- B) Mise en place d'un système de points-score
- C) Fiabilité de l'algorithme

II – Stratégies pour parier

- A) Différentes stratégies possibles
- B) Implémentation en Python
- C) Résultats graphiques
- D) Interprétation

Annexes

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Avantages

- Aucun risque
- Bénéfice quel que soit l'issue de l'événement

Inconvénients

- Rarement utilisable
- Difficile à mettre en place à cause de la fluctuation quasi-permanence des cotes
- Surveillance des bookmakers qui peuvent geler les comptes du parieur

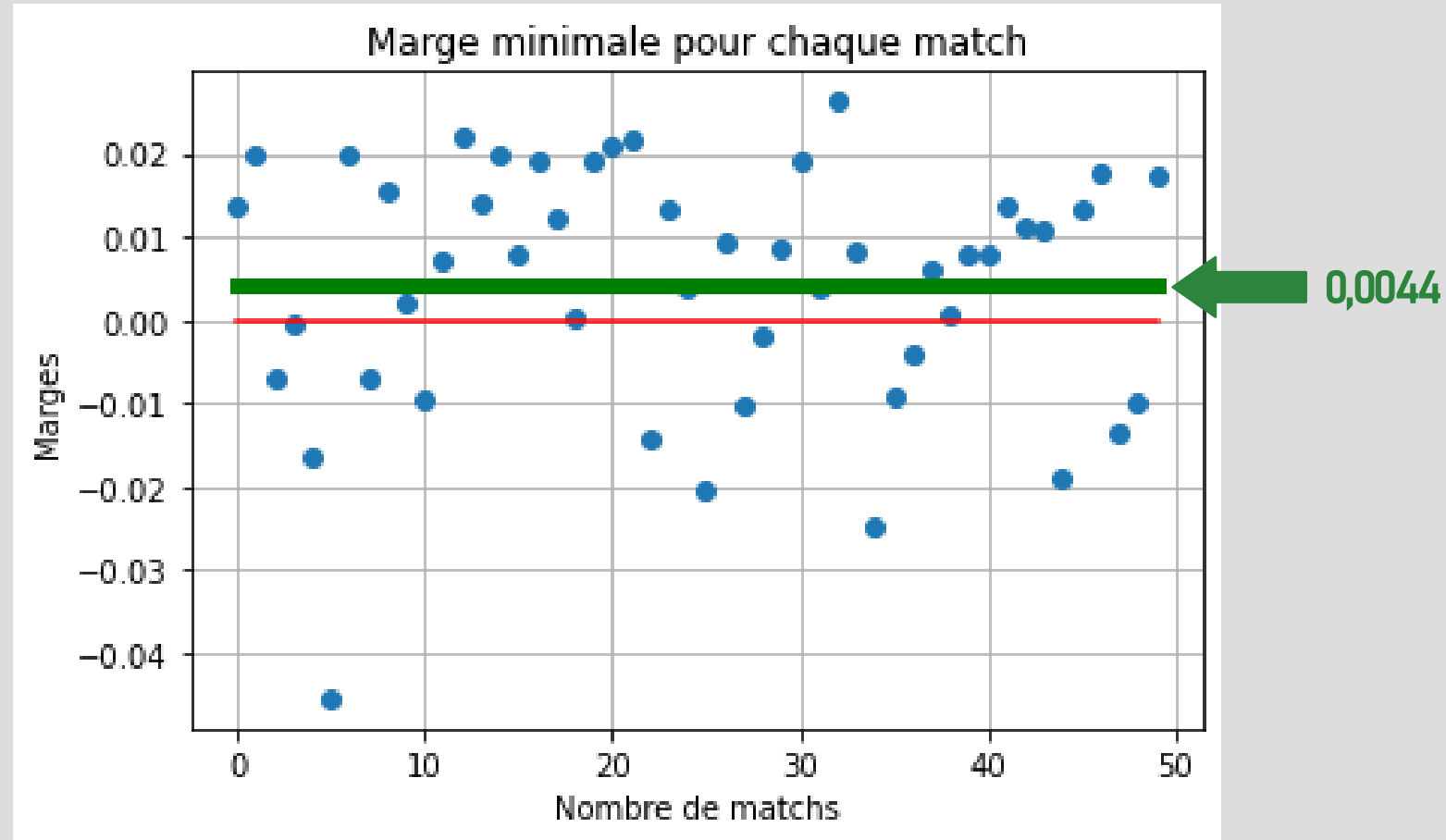
II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

```
"xYXGult3": {  
  "10x10bet": {  
    "Odds Player 1": 1.14,  
    "Odds Player 2": 5.25  
  },  
  "1xBet": {  
    "Odds Player 1": 1.15,  
    "Odds Player 2": 5.55  
  },  
  "Alphabet": {  
    "Odds Player 1": 1.15,  
    "Odds Player 2": 5.5  
  },  
  "bet-at-home": {  
    "Odds Player 1": 1.16,  
    "Odds Player 2": 5.0  
  },  
  "bet365": {  
    "Odds Player 1": 1.14,  
    "Odds Player 2": 5.5  
  },  
  "BetInAsia": {  
    "Odds Player 1": 1.16,  
    "Odds Player 2": 6.01  
  },  
  "GGBET": {  
    "Odds Player 1": 1.18,  
    "Odds Player 2": 5.45  
  },  
  "Lasbet": {  
    "Odds Player 1": 1.15,  
    "Odds Player 2": 5.5  
  },  
  "Pinnacle": {  
    "Odds Player 1": 1.17,  
    "Odds Player 2": 6.01  
  },  
  "Unibet": {  
    "Odds Player 1": 1.16,  
    "Odds Player 2": 5.4  
  },  
  "VOBET": {  
    "Odds Player 1": 1.15,  
    "Odds Player 2": 5.5  
  },  
  "Vulkan Bet": {  
    "Odds Player 1": 1.18,  
    "Odds Player 2": 5.45  
  },  
  "William Hill": {  
    "Odds Player 1": 1.14,  
    "Odds Player 2": 5.5  
  }  
}
```

II – STRATÉGIES POUR PARIER



A) DIFFÉRENTES STRATÉGIES POSSIBLES



Marges minimales possibles pour 50 matchs de tennis



II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Applicable lorsque : Marge < 0	 Joueur n°1	 Joueur n°2
Mise	$\frac{\text{Mise totale}}{(1 + \text{Marge}) \times \text{Cote du joueur n}^\circ 1}$	$\frac{\text{Mise totale}}{(1 + \text{Marge}) \times \text{Cote du joueur n}^\circ 2}$
Gain potentiel	$\frac{\text{Mise totale}}{1 + \text{Marge}}$	$\frac{\text{Mise totale}}{1 + \text{Marge}}$

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Applicable lorsque : Marge < 0	 Joueur n°1	 Joueur n°2
Mise	$\frac{\text{Mise totale}}{(1 + \text{Marge}) \times \text{Cote du joueur n}^\circ 1}$	$\frac{\text{Mise totale}}{(1 + \text{Marge}) \times \text{Cote du joueur n}^\circ 2}$
Gain potentiel	$\frac{\text{Mise totale}}{1 + \text{Marge}}$	$\frac{\text{Mise totale}}{1 + \text{Marge}}$

> Mise totale

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Existe-t-il une stratégie de mise qui puisse se substituer à la stratégie d'arbitrage ?

