Bandit Algorithm, Reinforcement Learning, and Horse Racing Result Prediction

LYU2103

Wong Tin Wang David(1155127053)

Sze Muk Hei(1155127477)

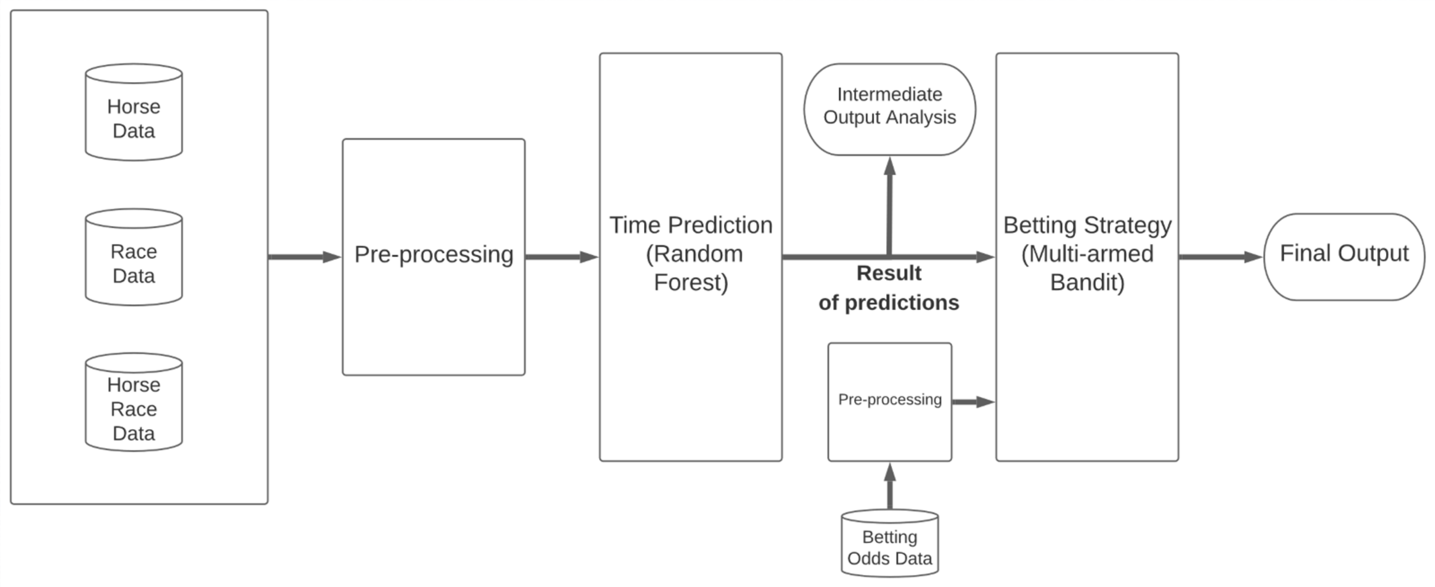
Supervised by Prof. Michael R. Lyu

0

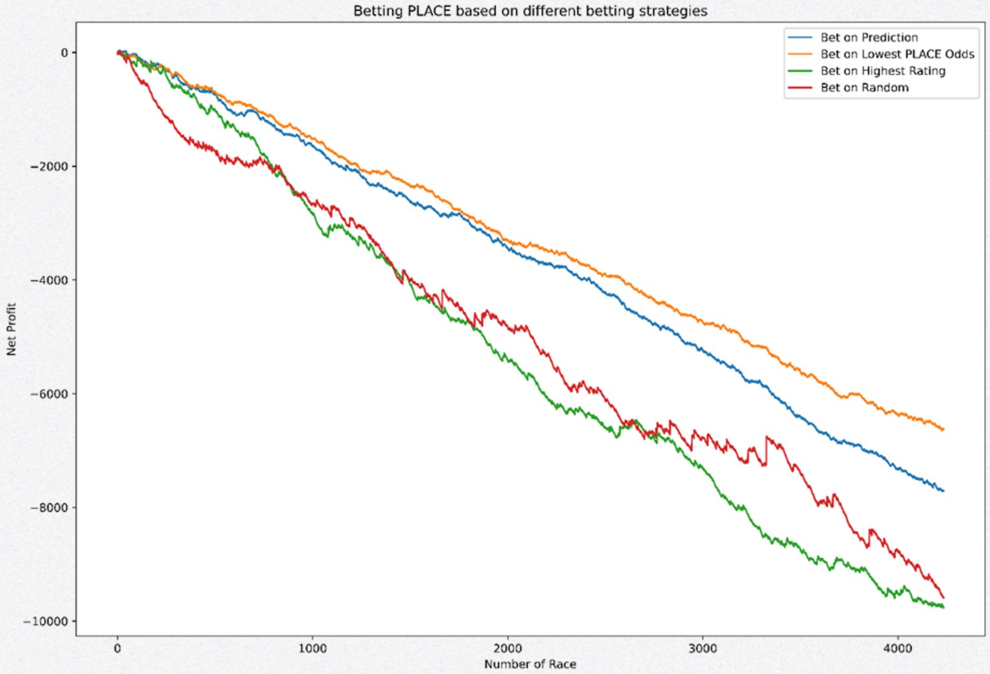
Review

Achievements in Last Semester

Recall that…

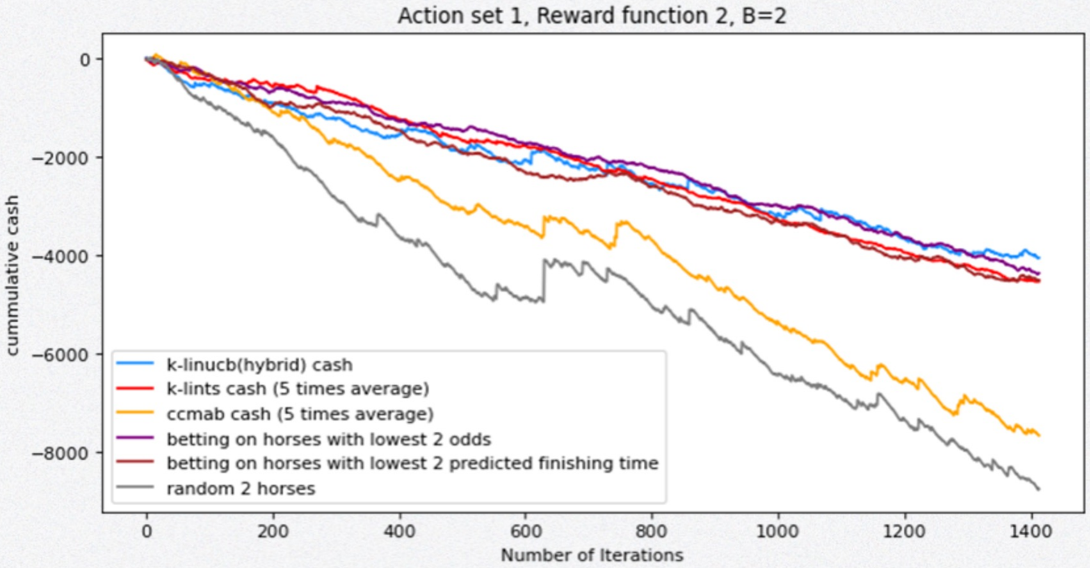

Horse Data
Race Data
Horse Race Data
Pre-processing
Time Prediction (Random Forest)
Intermediate Output Analysis
Result of predictions
Pre-processing
Betting Odds Data
Betting Strategy (Multi-armed Bandit)
Final Output

Recall that...



