What is the Optimum Foursome?

Riddler Classic – March 11, 2022

We're playing a game where you get to pick four whole numbers. Then I will roll four fair dice. If any two of the dice add up to any one of the four numbers you picked, then you win! Otherwise, you lose. To maximize your chances of winning, which four numbers should you pick? And what are your chances of winning with that pick?

I wrote Python code to evaluate each of the 330 combinations of four different picks chosen from the eleven numbers in the range of 2 to 12 (the possible sums of two dice), and see how often each set will win for all of the $6^4 = 1296$ equally likely rolls of four dice.

The program found that the optimal four number pick is $\boxed{\{4, 6, 8, 10\}}$. This pick will win on 1264 out of 1296 possible rolls of the dice, giving you winning chance of about $\boxed{97.53\%}$.

I was a bit surprised to see all even numbers here, but upon reflection it does make sense. Any roll of four dice is guaranteed to have at least two even numbers among the six possible two-dice sums, whereas one eighth of the possible rolls (either all dice even or all dice odd) has no odd two-dice sums at all.

The next best picks are {2, 6, 8, 10} or {4, 6, 8, 12}, either of which will win 1246/1296 ≈ 96.14% of the time, then {4, 6, 7, 9} or {5, 7, 8, 10}, which will win 1238/1296 ≈ 95.52% of the time.

The only losing rolls when using the {4, 6, 8, 10} pick are: {1, 1, 1, 1}, {1, 1, 1, 2}, {1, 1, 1, 4}, {1, 1, 1, 6}, {1, 1, 6, 6}, {1, 6, 6, 6}, {3, 6, 6, 6}, {5, 6, 6, 6}, and {6, 6, 6, 6}.

Here is the code to find the best pick:

```
from itertools import combinations

# return 1 if any pair of dice sum to a Picked value
def Score(lbPick, liDice):
    global glliPairs
    for i in range(len(glliPairs)):
        liPair = glliPairs[i]
        if (lbPick[liDice[liPair[0]] + liDice[liPair[1]]]):
            return 1
    return 0

# make a global list of all six pairs of four dice
glliPairs = list(combinations([0,1,2,3], 2))
# make a list of all possible Pick combinations
lliPicks = list(combinations([*range(2, 13)], 4))
liDice = [0]*4   # a list to store roll values
iBestScore = 0
liBestPick = []

# for every possible set of Picks (330 total)
for i in range(len(lliPicks)):
    # make a boolean array of the Pick values
    liPick = lliPicks[i]
    lbPick = [False]*13
    for j in range(4):
        lbPick[liPick[j]] = True

    # accumulate a score for all dice rolls (1296 total)
    iScore = 0
    for liDice[0] in range (1, 7):
        for liDice[1] in range (1, 7):
            for liDice[2] in range (1, 7):
                for liDice[3] in range (1, 7):
                    iScore += Score(lbPick, liDice)

    if (iScore > iBestScore):
        liBestPick = liPick
        iBestScore = iScore

print (liBestPick, iBestScore)
```

Dean Ballard
March 13, 2022