

Table of Contents

BLACKJACK.....	2
DEPENDENCIES AND SETUP.....	2
Installing Dependencies From requirements.txt.....	2
Creating Conda Environment.....	3
CONFIGURATION.....	3
Config.yml.....	4
STRATEGIES.....	6
Betting Strategies.....	6
MINIMUM.....	6
MAXIMUM.....	6
UNIFORM.....	7
NORMAL.....	8
EXPONENTIAL.....	8
TRIANGULAR.....	9
Playing Strategies.....	9
STAND.....	9
16+.....	12
17+.....	12
18+.....	12
50/50.....	13
D+10.....	14
RANDOM.....	15
OTHER PLAYERS.....	15
RUNNING SIMULATION.....	17
Known Bugs Using Conda Environments.....	17
DASHBOARD.....	17
Player Strategies.....	18
Chip Totals Per Hand.....	18
Card Totals Distribution.....	19
Wins/Losses.....	20
Maximum Winnings Per Trial.....	20
Winnings/Losses Per Hand.....	21

BLACKJACK

If you are new to black jack or need a refresher on what the game is and how it is played, I recommend going here (<https://www.pokernews.com/casino/how-to-play-blackjack-beginners.htm>) and getting familiar with the basic rules and intuition of blackjack.

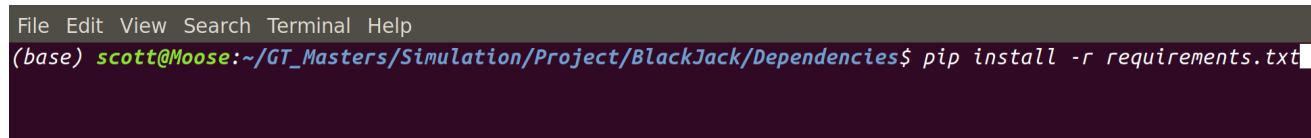
DEPENDENCIES AND SETUP

This BlackJack simulator was built utilizing Python 3.7 on an Ubuntu 18.04 Linux OS. It has not been tested on any other versions of Python. Therefore, it is recommended that a user create either a pip to virtual environment or conda virtual environment with the proper Python version. For your convenience, both a **requirements.txt** for pip virtual environment, and an **environment.yml** for a conda virtual environment has been provided for you to meet all the required dependencies. These items can be found in the Dependencies folder. It is also recommended that the user has either Google Chrome or Mozilla Firefox installed as well.

Installing Dependencies From requirements.txt

If you just want to use your base Python version, and install dependencies without dealing with virtual environments, you can use the requirements.txt in the Dependencies folder and simply pip install everything, then directly execute the simulation. You can do this by opening a terminal or command prompt and change working directory to the Dependencies directory in the BlackJack project folder. Then type

```
pip install -r requirements.txt
```

A screenshot of a terminal window with a dark background. The title bar at the top shows 'File Edit View Search Terminal Help'. The terminal text shows a prompt '(base) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack/Dependencies\$' followed by the command 'pip install -r requirements.txt' which has been executed, with a cursor at the end of the line.

```
File Edit View Search Terminal Help
(base) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack/Dependencies$ pip install -r requirements.txt
```

If you have multiple versions of Python installed on your system, you may need to explicitly state

```
pip3 install -r requirements.txt
```

Creating Conda Environment

In order to create a conda environment utilizing the yaml file be sure to have conda updated to the latest version by opening a terminal or command prompt. If a user is utilizing windows 10, it is recommended they run the shell in administrator mode. Then simply run:

conda update -n base -c defaults conda

```
File Edit View Search Terminal Help
(base) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ conda update -n base -c defaults conda
```

Next, from your terminal, change working directory to the Dependencies directory where the environment.yml file resides and type the following:

conda env create -n blackjack -f environment.yml

```
File Edit View Search Terminal Help
(base) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ conda env create -n blackjack -f environment.yml
```

