

확률 및 통계——

Blackjack

글로벌미디어학부

20172711

김민영



a presentation About

1
2
3
4
5

기본 게임 방식
무한대의 덱
하나의 덱
두 개의 덱
세 개의 덱

기본 방식

1. 플레이어 3명과 딜러 1명이 게임을 진행한다.
2. 모든 플레이어와 딜러는 카드 2장씩 지급, 그 후 딜러와 패를 비교해서 플레이어는 딜러보다 21에 가깝게 만들어야 한다. 더 필요하다고 생각된다면, 카드를 더 받을 수도 있지만, 카드를 한 장만 더 받도록 제한하기도 한다.
3. A는 기본적으로 11로 주어진다. 카드합이 21을 넘어갈 경우, A카드가 있다면, 하나의 A카드는 1이 된다. 카드 숫자 합에서 10을 빼주는 방식으로 A를 1로 만들어준다. (A카드가 두 장이어도 하나의 카드만 1로 바꿔준다.)
4. 처음 받은 2장을 합쳐 21이 나오면 A+K A+Q A+J A+10 4가지 경우 블랙잭, 블랙잭은 무조건 승리한다. 딜러와 플레이어가 동시에 블랙잭이라면, push 무승부)
5. 카드 숫자 합이 21을 초과하면 버스트, 딜러와 상관없이 플레이어 패배
6. 딜러는 카드숫자합계가 16 이하면 반드시 카드 한 장 추가로 받기 17이상이면 카드 받기 멈추기
7. A가 1이나 11이 되어 7이나 17이 되는 경우 소프트 17 (소프트 17 경우를 제한하기도 한다.)
8. 딜러가 21을 초과하고 플레이어가 21을 초과하지 않았다면, 플레이어 승리.
9. 플레이어가 21을 초과한다면 무조건 패배.

블랙잭 실험 목표

1. 플레이어의 승률을 높이는 방법을 알아본다.
2. 무한대의 덱, 하나의 덱, 두 개의 덱, 세 개의 덱 등을 비교해본다.
3. 플레이어가 받은 카드가 어떠할 때, 카드를 더 받으면 좋은지 알아본다.
4. 소프트17 규칙을 적용하는 것과 적용하지 않는 것을 비교해본다.
5. 몇 번의 블랙잭 게임을 실행했을 때, 승률이 수렴하는지 알아본다..



블랙잭 기본 코드

플레이어의 카드로 딜러를 이겼는지 판별해주는 함수

```
In [236]: def winorlose(card_arr):
    #딜러의 숫자
    deal = card_arr.pop() # 24
    #print("deal은 : ", deal)
    if deal > 21:
        print("모든 플레이어가 승리했습니다.")
        return [1, 1, 1]

    #WinOrLose
    wol = []
    for i in range(len(card_arr)):
        if card_arr[i] > 21:
            print(f"플레이어는 21이 초과하여 버스트, player{i} 패입니다.")
            wol.append(0)
        elif card_arr[i] == 21 and deal == 21:
            print(f"동시에 블랙잭이 되어 push하고, 무승부입니다.")
            wol.append(-1)
        elif card_arr[i] == 21 and deal < 21:
            print(f"플레이어가 블랙잭이 되어 player{i} 승입니다.")
            wol.append(1)
        elif card_arr[i] < 21 and deal == 21:
            print(f"플레이어는 딜러보다 값이 크지 않으므로 player{i} 패입니다.")
            wol.append(0)
        elif card_arr[i] < deal:
            print(f"딜러보다 값이 크지 않으므로 player{i} 패입니다.")
            wol.append(0)
        else: #card_arr[i] > deal 일 경우
            print(f"딜러보다 값이 크므로 player{i} 승입니다.")
            wol.append(1)
    return wol
```

블랙잭 기본 코드

플레이어의 카드 합을 계산하는 함수

```
In [246]: def sum_card(cards): #A는 일단 11로, JQK는 10으로 넣는다. 만약 카드들의 총합을 구했을 때, 21이 넘어가고 A카드가 한장 있다면 11이 10이 되므로  
#result에서 10을 빼주고, A가 두 장이라면, 11이 두 번 더 해졌으므로 20을 뺀다.  
#카드 경우의 수 : 'A', '2', '3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q', 'K'  
    result = 0  
  
    for i in range(len(cards)):  
        if cards[i] == 'A':  
            cards[i] = 11  
        elif cards[i] == 'J' or cards[i] == 'Q' or cards[i] == 'K':  
            cards[i] = 10  
        else:  
            cards[i] = int(cards[i])  
        result += cards[i]  
  
    a_cnt = cards.count(11)  
  
    #결과가 21보다 크고 A가 있다면, 개수에 따라 알맞게 1로 바꿔주기  
    if result > 21 and a_cnt == 1:  
        result -= 10  
  
    elif result > 21 and a_cnt != 0:  
        result -= (a_cnt-1) * 10  
  
    return result
```

블랙잭 기본 코드

```
: #카드 총합을 구하는 함수 #소프트 17 적용
def sum_card2(cards): #A는일단 11로, JQK는 10으로 넣는다. 만약 카드들의 총합을 구했을 때, 21이 넘어가고 A카드가 한장 있다면 11이 1이 되므로
    #result에서 10을 빼주고, A가 두 장이라면, 11이 두 번 더 해졌으므로 20을 뺀다.
    #카드 경우의 수 : 'A','2','3','4','5','6','7','8','9','10','J','Q','K'
    result = 0
    for i in range(len(cards)):
        if cards[i] == 'A':
            cards[i] = 11
        elif cards[i] == 'J' or cards[i] == 'Q' or cards[i] == 'K':
            cards[i] = 10
        else:
            cards[i] = int(cards[i])
        result += cards[i]

    a_cnt = cards.count(11)

    #결과가 21보다 크고 A가 0개가 아니라면 A하나를 1로 만들어준다.
    if result > 21 and a_cnt == 1:
        result -= 10

    elif result > 21 and a_cnt != 0:
        result -= (a_cnt-1) * 10

    #sum_card()에는 이 코드가 없다.
    #보통 딜러는 17이 되면, 룰에 따라 카드를 더 받지 않지만, A가 하나 있다면, 17이 7이 된다. 고로, 딜러는 카드를 더 받게 해야한다.
    #그렇지만, 플레이어는 이 소프트 17의 적용을 받지 않아야 한다. 고로 딜러의 카드 합을 계산할 때만 sum_card2() 함수를 쓴다.
    if result == 17 and a_cnt == 1: #11, 6인 경우
        result -= 10

    return result
```

블랙잭 기본 코드

각 플레이어들이 랜덤으로 카드를 받는 함수

```
In [277]: #처음 받은 카드 리스트를 받는 함수, 카드는 무한대데 #소프트17 적용
def receive_card2(num): #딜러포함 num명이 게임한다.
    make_card_deck = (string.ascii_uppercase + string.digits)
    card = list(set(make_card_deck+'10') - set('BCDEFGHILMNOPRSTUVWXYZ01')) * 4 #A,2,3,4,5,6,7,8,9,10,J,Q,K

    ccr = random.choices(card, k = num * 2 + 20) #20개 여유분까지 카드 뽑아놓기 #choices는 중복있게 여러개 뽑기
    print(ccr)
    #num-1명의 플레이어 리스트를 만든다. 나머지 하나는 딜러
    card_co = []
    j = 0
    for i in range(1, num):
        globals()['card_p'+str(i)] = ccr[j:j+2]
        j += 2
        card_co.append(globals()['card_p'+str(i)])

    deal = ccr[(num-1) * 2 : num * 2]
    print("dealer의 처음 카드 두 장은", deal)

    a_cnt = deal.count('A')
    print(a_cnt)

    #두 장씩 분배 후,
    a = 1
    while sum_card(deal) < 17: #기본 룰, 딜러는 카드 숫자 합이 17이하이면 더 받는다. #11,6
        deal = ccr[(num-1) * 2 : num * 2 + a]
        a += 1

    #어차피 소프트 17 규칙은 딜러에게 처음 한번만 적용되므로 if
    if sum_card(deal) == 17 and a_cnt == 1 and len(deal) == 2: #소프트 17 조건이 충족되면, # 11,6,10
        print(len(deal))
```

블랙잭 기본 코드

블랙잭 플레이 함수

```
In [263]: def blackjack_play2(): #소프트 17 적용
    #모든 플레이어들은 1000 coin을 가지고 시작한다.
    #베팅 금액은 계속 100 coin
    p1_account = 1000
    p2_account = 1000
    p3_account = 1000
    bet_amnt = betamount(100)

    #각 플레이어 계좌 잔고 리스트
    player_account = [p1_account, p2_account, p3_account]

    #플레이시작, 베팅
    p1_account = betting(p1_account, bet_amnt)
    p2_account = betting(p2_account, bet_amnt)
    p3_account = betting(p3_account, bet_amnt)

    #플레이어가 받은 카드
    play = receive_card2(4) #이 과정에서 카드 숫자 합을 구해서
    print(play)
    #플레이어 3명과 딜러 1명의 카드 묶음을 play에 담았다. ['2', '2', '2', '2', '9', '3', 'A', 'A', 'Q', '9', 'Q', '4', 'A', '6',
    cards = []

    for i, val in enumerate(play):
        if i == 3:
            val = sum_card(val)
            cards.append(val)
            print(i+1,"번째 카드 합은", val)
        else:
            val = sum_card(val)
            cards.append(val)
            print(i+1,"번째 카드 합은", val)
```