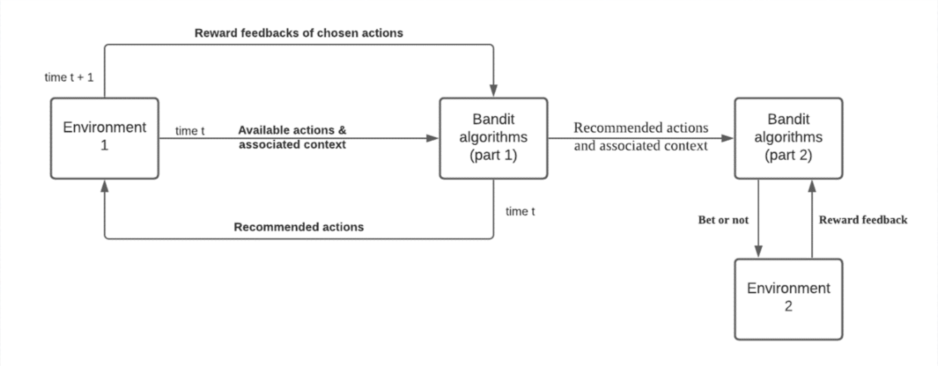
Simulated horse betting on the time prediction result from the random forest model

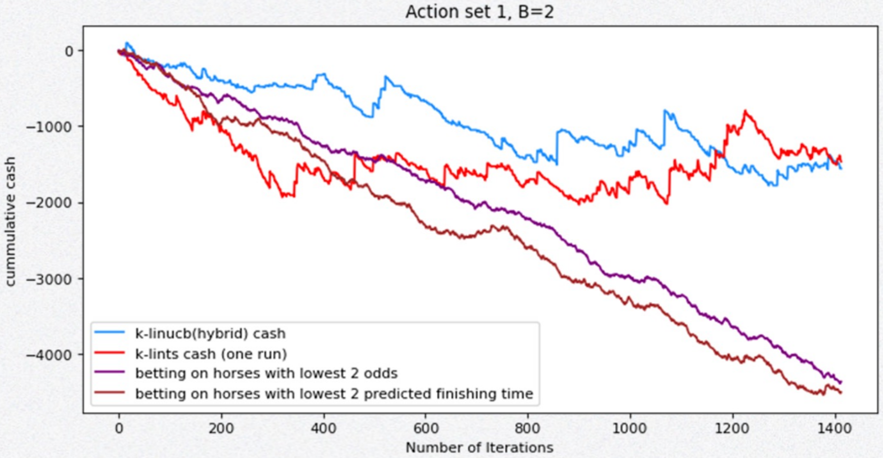
Recall that…



Explored different Bandit algorithms in many constructs (e.g. action sets, reward functions) On Horse betting

Recall that…





Applied a tricky technique to attempt to let bandit algorithm decide how much to bet

Agenda

1

Introduction

2

Data

3

Horse Racing Prediction

4

Betting Strategies

5

Conclusion

6

Q&A

1

Introduction

Objectives, Contribution

Objectives (2nd Semester)

- Improve **accuracy** and **interpretation** of time prediction model
- Explore new horse betting strategies using **new bandit algorithms** and **other** types of reinforcement learning algorithms
- Enable the agent **bet with different amount of money**
- Enhance the stability of horse betting strategies using **model selecting with EXP3**

WHAT’s NEW?

WHAT's NEW?

1. Improved Random Forest Model
2. Explored New Bandit Algorithms
3. Applied More RL Algorithms on Horse Betting
4. Model Selection using Bandit Algorithm

Contribution

1. Reduced loss of random forest betting
    - WIN bet
        i. **Reduced 87.162% loss**
    - PLACE bet
        i. **Reduced 46.008% loss**
2. Explored possible horse betting strategies generation **(PLACE)**
    - Neural Bandit / Neural UCB
    - Other reinforcement learning algorithms
3. **Enhanced stability** of horse betting strategies using model selection

2

Data

Descriptions, Analysis & Pre-processing

Sources & Descriptions

- **Data Sources**
  a. The Hong Kong Jockey Club
  b. Data Guru
  c. hkHorse
- **Datasets**
  - Ranged from 1979 to 2021
  - Tables:
    - Races data
    - Horses data
    - Horse-race data
    - Betting odds data

Input Data for Training

Features	Types	Encoding Methods
raceclass	Categorical	Ordinal
tracktype	Categorical	One-hot
racktrack	Categorical	One-hot
course	Categorical	One-hot
country	Categorical	One-hot
importtype	Categorical	One-hot
sex	Categorical	One-hot
colour	Categorical	One-hot
going	Categorical	One-hot
jockey	Categorical	Ordinal
trainer	Categorical	Ordinal
horseid	Categorical	Ordinal
dam	Categorical	Ordinal
sire	Categorical	Ordinal
damsire	Categorical	Ordinal
distance	Categorical	Ordinal
draw	Categorical	Ordinal
rating	Real Value	/
rating_rank	Real Value	/
last_rating	Real Value	/
avg_rating	Real Value	/
last_place	Real Value	/
winodds	Real Value	/
win_odds_rank	Real Value	/
actualweight	Real Value	/
declaredweight	Real Value	/
gear	Categorical	Customized Encoding
raceidseason	Real Value	/
count_{1 – 3}	Real Value	/
weight_diff	Real Value	/
avg_finishtime	Real Value	/
avg_pos{1 – 6}_pos	Real Value	/
avg_pos{1 – 6}_time	Real Value	/
last_pos{1 – 6}_pos	Real Value	/
last_pos{1 – 6}_time	Real Value	/

- Features included
  - Races data
  - Horses data
  - Horse-race data
  - Additional features
- Drop unnecessary , irrelevant features
- Split train and test data according to race season
  - **Training** data: **2008 - 2019**
  - **Testing** data: **2019 - 2021**