This will create the conda environment called **blackjack**. After the environment has been created, you can activate it by typing the following:

conda activate blackjack

```
File Edit View Search Terminal Help
(base) scott@Moose:~$ conda activate blackjack
(blackjack) scott@Moose:~$
```

CONFIGURATION

In the main directory of the BlackJack simulation project folder exists a config.yml file. This file is where you will setup all the necessary parameters and playing strategies for your simulation. This is where you will start for all simulations.

```
(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ tree
```

```
.
├── config.yaml
├── Dependencies
│   ├── environment.yml
│   └── requirements.txt
├── Game.py
├── main.py
├── Strategies
│   ├── BettingStrategies
│   │   ├── betting.py
│   │   ├── __init__.py
│   │   └── __pycache__
│   │       ├── betting.cpython-37.pyc
│   │       └── __init__.cpython-37.pyc
│   ├── __init__.py
│   ├── PlayingStrategies
│   │   ├── __init__.py
│   │   ├── __pycache__
│   │   │   ├── __init__.cpython-37.pyc
│   │   │   └── strategies.cpython-37.pyc
│   │   └── strategies.py
│   └── __pycache__
│       └── __init__.cpython-37.pyc
├── User Manual
├── Utils
│   ├── CustomLogger.py
│   ├── dashboard.py
│   ├── __init__.py
│   ├── __pycache__
│   │   ├── CustomLogger.cpython-37.pyc
│   │   ├── dashboard.cpython-37.pyc
│   │   ├── __init__.cpython-37.pyc
│   │   └── utils.cpython-37.pyc
│   └── utils.py
```

Config.yml

Within the config.yml file you will find the following

Parameter	Values	Description
RANDOM_SEED	Any Valid Integer	Sets a random seed generator for statistical repeatability
NUMBER_OF_TRIALS	Any Integer > 0	How many times to repeat the simulations
NUMBER_OF_DECKS	{1,2,3,4,5,6,7,8}	The number of standard 52 card decks to be used at the table. 1 is the minimum, resulting in a card shoe of 52 cards to deal from. 8 is maximum, resulting in a card shoe of 416 cards to deal from.

SHOE_CUT_PERCENTAGE	$0.0 \leq X \leq 1.0$	This specifies what percentage of cards must be dealt from the card shoe before the dealer reshuffles the deck. For example, a value of 0.75 means that for a shoe utilizing 8 decks of cards, 312 cards will be dealt before all cards are shuffled and reintroduced to the shoe to be dealt again.
TABLE_MIN_BET	$0 < X$	The required minimum bet to play at the table. Any valid integer greater than 0 is allowed
TABLE_MAX_BET	$\text{TABLE_MIN} \leq X$	The maximum allowable bet at the table. Any valid integer greater than TABLE_MIN_BET is allowed
PLAYER_NAME	<string>	The unique name to give your player at the table to disambiguate from any other players that may be sitting at the table
NUMBER_OF_HANDS	$0 < X$	The maximum number of hands to be dealt for each simulation trial. Each trial will progress until either the maximum hand has been reached or your player has gone broke. After either of these criteria have been met, everything will be reset and the next trial will commence
CHIPS	$0 < X$	The amount of chips every player sitting at the table will start with for each simulation trial
BETTING	See <i>Strategies</i> section	The betting strategy you wish to use for the simulation trials
PLAYING_STRATEGY	See <i>Strategies</i> section	The playing strategy you wish to use for the simulation trials
OTHER_PLAYERS	See <i>Other Players</i> section	Betting and Playing strategies for any active simulated players sitting at the table with your player.

STRATEGIES

Several betting and playing strategies have been created to explore various decision making schemes. Currently there are 6 different betting strategies, 3 of which have variable parameter inputs for further betting possibilities. Also, there are 7 different playing strategies. Between the variable parameter input betting strategies and 7 different playing strategies, the amount of possible combinations is too many to enumerate.