무한대의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

승률이 0.474~0.529 구간으로 나옵니다.
평균적으로 0.503이라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [139]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")
```

deal은 : 17
딜러보다 값이 크지 않으므로 player0 패입니다.
딜러보다 값이 크므로 player1 승입니다.
딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 1, 0]
['0', '6', '4', '7', 'A', 'J', 'J', '3', 'K', '4', '2',
'4', '7', 'K', 'J', 'A']
dealer의 처음 카드 두 장은 ['J', '3']
([10, 6], [4, 7, 11, 10], [11, 10], [10, 3, 10])
1 번째 카드 합은 16
2 번째 카드 합은 22
3 번째 카드 합은 21
4 번째 카드 합은 23
[16, 22, 21, 23]
deal은 : 23
모든 플레이어가 승리했습니다.
[1, 1, 1]
496
player의 승률은 0.496 입니다.

[0, 0, 0]
528
player의 승률은 0.528 입니다.
[0, 1, 0]
499
player의 승률은 0.499 입니다.
[0, 1, 0]
515
player의 승률은 0.515 입니다.
[1, 1, 1]
490
player의 승률은 0.49 입니다.
[1, 1, 1]
494
player의 승률은 0.494 입니다.
[1, 1, 1]
529
player의 승률은 0.529 입니다.
[0, 0, 0]
497
player의 승률은 0.497 입니다.
[-1, 0, 0]
485
player의 승률은 0.485 입니다.
[1, 0, 0]
503
player의 승률은 0.503 입니다.

미

무한대의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

승률이 0.534~0.569 구간으로 나옵니다.
평균적으로 0.551라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [207]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")
```

딜러보다 값이 크므로 player1 승입니다.
플레이어는 21이 초과하여 버스트, player2 패입니다.
[0, 1, 0]
['K', 'K', 'K', 'Q', '7', '2', 'K', 'A', '8', '5', '2', 'J', 'J', 'Q', '5', '8']
dealer의 처음 카드 두 장은 ['K', 'A']
([10, 10], [10, 10], [7, 2, 8], [10, 11])
1 번째 카드 합은 20
2 번째 카드 합은 20
3 번째 카드 합은 17
4 번째 카드 합은 21
[20, 20, 17, 21]
deal은 : 21
플레이어는 딜러보다 값이 크지 않으므로 player0 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
548
player의 승률은 0.548 입니다.

[0, 1, 0]	[0, 0, 0]
567	538
player의 승률은 0.567 입니다.	player의 승률은 0.538 입니다.
[0, 1, 1]	[1, 0, 0]
554	559
player의 승률은 0.554 입니다.	player의 승률은 0.559 입니다.
[1, 1, 0]	
569	
player의 승률은 0.569 입니다.	
[1, 1, 1]	
544	
player의 승률은 0.544 입니다.	
[0, 1, 0]	
534	
player의 승률은 0.534 입니다.	
[0, 0, 1]	
566	
player의 승률은 0.566 입니다.	
[0, 0, 0]	
539	
player의 승률은 0.539 입니다.	

무한대의 덱_카드 여러 장 더 받기

- 소프트 17 규칙을 적용 x
- 딜러와 플레이어 모두 카드를 여러 장 더 받는다.
- 플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다. 평균적으로 0.585이라는 승률이 나옵니다.
- 블랙잭을 1000번 시행했을 경우,

```
In [219]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

플레이어는 21이 초과하여 버스트, player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 0, 1]
['3', 'K', '6', '7', 'Q', '8', 'A', 'K', 'Q', '3', '8',
'8', '2', '5', '5', '8']
dealer의 처음 카드 두 장은 ['A', 'K']
([3, 10, 8], [6, 7, 5], [10, 8], [11, 10])
1 번째 카드 합은 21
2 번째 카드 합은 18
3 번째 카드 합은 18
4 번째 카드 합은 21
[21, 18, 18, 21]
deal은 : 21
동시에 블랙잭이 되어 push이고, 무승부입니다.
플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[-1, 0, 0]
593
player의 승률은 0.593 입니다.
```

승률이 0.564~0.612 구간으로 나옵니다.
승률이 더 높아지기도 하지만, 카드 숫자 합이 16일 때 카드를 더 받는 경우보다 범위가 넓은 것으로 보아 도박성이 더 강해진다고 볼 수 있습니다.
평균적으로 0.585이라는 승률이 나옵니다.

[1, 0, 0]	[1, 0, 1]
570	564
player의 승률은 0.57 입니다.	player의 승률은 0.564 입니다.
[0, 1, 0]	[1, 1, 1]
612	583
player의 승률은 0.612 입니다.	player의 승률은 0.583 입니다.
[0, 0, 0]	[1, 1, 1]
574	586
player의 승률은 0.574 입니다.	player의 승률은 0.586 입니다.
[1, 1, 1]	[1, 1, 1]
601	578
player의 승률은 0.601 입니다.	player의 승률은 0.578 입니다.
[0, 0, 0]	[1, 1, 1]
578	596
player의 승률은 0.578 입니다.	player의 승률은 0.596 입니다.

무한대의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

승률이 0.454~0.511 구간으로 나옵니다.
평균적으로 0.493라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [260]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

[4, 'Q', '2', '2', '7', '2', '9', '3', '9', 'K', '9', '7', '3', '7', 'K', '3', '2', 'K', '7', '7', '7', '7', '0', '9', 'Q', '5']
dealer의 처음 카드 두 장은 ['9', '3']
0
기본 룰이므로 더 받음
([4, 10], [2, 2, 5, 10], [7, 2, 10], [9, 3, 9])
1 번째 카드 합은 14
2 번째 카드 합은 19
3 번째 카드 합은 19
4 번째 카드 합은 21
[14, 19, 19, 21]
deal은 : 21
플레이어는 딜러보다 값이 크지 않으므로 player0 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
506
player의 승률은 0.506 입니다.
```

[0, 0, 0]	[0, 0, 0]
511	499
player의 승률은 0.511 입니다.	player의 승률은 0.499 입니다.
[1, 1, 0]	[0, 1, 1]
505	483
player의 승률은 0.505 입니다.	player의 승률은 0.483 입니다.
[0, 0, 0]	[0, 0, 0]
468	496
player의 승률은 0.468 입니다.	player의 승률은 0.496 입니다.
[0, 0, 0]	[0, 0, 0]
502	502
player의 승률은 0.502 입니다.	player의 승률은 0.502 입니다.
[0, 0, 0]	[0, 0, 0]
454	510
player의 승률은 0.454 입니다.	player의 승률은 0.51 입니다.

무한대의 덱_카드 여러 장 더 받기

- 소프트 17 규칙을 적용 o
- 딜러와 플레이어 모두 카드를 여러 장 더 받는다.
- 플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.
- 블랙잭을 1000번 시행했을 경우,

```
In [278]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

[1, 1, 1]
528
player의 승률은 0.528 입니다.

[1, 0, 0]
538
player의 승률은 0.538 입니다.

[0, 0, 0]
564
player의 승률은 0.564 입니다.

[1, 1, 1]
542
player의 승률은 0.542 입니다.

[0, 0, 1]
544
player의 승률은 0.544 입니다.

[1, 0, 1]
531
player의 승률은 0.531 입니다.

[1, 0, 0]
550
player의 승률은 0.55 입니다.

딜러보다 값이 크므로 player0 승입니다.
딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크지 않으므로 player2 패입니다.
```

승률이 0.538~0.564 구간으로 나옵니다.
운에 따라 0.538이나 0.585의 승률을 가질 수 있지만,
평균적으로 0.541라는 승률이 나옵니다.

```
[1, 0, 0]
545
player의 승률은 0.545 입니다.

[0, 0, 0]
545
player의 승률은 0.545 입니다.

[1, 1, 1]
542
player의 승률은 0.542 입니다.

[0, 0, 1]
544
player의 승률은 0.544 입니다.

[1, 0, 1]
531
player의 승률은 0.531 입니다.