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #1 – *fifty fifty*



Joueur n°1



Joueur n°2

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #1 – *fifty fifty*



Joueur n°1



50 %



Joueur n°2

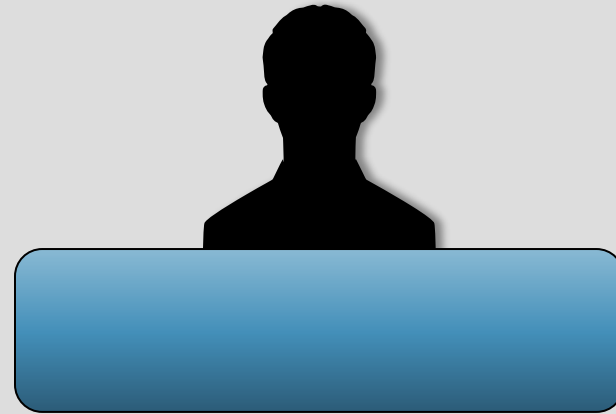
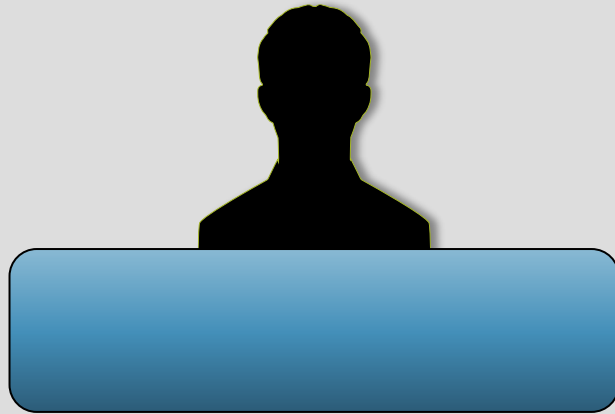


50 %

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #2 – *all on favor*



II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #2 – *all on favor*

Données du match



Cote du joueur n°1

<



Cote du joueur n°2

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #2 – *all on favor*

Données du match



Cote du joueur n°1

<



Cote du joueur n°2

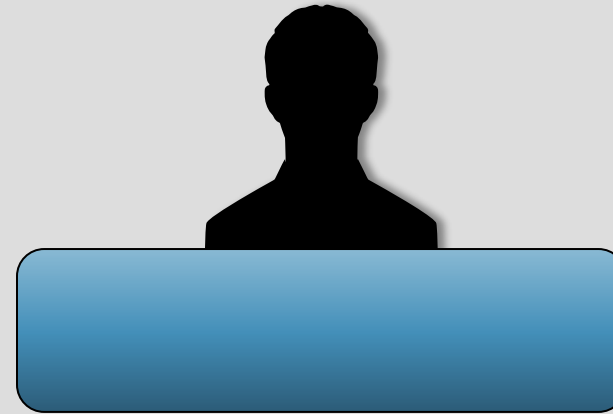
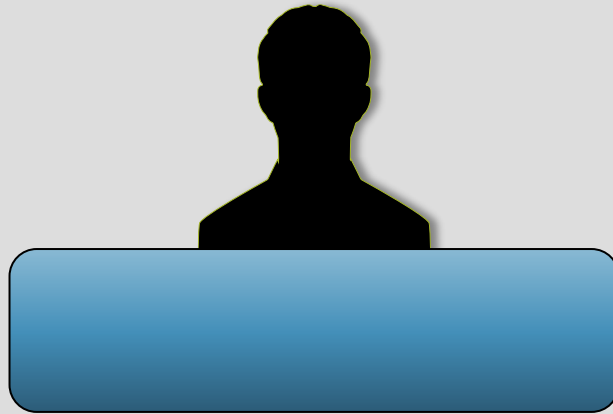


100 %

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #3 – *depending proba*



II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #3 – *depending proba*



Probabilité de
victoire du joueur n°1

>



Probabilité de
victoire du joueur n°2

Estimation de probabilités

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #3 – *depending proba*

Estimation de probabilités



Probabilité de
victoire du joueur n°1

>



Probabilité de
victoire du joueur n°2



Probabilité de
victoire du joueur n°1

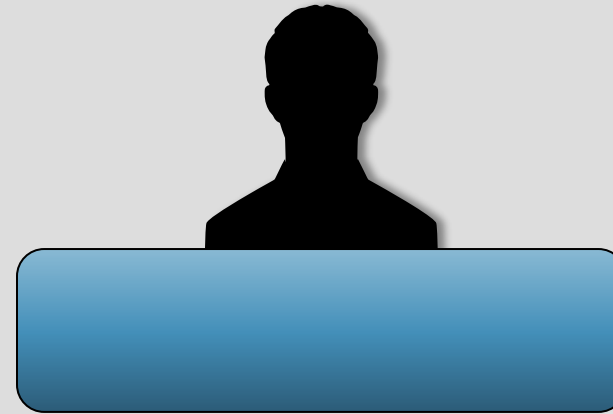
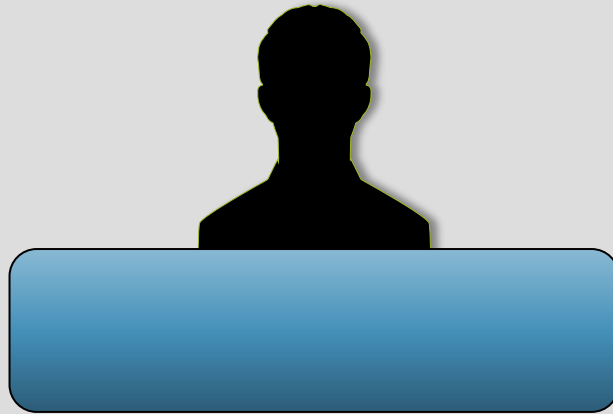


Probabilité de
victoire du joueur n°2

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #4 – *risky*



II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #4 – *risky*

Données du match



Cote du joueur n°1

<



Cote du joueur n°2

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #4 – *risky*

Données du match



Cote du joueur n°1

<



Cote du joueur n°2



100 %

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Stratégie #4 – *risky*

Données du match



Cote du joueur n°1

<




Cote du joueur n°2



100 %

PARTICULARITÉS

- 
- Il suffit de ne gagner qu'une seule fois pour faire un bénéfice
 - La mise n'est pas constante : elle dépend de la cote et du bénéfice souhaité

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Dans le cas où : Cote J1 < Cote J2	 Cote J1	 Cote J2
Mise	0	$\frac{\text{Somme souhaitée} - \text{Somme actuelle}}{\text{Cote J2} - 1}$
Gain potentiel	0	$\frac{\text{Somme souhaitée} - \text{Somme actuelle}}{\text{Cote J2} - 1} \times \text{Cote J2}$

II – STRATÉGIES POUR PARIER

A) DIFFÉRENTES STRATÉGIES POSSIBLES

Dans le cas où : Cote J1 < Cote J2	 Cote J1	 Cote J2
Mise	0	$\frac{\text{Somme souhaitée} - \text{Somme actuelle}}{\text{Cote J2} - 1}$
Gain potentiel	0	$\frac{\text{Somme souhaitée} - \text{Somme actuelle}}{\text{Cote J2} - 1} \times \text{Cote J2}$

Somme finale :
Somme souhaitée

II – STRATÉGIES POUR PARIER

B) IMPLÉMENTATION EN PYTHON

Etapas d'implémentation

Pour chaque stratégie :

II – STRATÉGIES POUR PARIER

B) IMPLÉMENTATION EN PYTHON

Etapes d'implémentation

Pour chaque stratégie :

- 1) En possession de 1 000 € au départ, on parie sur l'issue d'un match sélectionné au hasard selon la stratégie choisie

II – STRATÉGIES POUR PARIER

B) IMPLÉMENTATION EN PYTHON

Etapes d'implémentation

Pour chaque stratégie :

- 1) En possession de 1 000 € au départ, on parie sur l'issue d'un match sélectionné au hasard selon la stratégie choisie
- 2) On itère l'opération, tant que :
 - l'on possède toujours de l'argent
 - tant qu'on n'a pas parié sur plus de 500 matchs

II – STRATÉGIES POUR PARIER

B) IMPLÉMENTATION EN PYTHON

Etapes d'implémentation

Pour chaque stratégie :

- 1) En possession de 1 000 € au départ, on parie sur l'issue d'un match sélectionné au hasard selon la stratégie choisie
- 2) On itère l'opération, tant que :
 - l'on possède toujours de l'argent
 - tant qu'on n'a pas parié sur plus de 500 matchs
- 3) On trace le graphe représentant l'évolution de la somme en fonction du nombre de paris effectués

II – STRATÉGIES POUR PARIER

B) IMPLÉMENTATION EN PYTHON

Exemple

Stratégie #2 – *all on favor*

```
36 def all_on_favor(mise):
37     S = 1000
38     liste_sommes = [S]
39     while S > 0 and len(liste_sommes) < nb_match:
40         match = find_match()
41         M = mise
42         S -= M
43         odds = (match["Odds Player 1"], match["Odds Player 2"])
44         if odds[0] < odds[1] and match["Score Player 1"] > match["Score Player 2"]:
45             S += M*odds[0]
46         elif odds[0] > odds[1] and match["Score Player 1"] < match["Score Player 2"]:
47             S += M*odds[1]
48         liste_sommes.append(S)
49     return liste_sommes
```