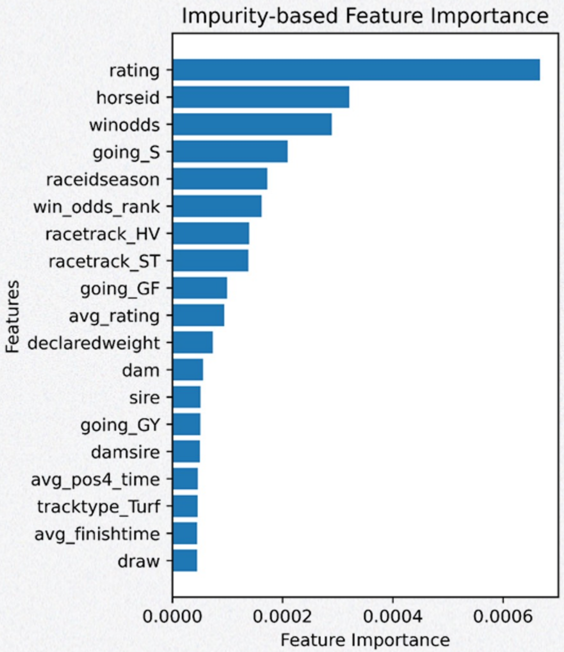
3

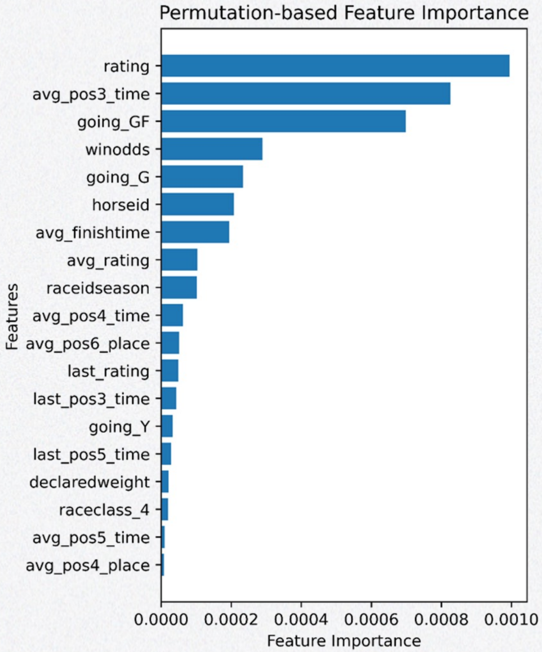
Horse Racing Prediction

Procedure, Evaluation & Performance

A New Random Forest

Why Need This?





- Some features are **not influential**
- Some features are **correlated**
- **Improve accuracy** of the model
- Investigate the importance of features **among the selected features**

Features of New Model

Features	Types	Encoding Methods
raceclass	Categorical	Ordinal
horseid	Categorical	Ordinal
distance	Categorical	Ordinal
rating	Real Value	/
winodds	Real Value	/
win_odds_rank	Real Value	/
raceidseason	Real Value	/

- Features included
  - Horses data
  - Horse-race data
- Extract features with **7 highest importance**
- Split train and test data according to race season
  - **Training** data: **2008 - 2019**
  - **Testing** data: **2019 - 2021**

Results and Analysis

Evaluation Metrics

- **Mean Squared Error (MSE)**
  - Accuracy of the prediction
  - Closer to 0, the better performance
  - **MSE of model: 1.7177 seconds**
    - **reduced by 24%** with value of **0.5472**
- **Explained Variance Score**
  - Discrepancy between the model and data
  - The closer to 1, the stronger association
  - **Explained Variance Score of model: 0.99547**
    - **increased by 0.00159**

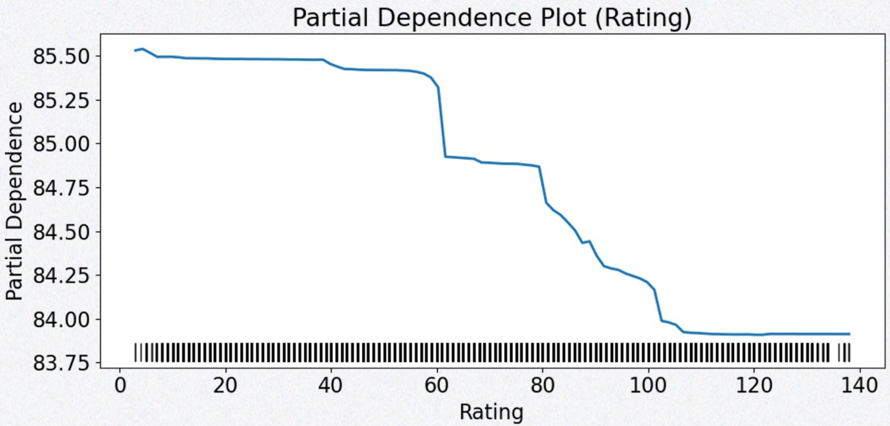
Betting Accuracy

- **WIN Betting**
  - Correctly predicted 24.331% of races
    - **Decreased by 0.206% from old model**
- **PLACE Betting**
  - Correctly predicted 47.108% of races
    - **Decreased by 0.499% from old model**

Partial Dependence Plot (Rating)

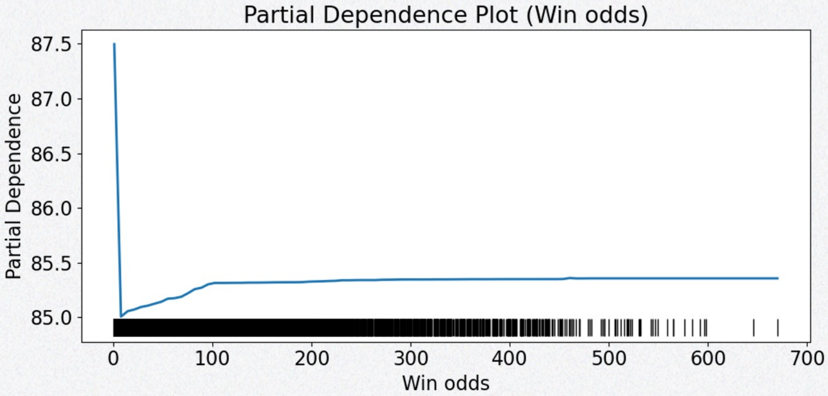
Table 1.1 Rating for Race classes [57]

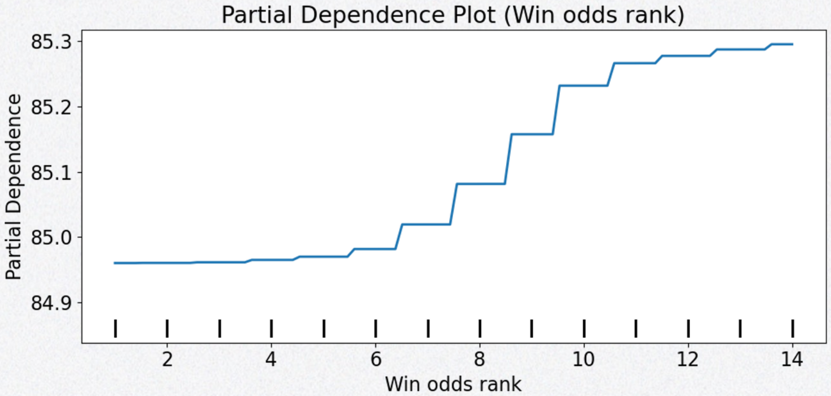
Rating upper bound	Race classes
120	1
100	2
80	3
60	4
40	5



