Shannon's Mind Reading Game

Himadri, Mrittika, Ayan, Drishti, Aman, Siddhartha

December 26, 2023

Table of Contents

Table of Contents

History

Mind Reader: The Game

The mind reader machine is a game where the user plays against the machine, the user selects the **right** or **left** keys, and the machine tries to predict which key the user will click. If the machine guesses correctly it gets a point, and otherwise the user. It was one of the earlier games developed by Claude Shannon, which gradually led to the Mind Reading Game.
If both the players have IID selections, chances of winning were equal. But here, the machine exploits the facts that human's aren't the best at generating IID values.

Mind Reader: Ideas

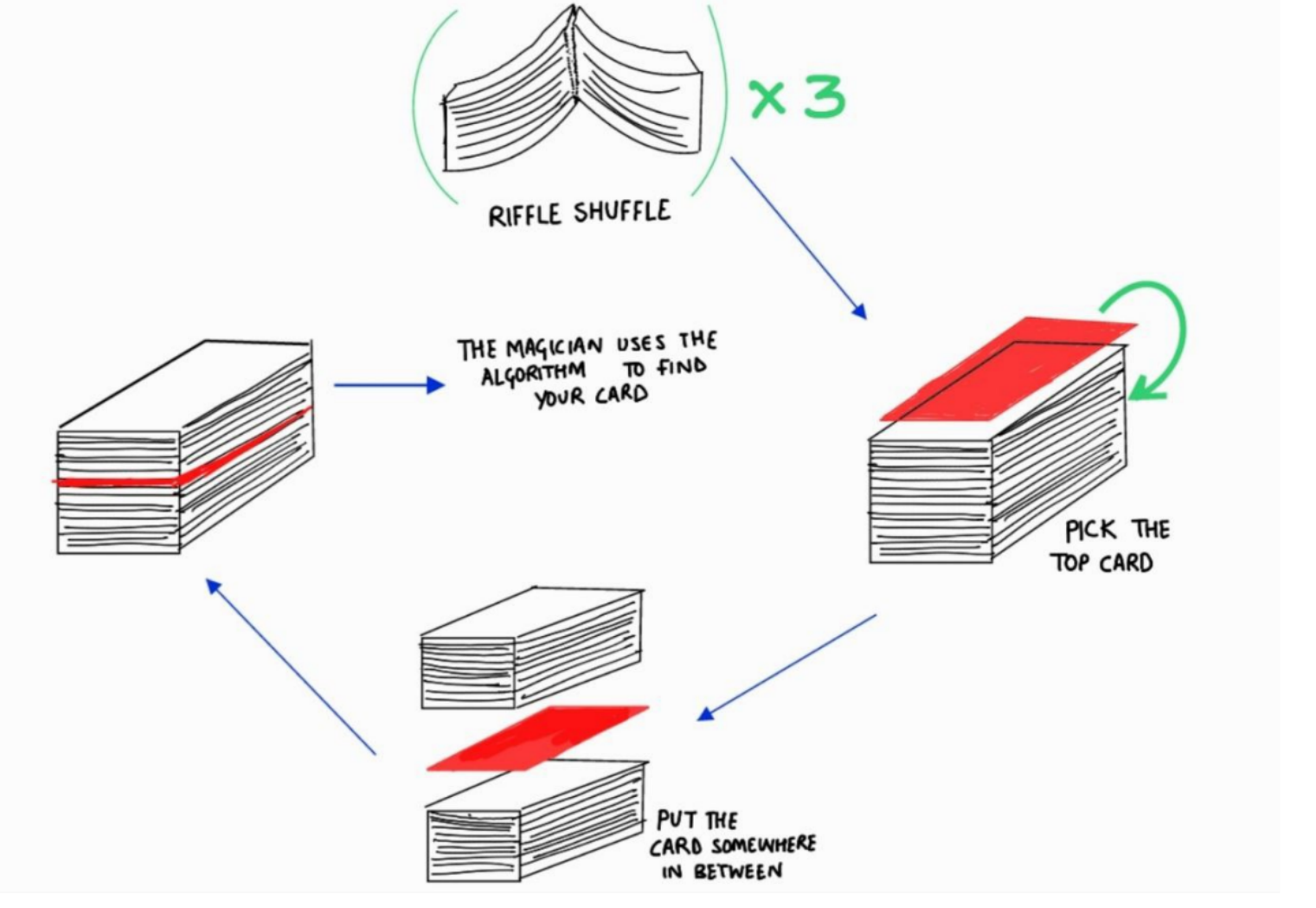
1. Bias Predictor: Detects bias in moves 'Left' and 'Right'
2. Flipping Bias Predictor: Detects bias in flipping of move or otherwise
3. Pattern Predictor: Detects repetition of moves after fixed length
4. Flipping Pattern Predictor: Detects repetition in occurence of flips after fixed length
5. Shannon Inspired Predictor: Detects user tendency after every win and loss
6. Hagelbarger's Inspired Predictor: Detects machine tendency after every win and loss