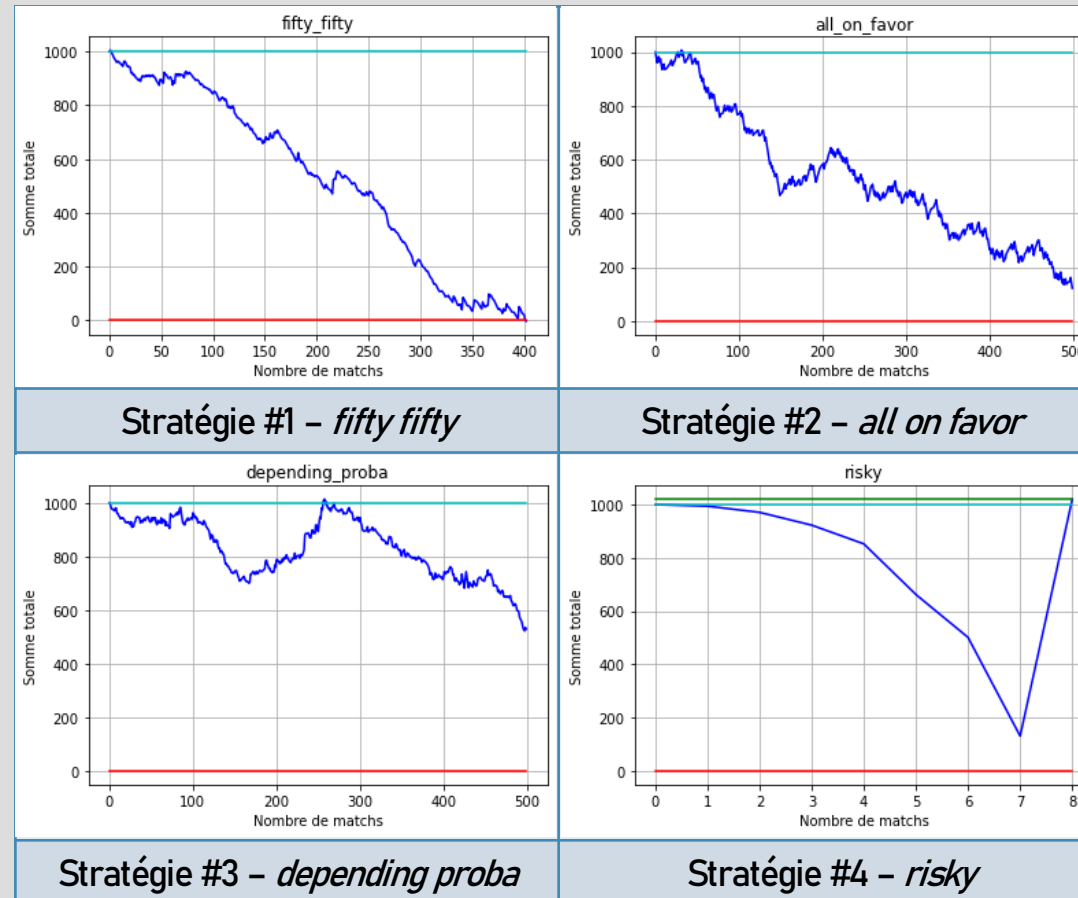
Code des étapes 1) et 2)

```
49 def curve_all_on_favor(mise):
50     values = all_on_favor(mise)
51     Y = np.array(values)
52     Yinit = np.array([1000 for i in range(len(values))])
53     Y0 = np.array([0 for i in range(len(values))])
54     X = np.array([i for i in range(len(values))])
55     plt.plot(X, Y, 'b')
56     plt.plot(X, Y0, 'r')
57     plt.plot(X, Yinit, 'c')
58     plt.xlabel("Nombre de matchs")
59     plt.ylabel("Somme totale")
60     plt.title("all_on_favor")
61     plt.grid()
62     plt.show()
```

Code de l'étape 3)