- Rating has **highest feature importance**
- Race classes determined by rating
- **Inversely proportional** to finishing time
- Clear intervals in PDP
  - Matches race classes
- **Race class 2** has the most varied results

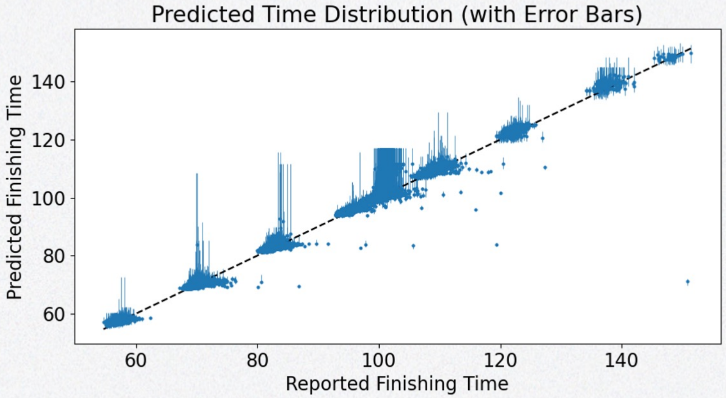
Partial Dependence Plot (Odds)

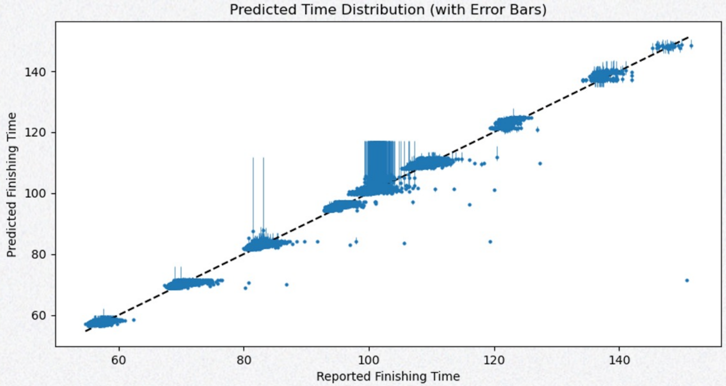




- Win odd regards as public intelligence
- **Greatly dropped at low win odds**
    - Horses with low odd may not always win
- Win odd rank shows clear intervals
    - Rank 5 - 11 has a large step up

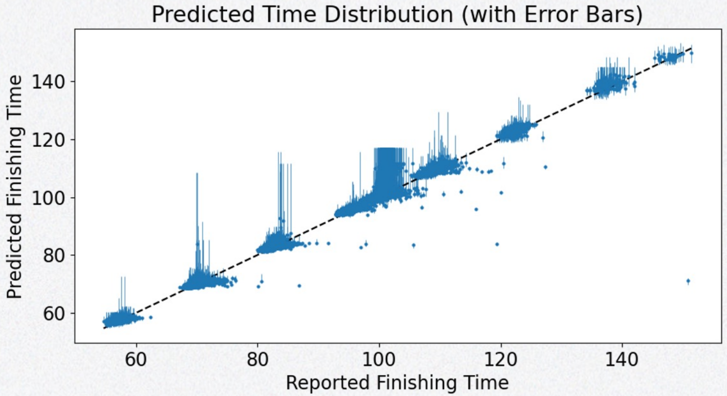
Error Range of Predictions

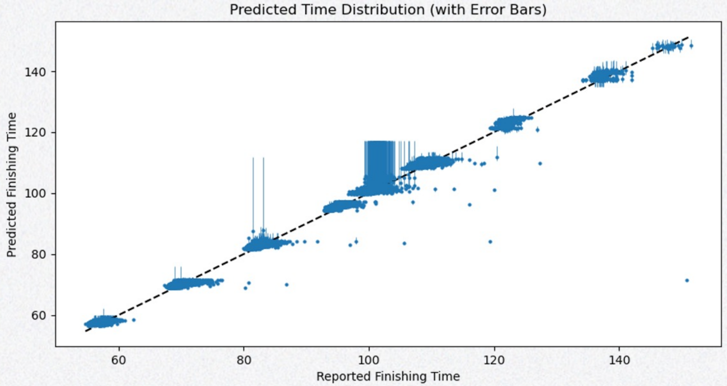




- **Variance of predictions better than old model**
- Average range of predictions: **0.851s**
  - Reduced by **48.188%** from old model
- 22.397% of predictions **range > mean**
  - Reduced by **4.08%** from old model

Error Range of Predictions





- Bet only predictions with small variance
    - **Range < mean**
    - Reduce loss
- **Change of number of correct predictions**
    - **WIN: Unchanged**
    - **PLACE: -5.2%**

Betting Simulation

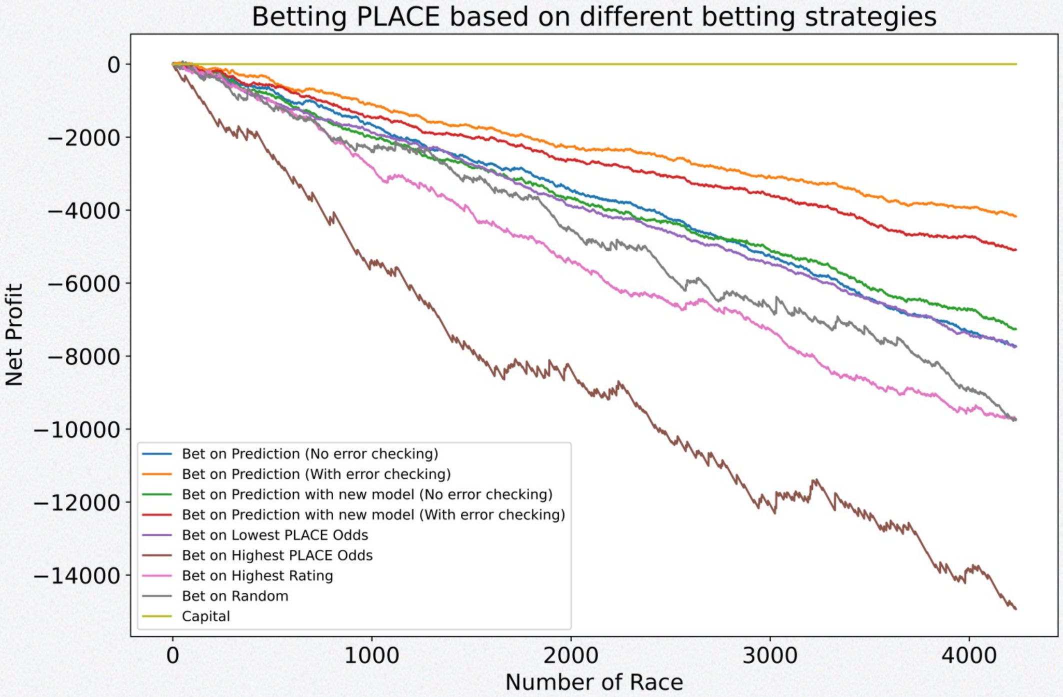
1. Group all the horses by the race
2. Order the horses by the predicted finishing time in ascending order
3. Assign a **predicted place** to each horse according to the ranking
4. Start Betting!

