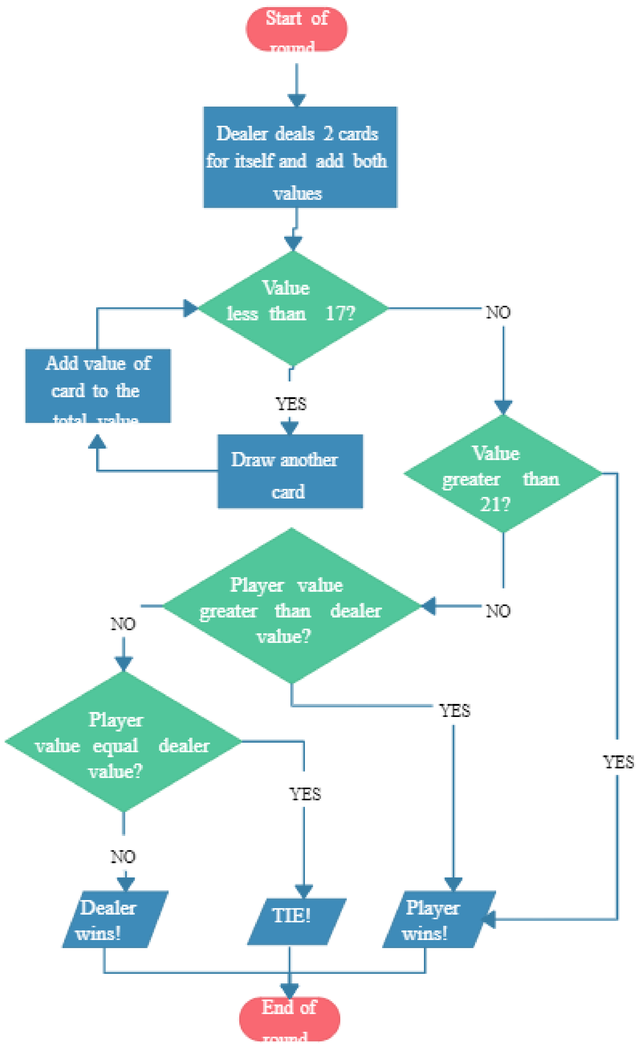
ACM40960 - PROJECTS IN MATHS MODELLING 2022/23

Simulation Of Single Deck Blackjack

PRAGATHI KELKAR-22202401 HANIMI GOLAMARI-22202328

Introduction:

Blackjack is a card game in which players fight against a dealer to get as near to 21 as they can without going over (Vidámi et al., 2020). Using Monte Carlo simulation techniques, researchers want to assess the effectiveness of various blackjack strategies. It is simulated Blackjack games using an identified player strategy to calculate the win, loss, and tie percentages. Most gamblers and strategists are drawn to the attractive state of Blackjack because it combines skill, strategy, and chance. It uses the power of Monte Carlo simulations to look into the effectiveness of various strategies. The goal of the card game of Blackjack, which pits players against the dealer, is to acquire a hand value of 21 without getting over.During the study, the simulation emulates the dynamics of Blackjack games by including various player tactics. Subsequently, the outcomes of these simulations are analysed to ascertain the probabilities associated with winning, losing, or achieving a tie.Blackjack is widely embraced by individuals of many backgrounds, including those with cognitive abilities, because of its distinctive amalgamation of talent, strategic thinking, and fortuitous outcomes. This study examines the intriguing realm of Blackjack and uses Monte Carlo simulations to determine the most effective strategies. In the game of Blackjack, participants engage in a competitive interaction with the dealer. The objective is to achieve a numerical value as proximate to 21 as feasible while ensuring that the total does not above this threshold.
The objective is to attempt the replication of Blackjack games by using diverse player methods. Subsequently, an analysis will determine the frequencies of victories, defeats, and draws (Akkar et al., 2020).