II – STRATÉGIES POUR PARIER

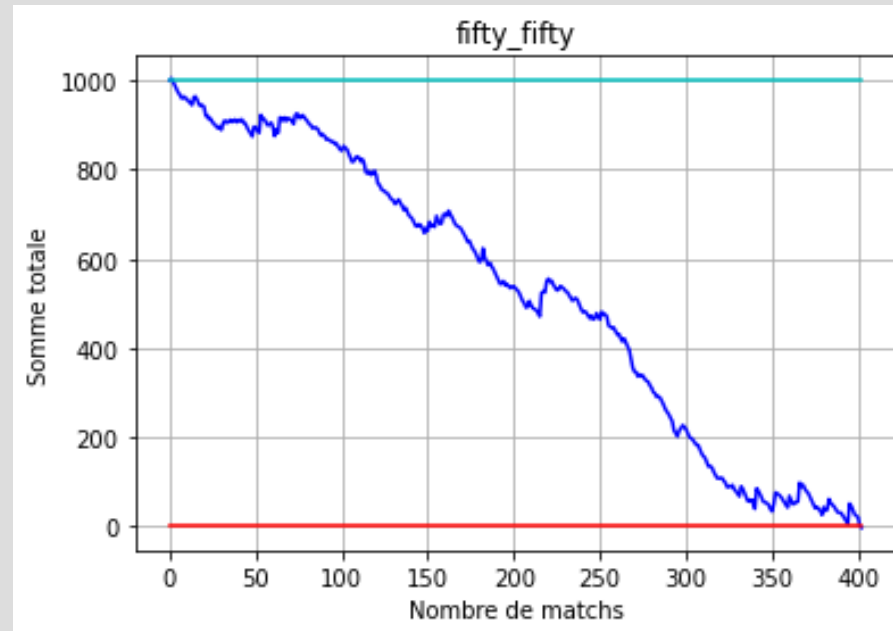
C) RÉSULTATS GRAPHIQUES



II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

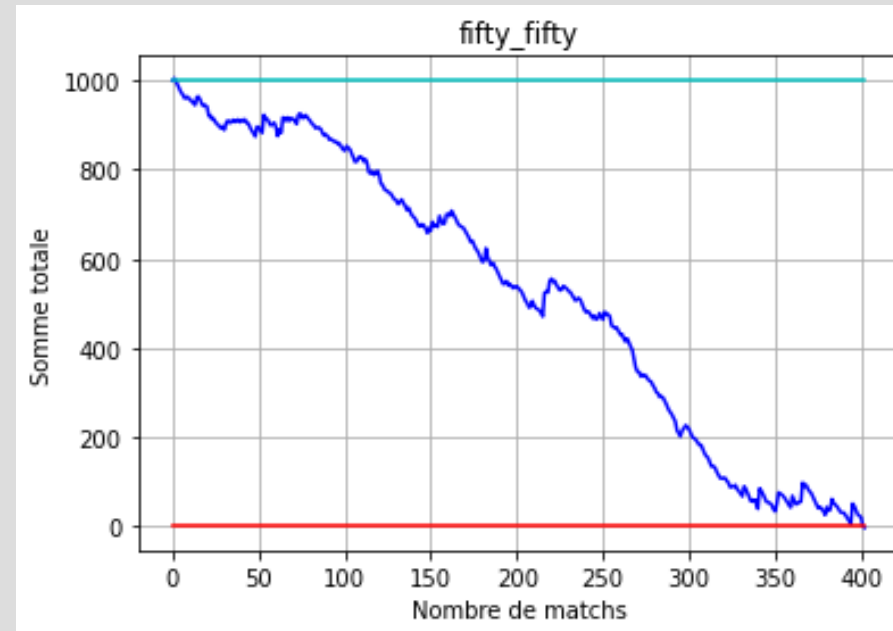
Stratégie #1 – *fifty fifty*



II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

Stratégie #1 – *fifty fifty*



- Tout l'argent est perdu au bout de 400 paris
- Chute est quasi-linéaire

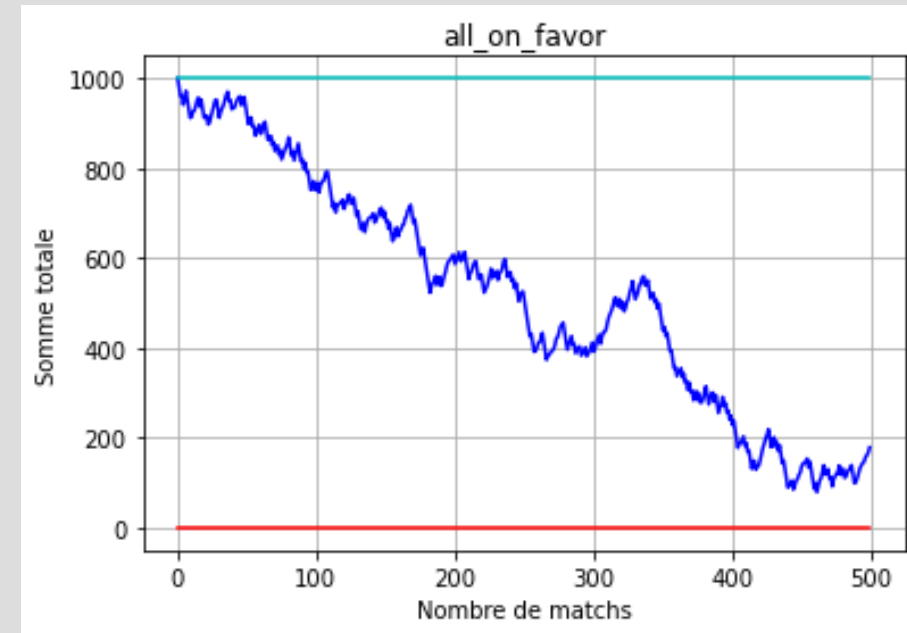
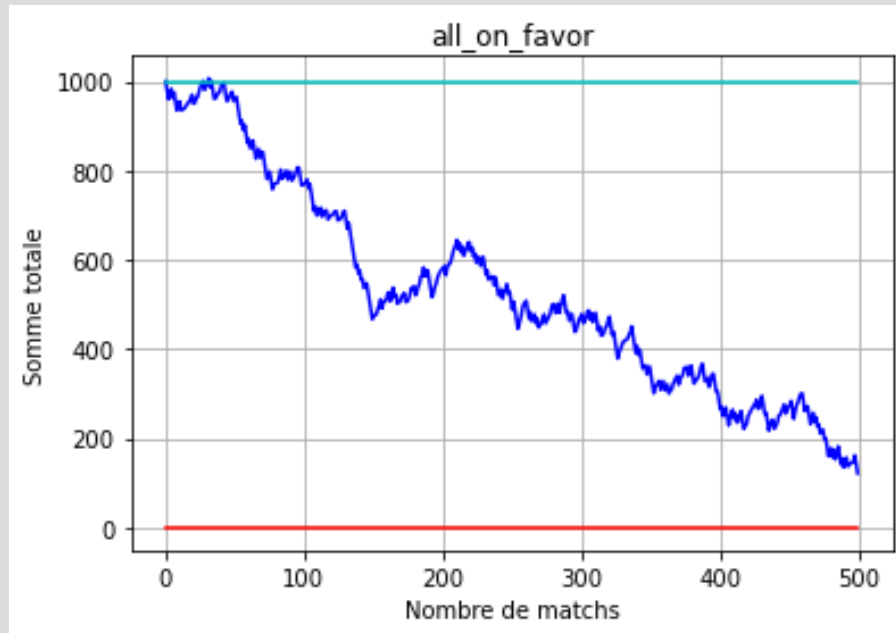


Très mauvaise stratégie

II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

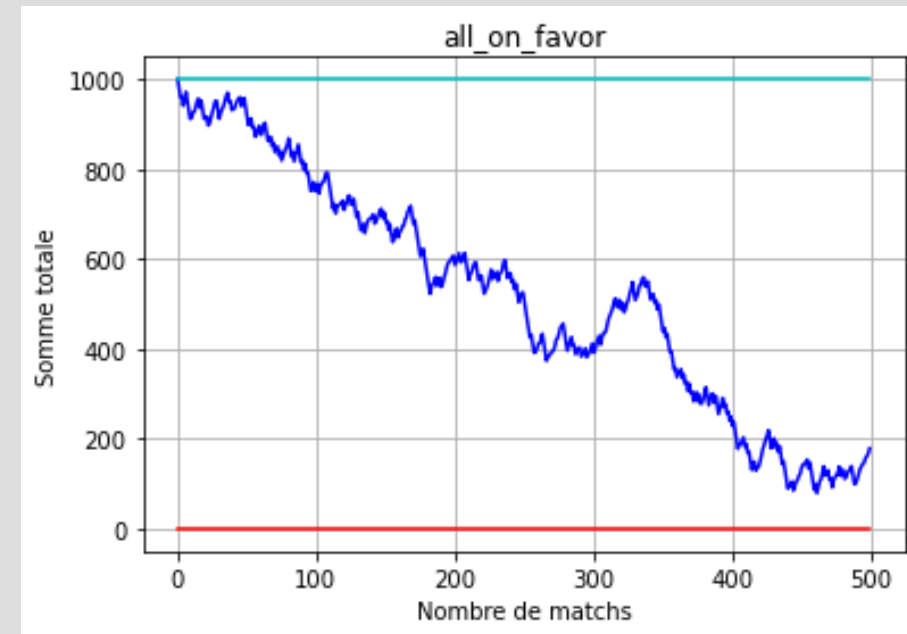
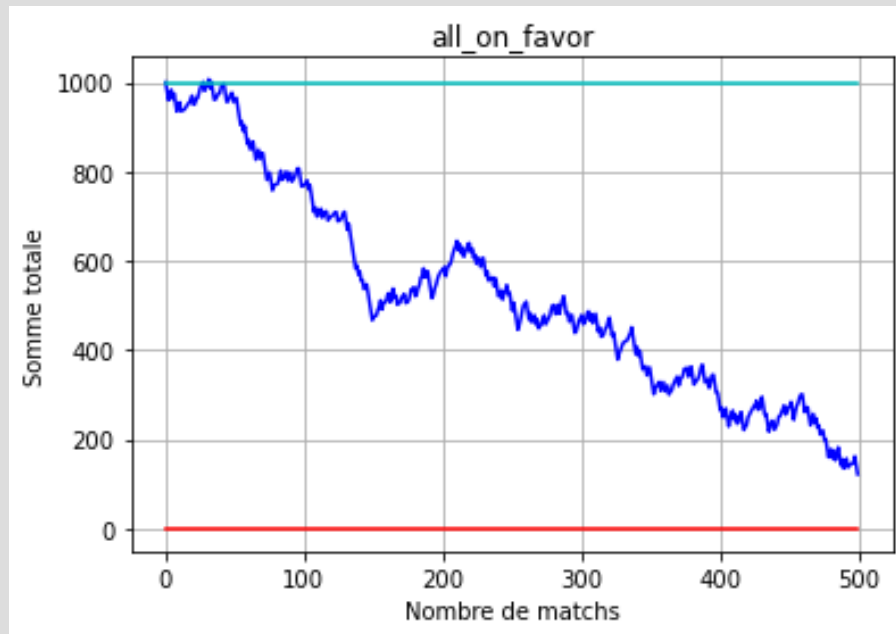
Stratégie #2 – *all on favor*



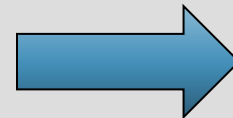
II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