	horseid	raceid	place	winodds	pred	pred_place	place_difference
17995	5265.0	2019-09-01-001-1600-Turf	1	2.2	95.84	1	0
17996	5186.0	2019-09-01-001-1600-Turf	2	4.9	95.89	4	2
17999	4302.0	2019-09-01-001-1600-Turf	3	18.0	96.16	5	2
17994	4268.0	2019-09-01-001-1600-Turf	4	5.7	95.86	3	-1
17993	4296.0	2019-09-01-001-1600-Turf	5	7.0	95.85	2	-3
18001	4809.0	2019-09-01-001-1600-Turf	6	19.0	96.21	7	1
17997	5103.0	2019-09-01-001-1600-Turf	7	50.0	96.35	9	2
17998	4845.0	2019-09-01-001-1600-Turf	8	14.0	96.33	8	0
18000	4982.0	2019-09-01-001-1600-Turf	9	21.0	96.17	6	-3

Betting Simulation

1. Assume $10 would be used for each bet
2. Gain **$10 * odds – 10** if correctly picked the horses
3. Lose $10 otherwise
4. **PLACE** betting would be simulated
5. Compare with different strategies
    - Based on lowest odds
    - **Based on highest odds**
    - **Based on error range**
    - Based on highest rating
    - Random

Betting Simulation



- Betting **PLACE**
- Based on **Highest odds** is the worst
- **New model performs better (No error checking)**
- Based on **Error range** is the best
  - **Old** Model has **39.873%** accuracy
  - **New** Model has **41.914%** accuracy

4

Horse Betting Strategies

Bandit part Improvements

Use better models

Last Sem: Linear models

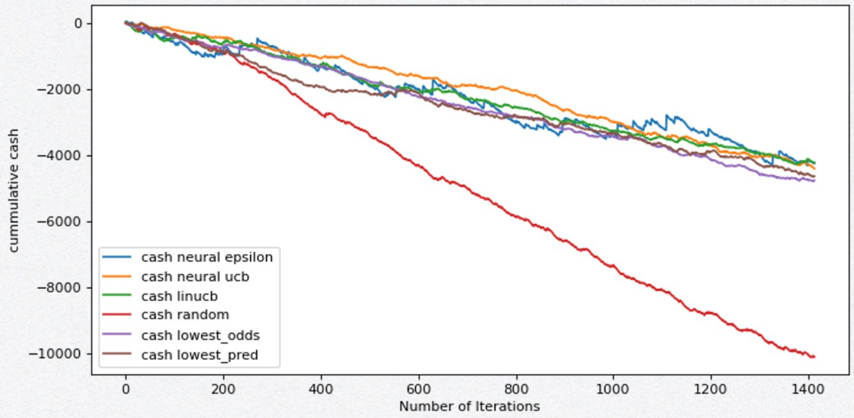
Possible Problem: lower accuracy

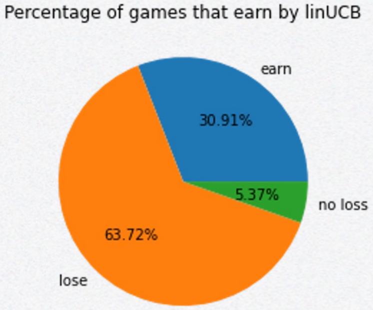
Attempt:

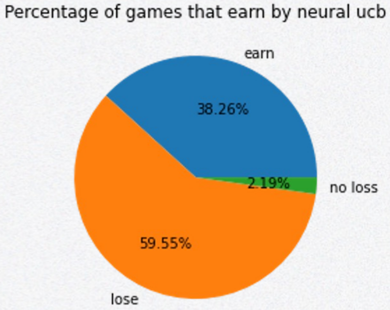
Use more complex models: **neural networks**

- **Neural UCB** (Single neural network & UCB exploration)
- **Neural Bandit** (neural network committee & epsilon greedy exploration)

Use better models



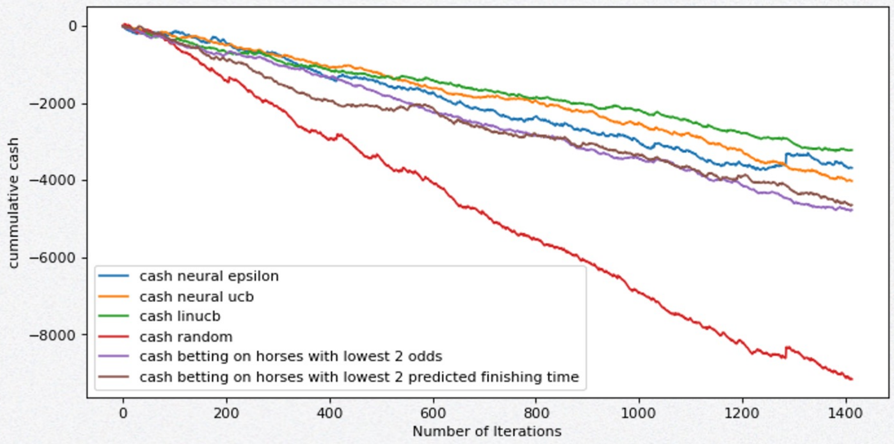




Doesn't show significant Improvement in terms of Cash balance

However, the earn rate is **8%** higher than that of linUCB

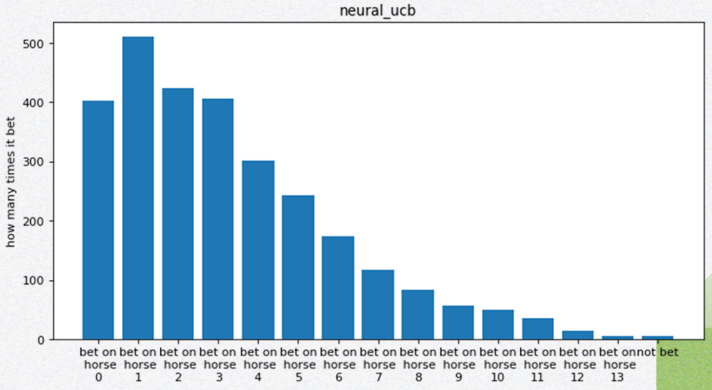
Bets on fewer options



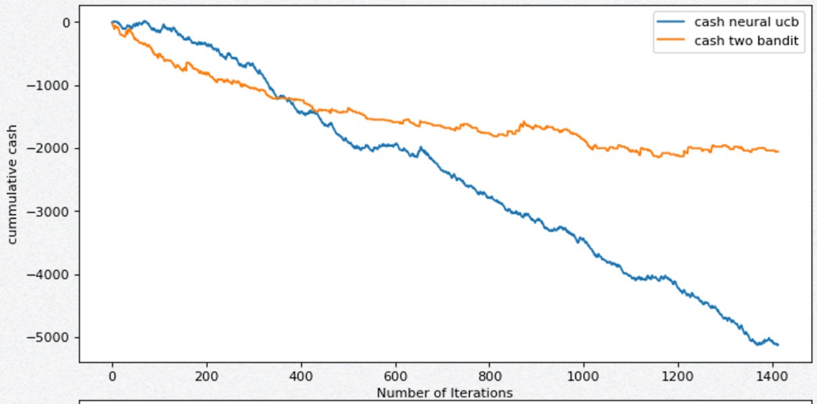
As top 5 horses occupy most out of all horses bet

