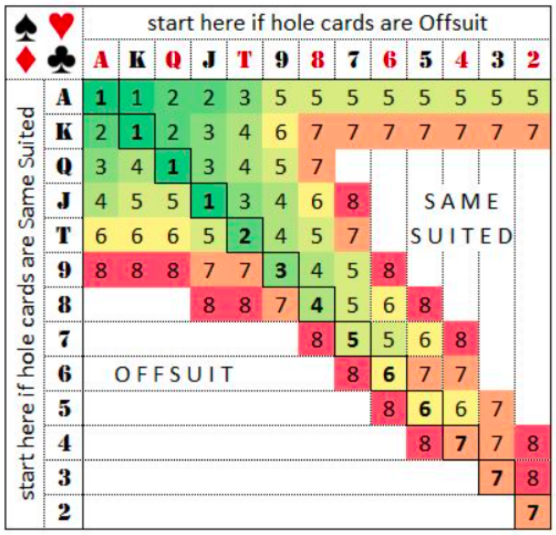
Evaluation of hold cards:

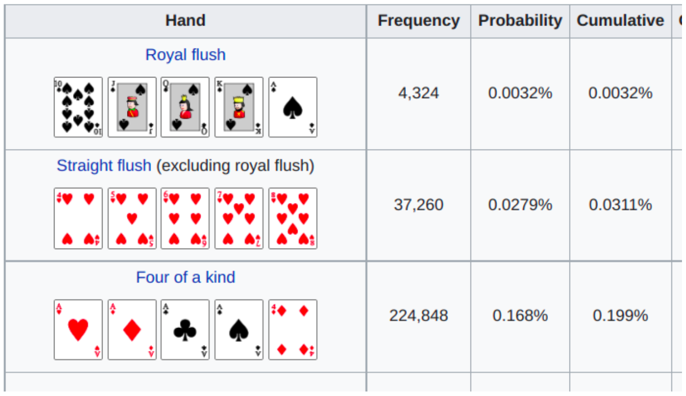
With the help of Texas hold'em starting hands provided in wiki, at first I wanted to build an evaluation function based on the given hold cards.


start here if hole cards are Offsuit
start here if hole cards are Same Suited
SAME SUITED
OFFSUIT

	A	K	Q	J	T	9	8	7	6	5	4	3	2
A	1	1	2	2	3	5	5	5	5	5	5	5	5
K	2	1	2	3	4	6	7	7	7	7	7	7	7
Q	3	4	1	3	4	5	7						
J	4	5	5	1	3	4	6	8					
T	6	6	6	5	2	4	5	7					
9	8	8	8	7	7	3	4	5	8				
8				8	8	7	4	5	6	8			
7							8	5	5	6	8		
6								8	6	7	7		
5									8	6	6	7	
4										8	7	7	8
3												7	8
2													7

Sklansky Hand Groups

It seemed like Sklanskly Hand Groups could be a nice starting point for evaluation. And here comes the next problem: How to construct an evaluation function with additional information about community cards. With the basic intuition of probability, I found Poker Probability in wiki. And cumulative in the chart could be regarded as approximation for *lose rate*.



Hand	Frequency	Probability	Cumulative
Royal flush	4,324	0.0032%	0.0032%
Straight flush (excluding royal flush)	37,260	0.0279%	0.0311%
Four of a kind	224,848	0.168%	0.199%

Poker Probability

Afterwards, I found that evaluation purely based on the aforementioned information has some flaws:

1. The community cards are not fully given, so it requires to consider all the combinations of community cards to obtain proper expected outcome.
2. Even if all the community cards are given, the cumulative chart does not consider the comparison between cards in the same type of hand.
3. The opponent share the same community cards, so taking the cumulative probability as lose rate would be inappropriate.
4. So,... why not just run the simulation to evaluate hold cards???

With the idea of simulation, the *win_rate* can be obtained by running the simulation for 1000 rounds. For each round, randomly fill up the remaining community cards and opponents, and then compare the strength of the two players. The *win_rate* is determined by the ratio of number of wins and number of simulations.

Follow the document, and the function to estimate *win_rate* of hole cards can be established. The estimated *win_rate* can be a useful tool for the agent to decide the next step.

Intuitive, the action can be determined by the following strategy:

$$action = \begin{cases} Fold & , \text{ if } win\_rate < 0.5 \\ Call & , \text{ if } win\_rate \geq 0.5 \end{cases}$$

However, the intuition does not work as well as expected. It still cannot win `CallPlayer` in most cases.

