Applying Reinforcement Learning to “Play” Horse Racing

Kam Fung CHEUNG, Ka Yui Or supervised by Prof. Michael Lyu
Outline

- Introduction (Motivation, Background, Objective)
- Data of Horse Racing (Collection, Description, Analysis, Preprocess)
- XGBoost (Reason using it, Progress, Result)
- Reinforcement Learning (A simple game, Horse Racing)
- Conclusion (Discussion, Problem, Future)
Introduction (Motivation)

- Reinforcement learning becomes popular in different areas especially gaming
- Horse racing and the related gambling is very popular and famous in Hong Kong
- Combine them to find out more ways to apply reinforcement learning and may make great profit
Introduction (Background)

- Reinforcement learning is one kind of machine learning and it is about how the agent learns to take actions in the environment to get the maximum reward.
- At this moment, we focus on choosing the winning horse in the race.
- Using reinforcement learning in horse racing means that the agent learns how to bet in horse racing to get the largest profit at the end.
Introduction (Objective)

- Our objective is to build a model to place the bet on the winning horse.
- The model gambles like a human. This means it can know when to bet and not to bet.
- Then, we want the model to gamble better than human beings.
Data (Collection)

- Use Python Beautifulsoup to do web scraping
- The horse racing data is collected from Hong Kong Jockey Club
- The weather data comes from Hong Kong Observatory
Data (Description)

- 3890 race records from 2014 to 2020 (End of 2019/2020 season)
- 2771 horse data
- 3890 weather data for each race day
- 47614 dataset for each horse in each race
## Data (Description)

- **Race data**

<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
<th>Type</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>race_date</td>
<td>The date of the race</td>
<td>Index</td>
<td>/</td>
</tr>
<tr>
<td>race_no</td>
<td>The number of a race in a day</td>
<td>Index</td>
<td>/</td>
</tr>
<tr>
<td>race_index</td>
<td>Unique id of the race</td>
<td>Index</td>
<td>/</td>
</tr>
<tr>
<td>location</td>
<td>Location of the race</td>
<td>Categorical</td>
<td>HV, ST</td>
</tr>
<tr>
<td>class</td>
<td>Class of the horses</td>
<td>Categorical</td>
<td>Class 1 to 5, Group 1 to 3</td>
</tr>
<tr>
<td>race_length</td>
<td>Distance of the race</td>
<td>Categorical</td>
<td>1000, 1200, 1400, 1600, 1650, 1800, 2000, 2200, 2400</td>
</tr>
<tr>
<td>course</td>
<td>Track of the race</td>
<td>Categorical</td>
<td>A, A+3, B, B+2, C, C+3</td>
</tr>
<tr>
<td>draw</td>
<td>Draw of the horse in a race</td>
<td>Categorical</td>
<td>14 distinct values, FAST, SLOW, WET FAST, WET SLOW, FIRM, GOOD TO FIRM, GOOD, GOOD TO YIELDING, YIELDING,</td>
</tr>
<tr>
<td>going</td>
<td>Condition of the track</td>
<td>Categorical</td>
<td></td>
</tr>
<tr>
<td>horse_id</td>
<td>Unique id of the horse</td>
<td>Categorical</td>
<td>2744 distinct values</td>
</tr>
<tr>
<td>jockey_name</td>
<td>Unique id of jockey</td>
<td>Categorical</td>
<td>113 distinct values</td>
</tr>
<tr>
<td>trainer_name</td>
<td>Unique id of trainer</td>
<td>Categorical</td>
<td>112 distinct values</td>
</tr>
<tr>
<td>actual_weight</td>
<td>Weight added to the horse</td>
<td>Real value</td>
<td>/</td>
</tr>
<tr>
<td>declared_horse_weight</td>
<td>Weight of the horse</td>
<td>Real value</td>
<td>/</td>
</tr>
<tr>
<td>win_odds</td>
<td>The odds of betting the horse</td>
<td>Real value</td>
<td>/</td>
</tr>
<tr>
<td>place</td>
<td>The final place of the horse in a race</td>
<td>Categorical</td>
<td>14 distinct values</td>
</tr>
<tr>
<td>finish_time_sec</td>
<td>Finishing time of the horse in a race</td>
<td>Real value</td>
<td>(Seconds)</td>
</tr>
<tr>
<td>Features</td>
<td>Description</td>
<td>Type</td>
<td>Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------------</td>
<td>-----------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>last_actual_weight</td>
<td>The actual weight of last race</td>
<td>Real Value</td>
<td>/</td>
</tr>
<tr>
<td>last_declared_horse_weight</td>
<td>The last weight in last race</td>
<td>Real Value</td>
<td>/</td>
</tr>
<tr>
<td>diff_actual_weight</td>
<td>Difference actual weight between present race and last race</td>
<td>Real Value</td>
<td>/</td>
</tr>
<tr>
<td>diff_declared_horse_weight</td>
<td>Difference declared weight between present race and last race</td>
<td>Real Value</td>
<td>/</td>
</tr>
<tr>
<td>country</td>
<td>The country of the horse</td>
<td>Categorical</td>
<td>US, AUS, etc.</td>
</tr>
<tr>
<td>age</td>
<td>The age of the horse</td>
<td>Real Value</td>
<td>/</td>
</tr>
<tr>
<td>colour</td>
<td>The colour of the horse</td>
<td>Categorical</td>
<td>Bay, Chestnut, etc.</td>
</tr>
<tr>
<td>sex</td>
<td>The sex of the horse</td>
<td>Categorical</td>
<td>Gelding</td>
</tr>
<tr>
<td>import_type</td>
<td>The import type of the horse</td>
<td>Categorical</td>
<td>PP, PPG</td>
</tr>
<tr>
<td>sire_name</td>
<td>The name of the horse's sire</td>
<td>Categorical</td>
<td>Acclamation, Patagan, etc.</td>
</tr>
<tr>
<td>last_plc</td>
<td>The place in the last race</td>
<td>Index</td>
<td>/</td>
</tr>
<tr>
<td>last_rating</td>
<td>The rating in the last race</td>
<td>Index</td>
<td>/</td>
</tr>
<tr>
<td>rating</td>
<td>The rating now</td>
<td>Index</td>
<td>/</td>
</tr>
</tbody>
</table>
Data (Description)