Stratégie #2 – *all on favor*



- Descente globalement constante
- Quelques remontées

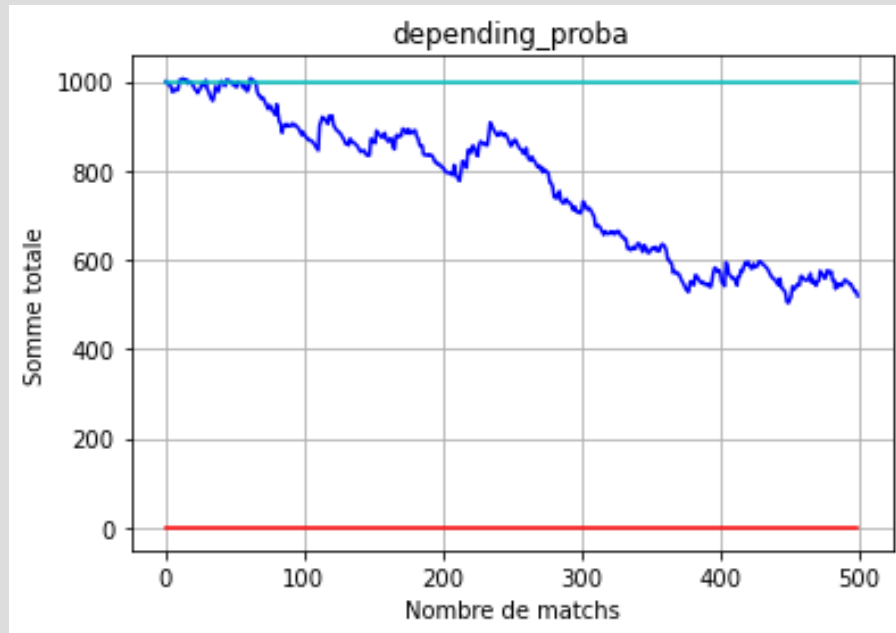


Stratégie peu fructueuse

II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

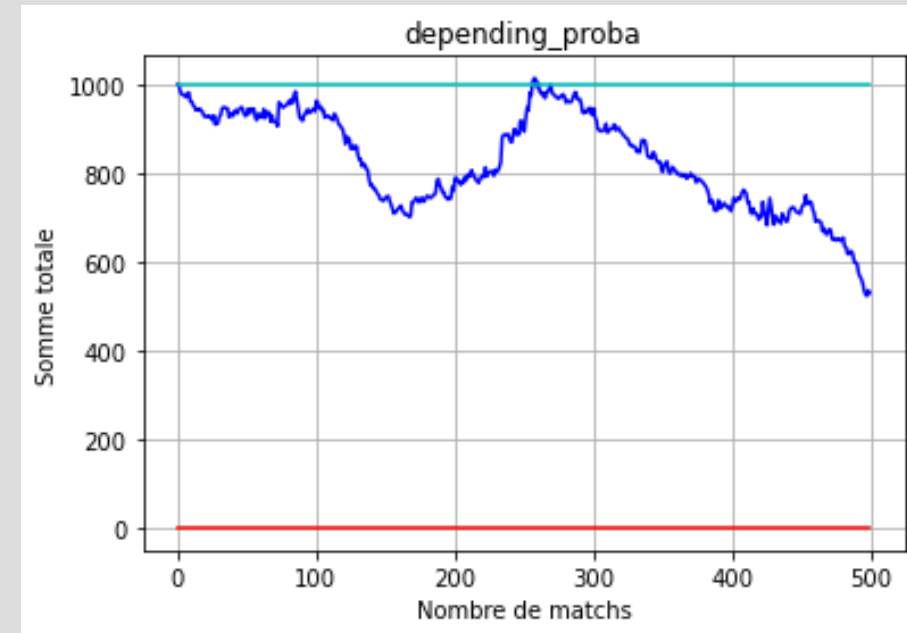
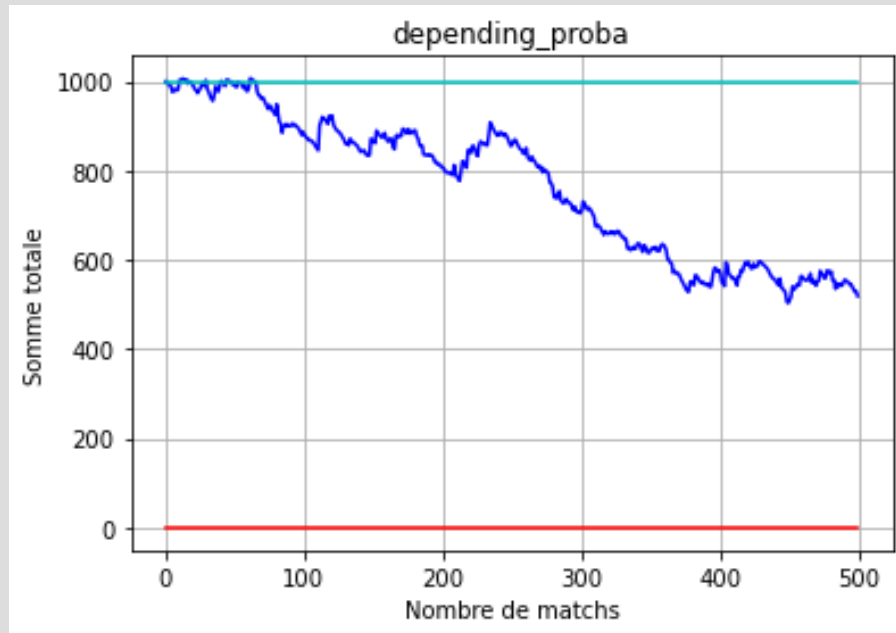
Stratégie #3 – *depending proba*



II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

Stratégie #3 – *depending proba*



- Descente assez faible, mais existante
- Augmentations notables
- Possibilité éventuelle de faire un bénéfice