	Win	Lose
`CallPlayer`	0	10

Determine the amount to bet

It could be a nice try to take *amount* into consideration for the agent. Generally, the following inequation is satisfied for a reasonable bet, with the exception of bluff.

$$pot\_amount \times win\_rate \geq cost \times lose\_rate$$
$$\Rightarrow cost \leq pot\_amount \times \frac{win\_rate}{lose\_rate}$$

Based on the above relation, the agent can develop the following strategy:

$$action = \begin{cases} Fold & , \text{if } amount_{call} > pot_{amount} \times \frac{win\_rate}{lose\_rate} \\ Call & , \text{otherwise} \end{cases}$$

	Win	Lose
`CallPlayer`	5	5

When competing `CallPlayer`, $amount_{call}$ is 0. That implies that the action will always be *Call*. That is, it would be a competition between 2 `CallPlayer`s, it sounds nonsense and the result did not work well. Then the strategy will be transformed to:

$$action = \begin{cases} Fold & \text{, if } amount_{call} > pot\_amount \times \frac{win\_rate}{lose\_rate} \\ Raise & \text{, if } amount_{raise\_min} < pot\_amount \times \frac{win\_rate}{lose\_rate} \\ Call & \text{, otherwise} \end{cases}$$

	Win	Lose
`CallPlayer`	7	3

However, a *Raise* in the situation that $win\_rate < 0.5$ can be quite risky, so the strategy should be modified to:

$$action = \begin{cases} Fold & \text{, if } amount_{call} > pot\_amount \times \frac{win\_rate}{lose\_rate} \\ Raise & \text{, if } amount_{raise\_min} < pot\_amount \times \frac{win\_rate}{lose\_rate} \text{ and } win\_rate > 0.5 \\ Call & \text{, otherwise} \end{cases}$$

	Win	Lose
`CallPlayer`	9	1
`Baseline0`	7	3
`Baseline1`	8	2
`Baseline2`	9	1
`Baseline3`	8	2

Intuitively, to make the opponent fold or make the opponent toss more tokens with *Call*, the *Raise amount* should be as high as possible. Therefore, in the above agents with `Raise`, it tries to maximize *amount* without violating the relation: $amount_{raise} < pot\_amount \times \frac{win\_rate}{lose\_rate}$

Discussion of Raise amount

Intuitively, the agent should maximize *Raise amount*. However, if the agent follows the rule, the opponent may identify the *win_rate*. And there are further discussion about *amount*.

The selection of *amount* of *Raise* can be a crucial problem. Higher *amount* and lower *amount* can bring different consequences. And they can be shown in the following table:

High *amount*	Low *amount*
Higher risk Induce the opponent to *Fold*	Lower risk Prevent the opponent from *Fold*

And the consequences of the opponent's *Fold* are discussed in the following table:

	Induce the opponent to *Fold*	Prevent the opponent from *Fold*
Pros	More likely to get the *pot* directly.	The opponent tends to participate more rounds, and may toss more coins in total.
Cons	The opponent does not toss more tokens.	The opponent may toss less coins in a single stage.

So it is reasonable to raise less in preflop and raise more in latter stages. And adopt the idea of conserve_factor and risk_factor.

$$\text{risk\_factor} = 0.05, \text{ and conserve\_factor} = \begin{cases} 0.1 & \text{, for river} \\ 0.4 & \text{, for turn} \\ 0.8 & \text{, for flop} \\ 1 & \text{, for preflop} \end{cases}$$

$$amount_0 = pot\_amount \times \frac{win\_rate}{1-win\_rate}$$

$$action = \begin{cases} Fold & \text{, if } amount_{call} > amount_0(1+risk\_factor) \\ Raise & \text{, if } amount_{raise\_min} < amount_0(1+risk\_factor) \text{ and } win\_rate > 0.5 \\ Call & \text{, otherwise} \end{cases}$$

$$amount_{raise} \sim U(amount_0 \times (1 - conserve\_factor), amount_0 \times (1 + risk\_factor))$$

A little trick

Since there will be 20 rounds, it is a wise strategy to fold the remaining rounds if we have enough tokens and *Call* if $amount_{call}$ is 0.

$$action = \begin{cases} Call & \text{, if } amount_{call} = 0 \text{ and } (20 - round) * 10 < token_{self} - 1000 \\ Fold & \text{, if } (20 - round) * 10 < token_{self} - 1000 \\ \ldots \end{cases}$$

	Win	Lose
`CallPlayer`	9	1
`Baseline0`	8	2
`Baseline1`	9	1
`Baseline2`	10	0
`Baseline3`	10	0