[1, 0, 0]
523
player의 승률은 0.523 입니다.
```

무한대의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

승률이 0.543~0.602 구간으로 나옵니다.
평균적으로는 0.570의 승률을 보입니다.

-플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [168]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

플레이어는 21이 초과하여 버스트, player2 패입니다.
[1, 1, 0]
['8', '5', 'Q', '2', '8', '6', 'K', '4', 'K', 'K', '2', '7'
'2', '7', '5', '6', '5', '2', '4', '5', '4', 'J', 'K', '7
'7', '7']
dealer의 처음 카드 두 장은 ['K', '4']
0
기본 룰이므로 더 받음
([8, 5, 7], [10, 2, 7], [8, 6, 2, 7], [10, 4, 10])
1 번째 카드 합은 20
2 번째 카드 합은 19
3 번째 카드 합은 23
4 번째 카드 합은 24
[20, 19, 23, 24]
deal은 : 24
모든 플레이어가 승리했습니다.
[1, 1, 1]
561
player의 승률은 0.561 입니다.
```

[1, 1, 1]	585	player의 승률은 0.585 입니다.
[1, 0, 0]	562	player의 승률은 0.562 입니다.
[1, 1, 1]	576	player의 승률은 0.576 입니다.
[0, 0, 1]	579	player의 승률은 0.579 입니다.
[0, 1, 0]	602	player의 승률은 0.602 입니다.
[1, 0, 0]	571	player의 승률은 0.571 입니다.
[0, 0, 0]	561	player의 승률은 0.561 입니다.

무한대의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 한 장만 더 받는다.

승률이 0.484~0.551 구간으로 나옵니다.
평균적으로 0.518이라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [56]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")
```

딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
['6', 'A', '7', '0', '4', '4', '2', 'K', '8', '5', '4',
'4', '7', '5', '5', '5']
dealer의 처음 카드 두 장은 ['2', 'K']
([6, 11], [7, 10], [4, 4, '5'], ['2', 'K', '8'])
1 번째 카드 합은 17
2 번째 카드 합은 17
3 번째 카드 합은 13
4 번째 카드 합은 20
[17, 17, 13, 20]
deal은 : 20
딜러보다 값이 크지 않으므로 player0 패입니다.
딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
520
player의 승률은 0.52 입니다.

[0, 0, 0]

503

player의 승률은 0.503 입니다.

[1, 1, 1]

551

player의 승률은 0.551 입니다.

[1, 1, 1]

484

player의 승률은 0.484 입니다.

[1, 1, 1]

515

player의 승률은 0.515 입니다.

[0, 0, 0]

503

player의 승률은 0.503 입니다.

[1, 0, 0]

527

player의 승률은 0.527 입니다.

[0, 0, 0]

536

player의 승률은 0.536 입니다.

무한대의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 한 장만 더 받는다.

승률이 0.51~0.567 구간으로 나옵니다.
평균적으로 0.541이라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [71]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

플레이어는 21이 초과하여 버스트, player1 패입니다.
동시에 블랙잭이 되어 push이고, 무승부입니다.
[-1, 0, -1]
['8', 'A', 'K', '4', 'K', '7', '2', 'Q', '5', '7', '5'
'K', '4', '5', '9', 'J']
dealer의 처음 카드 두 장은 ['2', 'Q']
([8, 11], [10, 4, 'J'], [10, 7], [2, 'Q', '5'])
1 번째 카드 합은 19
2 번째 카드 합은 24
3 번째 카드 합은 17
4 번째 카드 합은 17
[19, 24, 17, 17]
deal은 : 17
딜러보다 값이 크므로 player0 승입니다.
플레이어는 21이 초과하여 버스트, player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 0, 1]
527
player의 승률은 0.527 입니다.
```

[1, 1, 1]
537
player의 승률은 0.537 입니다.

[1, 1, 1]
556
player의 승률은 0.556 입니다.

[1, 1, 0]
510
player의 승률은 0.51 입니다.

[0, 1, 0]
524
player의 승률은 0.524 입니다.

[1, 1, 1]
541
player의 승률은 0.541 입니다.

[0, 0, 0]
554
player의 승률은 0.554 입니다.

[0, 0, 0]
567
player의 승률은 0.567 입니다.

[1, 0, 0]
554
player의 승률은 0.554 입니다.

[0, -1, 0]
548
player의 승률은 0.548 입니다.

[0, 1, 0]
524
player의 승률은 0.524 입니다.

무한대의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 x

승률이 0.504~0.561 구간으로 나옵니다.

-딜러와 플레이어 모두 카드를 한 장만 더 받는다.

평균적으로 0.533이라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [88]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

4 번째 카드 합은 25
[21, 24, 9, 25]
deal은 : 25
모든 플레이어가 승리했습니다.
[1, 1, 1]
['5', 'J', 'J', 'K', 'A', '5', 'Q', '4', '8', '3', ...
'8', '6', '0', '8', 'J']
dealer의 처음 카드 두 장은 ['Q', '4']
([5, 10, 'J'], [10, 10], [11, 5, '8'], ['Q', '4', ...
1 번째 카드 합은 25
2 번째 카드 합은 20
3 번째 카드 합은 14
4 번째 카드 합은 22
[25, 20, 14, 22]
deal은 : 22
모든 플레이어가 승리했습니다.
[1, 1, 1]
518
player의 승률은 0.518 입니다.
```

[1, 1, 0]

517

player의 승률은 0.517 입니다.

[1, 0, 0]

504

player의 승률은 0.504 입니다.

[0, 1, 1]

546

player의 승률은 0.546 입니다.

[1, 0, 0]

542

player의 승률은 0.542 입니다.

[1, 1, 1]

526

player의 승률은 0.526 입니다.

[0, 0, 0]

532

player의 승률은 0.532 입니다.

[0, 0, 0]

554

player의 승률은 0.554 입니다.

[0, 1, 1]

561

player의 승률은 0.561 입니다.

[0, 1, 1]

534

player의 승률은 0.534 입니다.

무한대의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 한 장만 더 받는다.

승률이 0.494~0.545 구간으로 나옵니다.
평균적으로 0.516라는 승률이 나옵니다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [27]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")
[1, 1, 1]
499
player의 승률은 0.499 입니다.
['A', '8', '7', '3', 'K', 'Q', '4', '4', 'J', 'A', '9', '8',
 '2', '5', 'A', 'Q', '8', 'A', '6', '6', '5', '7', '8', 'A',
 '7', 'A']
dealer의 처음 카드 두 장은 ['4', '4']
0
기본 룰이므로 더 받음
([11, 8], [7, 3, 'A'], [10, 10], [4, 4, 10])
1 번째 카드 합은 19
2 번째 카드 합은 21
3 번째 카드 합은 20
4 번째 카드 합은 18
[19, 21, 20, 18]
deal은 : 18
딜러보다 값이 크므로 player0 승입니다.
플레이어가 블랙잭이 되어 player1 승입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 1, 1]
494
player의 승률은 0.494 입니다.
```

[1, 1, 1]	[1, 0, 1]
499	505
player의 승률은 0.499 입니다.	player의 승률은 0.505 입니다.
[1, 1, 0]	[1, 1, 1]
512	513
player의 승률은 0.512 입니다.	player의 승률은 0.513 입니다.
[0, 0, 0]	[1, 1, 1]
520	513
player의 승률은 0.52 입니다.	player의 승률은 0.513 입니다.
[0, 1, 0]	[1, 1, 1]
521	513
player의 승률은 0.521 입니다.	player의 승률은 0.513 입니다.
[0, 1, 0]	[1, 1, 1]
524	513
player의 승률은 0.524 입니다.	player의 승률은 0.513 입니다.
[0, 0, 1]	[1, 1, 1]
535	513
player의 승률은 0.535 입니다.	player의 승률은 0.513 입니다.
[0, 0, 1]	[1, 1, 1]
545	513
player의 승률은 0.545 입니다.	player의 승률은 0.513 입니다.

무한대의 덱_카드 한 장만 더 받기

- 소프트 17 규칙을 적용 o
- 딜러와 플레이어 모두 카드를 한 장만 더 받는다.
- 플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.
- 블랙잭을 1000번 시행했을 경우,

```
In [45]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

['A', '2', '4', '8', 'K', 'K', '5', '7', '4', '5'
'5', 'Q', 'Q', '8', 'Q', '8', '6', '2', '3', 'J'
'7', '4']
dealer의 처음 카드 두 장은 ['5', '7']
0
기본 룰이므로 더 받음
([11, 2, '4'], [4, 8, '7'], [10, 10], [5, 7, 4])
1 번째 카드 합은 17
2 번째 카드 합은 19
3 번째 카드 합은 20
4 번째 카드 합은 16
[17, 19, 20, 16]
deal은 : 16
딜러보다 값이 크므로 player0 승입니다.
딜러보다 값이 크므로 player1 승입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 1, 1]
519
player의 승률은 0.519 입니다.
```

승률이 0.511~0.591 구간으로 나옵니다. 승률이 쪽에 치우쳐 있지 않고, 골고루 나타나는 것으로 운이 강하게 작용하고 있고, 도박성이 큰 것이라고 생각합니다. 운에 따라 0.561이나 0.529의 승률 가질 수 있지만, 평균적으로 0.540라는 승률이 나옵니다.