Betting Strategies

In this section we will go over in detail each of the 6 different betting strategies and how to configure them in the config.yml file

MINIMUM

When this betting strategy is utilized your player will only ever bet the table minimum (whatever you configure that to be). It is set in the config.yml as:

BETTING:
STRATEGY: MINIMUM

```
# Always Bet Minimum Strategy
#####
BETTING:
| STRATEGY: MINIMUM
```

MAXIMUM

When this betting strategy is utilized your player will only ever bet the table maximum (whatever you configure that to be). It is set in the config.yml as:

BETTING:
STRATEGY: MAXIMUM

```
# Always Bet Maximum Strategy
#####
BETTING:
| STRATEGY: MAXIMUM
```

UNIFORM

When this betting strategy is utilized, a random number will be sampled from uniform $[0,1]$ distribution. This value is then multiplied by the table maximum and applied as the player's bet. It is set in the config.yml as:

```
BETTING:  
  STRATEGY: UNIFORM
```

```
# Bet Uniform Random  
#####  
BETTING:  
| STRATEGY: UNIFORM
```

NORMAL

When this betting strategy is utilized, a random number will be sampled from a Normal distribution of user specified mean and standard deviation. The mean must be $0 \leq X \leq 1$. This number is then multiplied by the table maximum and applied as the player's bet. It is set in the config.yml as:

```
BETTING:  
  STRATEGY: NORMAL  
  MEAN: 0.5  
  STD: 0.15
```

```
# Normal Betting Strategy  
#####  
BETTING:  
  STRATEGY: NORMAL  
  MEAN: 0.5  
  STD: 0.15
```

EXPONENTIAL

When this betting strategy is utilized, a random number will be sampled from an Exponential distribution of user specified lambda. The lambda must be $0 \leq X \leq 1$. This number is then multiplied by the table maximum and applied as the player's bet. It is set in the config.yml as:

```
BETTING:  
  STRATEGY: EXPONENTIAL  
  LAMBDA: 0.5
```

```
# Exponential Betting Strategy  
#####  
BETTING:  
  STRATEGY: EXPONENTIAL  
  LAMBDA: 0.5
```


TRIANGULAR

When this betting strategy is utilized, a random number will be sampled from a Triangular distribution of user specified left threshold, median threshold and right threshold. The values must be $0 \leq \text{LEFT} < \text{MED} < \text{RIGHT}$. This number is then multiplied by the table maximum and applied as the player's bet. It is set in the config.yml as:

```
BETTING:  
  STRATEGY: TRIANGULAR  
  LEFT: 0.0  
  MED: 0.5  
  RIGHT: 1.0
```

```
# Triangular Betting Strategy  
#####  
BETTING:  
  STRATEGY: TRIANGULAR  
  LEFT: 0  
  MED: 0.5  
  RIGHT: 1.0
```

Playing Strategies

In this section we will go over in detail each of the 7 different playing strategies and how to configure them in the config.yml file

STAND

When this playing strategy is utilized, a player will always choose to stand after their 2 cards have been dealt. They will never choose to take a "hit". It is set in the config.yml as:

```
PLAYING_STRATEGY: STAND
```

```
# Always Stand Playing Strategy  
#####  
PLAYING_STRATEGY: STAND
```

16+

When this playing strategy is utilized, a player will always choose to “hit” if their card total is less than 16. For any total equal to or greater than 16, the player will elect to “stand”. It is set in the config.yml as:

PLAYING_STRATEGY: 16+

```
# Stand on 16 or Higher
#####
PLAYING_STRATEGY: 16+
```

17+

When this playing strategy is utilized, a player will always choose to “hit” if their card total is less than 17. For any total equal to or greater than 17, the player will elect to “stand”. It is set in the config.yml as:

PLAYING_STRATEGY: 17+

```
# Stand on 17 or Higher
#####
PLAYING_STRATEGY: 17+
```

18+

When this playing strategy is utilized, a player will always choose to “hit” if their card total is less than 18. For any total equal to or greater than 18, the player will elect to “stand”. It is set in the config.yml as:

PLAYING_STRATEGY: 18+

```
# Stand on 18 or Higher
#####
PLAYING_STRATEGY: 18+
```

