UNIVERSITÄT
BERN

Data Science Project

JassBot
Project Report

June 15, 2023

Raphael Ziegler
raphael.ziegler@gmail.com

Abstract

A prediction model for reinforced concrete columns is developed to facilitate cost estimation of construction projects. Different models are compared to find the most reliable prediction model. The analysis can be performed using software freely available on the Internet (FOSS).
The data are concrete column purchase prices from completed construction projects, which are manually entered into the source Excel file. The original values are checked for input errors as much as possible, sorted out, and serve as input for the Linear Regression Model. For the Neural Network, the data is additionally normalized so that the values are between 0 and 1. The models should reach a prediction accuracy of 70% to be used in general without further evaluation.
The analysis is performed for different data combinations to achieve the best result. Short-term market fluctuations (covid, war) cannot be considered and remain a risk in the prediction.

Contents

1 Project Objectives

For this project, a JassBot is trained to play against itself. For this purpose, a reinforcement learning algorithm will be trained to find the best game strategy. The baseline will be the random pick of a playable card for comparison.

1.1 Deck [1]

Jass is played with a deck of 36 cards (Ace, King, Queen, Jack, 10, 9, 8, 7, 6) Swiss-French or Swiss-German cards (Ass, König, Oober, Under, Banner (= 10), 9, 8, 7, 6). The Swiss-French cards are in the ordinary French suits but have a distinctive design. The Swiss-German cards use Swiss suits, a variant of German suits, and also have a distinctive design.

The game is traditionally played with Swiss suited playing cards east of the Brünig-Napf-Reuss line [2] and with the French in western Switzerland.
The Swiss suits are Rosen (roses) Eicheln (acorns), Schellen (bells) and Schilten (shields)