- Weather data
  - mean_degree: The mean of the temperature of the race day
  - mean_humidity: The mean of the humidity of the race day
  - mean_pressure: The mean of the air pressure of the race day

- Additional data
  - total_first_count: The total count of first place
  - total_second_count: The total count of second place
  - total_third_count: The total count of third place
  - total_race_count: The total count of joined race
Data (Analysis)

- Correlation between continuous data
- Show whether the data is valid
Data (Analysis)

- Association between categorical data
- Show whether the data is valid
Data (Analysis)

- Correlation ratio between continuous data and categorical data
- Show whether the data is valid
Data (Preprocess)

- Continuous data: z-score normalization

- Categorical data: One hot encoding (YES/NO → {1,0})

\[ z = \frac{x - \mu}{\sigma} \]
XGBoost (Regressor)

- Developed by Tianqi Chen in 2014
- A scalable end-to-end tree boosting system
- As a regressor to predict the finishing time of the horses
- Result will be used in reinforcement learning and compared to the result of reinforcement learning
XGBoost (Reason using it)

- A lot of people have used it to win machine learning challenges
- Proved as a high efficiency and high accuracy system
- Easy to use and tune for different purposes
XGBoost (Process)

- Separate the dataset to training set and testing set
- Training set is the dataset from 2014 to 2018
- Testing set is the dataset from 2019
XGBoost (Hyperparameter)

```python
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
importance_type='gain', interaction_constraints='',
learning_rate=0.05, max_delta_step=0, max_depth=5,
min_child_weight=1, missing=nan, monotone_constraints='()',
n_estimators=230, n_jobs=0, num_parallel_tree=1, random_state=42,
reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
tree_method='exact', validate_parameters=1, verbosity=None)
```
XGBoost (Result)

- The tree generated by XGBoost
- Prove the model is successful
XGBoost (Result)

- **R2 Score:** 0.9974
- **Accuracy of predicting the first place:** 30.37%
- **Accuracy of predicting the first, second, third place:** 7.16%