The Problem

The Problem

A magician hands you over a standard deck of 52 cards. You are asked to riffle shuffle the card thrice, draw the top card and place it anywhere in the middle. The magician claims that he knows which card you've drawn (without watching your entire performance, of course!). How is this possible?
Well, this is not always possible, but has a pretty high probability of success!

Broader Picture



Definitions

Riffle Shuffle

Riffle shuffle is a classic technique for randomised shuffling of cards. In this process:

1. Split the deck into (approximately) 2 halves.
2. Interleave the two halves

Index

Let us fix a bijection:

$$\psi : \mathit{Deck} \to \{1, 2, \cdots, 52\}$$

Once this is fixed, we no longer bother about the type of cards. (They play no role henceforth.)
The Index function tells the position of each card after a riffle shuffle.

$$\mathit{Index}(\sigma) = \{\sigma(1), \sigma(2), \cdots, \sigma(52)\}$$

given σ is a permutation of [52]

Rising Sequence

For a permutation σ:

$$S_a = \{a, a+1, \cdots, a+k-1\}$$

is a rising sequence of length k, if:

$$\sigma(a-1) > \sigma(a) < \sigma(a+1) < \cdots < \sigma(a+k-1) > \sigma(a+k)$$

Rising Sequence Singleton

A rising sequence singleton is essentially a rising sequence of length 1. This is what the magician actually tries to find at the end; i.e the drawn card is expected to be the ONLY rising singleton. Basically, **'a'** is the singleton card, if:

$$\sigma(a-1) > \sigma(a) > \sigma(a+1)$$

Switch

Switching is the move of pulling out the top card and placing it randomly in any of the remaining positions. Here, we have assumed the human tendency of putting the card in the middle.

The Trick

The Trick

Here's the catch: the drawn card is the one and only **rising singleton** pretty often!

Why? Well the reason is actually pretty intuitive!

After the *first* riffle shuffle, there are approximately 2 rising sequences of **length 26**.

After the *second* riffle shuffle, almost every card is a part of a rising sequence of **length 13**.

The Trick

After the *third* riffle shuffle, almost every card is part of a rising sequence of **length 7**. So when you change the position of the first card (which doesn't deviate much out of the set $\{1,2,3,4\}$ because of how riffle shuffle works) we almost always create a rising singleton.

The Models

The Models

We use the following models for the simulation of the situation:

1. Splitting the Deck
2. Riffle Shuffling
3. Switch

Splitting the Deck

The deck was split into two parts (a hundred times), and noted the value of the smaller deck (x). Each time, we tossed a coin to choose between (x) and (52-x) which got rid of any IID issues. We then fit a normal distribution over our data to obtain:

$$N(\mu, \sigma^2) = N(25.78, 2.3046^2)$$

Riffle Shuffling

Once we split the deck, we name the two arrays as Left and Right. At every moment:

$$P(Left) = \frac{n(Left)}{n(Left) + n(Right)}$$

$$P(Right) = \frac{n(Right)}{n(Left) + n(Right)}$$

We assume that the chance of the next card falling is proportional to the number of cards left in the array. This assumption is justified because the heavier the hand, the more probability that the next card will fall from it.