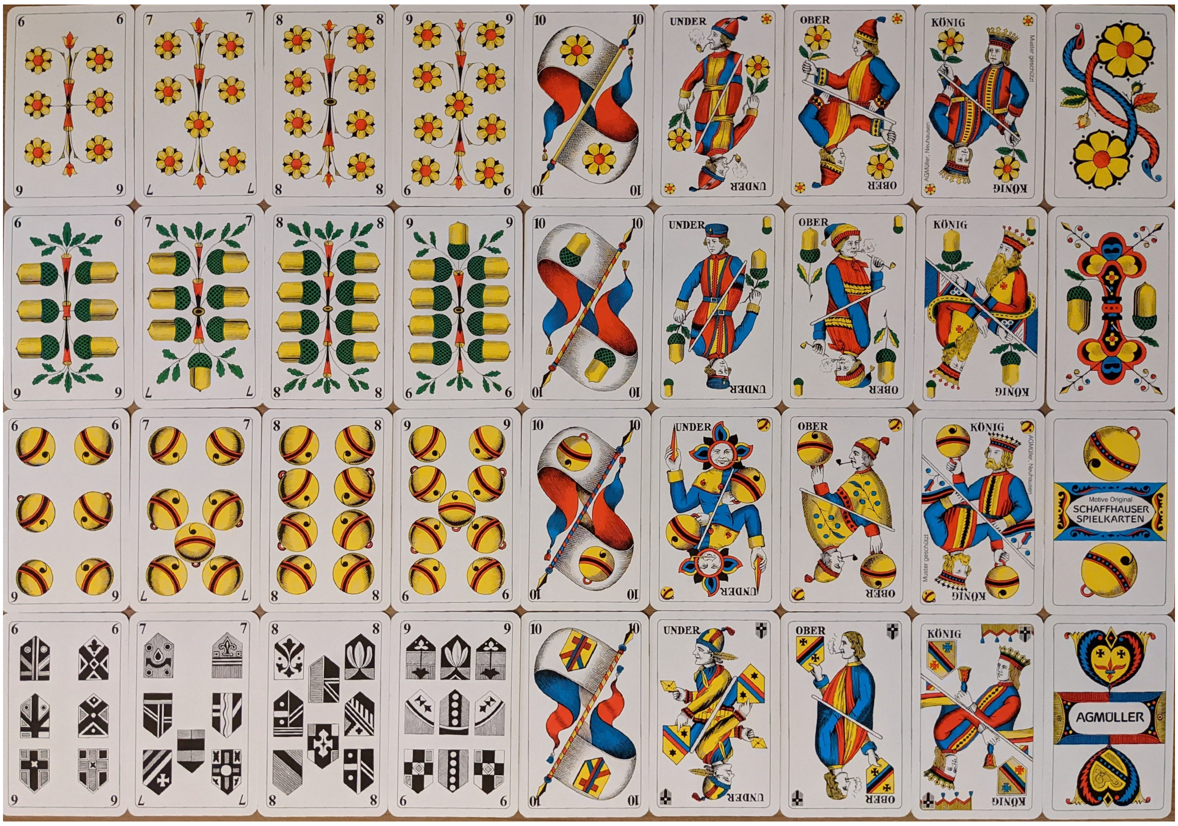


FIGURE 1.1: Swiss German Cards

1.2 Game variants

Jassen is the collective term for different game variants with different numbers of players. In most variants, each player receives nine cards (hand). The most popular variant is the slider, where two teams of 2 players play against each other until one team reaches the points needed to win. Another well-known game variant is the Differenzler (known from the TV show Samstagsjass). Here four players play against each other; before each deal, each player announces how many points he intends to make in this deal. The difference to the actual score is noted as penalty points and added at each round. The player with the least penalty points wins. Another game variant is the Handjass, which can be played in pairs, threes, or fours. In each deal, the player who takes the most points scores a stroke, and anyone who takes fewer than 26 points gets a Null - sometimes known as a Herdöpfel (potato) - which is worth minus one stroke. The player who achieves a net score of 7 strokes wins and retires from the game: the loser is the last player who fails to reach seven strokes. [3]

1.3 Rules [1]

Jass is essentially a game of points which are scored for three features known as Stöck, Wiis, Stich, respectively, "marriages, melds, tricks".

1.3.1 Tricks

The trump Jack, also called Puur, counts 20 and is the highest card in the game. The trump Nine, or Nell, Näll, is the second best card. Plain suit numerals below 10 count nothing. The total value of all counters in the pack is 152, that is, 62 in trumps plus 30 in each plain suit. Winning the last trick scores an additional 5 points. Hence the total possible for the third scoring feature, "tricks", is normally 157 points.

The rank of the cards, from highest to lowest, and their values in card points are shown in the following table:

TABLE 1.1: Card Values - trump games

Card Values											
Plain suit rank			A	K	O/Q	U/J	10	9	8	7	6
Value	20	14	11	4	3		10		0	0	0
Trump suit rank	J/U	9	A	K	O/Q	U/J	10	9	8	7	6

The no-trumps game called Obenabe and Undenufe, in which the ranks are reversed, are shown in the following table:

TABLE 1.2: Card Values - no-trump games

Card Values																		
Obenabe										Undenufe								
Rank	A	K	O/Q	U/J	10	9	8	7	6	6	7	8	9	10	U/J	O/Q	K	A
Value	11	4	3	2	10	0	8	0	0	11	0	8	0	10	2	3	4	0

1.3.2 Marriage

Marriage (Stöck): A marriage is the holding in one hand of the König and Ober (King and Queen) of trumps. Its holder claims it upon the second of them to a trick. Its score of 20 is recorded as if made before those for melds and tricks, even though it is not revealed until after melds have been declared.

1.3.3 Melds

Meld (Wys or Weis): A meld is a suit-sequence of three or more cards, or a quartet of Aces, Kings, Queens, or Jacks scoring as follows:

- Four Jacks: scores 200
- Four 9's: scores 150
- Five or more in suit sequence: scores 100
- Four A, K, Q, 10: scores 100
- Four in suit sequence: scores 50
- Three in suit sequence: scores 20

A card may not be used in two melds at once, though the trump King or Queen may belong to a meld in addition to being married, that is, a player holding four Kings and a sequence of four to the Ace or King would count only 100 for Kings, not also 50 for the sequence.