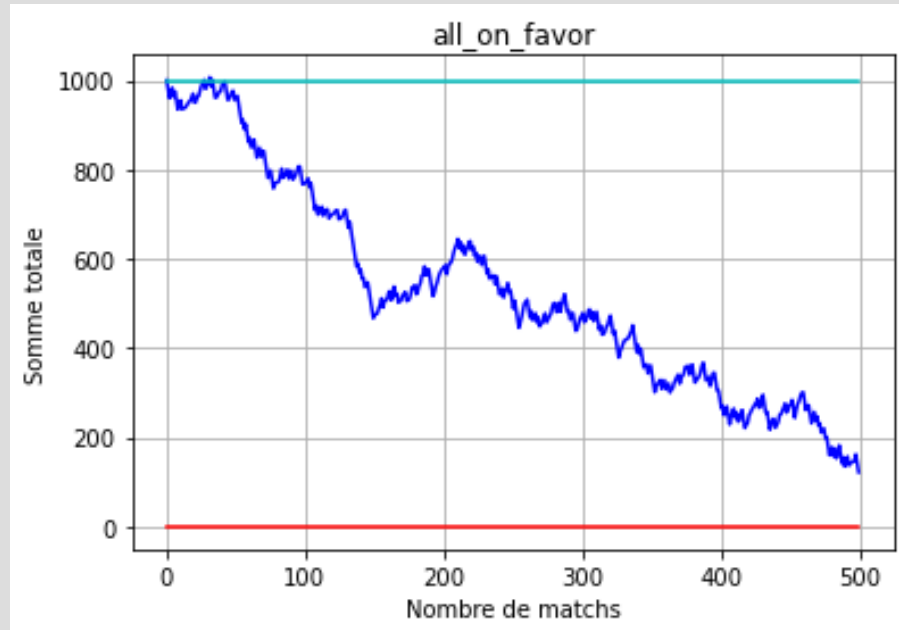


Stratégie pouvant être
bénéfique à court terme

II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

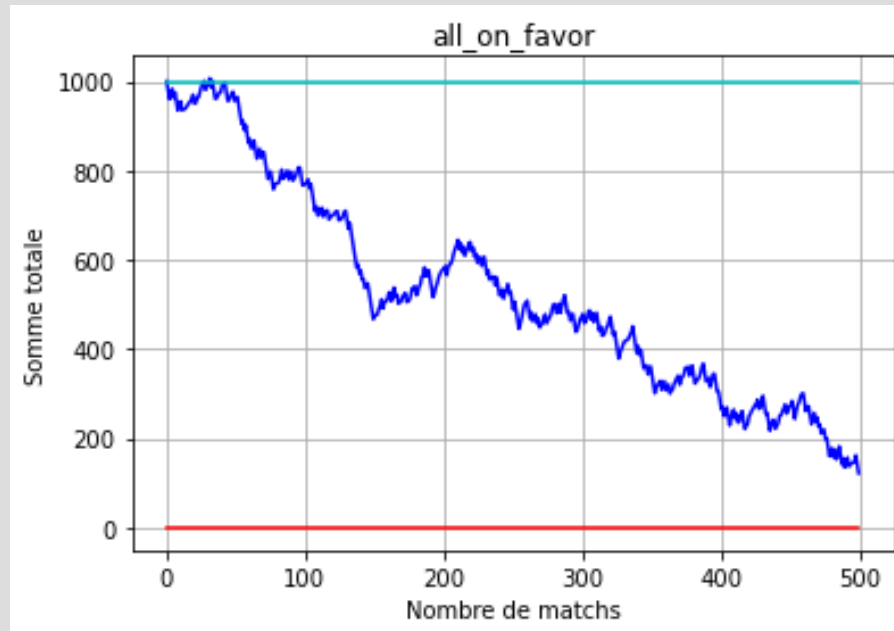
Comparaison des stratégies #2 et #3



II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

Comparaison des stratégies #2 et #3

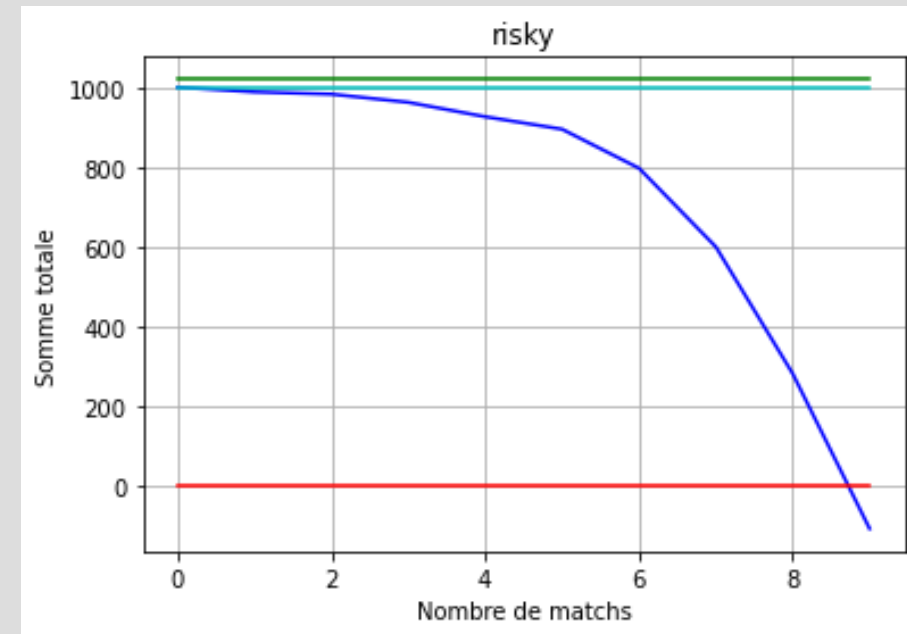
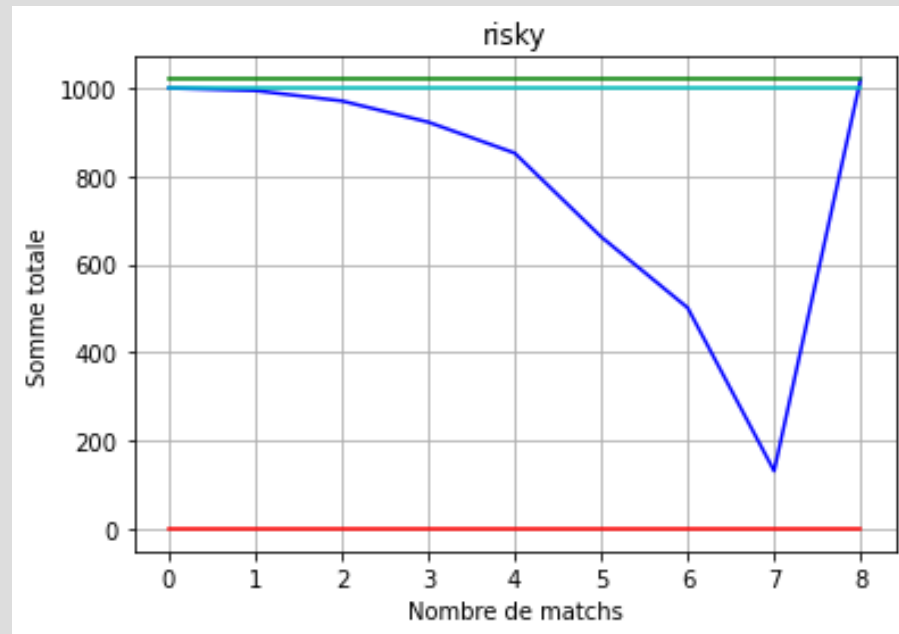


L'algorithme d'estimation de probabilités semble plutôt fiable

II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

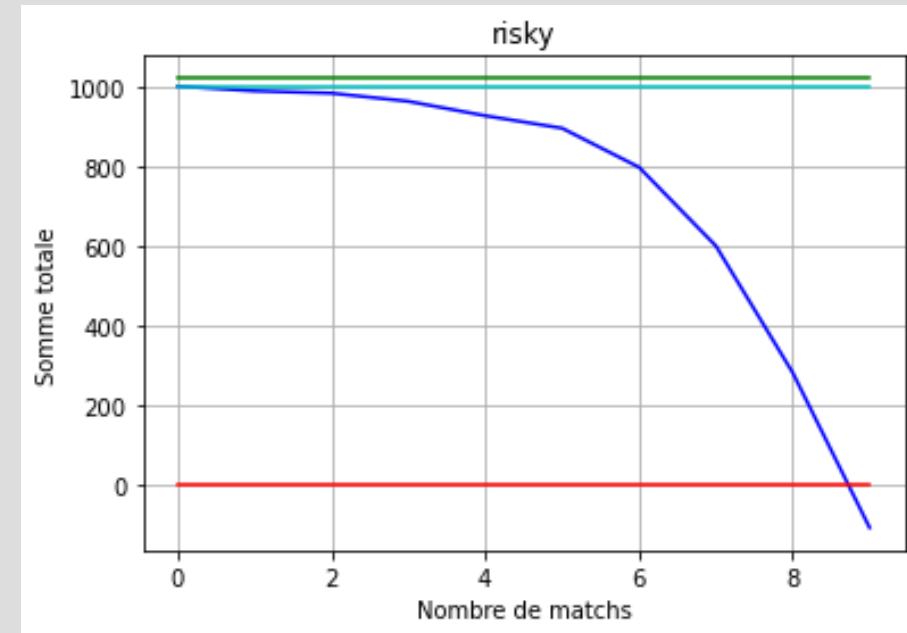
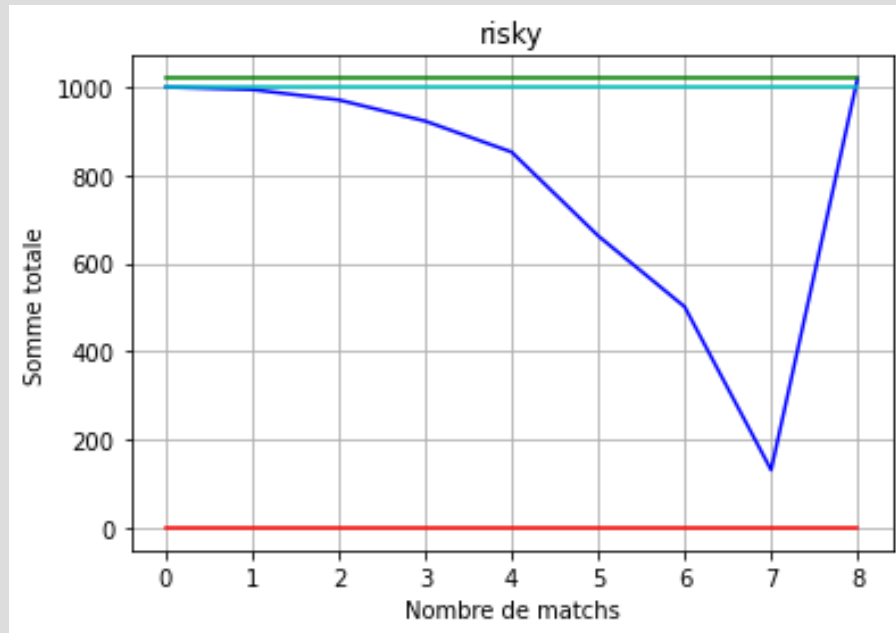
Stratégie #4 – *risky*