<table>
<thead>
<tr>
<th>0</th>
<th>df_date</th>
<th>race_date</th>
<th>season</th>
<th>race_index</th>
<th>horse_id</th>
<th>race_no</th>
<th>class</th>
<th>win_odds_</th>
<th>total_race_count_</th>
<th>finish_time_sec</th>
<th>rating</th>
<th>plc</th>
</tr>
</thead>
<tbody>
<tr>
<td>9856</td>
<td>99.338414</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2018_C107</td>
<td>10</td>
<td>Class 3</td>
<td>3.0</td>
<td>12</td>
<td>100.15</td>
<td>64.0</td>
<td>3</td>
</tr>
<tr>
<td>9849</td>
<td>99.410896</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2018_C443</td>
<td>10</td>
<td>Class 3</td>
<td>5.1</td>
<td>4</td>
<td>100.69</td>
<td>72.0</td>
<td>8</td>
</tr>
<tr>
<td>9851</td>
<td>99.426056</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_D023</td>
<td>10</td>
<td>Class 3</td>
<td>5.8</td>
<td>25</td>
<td>100.83</td>
<td>63.0</td>
<td>11</td>
</tr>
<tr>
<td>9845</td>
<td>99.687193</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_B189</td>
<td>10</td>
<td>Class 3</td>
<td>8.9</td>
<td>35</td>
<td>100.49</td>
<td>77.0</td>
<td>5</td>
</tr>
<tr>
<td>9858</td>
<td>99.632103</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2014_T098</td>
<td>10</td>
<td>Class 3</td>
<td>12.0</td>
<td>69</td>
<td>100.13</td>
<td>61.0</td>
<td>2</td>
</tr>
<tr>
<td>9854</td>
<td>99.680313</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_B161</td>
<td>10</td>
<td>Class 3</td>
<td>10.0</td>
<td>17</td>
<td>100.30</td>
<td>72.0</td>
<td>4</td>
</tr>
<tr>
<td>9855</td>
<td>99.829094</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2016_A193</td>
<td>10</td>
<td>Class 3</td>
<td>13.0</td>
<td>45</td>
<td>100.82</td>
<td>63.0</td>
<td>10</td>
</tr>
<tr>
<td>9852</td>
<td>99.874359</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_B317</td>
<td>10</td>
<td>Class 3</td>
<td>16.0</td>
<td>21</td>
<td>101.25</td>
<td>70.0</td>
<td>12</td>
</tr>
<tr>
<td>9847</td>
<td>99.891357</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_B353</td>
<td>10</td>
<td>Class 3</td>
<td>15.0</td>
<td>18</td>
<td>100.51</td>
<td>67.0</td>
<td>6</td>
</tr>
<tr>
<td>9857</td>
<td>100.045280</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_B203</td>
<td>10</td>
<td>Class 3</td>
<td>25.0</td>
<td>25</td>
<td>100.89</td>
<td>70.0</td>
<td>1</td>
</tr>
<tr>
<td>9848</td>
<td>100.065971</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2015_V338</td>
<td>10</td>
<td>Class 3</td>
<td>33.0</td>
<td>41</td>
<td>100.51</td>
<td>77.0</td>
<td>7</td>
</tr>
<tr>
<td>9850</td>
<td>100.538182</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2016_A127</td>
<td>10</td>
<td>Class 3</td>
<td>69.0</td>
<td>19</td>
<td>102.16</td>
<td>76.0</td>
<td>14</td>
</tr>
<tr>
<td>9853</td>
<td>100.536362</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2017_B330</td>
<td>10</td>
<td>Class 3</td>
<td>74.0</td>
<td>14</td>
<td>101.71</td>
<td>78.0</td>
<td>13</td>
</tr>
<tr>
<td>9846</td>
<td>100.768088</td>
<td>2019/12/29</td>
<td>2019/12/29</td>
<td>19/20</td>
<td>293 HK_2018_C409</td>
<td>10</td>
<td>Class 3</td>
<td>102.0</td>
<td>3</td>
<td>100.69</td>
<td>69.0</td>
<td>9</td>
</tr>
</tbody>
</table>
XGBoost (Simulation)

Betting Simulation:

- Each bet: $10
- Win: $10 * win_odds - $10 (Cost)
- Loss: -$10
XGBoost (Simulation)

Bet on every game

cash balance

races
XGBoost (Simulation)

Bet on horse that has participated in 10 races before

Cash balance vs. races
XGBoost (Simulation)

Bet on horse that has participated in 15 races before

cash balance

races
XGBoost (Simulation)

Bet on horse that has participated in 20 races before
XGBoost (Simulation)
XGBoost (Simulation)
XGBoost (Conclusion)

- Positive correlation between participation experience and the win rate
- Although there are criteria, the return is not high
- Prediction with 30% accuracy can help in reinforcement learning
- Great experience of studying machine learning in horse racing
Reinforcement Learning (Algorithm)
Reinforcement Learning (Algorithm)

The goal of reinforcement learning is to find the best policy which can bring the best expected total reward:

$$E_{\tau \sim p_{\theta}(\tau)} \left[ \sum_t r(s_t, a_t) \right]$$

Best policy:

$$\theta^* = \arg \max_{\theta} E_{\tau \sim p_{\theta}(\tau)} \left[ \sum_t r(s_t, a_t) \right]$$
Reinforcement Learning (Objective)

Value-based algorithm:

- Improve the policy based on value (reward)
- Similar to the situation of gambling
  - We place a bet on a horse since we believe the horse will bring us money
  - The agent place a bet on a horse since he believes the horse will bring him reward

Deep Q-learning!!!
Reinforcement Learning (Q-learning)

- Value-based algorithm
- Q-function: $Q(s,a)$
- Q-table:

<table>
<thead>
<tr>
<th>State \ Action</th>
<th>a1</th>
<th>a2</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>value1</td>
<td>value2</td>
</tr>
<tr>
<td>s2</td>
<td>value3</td>
<td>value4</td>
</tr>
</tbody>
</table>
Reinforcement Learning (Q-learning)

Algorithm of Q-learning:

1. Initialize Q function $Q(s,a)$ to some random values
2. Take an action from a state using epsilon-greedy policy from Q function
3. Observe the reward and the new state
4. Update the Q table by:
   \[
   Q(s, a) = Q(s, a) + \alpha(r + \gamma \max Q(s', a') - Q(s, a))
   \]
5. Repeat step 2 to step 4 until terminal state
Reinforcement Learning on playing games

- “Cartpole” from OpenAI gym
- Prevent the pole falling over the cart
Reinforcement Learning on playing games

Reason not using Q-learning:

- Only can be used in some simple problems or games
- Lots of data causing a huge Q-table and the efficiency is too low to complete the Q-table
- Impossible to test since it will only take random action while meeting new races with new horses.
Reinforcement Learning on horse racing

- Deep Q Learning with MLP policy
  - Use neural network to approximate Q-function
  - Loss Function: $E_{s,a,r,s'}[(r + \gamma \max_{a'} Q(s', a'; \theta')) - Q(s, a; \theta)]^2$
    - Stochastic gradient descent

- Moving target problem
  - change $\theta \rightarrow \theta'$ will be affected
    - Freeze the $\theta'$ long enough from DeepMind
Reinforcement Learning on horse racing

Environment

- **Observation Space**
  - the features of 14 horses, including invalid horses (set to -99)

- **Action Space**
  - 15 actions: {'bet on horse 1', ... , 'bet on horse 14', 'do not bet'}, referring to the input order
  - bet with a fixed amount 10 dollars

- **State**
  - 1 state = 1 horse racing game

- **Termination state**
  - Lose more than $1000
  - goes through all the horse racing games
Reinforcement Learning on horse racing

Reward Function

- Idea
  - Reward of winning $100 > Reward of winning $11
  - \[
    R(\text{bet and win}) = C_1 \cdot \Delta\text{Cash Balance}, \text{ where } C_1 > 0
    \]
  - \[
    R(\text{bet but lose}) = C_2 \cdot \Delta\text{Cash Balance}, \text{ where } C_2 < 0
    \]
  - \[
    R(\text{do not bet}) = C_3 \cdot \text{win odds of the true first place}, \text{ where } C_2 < C_3 < 0
    \]
Reinforcement Learning on horse racing