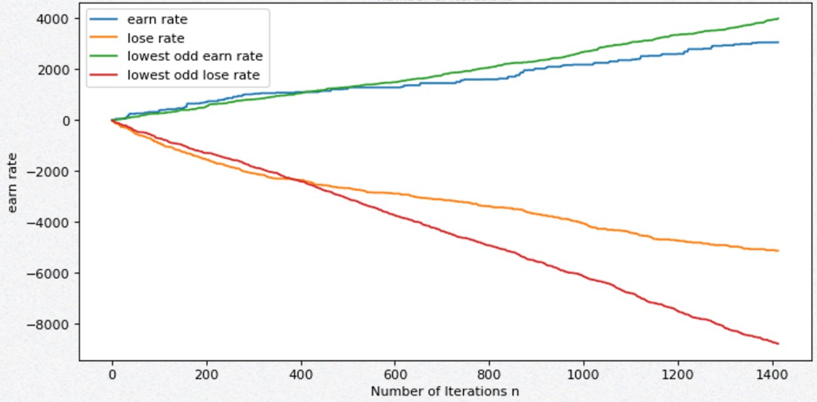
Bet on **only top 5**

- Slight improvements but still losing



Redoing previous approach with these improvement





- Maintain its balance
  But doesn't earn
- Earn rate still grow overtime
- Lose rate reduces over time

Concluding bandit parts

Problems of directly using bandit algorithms on horse betting:

- **Not flexible**
  - unable to consider state information like remaining balance
  - Not easy to make variable amount of bet
    (Directly set as actions: **Failed**, always fall to **safest** option which is $10 bet)
- **Not work for very low odds and insufficient accuracy**
  - Low expected return

Might be better to use more common RL algorithms

Other Algorithms

Why using other algorithms?

- Explore the possibility of finding horse betting strategies using different algorithms
    - Enhance the profitability
    - Able to bet with different amounts of money
- Evaluate the performance of multi-armed bandit by comparing all results

Algorithms used

- Selected from previous projects
  - Deep Q Network
  - Proximal Policy Optimization
- Other model-free, policy-based algorithms
  - Augmented Random Search
  - Cross Entropy Method

Environments

Type 1: only bet with $10
Type 2: bet with different amount of money ($10 - $50)
Data to Use

- Split into train and test set with 707 records each

Features (for each horse):

- Last moment place odds
- Last 10 minutes EMA of odds
- Rankings (odds, predicted finishing time)
- Ratio of finishing time between each horse with the horse ranked 1 place ahead (finishing time)
- Confidence level related (error range, upper and lower bound)

Environments (Type 1)