Only the holder of the best meld may score for it, but he may also score for any other melds he holds involving entirely different cards, and in a partnership game, his partner also scores for those held by his partner. The holder of the best meld is found in the following way as each player contributes a card to the first trick. The leader declares the value of his best meld. The next, upon playing his card says "good" (gut) if he can't beat; otherwise he declares a higher value or the same value and the number of cards it contains. A longer meld beats a shorter, so the previous player then says "not good" if he can beat it, "good" if he can't, or "equal" (gleich). If equal, the next states its rank if a quartet, or its top card if a sequence. A higher rank beats a lower, and the previous player again says "not good". "good", or "equal". Equality must mean a sequence is in question, which the second player can then only win by truthfully announcing "in trumps". Otherwise, all else being equal, the previous player wins by prior position. The pecking order is: value, length, height, trump, position.

1.4 Chosen Game and Rules for this Project

For this project, we used a two-player Handjass, hoping to minimize the game's complexity. We also decided not to use Marriage and Melds, as these scores are obtained randomly and do not contribute to learning a game strategy. We also decided not to implement the 5 scores for the last-round winner to minimize the programming effort. If this project finds a successful game strategy, the remaining game elements can be added in a later version.

2 Descriptive Statistics

The number of different possible hands is:

$$N = \binom{36}{9} = 94'143'280$$

Using all combinations in a reasonable time for training will not be feasible because the computational cost can not be satisfied with the hardware at hand. $1'000'000$ hands were generated to be used for the descriptive statistics.

As expected, the cards are completely random, and no pattern can be seen. (Figure 2.1)

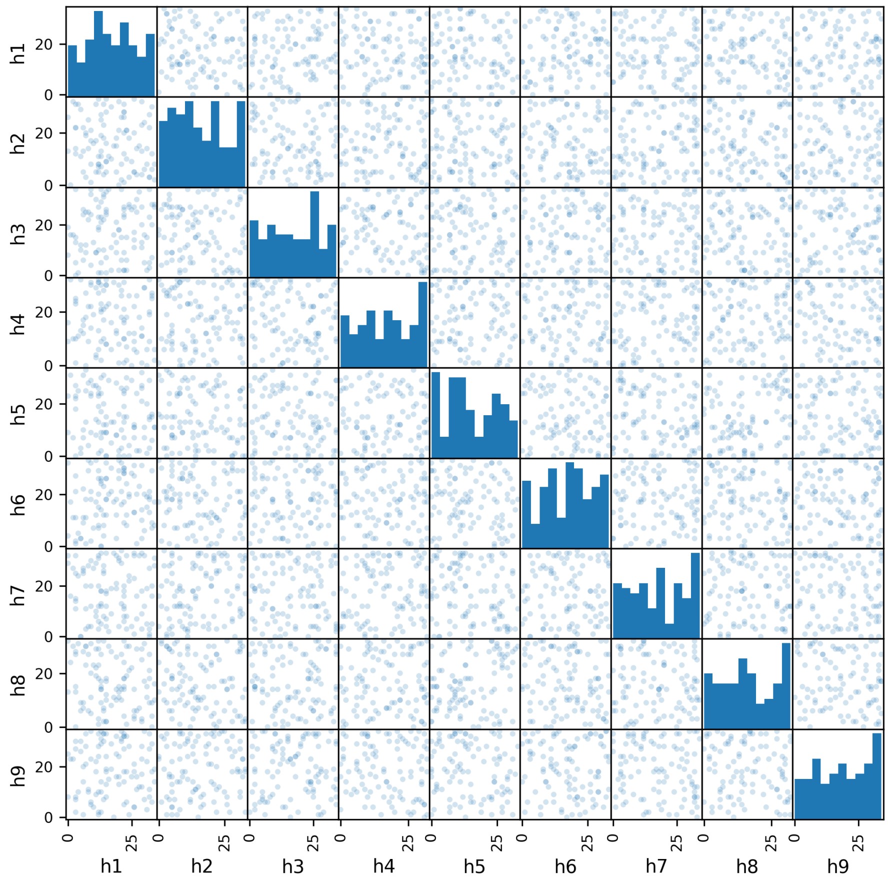


FIGURE 2.1: Scatter Matrix of $1'000'000$ hands

3 Reinforcement Learning

Reinforcement learning (RL) is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning. [4]

For this project, I was looking for an RL example that is well-documented so it can be customized for the project's needs. The choice fell on the article "Playing Cards with Reinforcement Learning" [5], which describes Reinforcement Learning and the principles behind it; also, the Jupyter Notebook is provided [6]. The article uses the game Easy 21 (a variant of BlackJack) to explain the basics of reinforcement learning and three approaches for estimating the best playing strategy.