50/50

When this playing strategy is utilized, a player will choose with 50/50 odds to either “hit” or “stand” regardless of whatever their current card total is. It is set in the config.yml as:

PLAYING_STRATEGY: 50/50

```
# 50/50 Chance Hit or Stand
#####
PLAYING_STRATEGY: 50/50
```

D+10

When this playing strategy is utilized, a player will attempt to beat whatever the dealer is currently showing + 10. E.g. after dealing, if the dealer is showing a 9, the player will choose to “hit” until their total is greater than or equal to 19. However, if a player is in possession of 4 cards, they will elect to stand so as to prevent any further chances of going bust. It is set in the config.yml as:

PLAYING_STRATEGY: D+10

```
# Dealer Show Plus 10
#####
PLAYING_STRATEGY: D+10
```

RANDOM

When this playing strategy is utilized, a player will randomly select one of the above playing strategies and utilize it for the entire simulation. It is set in the config.yml as:

PLAYING_STRATEGY: RANDOM

```
# Random
#####
PLAYING_STRATEGY: RANDOM
```

OTHER PLAYERS

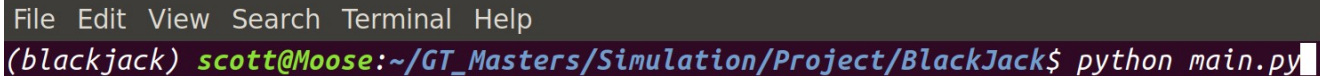
You have the capability to simulate the black jack games either by yourself or with up to 7 other players at the table. This induces more randomness as to which cards you may get dealt due to more cards being dealt out to other players. You can specifically set both the betting and playing strategies for all players or simply set them to RANDOM. All strategies, both betting and playing, are available to be used for “other” players sitting at the table. The image below shows how to configure a simulation with 4 other players at the table.

```
# Other players at the table and their strategies
# 7 Maximum other players
# Set to NONE if no other players desired
# OTHER_PLAYERS: NONE
OTHER_PLAYERS:
    ONE:
        BETTING:
            STRATEGY: RANDOM
        PLAYING_STRATEGY: RANDOM
    TWO:
        BETTING:
            STRATEGY: RANDOM
        PLAYING_STRATEGY: RANDOM
    THREE:
        BETTING:
            STRATEGY: RANDOM
        PLAYING_STRATEGY: RANDOM
    FOUR:
        BETTING:
            STRATEGY: RANDOM
        PLAYING_STRATEGY: RANDOM
# FIVE:
#     BETTING:
#         STRATEGY: MINIMUM
#     PLAYING_STRATEGY: RANDOM
# SIX:
#     BETTING:
#         STRATEGY: MINIMUM
#     PLAYING_STRATEGY: RANDOM
# SEVEN:
#     BETTING:
#         STRATEGY: MINIMUM
#     PLAYING_STRATEGY: RANDOM
```

RUNNING SIMULATION

After you have configured the config.yml with all the desired parameters and strategies, within the main directory of the BlackJack project folder exists a file called main.py. This file is what is called to run the simulation.

`python main.py`

A terminal window with a dark background and light text. The menu bar at the top shows 'File Edit View Search Terminal Help'. The prompt is '(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack\$' and the command 'python main.py' is entered.

```
File Edit View Search Terminal Help
(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ python main.py
```

Known Bugs Using Conda Environments

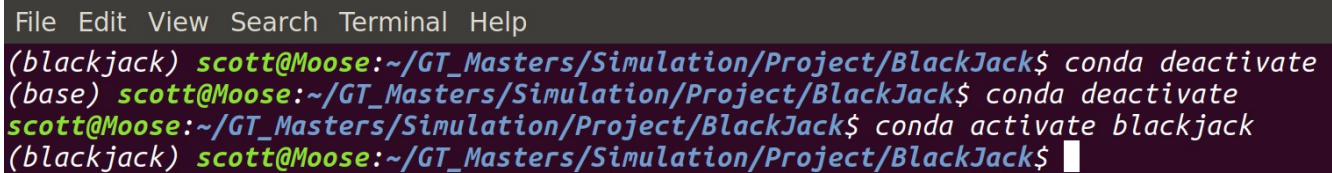
If for some reason you run the main.py file but get errors that dependencies are missing that should already be a part of your environment, keep calling

`conda deactivate`

until even your base environment disappears, then call:

`conda activate`

again.