Invalid action

- Ignore it
  - not reasonable, cheaper version of ‘do not bet’
- Same penalty as ‘do not bet’
  - it will treat betting on invalid horse as ‘do not bet’ if there are less than 14 horses
- Large penalty
  - Bet only on races with 5 horses OR ‘do not bet’

\[ R(\text{bet on invalid horse}) = R(\text{do not bet}) \text{ is chose} \]
Reinforcement Learning on horse racing

Input Order

- Shuffle the order of the horse -> do not converge on reward

- We use the prediction from XGBoost to order the input
  - The horse with largest number is the fastest horse we predicted in a race
  - If there are only 13 horses, then the horse 14 is a invalid horse
    - to see if the agent learns the exist of ‘invalid horse’
Convergency
How agent bet

Training set
Invalid and valid betting

training set

How many times it bet

- Valid bet: 69%
- Invalid bet/Do not bet: 31%

testing set

How many times it bet

- Valid bet: 70%
- Invalid bet/Do not bet: 30%
Win ratio

Training set:
- How many times it wins:
  - Win: 70%
  - Lose: 30%

Testing set:
- How many times it wins:
  - Win: 73%
  - Lose: 27%
Cash balance in training set

Largest win: 10 * 140
Cash balance in testing set

Largest win : $10 \times 34$
Result analysis

- 0 ‘do not bet’ action
  - Same as our expectation,
    - The agent treat ‘bet on invalid horse’ as ‘do not bet’
    - It means the agent bets on all the races with 14 horses
      - can't learn the meaning of the action ‘do not bet’?
      - betting on every race with 14 horses is a nice choice?

- Large win odds
  - Bet on those horses with high win odds to gain a larger reward
Result analysis

- **Bet on races with 14 horses ONLY**
  - Reward is maximized when it only bet on the races with 14 horses
  - Penalty of losing is larger than betting on a invalid horse
  - More explanation in the next result

- ‘Bet on horse 14’ most of the time
  - The horse 14 is the most likely the winning horse in races with 14 horses
  - This is the most safe action
    - 30% accuracy in races with 14 horses
    - ‘invalid horse’ most of the time, which is do not bet
      - the penalty is less than ‘losing’
Improve the reward function

We want to encourage the agent to bet more, not just limited to races with 14 horses.

Solution:

- Increase the reward of ‘winning’
- Decrease the penalty of ‘losing’
- Decrease the penalty of ‘invalid betting’
Comparison of how agent bet
Comparison of Invalid or valid betting

How many times it bet

old reward function

69%

31%

new reward function

63%

37%

Valid bet
Invalid bet/ 'Do not bet'
Comparison of Win ratio

How many times it wins

- Old reward function: 73% Win, 27% Lose
- New reward function: 77% Win, 23% Lose
Comparison of cash balance
Result analysis

● 0 ‘do not bet’ action
  ○ Same as our expectation,
    ■ The agent treat betting on invalid horse as ‘do not bet’
    ■ It still bet on all the races with 13 and 14 horses
      • Best choice?
      • can’t learn well?

● Win ratio is decreased
  ○ It is the reason why the old agent refused to bet on races less than 14 horses
    ■ It will lower the win rate and the reward is less than before

● ‘Bet on horse 13’ most of the time
  ○ Refuse to bet more
    ■ The win ratio is decreased
    ■ The reward is maximized
  ○ So It is still limited to bet only on races with 13 and 14 horses
Betting strategy
Conclusion

● Input order/format matters!
  ○ Invalid horse
  ○ ordered by the win odds?
● Bet on races with particular number of candidates only
  ○ maybe build up a specified model
    ■ 14 horses RL model
    ■ 13 horses RL model ...
● Combinining all the races
  ○ races with different counts of candidates may become the ‘noise’ to each other
● Construction of reward function
  ○ It is hard to balance the reward and penalty
  ○ It will affect how the agent bet
Conclusion

● More Betting types
  ○ Quinella
  ○ Place
  ○ ...

● Different betting amount
  ○ The betting amount should not be fixed

● Based on the needs above
  ○ different model is required.
    ■ policy gradient or actor-critic
    ■ continuous output
Thank You!

Q&A