The goal of the following algorithms is to learn the Q value function. The Q value is the estimated potential reward in a given game situation.

3.1 Monte Carlo Control (MCC)

Monte-Carlo means that we are going to play game sequences (play a round of Jass). The **Control** part means that we are going to find the optimal policy (best action to pick to maximize our winning chances). After each round, the Q value gets updated. If the played game was a win (resp. loss), the values of the state-action pair should be increased (resp. decreased)

3.2 SARSA(λ)

SARSA is the acronym of State, Action, Revard, State', Action', which is related to the way the algorithm updates the Q value function. The λ is a model's parameter called the trace factor, scaling from 0 to 1. If λ=1, we have the classic MCC approach, and if λ=0, it is a Temporal-Difference learning update. This means we adjust our current Q value to better match a future Q value estimation.

3.3 Value Approximation (VA)

Value Approximation means that we are going to approximate the Q value function rather than computing it for all the possible state-action pairs.

3.4 Adjustments for the JassBot

For the JassBot, some adjustments to the Easy 21 example were necessary. First of all, a "new" game engine (enviroment) had to be written which could interact with the RL algorithms. The number of possible actions was not changed because in Jass (as well as in Easy 21) you can only take two actions (take the trick or not take the trick). The code for the RL algorithms had to be adapted only slightly and could be copied as far as possible from the example. The value approximation was not pursued further since different typical game situations must be represented. Because of the more than 90 million possible hand combinations, hand coding of significant game situations did not seem purposeful.

3.4.1 Game Environment

The game environment implements the Jass mechanisms. Here the deck of cards and the two players are created.
A game lasts until one player has reached 7 points. A game consists of several rounds. A round consists of 9 cards per player (All nine cards in the hand are played). Different functions are implemented (see 3.4.2) to decide which card to play.

3.4.2 Agents and Actions

The two players are represented by an AGENT, and this agent can choose different functions for selecting a card to play, depending on the situation.

For the RL of the first player, two actions are available (take: take the trick or leave: leave the cards on the table/let the other player win the cards).

- take will always play the strongest card possible
- leave will always play the weakest card possible

For the second player, no RL is used, here the strategy can be defined once in the code, which will always be used (e.g., random, probabilistic, card selection based on the previously learned Q values)

3.4.3 Card encoding

With different encodings, it was examined whether the learning success can be improved if more information is encoded in the numbers.
Several methods have been implemented for encoding the cards/game states:

- simple: each card has a fixed number (the Trump cards have the highest numbers).
- linear numbering: from weak (0) to strong (26)
- card strength: Different cards can have the same value
- digits1: see Figure 3.1a
- digits2: see Figure 3.1b

```
"""
first digit:                second digit:
------------                -------------
1 if Rose                   0 if card 6
2 if Eichel                 1 if card 7
3 if Schellen               2 if card 8
4 if Schilten               3 if card 9
                            4 if card Banner
                            5 if card Under
                            6 if card Ober
                            7 if card König
                            8 if card Ass

third digit:
------------
0 if card suit an played card suit doesn't march
1 if played card is None (we play first)
2 if card suit and played suit match
3 if card suit is trump suit and trump is played
4 if card suit is trump suit
```

(A) Digits1

```
"""                             third digit:
first digit:                    -----------
-----------                     0 if we play first
1 if Rose                       1 if no trump was played
2 if Eichel                     2 if trump was played
3 if Schellen
4 if Schilten                   fourth digit:
                                ------------
second digit:                   0 if player 2 takes the trick
------------                    1 if take the trick
0 if card 6                     2 if play suit
1 if card 7                     3 if discard
2 if card 8                     4 if we played first
3 if card 9                     """
4 if card Banner
5 if card Under
6 if card Ober
7 if card König
8 if card Ass
```