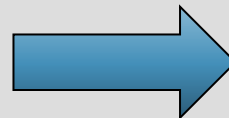
II – STRATÉGIES POUR PARIER

D) INTERPRÉTATION

Stratégie #4 – *risky*

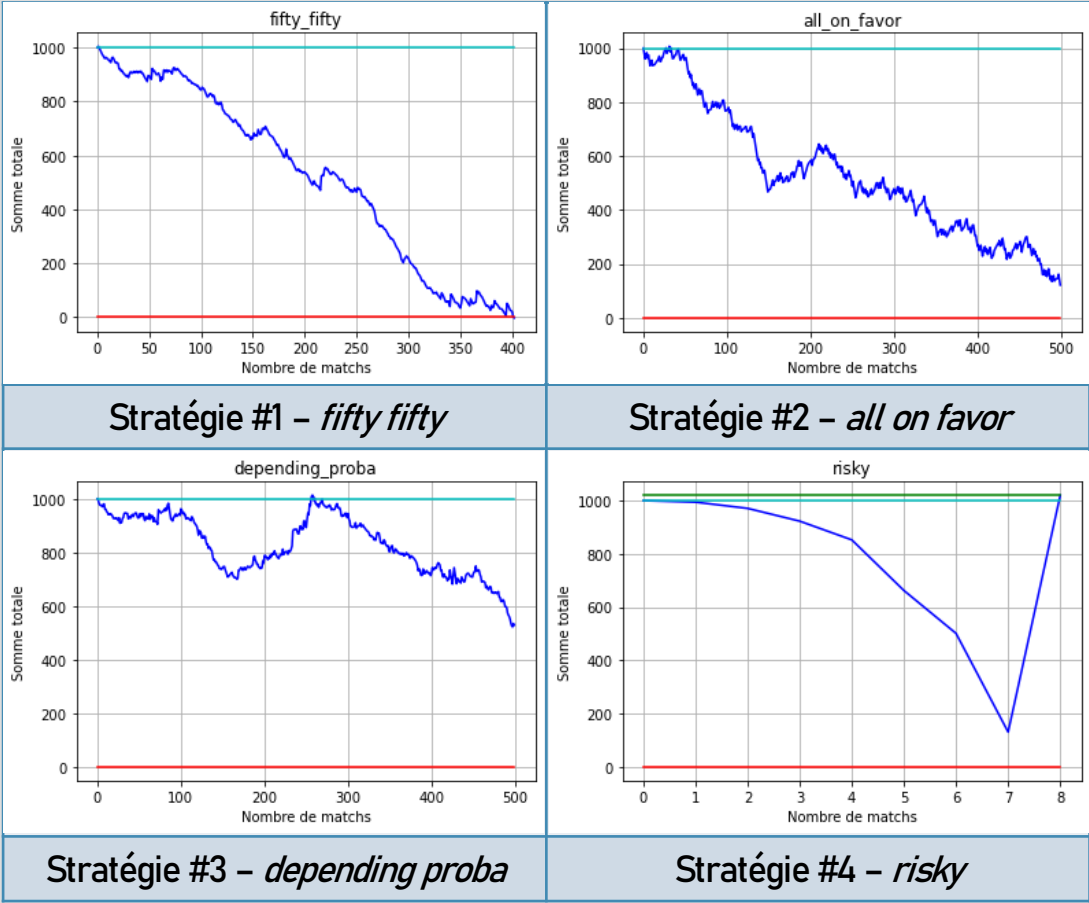


- Nombre de matchs très restreint
- Chute brutale
- Cas de figure n°1 majoritaire en pratique



Stratégie abordable pour
des bénéfices très minimes

CONCLUSION



FIN

MERCI DE VOTRE ATTENTION

SOMMAIRE

I – Estimation de probabilités

- A) Récolte de données
- B) Mise en place d'un système de points-score
- C) Fiabilité de l'algorithme

II – Stratégies pour parier

- A) Différentes stratégies possibles
- B) Implémentation en Python
- C) Résultats graphiques
- D) Interprétation

Annexes

ANNEXES

A) DONNÉES

```
"Djokovic N.": {  
  "Rank (YTD)": 1,  
  "Titles (Career)": 98,  
  "Wins (YTD)": 12,  
  "Loses (YTD)": 5,  
  "Wins (Career)": 1099,  
  "Loses (Career)": 218,  
  "Preferred Surface": "Grass",  
  "Dominant Hand": "Right-Handed",  
  "1st Serve Points Won": 74,  
  "1st Serve Return Points Won": 34,  
  "2nd Serve Points Won": 56,  
  "2nd Serve Return Points Won": 55,  
  "Break Points Converted": 44,  
  "Break Points Saved": 65,  
  "Clay Index (Career)": 0.801,  
  "Grass Index (Career)": 0.858,  
  "Hard Index (Career)": 0.847,  
  "VS Right-Handers Index (Career)": 0.845,  
  "VS Left-Handers Index (Career)": 0.768  
},
```

ANNEXES

A) DONNÉES

```
"2024C-MiHa0": {  
  "Date": 2024,  
  "Surface": "Clay",  
  "Player 1": "Misolic F.",  
  "Player 2": "Halys Q.",  
  "Odds Player 1": 2.24,  
  "Odds Player 2": 1.62,  
  "Score Player 1": 2,  
  "Score Player 2": 1  
},
```

ANNEXES

A) DONNÉES

```
"xYXGult3": {
  "10x10bet": {
    "Odds Player 1": 1.14,
    "Odds Player 2": 5.25
  },
  "1xBet": {
    "Odds Player 1": 1.15,
    "Odds Player 2": 5.55
  },
  "Alphabet": {
    "Odds Player 1": 1.15,
    "Odds Player 2": 5.5
  },
  "bet-at-home": {
    "Odds Player 1": 1.16,
    "Odds Player 2": 5.0
  },
  "bet365": {
    "Odds Player 1": 1.14,
    "Odds Player 2": 5.5
  },
  "BetInAsia": {
    "Odds Player 1": 1.16,
    "Odds Player 2": 6.01
  },
  "GGBET": {
    "Odds Player 1": 1.18,
    "Odds Player 2": 5.45
  },
  "Lasbet": {
    "Odds Player 1": 1.15,
    "Odds Player 2": 5.5
  },
  "Pinnacle": {
    "Odds Player 1": 1.17,
    "Odds Player 2": 6.01
  },
  "Unibet": {
    "Odds Player 1": 1.16,
    "Odds Player 2": 5.4
  },
  "VOBET": {
    "Odds Player 1": 1.15,
    "Odds Player 2": 5.5
  },
  "Vulkan Bet": {
    "Odds Player 1": 1.18,
    "Odds Player 2": 5.45
  },
  "William Hill": {
    "Odds Player 1": 1.14,
    "Odds Player 2": 5.5
  }
}
```

ANNEXES

B) STRATÉGIES

```
52     def fifty_fifty(mise):
53         S = 1000
54         liste_sommes = [S]
55         while S > 0 and len(liste_sommes) < nb_match:
56             match = find_match()
57             M = mise
58             S -= M
59             odds = (match["Odds Player 1"], match["Odds Player 2"])
60             if match["Score Player 1"] > match["Score Player 2"]:
61                 S += (M/2) * odds[0]
62             elif match["Score Player 1"] < match["Score Player 2"]:
63                 S += (M/2) * odds[1]
64             liste_sommes.append(S)
65         return liste_sommes
```

ANNEXES

B) STRATÉGIES

```
36 def all_on_favor(mise):
37     S = 1000
38     liste_sommes = [S]
39     while S > 0 and len(liste_sommes) < nb_match:
40         match = find_match()
41         M = mise
42         S -= M
43         odds = (match["Odds Player 1"], match["Odds Player 2"])
44         if odds[0] < odds[1] and match["Score Player 1"] > match["Score Player 2"]:
45             S += M*odds[0]
46         elif odds[0] > odds[1] and match["Score Player 1"] < match["Score Player 2"]:
47             S += M*odds[1]
48         liste_sommes.append(S)
49     return liste_sommes
```


ANNEXES

B) STRATÉGIES

```
68     def depending_proba(mise):
69         S = 1000
70         liste_sommes = [S]
71         while S > 0 and len(liste_sommes) < nb_match:
72             match = find_match()
73             M = mise
74             S -= M
75             proba = estim_proba(match)
76             odds = (match["Odds Player 1"], match["Odds Player 2"])
77             if match["Score Player 1"] > match["Score Player 2"]:
78                 S += odds[0] * M * proba[0]
79             elif match["Score Player 1"] < match["Score Player 2"]:
80                 S += odds[1] * M * proba[1]
81             liste_sommes.append(S)
82         return liste_sommes
```

ANNEXES

B) STRATÉGIES

```
19 def risky(benefice):
20     s0 = 1000
21     S = s0
22     liste_sommes = [S]
23     while S > 0 and len(liste_sommes) < nb_match and S <= s0:
24         match = find_match()
25         odds = (match["Odds Player 1"], match["Odds Player 2"])
26         M = (benefice + s0 - S) / (max(odds) - 1)
27         S -= M
28         if odds[0] < odds[1] and match["Score Player 2"] > match["Score Player 1"]:
29             S += M*odds[1]
30         elif odds[0] > odds[1] and match["Score Player 1"] > match["Score Player 2"]:
31             S += M*odds[0]
32         liste_sommes.append(S)
33     return liste_sommes
```

ANNEXES

C) TRACÉS

```
15     def curve_fifty_fifty(mise):
16         values = fifty_fifty(mise)
17         Y = np.array(values) # strat
18         X = np.array([i for i in range(len(values))])
19         Y0 = np.array([0 for i in range(len(values))])
20         Yinit = np.array([1000 for i in range(len(values))])
21         plt.plot(X,Y,'b')
22         plt.plot(X,Y0,'r')
23         plt.plot(X,Yinit,'c')
24         plt.xlabel("Nombre de matchs")
25         plt.ylabel("Somme totale")
26         plt.title("fifty_fifty")
27         plt.grid()
28         plt.show()
```

ANNEXES

C) TRACÉS

```
49     def curve_all_on_favor(mise):
50         values = all_on_favor(mise)
51         Y = np.array(values)
52         Yinit = np.array([1000 for i in range(len(values))])
53         Y0 = np.array([0 for i in range(len(values))])
54         X = np.array([i for i in range(len(values))])
55         plt.plot(X,Y, 'b')
56         plt.plot(X,Y0, 'r')
57         plt.plot(X,Yinit, 'c')
58         plt.xlabel("Nombre de matchs")
59         plt.ylabel("Somme totale")
60         plt.title("all_on_favor")
61         plt.grid()
62         plt.show()
```