A terminal window with a dark background and light text. The menu bar at the top shows 'File Edit View Search Terminal Help'. The prompt is '(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack\$' and the command 'conda deactivate' is entered. The prompt changes to '(base) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack\$' and the command 'conda deactivate' is entered. The prompt changes back to 'scott@Moose:~/GT_Masters/Simulation/Project/BlackJack\$' and the command 'conda activate blackjack' is entered. The prompt changes back to '(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack\$' and the command 'python main.py' is entered.

```
File Edit View Search Terminal Help
(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ conda deactivate
(base) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ conda deactivate
scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ conda activate blackjack
(blackjack) scott@Moose:~/GT_Masters/Simulation/Project/BlackJack$ python main.py
```

Also note that if at any time the user is warned that a specific library is missing, even though it should have been installed with all dependencies, they can simply call `pip install` themselves for that specific instance.

DASHBOARD

Once you execute the main script, you should see some output to std out scroll across the terminal. This will continue until all trials have finished. Once the simulation is complete, a web based Dashboard application should open in your web browser. Mozilla Firefox or Google Chrome is strongly recommended. If for whatever reason your browser doesn't automatically open, you can manually navigate to the app by opening your browser and typing in the URL search bar.

`http://127.0.0.1:8050/`

This dashboard will show you results and stats accumulated for all trials. Once you are done with the Dashboard, to fully terminate the program, in your terminal type:

Ctrl + c

For the rest of this section, the different tabs available in the Dashboard will be discussed

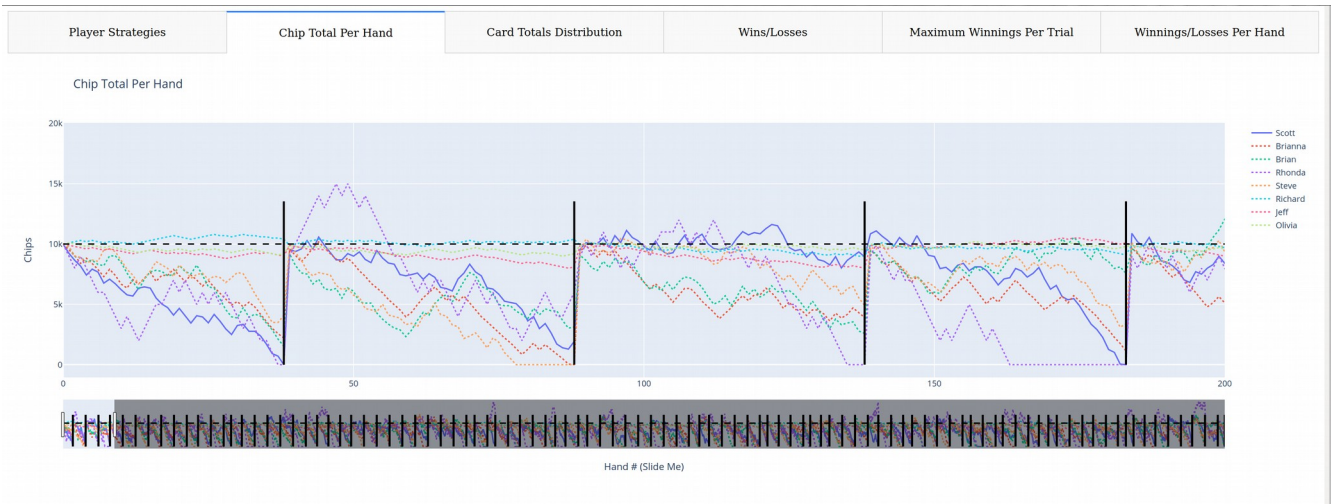
Player Strategies



PLAYER STRATEGIES shows the betting and playing strategies for every player sitting at the table. Your player will always be the first tile on the tab.