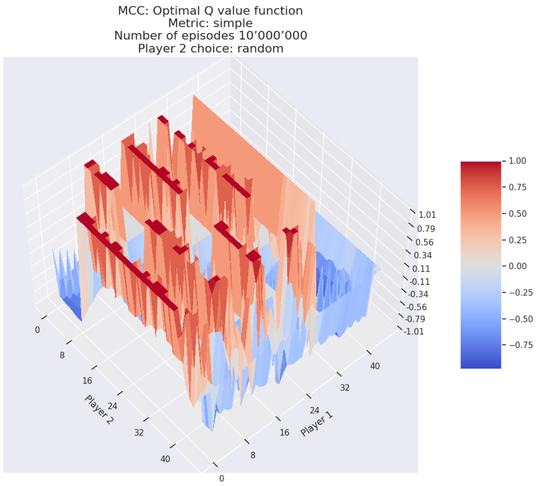
(B) Digits2

FIGURE 3.1: Card encoding

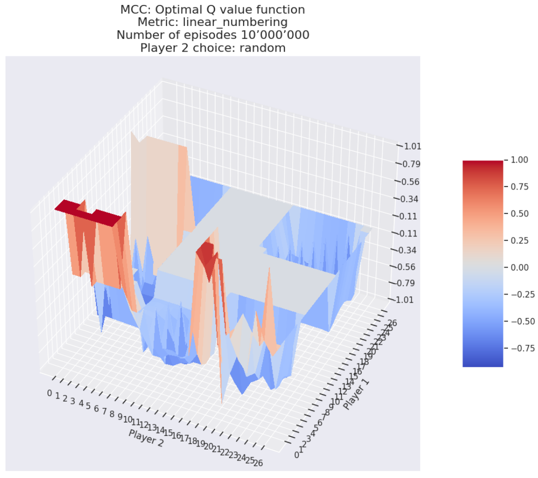
4 Results

4.1 Monte Carlo Controll (MCC

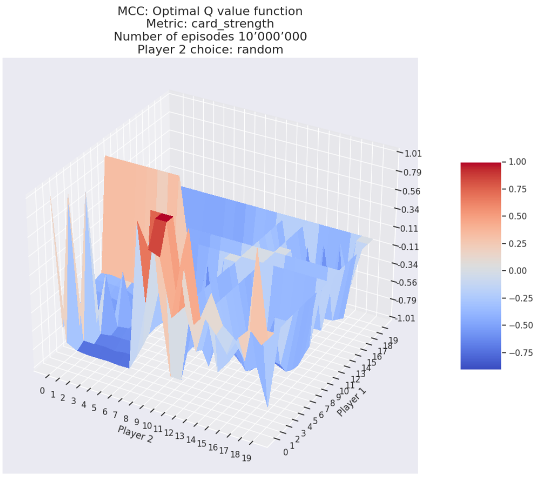
asrffg



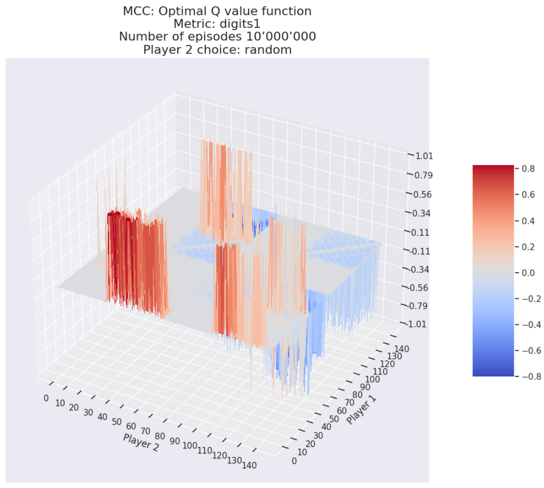
(A) Simple



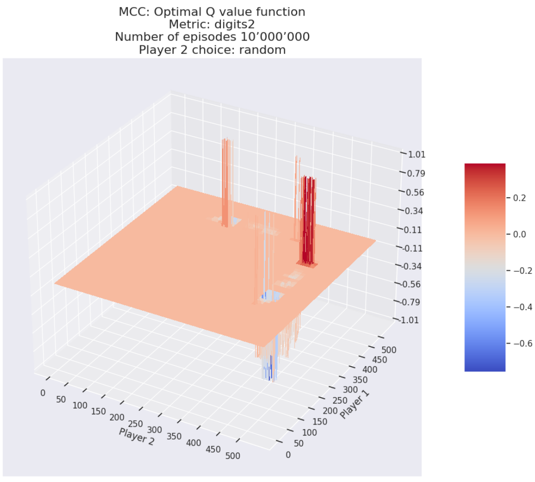
(B) Linear Numbering



(C) Card Strength



(D) Digits1



(E) Digits2

FIGURE 4.1: Monte Carlo Control: Q value function with different card encodings

4.2 SARSA(λ)

asrffg

5 Conclusion

The two ML algorithms (MCC, SARSA) could not achieve any meaningful learning success, or more precisely, the learned Q-Value function does only achive a 1% better result than a randomly played card. A simple approach (always take the trick if possible) shows better results (XXX win rate).

Our considerations to encode the game states were not successful.

- One possible reason is that "only" two actions are available (take, leave). However, it cannot be guaranteed that the action can be executed (e.g., a take action cannot have a card in hand to take the trick).
- The changing of the dealer (the winner of the last round plays the first card in the current round) leads to the fact that the actions do not reflect the actual state of the game because the player who deals can only partially judge whether an action (take/leave) is possible.
- If the desired action was impossible, a playable card was randomly selected. This should have been considered a learning termination (terminal).

The example of Easy 21 was not transversally for Jass, which allows more gaming states. It should be possible to consider more than one game state:

- Consideration of trump suit
- Who plays the first card in a round (for the first and second players, there are two different strategies to learn).