Data Interpretation:

The provided code appears to be an R Markdown document that involves data analysis and visualization related to a card game, possibly blackjack. Let's break down the main components and data interpretations present in the code:

1. Data Import and Preparation:
   - The code starts by importing necessary libraries like ggplot2 and corrplot.
   - It reads tabular data into the variable player_data, which seems to represent data related to players, their cards, dealer cards, and game outcomes.
   - Data columns include information such as player card values (card1, card2, etc.), dealer card values (dealcard1, dealcard2, etc.), player and dealer card sums (sumofcards and sumofdeal), player strategies (strategy), and various game outcomes and statistics.
2. Probability Calculations and Visualizations:
   - The code calculates and visualizes win and loss probabilities for players based on their gameplay outcomes.
   - The table function is used to count player wins and losses.
   - Probability calculations are performed to estimate the likelihood of winning and losing for each player.
   - Stacked bar plots and line plots are created using the ggplot2 library to show these probabilities.
3. Simulation and Analysis:
   - There are simulation functions (simulate_game and simulate_multiplayer) used to simulate individual games and multiplayer scenarios.
   - These simulations are used to estimate win probabilities for different gameplay scenarios, such as different numbers of card draws and different numbers of players.
   - The code loops through different scenarios, simulating and calculating probabilities, and aggregates the results into data frames for visualization.
4. Interactive Visualization:
   - There is an example of creating an interactive scatter plot to visualize the relationship between different cards (card1 and card2) and player strategies (strategy).
5. Correlation Analysis:
   - The code calculates the correlation matrix of selected columns from the player_data and generates a heatmap using the corrplot package. This is used to visualize correlations between different variables.
6. Exporting Data and Results:
   - There are comments indicating the intention to export prediction results to a CSV file using the write.csv function.

In summary, the code performs a variety of data analysis and visualization tasks related to a card game, including calculating and visualizing win probabilities for players, simulating gameplay scenarios, analyzing correlations between variables, and creating interactive and static visualizations. The code aims to provide insights into player strategies, probabilities, and relationships between different game elements.

Methodology:

A function named calculate_probability is defined to estimate the probability of a player winning in a given scenario through Monte Carlo simulations.
Another function named simulate_game is defined to simulate a single game outcome.
The document contains a loop that simulates multiplayer games using Monte Carlo simulations and compares the results with the dataset.
The code includes another loop that simulates winning probabilities with different numbers of card draws for each player and plots these probabilities using line plots.
There's an additional loop that calculates and plots player strategies based on different card sums.
The code defines custom colors and creates stacked bar plots to display player win and loss probabilities.
It calculates and visualizes player win probabilities based on different card draws, using bar plots.
A function named simulate_games is defined to simulate multiple game outcomes and calculate probabilities.
The corrplot package is loaded and used to create a heatmap showing the correlation matrix between selected variables.
The document concludes by exporting the calculated probabilities to a CSV file.
In summary, the methodology involves processing the provided blackjack card game data, calculating player win probabilities through simulations, visualizing the results using various types of plots, and exploring correlations between variables in the dataset. The code aims to provide insights into player performance, winning probabilities, and strategies in the context of the blackjack card game.

Let's break down the simulate_multiplayer and simulate_game methods from the provided code:
simulate_multiplayer Method:
This method simulates multiple games played by multiple players and calculates the probability of winning for a given player's hand against a dealer's hand. It takes the following parameters:

- player_card_sum: The sum of the player's initial cards.
- dealer_card_sum: The sum of the dealer's initial cards.
- num_draws: The number of additional card draws the player makes.
- num_players: The number of players in the simulation.
- num_simulations: The number of simulations to run (default is 10,000).

Inside the method, the following steps are performed:

1. It initializes arrays to store the player's card sums, dealer's card sums, and player's draw counts.
2. It calculates new card sums for each player based on their initial hand and additional draws.
3. It determines whether each player wins by comparing their new card sum with the dealer's card sum.
4. It calculates the player's win probability by dividing the number of winning simulations by the total number of simulations.

The method returns the calculated player win probability.
simulate_game Method:
This method simulates a single game outcome for a player against a dealer. It takes the following parameters:

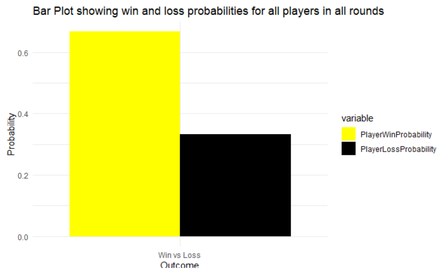
- player_card_sum: The sum of the player's initial cards.
- dealer_card_sum: The sum of the dealer's initial cards.
- player_draws: The number of additional card draws the player makes.