This graphic is a summary table of all the players you configured the simulation with and what their respective betting and playing strategy was. Your player will always be the first tile on the page.

Chip Totals Per Hand



This graphic shows the chip count for each player after each hand. The trials are separated by the black vertical lines. The horizontal dashed line represents the starting chip amount for each player. In instances that curves are above the dashed line, this means the player has positive winnings. Anytime it is below, this signifies a player has negative winnings (losing money). You can slide the ruler at the bottom to shift the graph to focus on specific trials. Double click on the slider to move as is, or move the individual thresholds. You can hide plotted curves by clicking a players name in the legend.

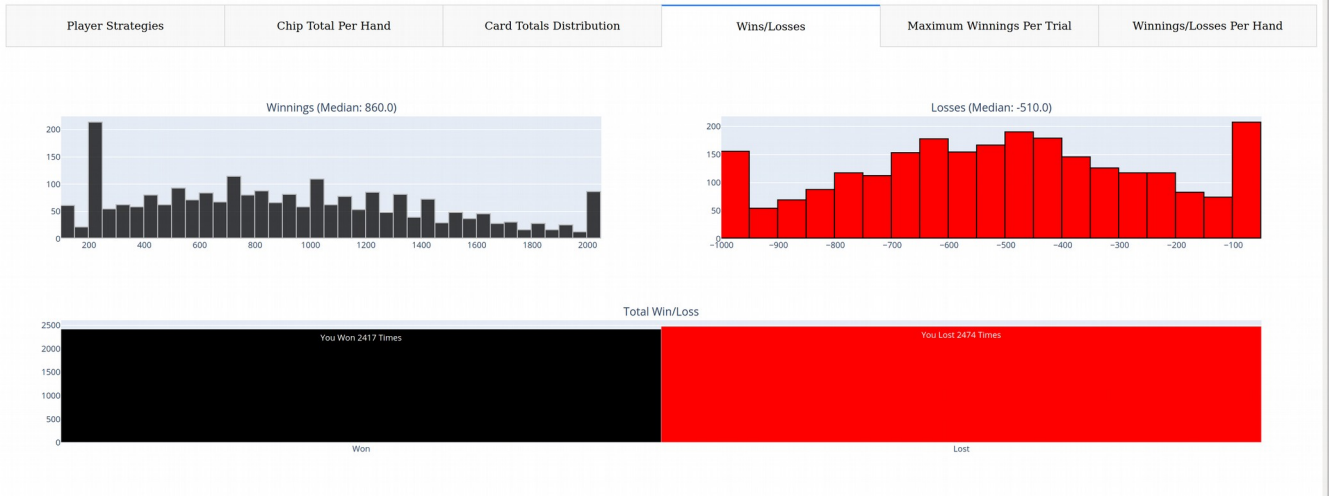


Card Totals Distribution



This graphic shows the frequencies of card totals seen throughout all hands of all trials for your player. This allows a user analyze how often they are busting. Conversely, it allows a user to analyze how close they are getting to 21 with the current playing strategy, as well as how often they do it.

Wins/Losses



This graphic shows the distributions of the amount of chips won when their player does win vs the distribution of the amount of chips lost when their player loses. It also shows for the given strategy, how many times they lost and won for all hands under all trials

Maximum Winnings Per Trial



This graphic shows the distribution of the maximum amount of winnings achieved at any time during a trial, for all trials. This gives a user an idea of the overall maximum amount of winnings a person could walk away from the table with for the given betting and playing strategy.

Winnings/Losses Per Hand



This graphic shows the amount won or lost for every single hand played across all trials. You can slide the ruler at the bottom to shift the graph to focus on specific trials. Double click on the slider to move as is, or move the individual thresholds.