- Does the player have the cards to decide if he can take or leave? Only if both possibilities exist can something be learned about the game strategy (take/leave).

Acknowledgements

I want to take this opportunity to thank everyone who accompanied me during the CAS Applied Data Science.

Special thanks go to Sigve Haug and Mykhailo Vladymyrov, with whom we often had exciting conversations about all and everything and, of course, also in professional matters.

Furthermore, I would like to thank all the other lecturers/referees who always actively supported us and patiently answered our questions.

Another thank goes to my fellow students; seeing and exchanging ideas with them was always a pleasure.

Last but not least, I would like to thank Matyas Amrouche for writing the article *"Playing cards with Reinforcement Learning"* [6] and providing the JupyterNotebook [5].

Raphael Ziegler, JassBot

Reference

[1] Jass. *in Wikipedia*. URL: `https://en.wikipedia.org/wiki/Jass` (visited on 05/24/2023).

[2] Brünig-Napf-Reuss line. *in Wikipedia*. URL: `https://en.wikipedia.org/wiki/Br%C3%BCnig-Napf-Reuss_line` (visited on 03/14/2023).

[3] Handjass. URL: `https://www.pagat.com/jass/handjass.html` (visited on 05/28/2023).

[4] Reinforcement learning. *in Wikipedia*. URL: `https://en.wikipedia.org/wiki/Reinforcement_learning` (visited on 05/18/2023).

[5] Matyas Amrouche. *Playing cards with Reinforcement Learning*. URL: `https://towardsdatascience.com/playing-cards-with-reinforcement-learning-1-3-c2dbabcf1df0` (visited on 03/27/2023).

[6] Matyas Amrouche. *Github/Easy 21*. URL: `https://github.com/Matyyas/Easy21` (visited on 03/27/2023).

List of Figures

List of Tables

List of Abbreviations

ML	Machine Learning
RL	Reinforcement Learning
MCC	Monte Carlo Control
SARSA	State, Action, Revard, State', Action'