Action Set

- 14 horses (at most) ordered by predicted finishing time + not to bet

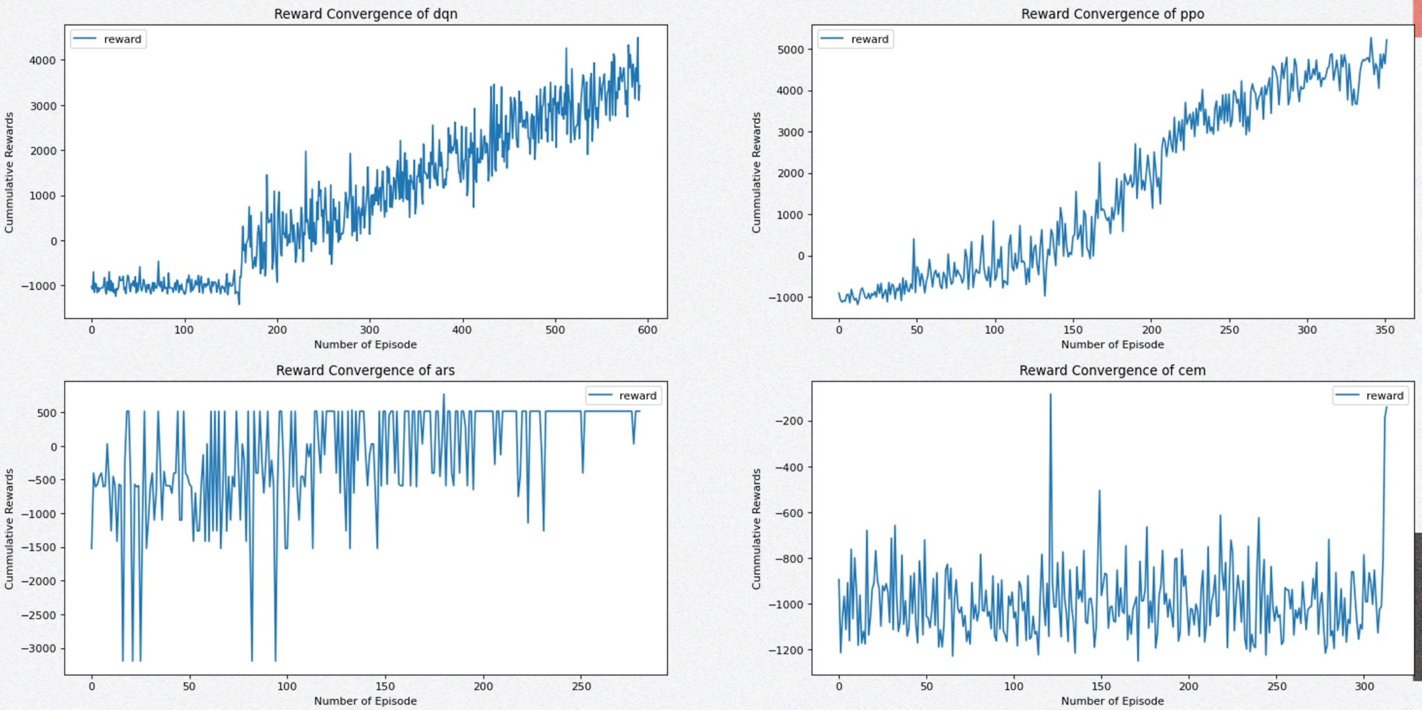
Terminating State

- No more races
- **Cash balance < 9000**

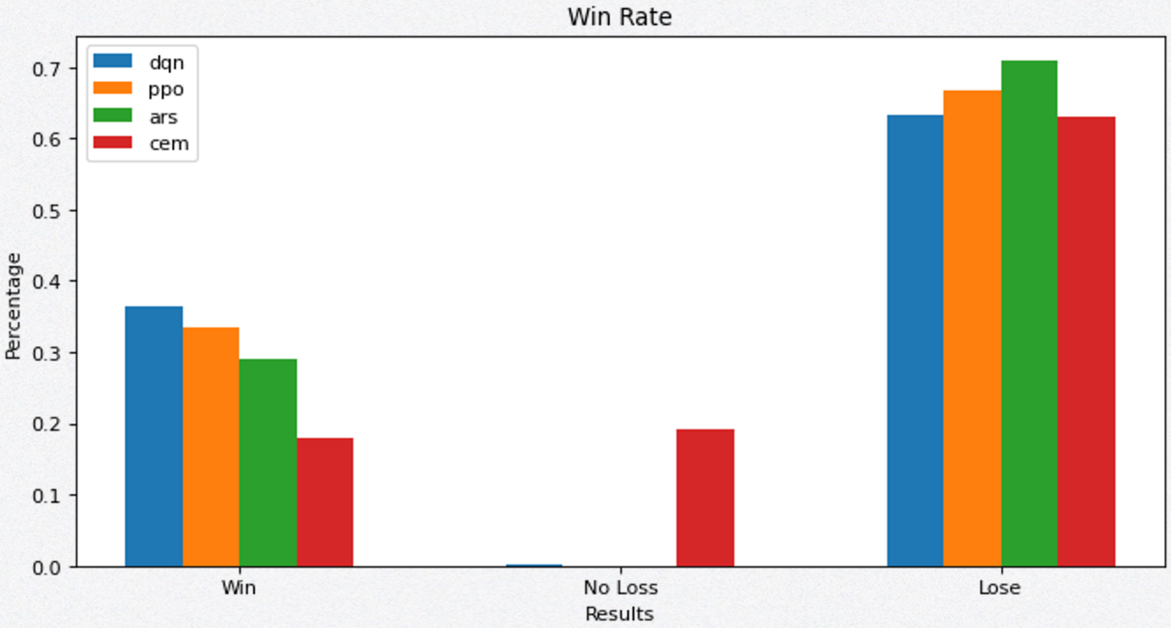
Reward Functions (Type 1 & 2)

- R(Bet **any of top 3** horses correctly and **error range < mean**) = (dollar bet * betting odd) *** ((dollar bet / 10) + 0.5)**
- R(Bet **any of top 3** horses correctly) = dollar bet * betting odd of betted horse
- R(Bet wrong and **error range > mean**) = -dollar bet *** ((dollar bet / 10) + 0.5)**
- R(Bet wrong) = -dollar bet
- R(Not bet) = - 3

Reward Convergence (Type 1)

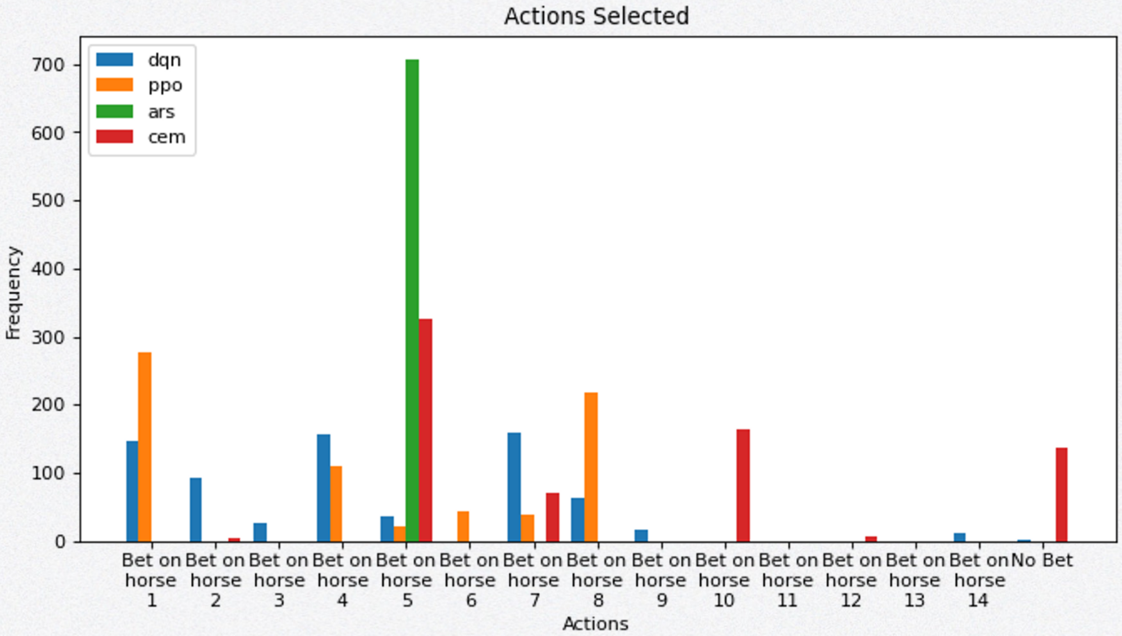


Win Rate (Type 1)



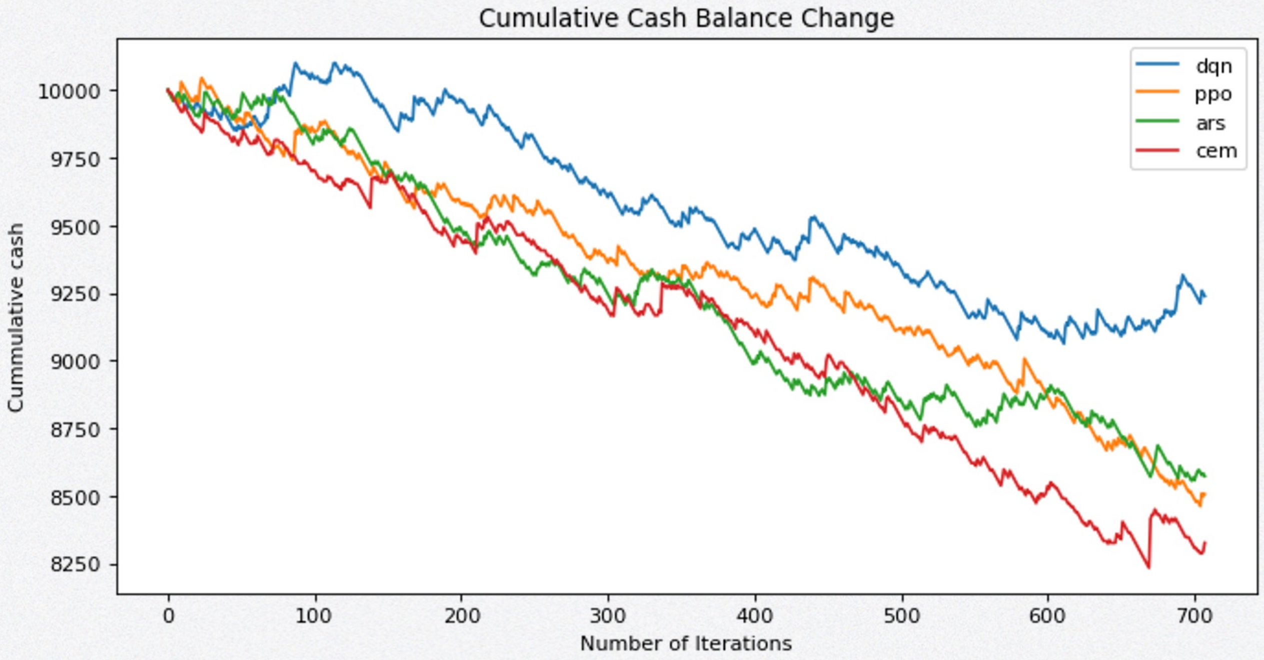
- **DQN** has **highest** win rate
- **ARS** has **highest** loss rate
- **Majority** of selection are **betting**