ANNEXES

C) TRACÉS

```
65     def curveDepending_proba(mise):
66         values = depending_proba(mise)
67         Y = np.array(values)
68         Yinit = np.array([1000 for i in range(len(values))])
69         Y0 = np.array([0 for i in range(len(values))])
70         X = np.array([i for i in range(len(values))])
71         plt.plot(X,Y,'b')
72         plt.plot(X,Y0,'r')
73         plt.plot(X,Yinit,'c')
74         plt.xlabel("Nombre de matchs")
75         plt.ylabel("Somme totale")
76         plt.title("depending_proba")
77         plt.grid()
78         plt.show()
```

ANNEXES

C) TRACÉS

```
31 def curve_risky(benefice):
32     values = risky(benefice)
33     Y = np.array(values)
34     X = np.array([i for i in range(len(values))])
35     Y0 = np.array([0 for i in range(len(values))])
36     Yinit = np.array([1000 for i in range(len(values))])
37     Ygain = np.array([(1000 + benefice) for i in range(len(values))])
38     plt.plot(X,Y, 'b')
39     plt.plot(X,Y0, 'r')
40     plt.plot(X,Yinit, 'c')
41     plt.plot(X,Ygain, 'g')
42     plt.xlabel("Nombre de matchs")
43     plt.ylabel("Somme totale")
44     plt.title("risky")
45     plt.grid()
46     plt.show()
```

ANNEXES

C) TRACÉS

```
81     def curve_marge_moyenne():
82         values = liste_marge()
83         average = 0
84         for i in values:
85             average += i
86         average /= len(values)
87         Y = np.array(values)
88         X = np.array([i for i in range(len(values))])
89         Y0 = np.array([0 for i in range(len(values))])
90         Ymoy = np.array([average for i in range(len(values))])
91         plt.plot(X,Y, '')
92         plt.plot(X,Y0, 'r')
93         plt.plot(X,Ymoy, 'g', linewidth = 5)
94         plt.xlabel("Nombre de matchs")
95         plt.ylabel("Marges")
96         plt.title("Marge moyenne pour chaque match")
97         plt.grid()
98         plt.show()
```

ANNEXES

C) TRACÉS

```
101     def curve_marge_minimale():
102         values = liste_arbitrage()
103         average = 0
104         for i in values:
105             average += i
106         average /= len(values)
107         Y = np.array(values)
108         Y0 = np.array([0 for i in range(len(values))])
109         X = np.array([i for i in range(len(values))])
110         Ymoy = np.array([average for i in range(len(values))])
111         print(Ymoy)
112         plt.plot(X,Y,'o')
113         plt.plot(X,Y0,'r')
114         plt.plot(X,Ymoy,'g', linewidth = 5)
115         plt.xlabel("Nombre de matchs")
116         plt.ylabel("Marges")
117         plt.title("Marge minimale pour chaque match")
118         plt.grid()
119         plt.show()
```


ANNEXES

D) ESTIMATION DE PROBABILITÉS

```
16 def calc_pts_score_joueur1(infoJ1,infoJ2,surface):
17     pts_rank = infoJ2["Rank (YTD)"]-infoJ1["Rank (YTD)"]
18     pts_titles = infoJ1["Titles (Career)"]
19     if infoJ2["Dominant Hand"] == "Left-Handed":
20         pts_dom_hand = infoJ1["VS Left-Handers Index (Career)"]
21     else:
22         pts_dom_hand = infoJ1["VS Right-Handers Index (Career)"]
23     if surface == "Clay":
24         pts_surface = infoJ1["Clay Index (Career)"]
25     elif surface == "Grass":
26         pts_surface = infoJ1["Grass Index (Career)"]
27     else:
28         pts_surface = infoJ1["Hard Index (Career)"]
29
30     pts_winlose = infoJ1["Wins (YTD)"]-infoJ1["Loses (YTD)"]
31     pts_serve = (infoJ1["1st Serve Points Won"] + infoJ1["1st Serve Return Points Won"] + infoJ1["2nd Serve Points Won"] + infoJ1["2nd Serve Return Points Won"]) / 100
32     pts_break = (infoJ1["Break Points Converted"] + infoJ1["Break Points Saved"]) / 100
33     pts_score = 0.015*pts_rank + 0.07*pts_titles + 2*pts_dom_hand + 3*pts_surface + 0.1*pts_winlose + 2*pts_serve + 6*pts_break
34     if pts_score <= 0:
35         return 0.01
36     else:
37         return pts_score
```

ANNEXES

D) ESTIMATION DE PROBABILITÉS

```
40 def estim_proba(match):
41     player1,player2 = match["Player 1"],match["Player 2"]
42     surface = match["Surface"]
43     infoJ1,infoJ2 = joueurs[player1],joueurs[player2]
44     pts_score1 = calc_pts_score_joueur1(infoJ1,infoJ2,surface)
45     pts_score2 = calc_pts_score_joueur1(infoJ2,infoJ1,surface)
46     total_pts = pts_score1 + pts_score2
47     return (pts_score1/total_pts,pts_score2/total_pts)
```

ANNEXES

E) ÉVALUATION DE L'IMPORTANCE DES CRITÈRES

```
16 def importance_rank():
17     i = 0
18     total = 0
19     for match in matches.values():
20         total += 1
21         if joueurs[match["Player 1"]]["Rank (YTD)"] < joueurs[match["Player 2"]]["Rank (YTD)"] and match["Score Player 1"] > match["Score Player 2"]:
22             i += 1
23         elif joueurs[match["Player 1"]]["Rank (YTD)"] > joueurs[match["Player 2"]]["Rank (YTD)"] and match["Score Player 1"] < match["Score Player 2"]:
24             i += 1
25     return (100*i/total, total)
```

ANNEXES

E) ÉVALUATION DE L'IMPORTANCE DES CRITÈRES

```
28 def importance_titles():
29     i = 0
30     total = 0
31     for match in matches.values():
32         total += 1
33         if joueurs[match["Player 1"]]["Titles (Career)"] > joueurs[match["Player 2"]]["Titles (Career)"] and match["Score Player 1"] > match["Score Player 2"]:
34             i += 1
35         elif joueurs[match["Player 1"]]["Titles (Career)"] < joueurs[match["Player 2"]]["Titles (Career)"] and match["Score Player 1"] < match["Score Player 2"]:
36             i += 1
37     return (100*i/total, total)
```

ANNEXES

E) ÉVALUATION DE L'IMPORTANCE DES CRITÈRES

```
40 def importance_win_lose():
41     i = 0
42     total = 0
43     for match in matches.values():
44         total += 1
45         win_lose1 = joueurs[match["Player 1"]]["Wins (YTD)"] - joueurs[match["Player 1"]]["Loses (YTD)"]
46         win_lose2 = joueurs[match["Player 2"]]["Wins (YTD)"] - joueurs[match["Player 2"]]["Loses (YTD)"]
47         if win_lose1 > win_lose2 and match["Score Player 1"] > match["Score Player 2"]:
48             i += 1
49         elif win_lose1 < win_lose2 and match["Score Player 1"] < match["Score Player 2"]:
50             i += 1
51     return (100*i/total, total)
```

ANNEXES

F) AUTRES

```
9 matches = read_json("save_matches.json")
10
11 def find_match():
12     cle = random.choice(list(matches.keys()))
13     return matches[cle]
```

ANNEXES

F) AUTRES

```
61 def liste_marge():
62     L = []
63     for match in matches.values():
64         L.append((1/match["Odds Player 1"]) + (1/match["Odds Player 2"]) - 1)
65     return L
```

ANNEXES

F) AUTRES

```
60 def liste_arbitrage():
61     arbitrage = []
62     for match in odds.values():
63         cote_joueur1 = []
64         cote_joueur2 = []
65         for site in match.values():
66             cote_joueur1.append(site["Odds Player 1"])
67             cote_joueur2.append(site["Odds Player 2"])
68         cote_max = (max(cote_joueur1), max(cote_joueur2))
69         arbitrage.append(1/cote_max[0] + 1/cote_max[1] - 1)
70     return arbitrage
```


ANNEXES

F) AUTRES

```
38 def test_coherence_estim_proba():
39     i = 0
40     total = 0
41     for match in matches.values():
42         total += 1
43         if estim_proba(match)[0] < estim_proba(match)[1] and match["Odds Player 1"] > match["Odds Player 2"]:
44             i += 1
45         elif estim_proba(match)[0] > estim_proba(match)[1] and match["Odds Player 1"] < match["Odds Player 2"]:
46             i += 1
47     return i/total
```