The Final Switch

We draw the top card and place it anywhere. For this, we use a binomial distribution (keeping in mind the human bias of putting it in the middle):

$$Binom(n, p) = Binom(52, \frac{1}{2})$$

Algorithm

Riffle Shuffle

Algorithm 1 Riffle Shuffle the deck of cards

```
1:  procedure RiffleShuffle(Cards)
2:      Deck := {}
3:      split := floor(rnorm(1, mean = 25.78, sd = 2.3046))
4:      LeftHand := Cards[1 : split]
5:      RightHand := Cards[split + 1 : 52]
6:      for i in {1, 2, ··· , 52} do
7:          hand := sample((L, R), prob = (size(LeftHand), size(RightHand)))   ▷ hand the Card falls from
8:          if hand == L then
9:              Deck ← LeftHand[1]
10:             pop(LeftHand)                                                  ▷ remove the added Card
11:         else
12:             Deck ← RightHand[1]
13:             pop(RightHand)
14:         end if
15:     end for
16:     return Deck
17: end procedure
```

Count of Rising Sequence Singletons

Algorithm 2 Count number of Rising Sequences of size 1 and check if it's unique

```
1:  procedure RisingSingleton(S)
2:      Index := {}
3:      for i in {1, 2, ··· 52} do
4:          Index[S[i]] = i                          ▷ Define the Index array.
5:      end for
6:      Sum := 0, SingletonCard := 0
7:      for i in {1, 2, ··· , 52} do
8:          if Index[i] ≥ Index[i + 1] and Index[i] ≤ Index[i − 1] then   ▷ Singleton Rising Sequence
9:              Sum = Sum + 1
10:             SingletonCard = i
11:         end if
12:     end for
13:     return (Sum == 1), SingletonCard
14: end procedure
```

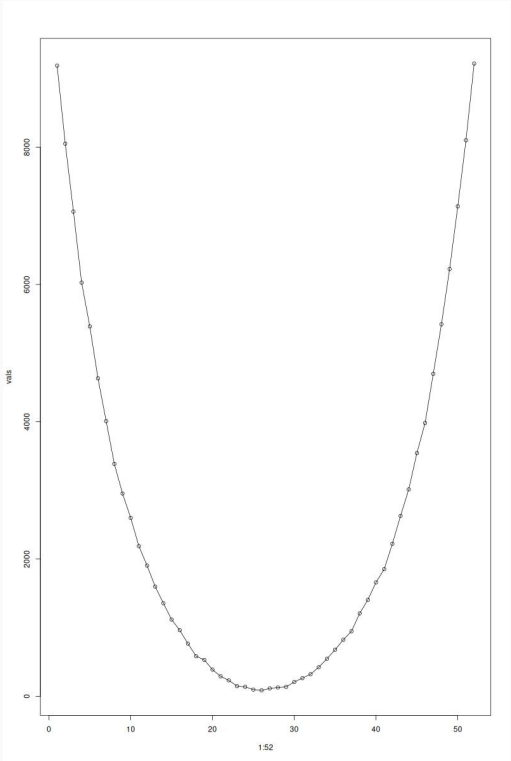
Beyond

Beyond

The original problem statement asked us to pick a random card and put it anywhere. That is a terrible idea. We will show why.

We decided to pick the top card. However, if we were free to choose the index, what happens to the probability? Here's what we found!

Beyond



Observe that

$$\mathbb{P}(\text{Success} \mid i^{\text{th}} \text{ card is picked})$$

is maximized for $\mathrm{i} = 1, 52$

Too much randomness isn't good.

So what goes wrong when we *randomly pick* the card and put it back anywhere? We simulated this as well.