[0, 0, 0]
544
player의 승률은 0.544 입니다.

[1, 1, 1]
553
player의 승률은 0.553 입니다.

[0, 1, 1]
555
player의 승률은 0.555 입니다.

[0, -1, 0]
591
player의 승률은 0.591 입니다.

[1, 1, 1]
511
player의 승률은 0.511 입니다.

[1, 1, 1]
521
player의 승률은 0.521 입니다.

[0, 0, 1]
542
player의 승률은 0.542 입니다.

[0, 0, 1]
541
player의 승률은 0.541 입니다.

[0, 0, 0]
529
player의 승률은 0.529 입니다.

무한대의 덱_카드 한 장만 더 받기

- 소프트 17 규칙을 적용 o
- 딜러와 플레이어 모두 카드를 한 장만 더 받는다.
- 플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.
- 블랙잭을 1000번 시행했을 경우,

```
In [56]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

['2', '5', 'J', '3', '9', '4', '6', '8', '3', '8', '2',
'9', 'J', 'A', '7', '2', '5', 'A', '5', '8', '2', 'A',
'2', 'A']
dealer의 처음 카드 두 장은 ['6', '8']
0
기본 룰이므로 더 받음
([2, 5, 'A'], [10, 3, '2'], [9, 4, '7'], [6, 8, 3])
1 번째 카드 합은 18
2 번째 카드 합은 15
3 번째 카드 합은 20
4 번째 카드 합은 17
[18, 15, 20, 17]
deal은 : 17
딜러보다 값이 크므로 player0 승입니다.
딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 0, 1]
557
player의 승률은 0.557 입니다.
```

승률이 0.505~0.562 구간으로 나옵니다.
평균적으로는 0.539의 승률을 보입니다.
기준점 18인 경우에도 소프트17

[1, 1, 1]	[0, 1, 1]
562	505
player의 승률은 0.562 입니다.	player의 승률은 0.505 입니다.
[1, 0, 1]	[1, 1, 1]
558	536
player의 승률은 0.558 입니다.	player의 승률은 0.536 입니다.
[1, 0, 0]	[1, 0, 1]
516	516
player의 승률은 0.516 입니다.	player의 승률은 0.516 입니다.
[1, 1, 1]	[1, 1, 1]
545	545
player의 승률은 0.545 입니다.	player의 승률은 0.545 입니다.
[0, 1, 1]	[0, 1, 1]
531	531
player의 승률은 0.531 입니다.	player의 승률은 0.531 입니다.
[0, 0, 0]	[0, 0, 0]
538	538
player의 승률은 0.538 입니다.	player의 승률은 0.538 입니다.
[1, 1, 1]	[1, 1, 1]
548	548
player의 승률은 0.548 입니다.	player의 승률은 0.548 입니다.

무한대덱에서의 실험 결과

무한대의 덱에서 카드를 여러 번 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때, **기준점이 커질수록 승률이 오르는 것을 알 수 있습니다.**

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **소프트 17을 적용했을 경우, 승률이 떨어지는 경향을 보입니다.**

무한대의 덱에서 카드를 한 장만 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때, **14일 때와 16일 때는, 카드를 여러번 받을 때와 같이 기준점이 커질수록 승률이 오르는 경향을 보이지만, 16과 18을 비교했을 때는 앞과 같은 경향을 보이지 않습니다.** 딜러가 한 장의 카드만 추가로 받게 되기 때문에 딜러의 버스트 가능성이 줄어들기 때문에 이러한 결과가 나온 것이라 생각합니다. 14와 16의 경우, 플레이어가 버스트가 될 가능성이 현저히 줄기 때문에 딜러의 버스트 가능성이 줄어들었다 해도 큰 영향을 받지 않고 경향성을 따라가는 것으로 보입니다.

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, 뚜렷한 비교 기준이 보이지 않습니다. **소프트 17 규칙은 카드를 여러 장 받아야, 적용하지 않았을 때보다 승률이 높아지는 것이라 추측해봅니다.**

무한대덱에서의 실험 결과

카드 여러 장 뽑기		
	소프트 17 적용 x	소프트 17 적용 o
14	0.503	0.493
16	0.551	0.541
18	0.585	0.570

카드 한 장만 뽑기		
	소프트 17 적용 x	소프트 17 적용 o
14	0.518	0.516
16	0.541	0.540
18	0.533	0.539

- ⇒ 카드를 여러 장 받을 수 있는 경우에는, 플레이어가 두 장의 카드 숫자 합이 18보다 작을 때 카드를 추가로 더 받고, 소프트17 규칙을 적용하지 않는다면, 승률을 높일 수 있다.
- ⇒ 카드를 추가로 한 장 받는 것보다는 여러 장 받을 수 있는 경우가 플레이어에게 유리하다.

무한대덱에서의 실험 결과

```
In [44]: win_ra = []
win_ra_ap = win_ra.append

In [45]: p1_win_cnt = 0
i = 0
while(i < 100):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/100, "입니다.")
win_ra_ap(p1_win_cnt/100)

In [46]: p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")
win_ra_ap(p1_win_cnt/1000)

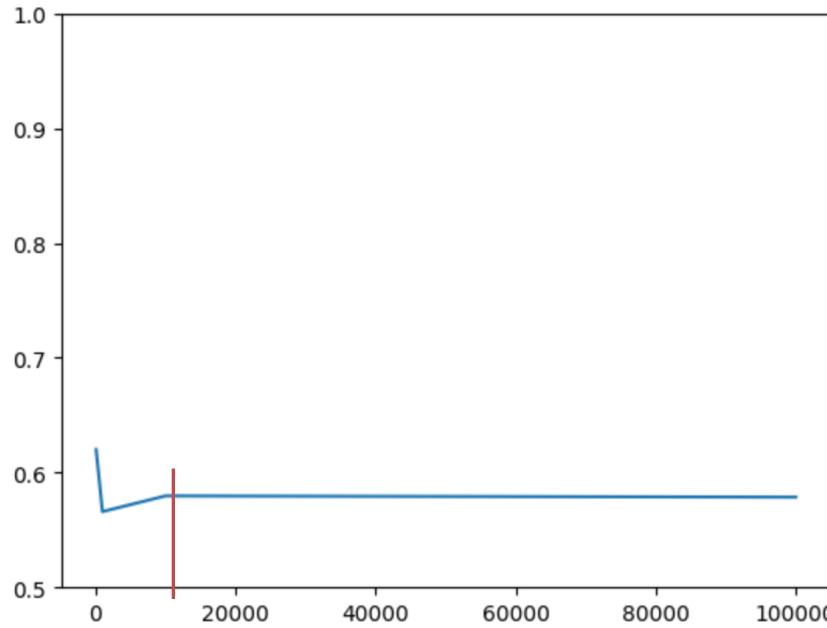
In [47]: p1_win_cnt = 0
i = 0
while(i < 10000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/10000, "입니다.")
win_ra_ap(p1_win_cnt/10000)

In [48]: p1_win_cnt = 0
i = 0
while(i < 100000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/100000, "입니다.")
win_ra_ap(p1_win_cnt/100000)
```

블랙잭을 100번, 1000번, 10000번, 100000번 실행해본 결과,
10000과 100000의 승률이 비슷해서 아래의 그래프가 10000 부터 100000까지
일자로 이어지고 있습니다.
우연히 10000번 시행했을 때와 100000번 시행 했을 때가 승률이 비슷해서 아래와
같은 그래프가 나온 것이고, 100000번의 시행에서 승률이 수렴하는지 아닌지는
아직 알 수 없습니다

```
In [50]: plt.plot([10**x for x in range(2, 6)], win_ra)
plt.ylim([0.5, 1])
```

Out[50]: (0.5, 1.0)

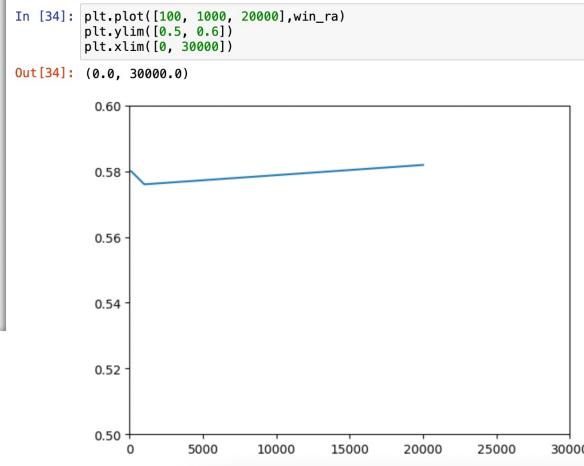


무한대덱에서의 실험 결과

```
In [12]: #전체 블랙잭 플레이 횟수는 100000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 100000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/100000, "입니다.")

플레이어는 21이 초과하여 버스트, player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
['3', 'A', '4', 'A', '6', 'A', 'A', '8', '6', '8', '6', '6', '6', '6', '8', '9', '5', '7', '4']
dealer의 처음 카드 두 장은 ['A', '8']
([3, 11, 4], [4, 11, 7, 5, 9], [6, 11, 8, 6], [11, 8])
1 번째 카드 합은 18
2 번째 카드 합은 26
3 번째 카드 합은 21
4 번째 카드 합은 19
[18, 26, 21, 19]
deal은 : 19
딜러보다 값이 크지 않으므로 player0 패입니다.
플레이어는 21이 초과하여 버스트, player1 패입니다.
플레이어가 블랙잭이 되어 player2 승입니다.
[0, 0, 1]
58040
player의 승률은 0.5804 입니다.
```

따라서 몇 번의 시행으로 승률이 수렴하는지 알아보기 위해서 일단 10000번에서 100000번까지 10000 step으로 블랙잭을 시행해보았습니다.