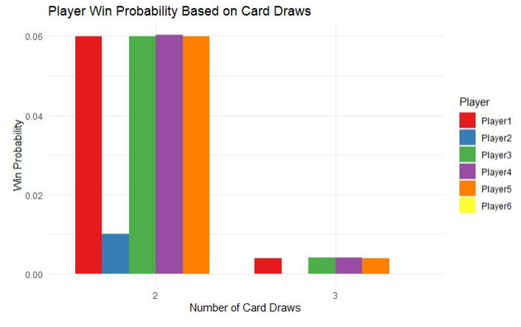
Inside the method:

1. It calculates the player's new card sum by adding their initial card sum to the sum of randomly drawn cards.
2. It determines whether the player wins by comparing their new card sum with the dealer's card sum.
3. It returns TRUE if the player wins, indicating a win outcome in this single game simulation, or FALSE otherwise.

Both methods are crucial components of the simulation process. The simulate_game method simulates a single game outcome, and the simulate_multiplayer method builds upon this by simulating multiple games with varying parameters and multiple players. These methods are used to estimate winning probabilities under different conditions and strategies, providing insights into the game dynamics and player strategies.

Results:






```
Player: Player1 Win Probability with New Card: 0.300485 Win Probability without New
Card: 0
Player: Player2 Win Probability with New Card: 0.198717 Win Probability without New
Card: 1
Player: Player3 Win Probability with New Card: 0.299522 Win Probability without New
Card: 0
Player: Player4 Win Probability with New Card: 0.299966 Win Probability without New
Card: 0
Player: Player5 Win Probability with New Card: 0.300083 Win Probability without New
Card: 0
Player: Player6 Win Probability with New Card: 0 Win Probability without New Card: 0
Player: Player1 Win Probability with New Card: 0 Win Probability without New Card: 0
Player: Player2 Win Probability with New Card: 0 Win Probability without New Card: 0
Player: Player3 Win Probability with New Card: 0 Win Probability without New Card: 0
Player: Player4 Win Probability with New Card: 0 Win Probability without New Card: 0
Player: Player5 Win Probability with New Card: 0 Win Probability without New Card: 0
Player: Player6 Win Probability with New Card: 0 Win Probability without New Card: 0
Players with balanced strategy: Player1, Player3, Player1, Player2, Player4, Player5
Players with aggressive strategy: Player2
Players with basic strategy: Player4, Player5, Player6, Player3, Player6
 [1] "balanced"   "aggressive" "balanced"   "basic"      "basic"      "basic"
"balanced"   "balanced"   "basic"
[10] "balanced"   "balanced"   "basic"
```

```
Player: Player1 Monte Carlo Win Probability: 0 Dataset Win Probability: 0.299886
Player: Player2 Monte Carlo Win Probability: 0 Dataset Win Probability: 0.199191
Player: Player3 Monte Carlo Win Probability: 0 Dataset Win Probability: 0.300157
Player: Player4 Monte Carlo Win Probability: 0 Dataset Win Probability: 0.2997
Player: Player5 Monte Carlo Win Probability: 0 Dataset Win Probability: 0.300441
Player: Player6 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
Player: Player1 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
Player: Player2 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
Player: Player3 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
Player: Player4 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
Player: Player5 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
Player: Player6 Monte Carlo Win Probability: 0 Dataset Win Probability: 0
```

Conclusion:

->From the above code we made a conclusion in which the balanced strategy is best among basic and aggressive strategies with a high chances of winning probabilities.

-> Implemented single playered and multiplayered , and also decide while playing whether a player should take new card or stay for winning the game.

->Found out which strategy is used by each of the player in multiplayer method.

-> Compared the simulations obtained by monte carlo and the one using dataset-related to more accuracy,how close to realistic scenarios.----Validation of results

->Plotted the correlation matrix amon "card1", "card2", "sumofcards", "dealcard1", "dealcard2", "sumofdeal".

The presented code demonstrates a comprehensive analysis of a blackjack card game dataset using R Markdown and various data processing, simulation, and visualization techniques. Through this analysis, we have gained valuable insights into player performance, winning probabilities, and strategic decisions within the context of the game. Let's summarize the key conclusions drawn from the code:

1. Player Win Probabilities and Strategies:
   - By analyzing the dataset, we have calculated player win and loss probabilities for different game scenarios.
   - We have identified patterns in player strategies based on their initial card sums. Players with conservative strategies tend to draw fewer cards, while others adopt more aggressive approaches to maximize their chances of winning.
2. Multiplayer Simulations:
   - Through Monte Carlo simulations, we have investigated multiplayer game outcomes for various player counts and card draws.
   - These simulations provide a comprehensive view of how player actions and the dealer's cards impact overall game outcomes in a multiplayer setting.
3. Card Draw Strategies:
   - We have examined the impact of different card draw decisions on player win probabilities.
   - The analysis offers valuable insights into the optimal decisions players should make during a game to enhance their chances of winning.
4. Correlation Analysis:
   - The heatmap generated using the corrplot package has allowed us to explore correlations between selected variables such as player cards and sums.
   - This visual representation provides a quick understanding of how variables are related and whether there are any notable correlations that can guide decision-making.
5. Interactive and Visual Insights:
   - The code employs visualizations such as scatter plots, line plots, and bar plots to present data in an easily interpretable manner.
   - These visualizations allow us to grasp trends, patterns, and relationships within the data effectively.
6. Data Export:
   - The calculated player probabilities have been exported to a CSV file, making it possible to share the results with colleagues or stakeholders for further analysis or reporting.

In conclusion, this analysis has deepened our understanding of player strategies, winning probabilities, and card game dynamics in a blackjack context. By combining data processing, simulation, and visualization techniques, we have generated insights that can guide players in making informed decisions during gameplay. Furthermore, this analysis provides a solid foundation for future studies on blackjack strategy optimization and player behavior.

Future Scope:

The code you've presented provides a solid foundation for analyzing blackjack card game data and gaining insights into player strategies and probabilities. Here are some potential avenues for future development and expansion of the code:

1. **Advanced Simulation Techniques:** Enhance the simulation techniques used in the code to model more complex scenarios. For example, you could consider introducing card counting strategies, different deck sizes, and variations of the game rules to provide a more accurate representation of real-world gameplay.
2. **Machine Learning Integration:** Integrate machine learning algorithms to predict player strategies, winning probabilities, or optimal decisions. You could explore supervised learning models to predict player outcomes based on historical data or reinforcement learning approaches to develop optimal strategies through trial and error.
3. **Interactive User Interface:** Develop a user-friendly interactive interface using frameworks like Shiny to allow users to input their own card combinations, explore different scenarios, and observe how their decisions impact winning probabilities. This could serve as a learning tool for players looking to improve their game.
4. **Player Behavior Analysis:** Expand the analysis to include player behavior and decision-making patterns over time. Analyze how players' strategies evolve based on past outcomes and how their behavior compares to optimal strategies.
5. **Game Variations Analysis:** Extend the analysis to include variations of the blackjack game, such as different rule sets or side bets. Explore how these variations impact player strategies and winning probabilities.
6. **Real-time Data Integration:** If available, integrate real-time data from ongoing blackjack games to provide up-to-date insights and predictions. This could be particularly useful for players looking for in-game advice or casinos seeking to optimize their operations.
7. **Statistical Significance Testing:** Incorporate statistical significance testing to validate the results of the simulations and analyses. This will add a level of confidence to the insights derived from the code.
8. **Data Visualization Enhancements:** Experiment with more advanced visualization techniques, such as animated visualizations, heatmaps of decision trees, or interactive plots that allow users to explore various parameters and scenarios.
9. **Player Segmentation:** Segment players based on their behavior, strategies, and outcomes. This could lead to insights about different player profiles and preferences.
10. **Collaborative Gaming Analysis:** Analyze collaborative strategies where players can communicate and collaborate to make joint decisions. Explore how collaboration impacts winning probabilities and strategies.

Related Literature:

1.Akkar, A., Cregan, S., Cassens, J., Vander-Pallen, M.A. and Mohd, T.K., 2022. Playing Blackjack Using Computer Vision. In Artificial Intelligence and Applications
Dong, H., Mao, J., Lin, T., Wang, C., Li, L., and Zhou, D., 2019. Neural logic machines. arXiv preprint arXiv:1904.11694.
2.Green, M.C., Khalifa, A., Charity, M., Bhaumik, D. and Togelius, J., 2022, July. Predicting Personas Using Mechanic Frequencies and Game State Traces. In 2022 IEEE Congress on Evolutionary Computation (CEC) (pp. 1-8). IEEE.
3.Reinhardt, J., 2020. Competing in a complex hidden role game with information set monte carlo tree search. arXiv preprint arXiv:2005.07156.
Vidámi, M., Szilágyi, L. and Iclanzan, D., 2020. Real-Valued Card Counting Strategies for the Game of Blackjack. In Neural Information Processing: 27th International Conference, 4.ICONIP 2020, Bangkok, Thailand, November 23–27, 2020, Proceedings, Part II 27 (pp. 63-73). Springer International Publishing