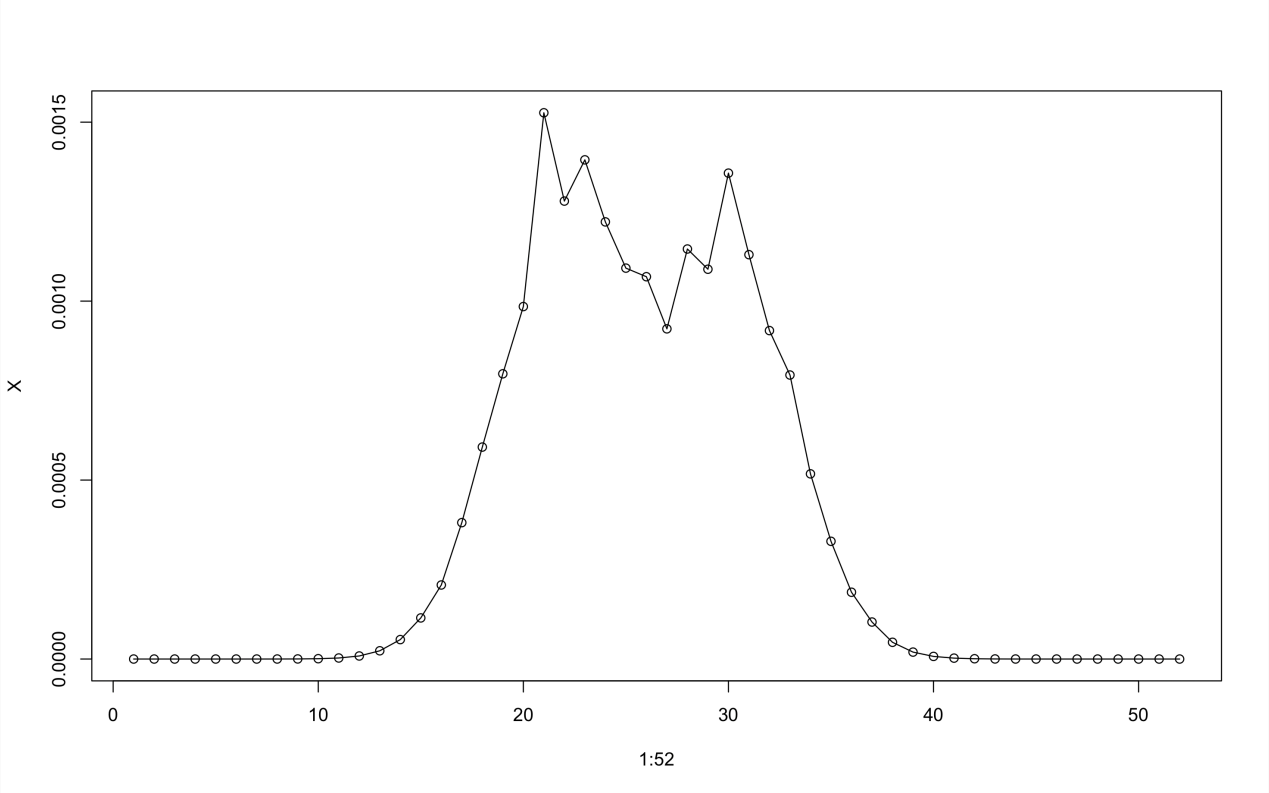
Ofcourse we use the same model again.

$$\text{Index} \sim \text{Bin}\left(52, \frac{1}{2}\right)$$

We plotted

$$\mathbb{P}\left(\text{Success} \,\middle|\, i^{\text{th}} \text{ card is picked}\right) \cdot \mathbb{P}\left(i^{\text{th}} \text{ card is picked}\right)$$

Too much randomness isn't good.



Summing the values we find this is barely above 1.75% which justifies why this was a terrible idea.

Conclusion

We were unable to work with the original statement beacuse of reasons previously mentioned. However, when we did run our algorithm with the associated models in the modified statement, we achieved a result of $\boxed{91\%}$.