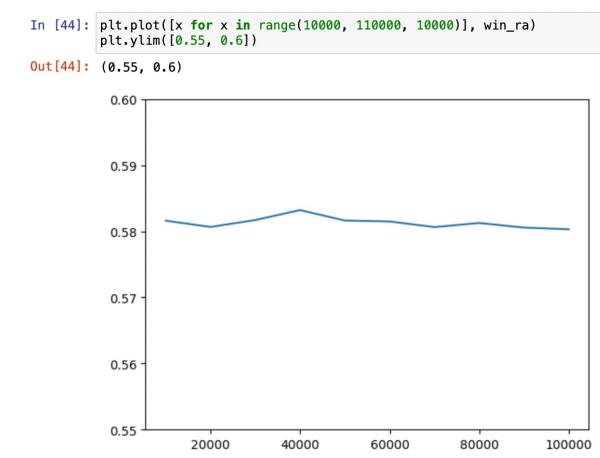
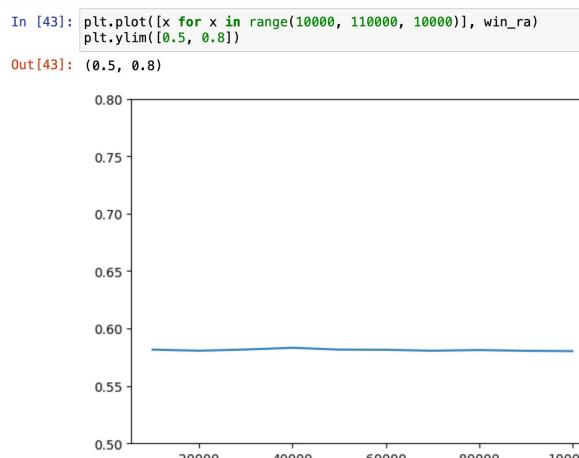
무한대덱에서의 실험 결과

```
In [40]: p1_win_cnt = 0
i = 0
for j in range(10000, 110000, 10000):# j는 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000
    while(i < j): #i=100, j=100 //i=100, j =200
        p1_result = blackjack_play()
        if p1_result == 1:
            p1_win_cnt += 1
        i += 1
    print(p1_win_cnt, j)
    print(i, "player의 승률은", p1_win_cnt / j, "입니다.")
    win_ra.append(p1_win_cnt / j)
    i += 1

win_lst = win_ra[30:]
print(win_lst)
```

```
플레이어는 21이 초과하여 버스트, player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 0, 1]
['1', '0', 'A', 'Q', 'Q', '8', '8', 'Q', 'Q', '6', '7', 'J', '6', '5']
dealer의 처음 카드 두 장은 ['8', 'Q']
([9, 8, 6], [11, 10], [10, 8], [8, 10])
1 번째 카드 합은 23
2 번째 카드 합은 21
3 번째 카드 합은 18
4 번째 카드 합은 18
[23, 21, 18, 18]
deal은 : 18
플레이어는 21이 초과하여 버스트, player0 패입니다.
```

따라서 몇 번의 시행으로 승률이 수렴하는지 알아보기 위해서
일단 10000번에서 100000번까지 10000 step으로
블랙잭을 시행해보았습니다.



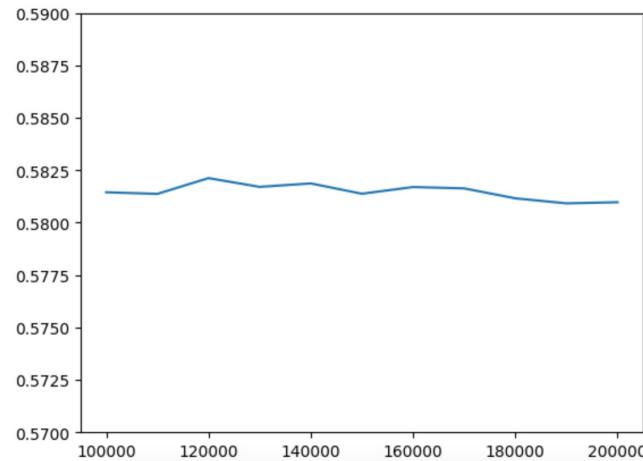
=> 그렇지만, 아직 수렴하는지 확실해보이지 않으므로 더 실험을 해보았습니다.

무한대덱에서의 실험 결과

몇 번의 시행으로 승률이 수렴하는지 알아보기 위해서 일단 10000번에서 200000번까지 10000 step으로 블랙잭을 시행해보았습니다.

```
In [49]: plt.plot([x for x in range(100000, 210000, 10000)], win_ra[10:])
plt.ylim([0.57, 0.59])
```

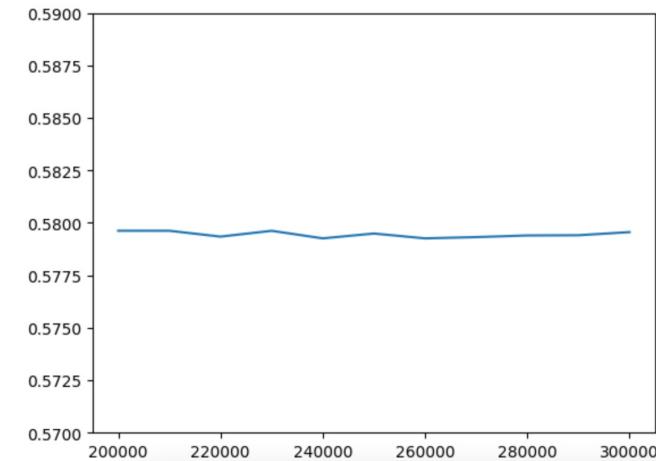
```
Out[49]: (0.57, 0.59)
```



몇 번의 시행으로 승률이 수렴하는지 알아보기 위해서 일단 10000번에서 300000번까지 10000 step으로 블랙잭을 시행해보았습니다.

```
In [53]: plt.plot([x for x in range(200000, 310000, 10000)], win_ra[21:])
plt.ylim([0.57, 0.59])
```

```
Out[53]: (0.57, 0.59)
```



=> 시행횟수를 늘릴수록 승률이 0.58에 수렴하는 듯 보이지만, 정확한 시점을 알고 싶어 시행횟수를 600000번까지 늘려서 진행해보려 했으나 주피터가 돌아가지 않아 더 이상 실험해볼 수 없었습니다.

하나의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [19]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.

[0, 0, 1]
['9', 'A', '5', 'J', '5', '4', '2', '9', 'J', '8',
'2', '8', '7', 'A', 'K']
dealer의 처음 카드 두 장은 ['2', '9']
([9, 11], [5, 10], [5, 4, 10], [2, 9, 10])
1 번째 카드 합은 20
2 번째 카드 합은 15
3 번째 카드 합은 19
4 번째 카드 합은 21
[20, 15, 19, 21]
deal은 : 21
플레이어는 딜러보다 값이 크지 않으므로 player0 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
522
player의 승률은 0.522 입니다.
```

승률이 0.484~0.522 구간으로 나옵니다.
평균적으로는 0.500의 승률을 보입니다.

[0, 0, 0]	[1, 1, 1]
511	501
player의 승률은 0.511 입니다.	player의 승률은 0.501 입니다.
[1, 1, 1]	[0, 1, 0]
489	516
player의 승률은 0.489 입니다.	player의 승률은 0.516 입니다.
[1, 0, 0]	
503	
player의 승률은 0.503 입니다.	
[0, 0, 0]	
484	
player의 승률은 0.484 입니다.	
[1, 1, 1]	
487	
player의 승률은 0.487 입니다.	
[1, 1, 1]	
485	
player의 승률은 0.485 입니다.	
[1, 1, 0]	
502	
player의 승률은 0.502 입니다.	

하나의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

-플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [46]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

deal은 : 25
모든 플레이어가 승리했습니다.
[1, 1, 1]
['Q', '5', '4', '7', '7', 'Q', 'Q', '7', '8', '2', '4',
 'J', '2', '3', '6', '3']
dealer의 처음 카드 두 장은 ['Q', '7']
([10, 5, 3], [4, 7, 6], [7, 10], [10, 7])
1 번째 카드 합은 18
2 번째 카드 합은 17
3 번째 카드 합은 17
4 번째 카드 합은 17
[18, 17, 17, 17]
deal은 : 17
딜러보다 값이 크므로 player0 승입니다.
딜러보다 값이 크므로 player1 승입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 1, 1]
565
player의 승률은 0.565 입니다.
```

승률이 0.529~0.58구간으로 나옵니다.
평균적으로는 0.562의 승률을 보입니다.

```
[1, 0, 1]      [0, 0, 1]
546           529
player의 승률은 0.546 입니다. player의 승률은 0.529 입니다.
[0, 0, 1]      [0, 1, 1]
551           564
player의 승률은 0.551 입니다 player의 승률은 0.564 입니다.
[1, 0, 0]      [0, 0, 1]
580           529
player의 승률은 0.58 입니다.
[0, 0, 1]      [0, 1, 1]
558           558
player의 승률은 0.558 입니다.
[1, 1, 1]      [0, 0, 1]
574           558
player의 승률은 0.574 입니다.
[1, 1, 0]      [0, 0, 1]
554           560
player의 승률은 0.554 입니다.
[0, 0, 0]      [0, 0, 1]
560           560
player의 승률은 0.56 입니다.
```

하나의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

-플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [21]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

deal은 : 23
모든 플레이어가 승리했습니다.
[1, 1, 1]
['3', '6', '3', '8', '4', '5', '4', 'A', '2', '2', '9', 'J', 'Q', 'K', '8', 'A']
dealer의 처음 카드 두 장은 ['4', 'A']
([3, 6, 11], [3, 8, 8], [4, 5, 10], [4, 11, 2])
1 번째 카드 합은 20
2 번째 카드 합은 19
3 번째 카드 합은 19
4 번째 카드 합은 17
[20, 19, 19, 17]
deal은 : 17
딜러보다 값이 크므로 player0 승입니다.
딜러보다 값이 크므로 player1 승입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 1, 1]
574
player의 승률은 0.574 입니다.
```

승률이 0.568~0.597 구간으로 나옵니다.
평균적으로는 0.578의 승률을 보입니다.

```
[0, 1, 0]          [1, 1, 1]
572               595
player의 승률은 0.572 입니다. player의 승률은 0.595 입니다.

[1, 1, 0]          [1, 1, 1]
570               595
player의 승률은 0.57 입니다. player의 승률은 0.592 입니다.

[0, 1, 0]          [1, 1, 1]
592               559
player의 승률은 0.592 입니다. player의 승률은 0.559 입니다.

[1, 1, 1]          [1, 1, 1]
559               568
player의 승률은 0.559 입니다. player의 승률은 0.568 입니다.

[0, 0, 0]          [1, 1, 1]
569               589
player의 승률은 0.569 입니다. player의 승률은 0.589 입니다.
```

하나의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [11]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

#는 플레이어가 승리했습니다.
[1, 1, 1]
['J', '5', '6', '9', '2', '4', '9', '7', '7', '7', '2'
'2', 'J', 'K', 'A', '4', 'Q', 'Q', '4', '9', '4', '5'
'Q', '8']
dealer의 처음 카드 두 장은 ['9', '7']
0
기본 룰이므로 더 받음
([10, 5], [6, 9], [2, 4, 8], [9, 7, 7])
1 번째 카드 합은 15
2 번째 카드 합은 15

3 번째 카드 합은 14
4 번째 카드 합은 23
[15, 15, 14, 23]
deal은 : 23
모든 플레이어가 승리했습니다.
[1, 1, 1]
488
player의 승률은 0.488 입니다.
```

승률이 0.485~0.522 구간으로 나옵니다.
평균적으로는 0.499의 승률을 보입니다.

[0, 0, 0]	[0, 0, 0]
522	508
player의 승률은 0.522 입니다.	player의 승률은 0.508 입니다.
[1, 1, 1]	[0, 0, 1]
500	485
player의 승률은 0.5 입니다.	player의 승률은 0.485 입니다.
[1, 0, 0]	
509	
player의 승률은 0.509 입니다.	
[0, 0, 0]	
490	
player의 승률은 0.49 입니다.	
[0, 1, 0]	
486	
player의 승률은 0.486 입니다.	
[1, 0, 0]	
491	
player의 승률은 0.491 입니다.	
[0, 1, 1]	
514	
player의 승률은 0.514 입니다.	

하나의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

-플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [22]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

['9', '5', 'Q', 'K', 'Q', '9', '5', '7', '9', '4', '2', 'J',
 'J', '5', '3', '4', '4', '8', '7', 'Q', 'A', '2', 'A', '7',
 '6', 'A']
dealer의 처음 카드 두 장은 ['5', '7']
0
기본 룰이므로 더 받음
([9, 5, 11, 6], [10, 10], [10, 9], [5, 7, 9])
1 번째 카드 합은 21
2 번째 카드 합은 20
3 번째 카드 합은 19
4 번째 카드 합은 21
[21, 20, 19, 21]
deal은 : 21
동시에 블랙잭이 되어 push이고, 무승부입니다.
플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[-1, 0, 0]
530
player의 승률은 0.53 입니다.
```

승률이 0.522~0.576 구간으로 나옵니다.
평균적으로는 0.544의 승률을 보입니다.

[0, 0, 0]	[0, 1, 1]
556	536
player의 승률은 0.556 입니다.	player의 승률은 0.536 입니다.
[1, 1, 1]	[1, 0, 0]
555	530
player의 승률은 0.555 입니다.	player의 승률은 0.53 입니다.
[1, 1, 1]	[1, 0, 0]
563	530
player의 승률은 0.563 입니다.	player의 승률은 0.53 입니다.
[0, 0, 1]	[0, 1, 1]
543	522
player의 승률은 0.543 입니다.	player의 승률은 0.522 입니다.
[1, 1, 0]	[0, 1, 1]
576	531
player의 승률은 0.576 입니다.	player의 승률은 0.531 입니다.

하나의 덱_카드 여러 장 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 여러 장 더 받는다.

-플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [33]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

[-1, 0, 0]
['8', '5', 'K', 'A', '2', 'A', 'J', 'Q', 'J', '4', '7',
'K', '8', '5', '2', '3', '3', '7', '9', 'K', '6', '9',
'9', '7']
dealer의 처음 카드 두 장은 ['J', 'Q']
0
([8, 5, 7], [10, 11], [2, 11, 9, 9], [10, 10])
1 번째 카드 합은 20
2 번째 카드 합은 21
3 번째 카드 합은 21
4 번째 카드 합은 20
[20, 21, 21, 20]
deal은 : 20
딜러보다 값이 크므로 player0 승입니다.
플레이어가 블랙잭이 되어 player1 승입니다.
플레이어가 블랙잭이 되어 player2 승입니다.
[1, 1, 1]
585
player의 승률은 0.585 입니다.
```

승률이 0.563~0.591 구간으로 나옵니다.
평균적으로는 0.577의 승률을 보입니다.

```
[1, 1, 1]
591
player의 승률은 0.591 입니다. [1, 1, 1]
563
player의 승률은 0.563 입니다.

[1, 1, 0]
577
player의 승률은 0.577 입니다.

[0, 0, 1]
586
player의 승률은 0.586 입니다.

[0, 0, 1]
563
player의 승률은 0.563 입니다.

[0, 0, 0]
591
player의 승률은 0.591 입니다.

[0, 0, 0]
573
player의 승률은 0.573 입니다.

[1, 1, 1]
578
player의 승률은 0.578 입니다.

[0, 1, 1]
564
player의 승률은 0.564 입니다.
```

하나의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 한 장 더 받는다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우

```
In [10]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
['3', '7', '5', '6', '2', 'A', '0', '3', '4', '3',
'A', 'J', '6', '4', 'A']
dealer의 처음 카드 두 장은 ['Q', '3']
([3, 7, 'A'], [5, 6, '4'], [2, 11, '6'], ['Q', '3'])
1 번째 카드 합은 21
2 번째 카드 합은 15
3 번째 카드 합은 19
4 번째 카드 합은 17
[21, 15, 19, 17]
deal은 : 17
플레이어가 블랙잭이 되어 player0 승입니다.
딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 0, 1]
509
player의 승률은 0.509 입니다.
```

승률이 0.481~0.538 구간으로 나옵니다.
평균적으로는 0.509의 승률을 보입니다.

```
[0, 1, 1]
538
player의 승률은 0.538 입니다.
[1, 0, 1]
495
player의 승률은 0.495 입니다.

[0, 0, 0]
519
player의 승률은 0.519 입니다.
[0, 0, 0]
499
player의 승률은 0.499 입니다.

[1, 1, 1]
519
player의 승률은 0.519 입니다.
[0, 0, 1]
481
player의 승률은 0.481 입니다.

[1, 1, 1]
502
player의 승률은 0.502 입니다.
[1, 0, 0]
525
player의 승률은 0.525 입니다.

[0, 0, 0]
512
player의 승률은 0.512 입니다.
```

하나의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 한 장 더 받는다.

-플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [29]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크므로 player2 승입니다.
[0, 0, 1]
['5', '0', 'K', '7', '9', '7', '8', '2', '4', '5',
 '3', 'K', '8', '7', '6']
dealer의 처음 카드 두 장은 ['8', '2']
([5, 10, '6'], [10, 7], [9, 7], ['8', '2', '4'])
1 번째 카드 합은 21
2 번째 카드 합은 17
3 번째 카드 합은 16
4 번째 카드 합은 14
[21, 17, 16, 14]
deal은 : 14
플레이어가 블랙잭이 되어 player0 승입니다.
딜러보다 값이 크므로 player1 승입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 1, 1]
570
player의 승률은 0.57 입니다.
```

승률이 0.535~0.57 구간으로 나옵니다.
평균적으로는 0.543의 승률을 보입니다.

```
[1, 1, 1] [0, 0, 1]
538 540
player의 승률은 0.538 입니다. player의 승률은 0.54 입니다.
[0, 0, 1] [1, 1, 1]
539 553
player의 승률은 0.539 입니다. player의 승률은 0.553 입니다.
[0, 0, 0]
542
player의 승률은 0.542 입니다.
[0, 0, 0]
536
player의 승률은 0.536 입니다.
[1, 1, 1]
535
player의 승률은 0.535 입니다.
[0, 0, 0]
539
player의 승률은 0.539 입니다.
[0, 0, 0]
541
player의 승률은 0.541 입니다.
```

하나의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 x

-딜러와 플레이어 모두 카드를 한 장 더 받는다.

-플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [41]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다")
```

딜러보다 값이 크므로 player1 승입니다.
딜러보다 값이 크므로 player2 승입니다.
[1, 1, 1]
['9', '4', '4', '5', 'A', 'K', '6', '2', '3'
'Q', 'A', '9', '2', '9']
dealer의 처음 카드 두 장은 ['6', '2']
([9, 4, '9'], [4, 5, '2'], [11, 10], ['6',
1 번째 카드 합은 22
2 번째 카드 합은 11
3 번째 카드 합은 21
4 번째 카드 합은 11
[22, 11, 21, 11]
deal은 : 11
플레이어는 21이 초과하여 버스트, player0 패입니다.
딜러보다 값이 크므로 player1 승입니다.
플레이어가 블랙잭이 되어 player2 승입니다.
[0, 1, 1]
529
player의 승률은 0.529 입니다.

승률이 0.509~0.552 구간으로 나옵니다.
평균적으로는 0.531의 승률을 보입니다.

[0, 1, 0]
534
player의 승률은 0.534 입니다
[1, 0, 1]
553
player의 승률은 0.553 입니다
[0, 0, 0]
534
player의 승률은 0.534 입니다
[1, 0, 1]
552
player의 승률은 0.552 입니다.
[0, 0, 0]
509
player의 승률은 0.509 입니다.
[1, 1, 1]
527
player의 승률은 0.527 입니다.

하나의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 한 장 더 받는다.

-플레이어는 카드 숫자 합이 14보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [11]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다")
[1, 1, 0]
['2', '5', '8', 'A', '3', '9', 'Q', 'A', 'K',
'8', '5', 'J', '3', '4', '3', 'K', 'J', '2',
'J', '9']
dealer의 처음 카드 두 장은 ['Q', 'A']
1
([2, 5, '9'], [8, 11], [3, 9, 'J'], [10, 11])
1 번째 카드 합은 16
2 번째 카드 합은 19
3 번째 카드 합은 22
4 번째 카드 합은 21
[16, 19, 22, 21]
deal은 : 21
플레이어는 딜러보다 값이 크지 않으므로 player0 패입니다.
플레이어는 딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어는 21이 초과하여 버스트, player2 패입니다.
[0, 0, 0]
488
player의 승률은 0.488 입니다.
```

승률이 0.484~0.531 구간으로 나옵니다.
평균적으로는 0.508의 승률을 보입니다.

```
[1, 1, 1]
531
player의 승률은 0.531 입니다.
[1, 0, 1]
518
player의 승률은 0.518 입니다.
[1, 1, 0]
513
player의 승률은 0.513 입니다.
[0, 0, 0]
513
player의 승률은 0.513 입니다.
[1, 1, 0]
498
player의 승률은 0.498 입니다.
[1, 0, 0]
484
player의 승률은 0.484 입니다.
[1, 1, 1]
517
player의 승률은 0.517 입니다.
[0, 0, 0]
494
player의 승률은 0.494 입니다.
[0, 0, 0]
527
player의 승률은 0.527 입니다.
```

하나의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 한 장 더 받는다.

-플레이어는 카드 숫자 합이 16보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [23]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다.")

[4, 2, 'K'], [9, 7], [9, 5, '7'], [8, 7, 2]
1 번째 카드 합은 16
2 번째 카드 합은 16
3 번째 카드 합은 21
4 번째 카드 합은 17
[16, 16, 21, 17]
deal은 : 17
딜러보다 값이 크지 않으므로 player0 패입니다.
딜러보다 값이 크지 않으므로 player1 패입니다.
플레이어가 블랙잭이 되어 player2 승입니다.
[0, 0, 1]
552
player의 승률은 0.552 입니다.
```

승률이 0.503~0.568 구간으로 나옵니다.
평균적으로는 0.542의 승률을 보입니다.

```
[1, 1, 1] [1, 0, 0]
540 504
player의 승률은 0.54 입니다 player의 승률은 0.504 입니다

[1, 0, 0] [1, 1, 1]
549 558
player의 승률은 0.549 입니다 player의 승률은 0.558 입니다

[-1, 0, 0] [0, 0, 0]
547 568
player의 승률은 0.547 입니다 player의 승률은 0.568 입니다

[0, 0, 0] [0, 0, 0]
549 530
player의 승률은 0.549 입니다 player의 승률은 0.53 입니다

[0, 0, 0]
531

player의 승률은 0.531 입니다.
```

하나의 덱_카드 한 장만 더 받기

-소프트 17 규칙을 적용 o

-딜러와 플레이어 모두 카드를 한 장 더 받는다.

-플레이어는 카드 숫자 합이 18보다 작을 때 카드를 더 받는다.

-블랙잭을 1000번 시행했을 경우,

```
In [34]: #전체 블랙잭 플레이 횟수는 1000
#그 중 p1이 승리한 횟수는 p1_win_cnt
p1_win_cnt = 0
i = 0
while(i < 1000):
    p1_result = blackjack_play2()
    if p1_result == 1:
        p1_win_cnt += 1
    i += 1
print(p1_win_cnt)
print("player의 승률은", p1_win_cnt/1000, "입니다")
[3, 6, 'J'], [10, 7, '6'], [3, 9, 'J'], [4, 10, 10, 10], [10, 10, 10, 10]
1 번째 카드 합은 19
2 번째 카드 합은 23
3 번째 카드 합은 22
4 번째 카드 합은 20
[19, 23, 22, 20]
deal은 : 20
딜러보다 값이 크지 않으므로 player0 패입니다.
플레이어는 21이 초과하여 버스트, player1 패입니다.
플레이어는 21이 초과하여 버스트, player2 패입니다.
[0, 0, 0]
546
player의 승률은 0.546 입니다.
```

승률이 0.492~0.546 구간으로 나옵니다.
평균적으로는 0.522의 승률을 보입니다.

[1, 0, 1]	[1, 1, 1]
492	524
player의 승률은 0.492 입니다.	player의 승률은 0.524 입니다
[1, 0, 0]	[0, 0, 1]
534	525
player의 승률은 0.534 입니다.	player의 승률은 0.525 입니다
[0, 1, 1]	[1, 1, 1]
511	541
player의 승률은 0.511 입니다	player의 승률은 0.541 입니다
[0, 1, 0]	[0, 0, 0]
542	504
player의 승률은 0.542 입니다.	player의 승률은 0.504 입니다
[0, 0, 0]	
506	
player의 승률은 0.506 입니다.	

하나의 덱에서의 실험 결과

하나의 덱에서 카드를 여러 번 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때,
무한대 덱에서 실험했을 때와 동일하게, **기준점이 커질수록 승률이 오르는 것**을 알 수 있습니다.

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **소프트 17을 적용했을 경우, 승률이 떨어지는 경향**을 보입니다. 여기서 소프트 17을 적용했을 경우와 아닌 경우의 승률 차가 미미해 보이는데, 이것은 1000번 시행했을 경우의 승률을 10번 구해서 평균을 취했기 때문이라고 생각합니다. 블랙잭 1000번 시행을 10번이 아니라 무한대만큼 반복해서 평균을 취해준다면 더 큰 승률 차이가 날 것이라고 생각합니다.

하나의 덱에서 카드를 한 장만 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때, 무한대덱에서의 결과와 동일합니다. **14일 때와 16일 때는, 카드를 여러번 받을 때와 같이 기준점이 커질수록 승률이 오르는 경향을 보이지만, 16과 18을 비교했을 때는 앞과 같은 경향을 보이지 않습니다.** 딜러가 한 장의 카드만 추가로 받게 되기 때문에 딜러의 버스트 가능성이 줄어들기 때문에 이러한 결과가 나온 것이라 생각합니다. 14와 16의 경우, 플레이어가 버스트가 될 가능성이 현저히 줄기 때문에 딜러의 버스트 가능성이 줄어들었다 해도 큰 영향을 받지 않고 경향성을 따라가는 것으로 보입니다.

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **미세하게 소프트 17을 적용했을 경우가 승률이 더 떨어지는 것을 알 수 있습니다.** 승률차이가 크게 나지 않는 것이 게임 시행 횟수가 적어서 정확한 값이 나오지 않기 때문이라고 생각합니다.

하나의 덱에서의 실험 결과

카드 여러 장 뽑기		
	소프트 17 적용 x	소프트 17 적용 o
14	0.500	0.499
16	0.562	0.544
18	0.578	0.577

카드 한 장만 뽑기		
	소프트 17 적용 x	소프트 17 적용 o
14	0.509	0.508
16	0.543	0.542
18	0.531	0.522

- ⇒ 무한대 덱과 마찬가지로, 카드를 여러 장 받을 수 있는 경우에는, 플레이어가 두 장의 카드 숫자 합이 18보다 작을 때 카드를 추가로 더 받고, 소프트17 규칙을 적용하지 않는다면, 승률을 높일 수 있다.
- ⇒ 무한대 덱과 마찬가지로, 카드를 한 장만 추가로 받을 수 있는 경우보다 여러 장 추가로 받을 수 있는 경우가 플레이어에게 유리하다.
- ⇒ 무한대 덱과 하나의 덱을 비교해 보았을 때, 어떠한 규칙성이 보이지는 않고, 따라서 무한대덱과 하나의 덱 중 어떤 덱이 더 플레이어에게 유리한지는 알 수 없습니다.

두 개의 덱_카드 여러 장 더 받기

위와 같은 방식으로 진행하기에는 너무 시간 낭비인 것 같아
코드를 통해 평균 승률을 구하도록 했습니다.
다음 장에서는 바로 표로 보여드리겠습니다.

```
for j in range(10):
    #전체 블랙잭 플레이 횟수는 1000 * 10
    #그 중 p1이 승리한 횟수는 p1_win_cnt
    p1_win_cnt = 0
    i = 0
    while(i < 1000):
        p1_result = blackjack_play()
        if p1_result == 1:
            p1_win_cnt += 1
        i += 1
        print(p1_win_cnt/1000)
        win_rate += p1_win_cnt/1000
    print("평균적으로 ", win_rate/10, "이라는 승률을 보입니다.")

test1000()
```

```
deal은 : 22
모든 플레이어가 승리했습니다.
[1, 1, 1]
['4', 'J', '8', '6', 'Q', '4', '5', '3', '6', '4', '4', '7', '9',
'5', '9', 'A', '9', '6']
dealer의 처음 카드 두 장은 ['5', '3']
([4, 10], [8, 6], [10, 4], [5, 3, 6, 4])
1 번째 카드 합은 14
2 번째 카드 합은 14
3 번째 카드 합은 14
4 번째 카드 합은 18
[14, 14, 14, 18]
deal은 : 18
딜러보다 값이 크지 않으므로 player0 패입니다.
딜러보다 값이 크지 않으므로 player1 패입니다.
딜러보다 값이 크지 않으므로 player2 패입니다.
[0, 0, 0]
0.476
평균적으로 0.4920999999999999 이라는 승률을 보입니다.
```

평균적으로 0.492이라는 승률이 나옵니다.

두 개의 덱에서의 실험 결과

두 개의 덱에서 카드를 여러 번 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때,
무한대 덱에서 실험했을 때와 동일하게, **기준점이 커질수록 승률이 오르는 것을 알 수 있습니다.**

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **두 경우 사이의 어떠한 규칙성이 보이지 않습니다.**

두 개의 덱에서 카드를 한 장만 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때, 무한대덱에서의 결과와 동일합니다. **14일 때와 16일 때는, 카드를 여러번 받을 때와 같이 기준점이 커질수록 승률이 오르는 경향을 보이지만, 16과 18을 비교했을 때는 앞과 같은 경향을 보이지 않습니다.** 딜러가 한 장의 카드만 추가로 받게 되기 때문에 딜러의 버스트 가능성이 줄어들기 때문에 이러한 결과가 나온 것이라 생각합니다. 14와 16의 경우, 플레이어가 버스트가 될 가능성이 현저히 줄기 때문에 딜러의 버스트 가능성이 줄어들었다 해도 큰 영향을 받지 않고 경향성을 따라가는 것으로 보입니다.

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **미세하게 소프트 17을 적용했을 경우가 승률이 더 떨어지는 것을 알 수 있습니다.** 승률차이가 크게 나지 않는 것이 게임 시행 횟수가 적어서 정확한 값이 나오지 않기 때문이라고 생각합니다

두 개의 덱에서의 실험 결과

카드 여러 장 뽑기

	소프트 17 적용 x	소프트 17 적용 o
14	0.492	0.493
16	0.547	0.550
18	0.583	0.571

카드 한 장만 뽑기

	소프트 17 적용 x	소프트 17 적용 o
14	0.509	0.521
16	0.542	0.549
18	0.535	0.528

- ⇒ 무한대 덱, 하나의 덱과 마찬가지로, 카드를 여러 장 받을 수 있는 경우에는, 플레이어가 두 장의 카드
- ⇒ 숫자 합이 18보다 작을 때 카드를 추가로 더 받고, 소프트17 규칙을 적용하지 않는다면,
- ⇒ 승률을 높일 수 있다.
- ⇒ 무한대덱, 하나의 덱, 두 개의 덱은 여러 가지 조건 하에 승률을 비교해보았을 때, 승률이 비슷합니다.
- ⇒ 따라서 셋 중 어떠한 덱이 플레이어에게 더 유리할지는 알 수 없습니다.

세 개의 덱에서의 실험 결과

세 개의 덱에서 카드를 여러 번 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때,
무한대 덱에서 실험했을 때와 동일하게, **기준점이 커질수록 승률이 오르는 것을 알 수 있습니다.**

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **두 경우 사이의 어떠한 규칙성이 보이지 않습니다.**

세 개의 덱에서 카드를 한 장만 더 받을 수 있다고 가정하고 실험했을 때,

플레이어가 처음으로 받은 자신의 카드 합에 따라 카드를 더 받을지 말지 결정한다고 하면, 그 기준점이 14, 16, 18일 때 카드를 추가로 받는 경우들을 비교해보았을 때, 무한대덱에서의 결과와 동일합니다. **14일 때와 16일 때는, 카드를 여러번 받을 때와 같이 기준점이 커질수록 승률이 오르는 경향을 보이지만, 16과 18을 비교했을 때는 앞과 같은 경향을 보이지 않습니다.**
딜러가 한 장의 카드만 추가로 받게 되기 때문에 딜러의 버스트 가능성이 줄어들기 때문에 이러한 결과가 나온 것이라 생각합니다. 14와 16의 경우, 플레이어가 버스트가 될 가능성이 현저히 줄기 때문에 딜러의 버스트 가능성이 줄어들었다 해도 큰 영향을 받지 않고 경향성을 따라가는 것으로 보입니다.

소프트 17을 적용했을 경우와 적용하지 않았을 경우를 비교해보았을 때, **두 경우 사이의 어떠한 규칙성이 보이지 않습니다.**

세 개의 덱에서의 실험 결과

카드 여러 장 뽑기

	소프트 17 적용 x	소프트 17 적용 o
14	0.508	0.493
16	0.539	0.541
18	0.588	0.574

카드 한 장만 뽑기

	소프트 17 적용 x	소프트 17 적용 o
14	0.516	0.519
16	0.539	0.543
18	0.533	0.533

⇒ 무한대 덱, 하나의 덱, 두 개의 덱과 마찬가지로, 카드를 여러 장 받을 수 있는 경우에는, 플레이어가
⇒ 두 장의 카드 숫자 합이 18보다 작을 때 카드를 추가로 더 받고, 소프트17 규칙을 적용하지 않는다면,
⇒ 승률을 높일 수 있다.

⇒ 무한대덱, 하나의 덱, 두 개의 덱은 여러 가지 조건 하에 승률을 비교해보았을 때, 승률이 비슷합니다.
⇒ 따라서 셋 중 어떠한 덱이 플레이어에게 더 유리할지는 알 수 없습니다.

About the people