Actions Selected (Type 1)



- Only **ARS** bet on **single option**
- **DQN & PPO** has a **safer** strategy
- **CEM**'s strategy involves **different risks**

Profitability (Type 1)



- **All losing money**
- **DQN** perform **significantly better** than others

Environments (Type 2)

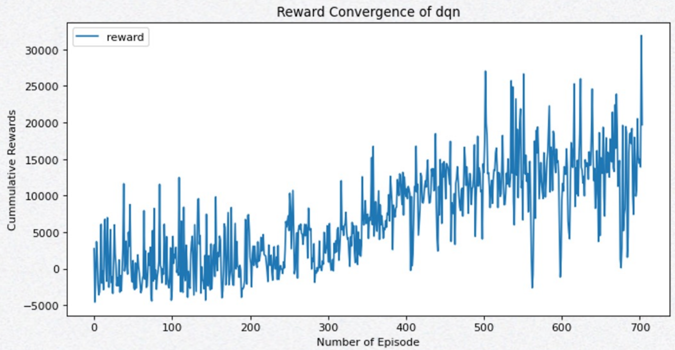
Action Set

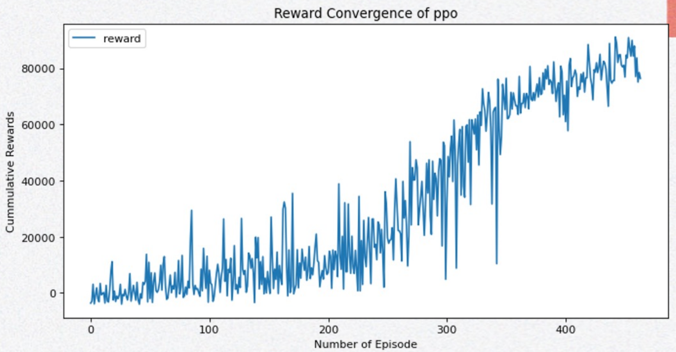
- 14 horses (at most) ordered by predicted finishing time + not bet
- 5 different amount of dollar bets ($10, $20, $30, $40, $50)
- Total actions: 15 * 5 = 75

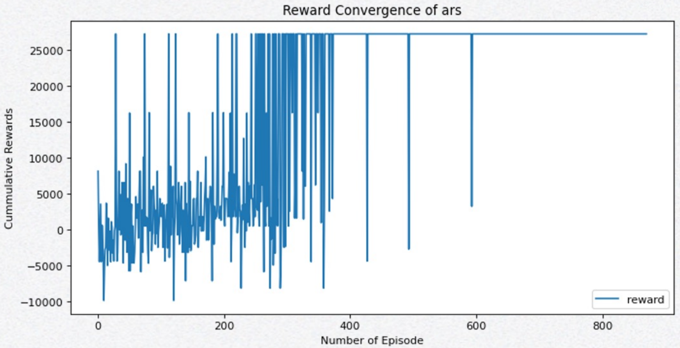
Terminating State

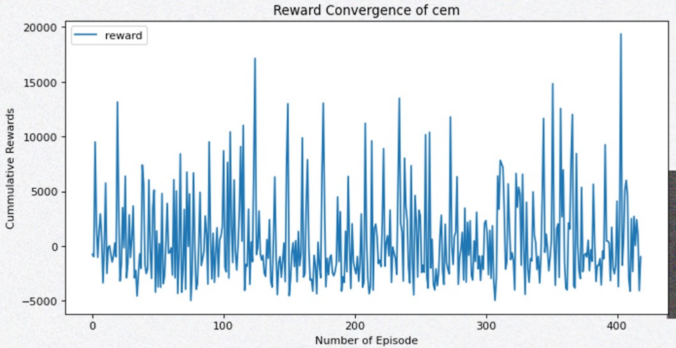
- No more races
- **Cash balance < 8000**

Reward Convergence (Type 2)

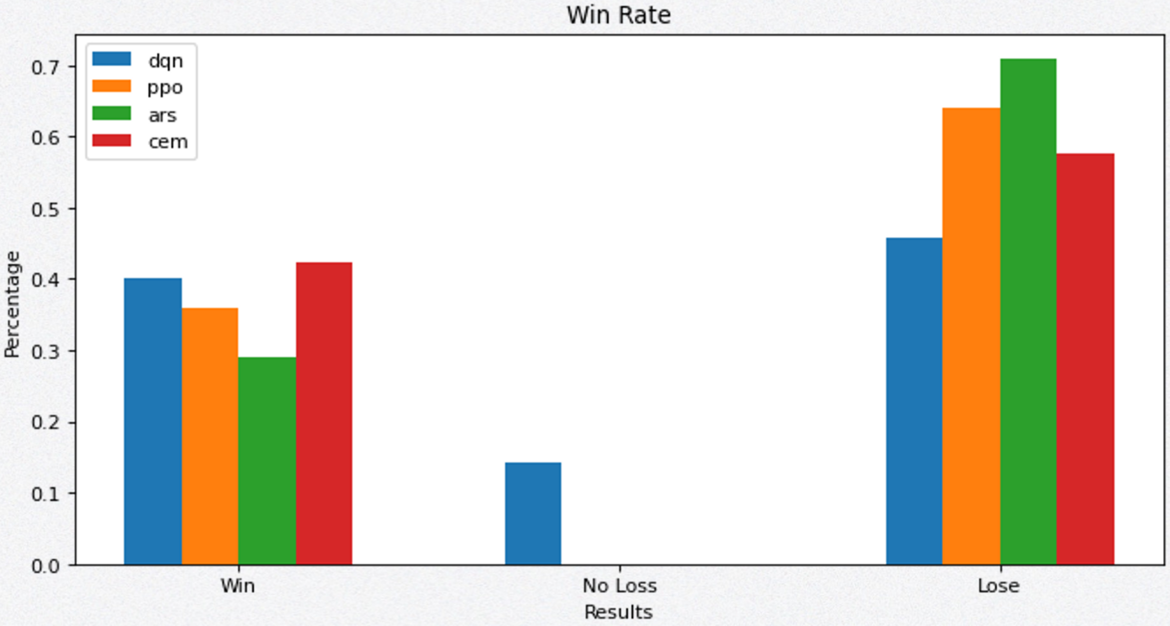






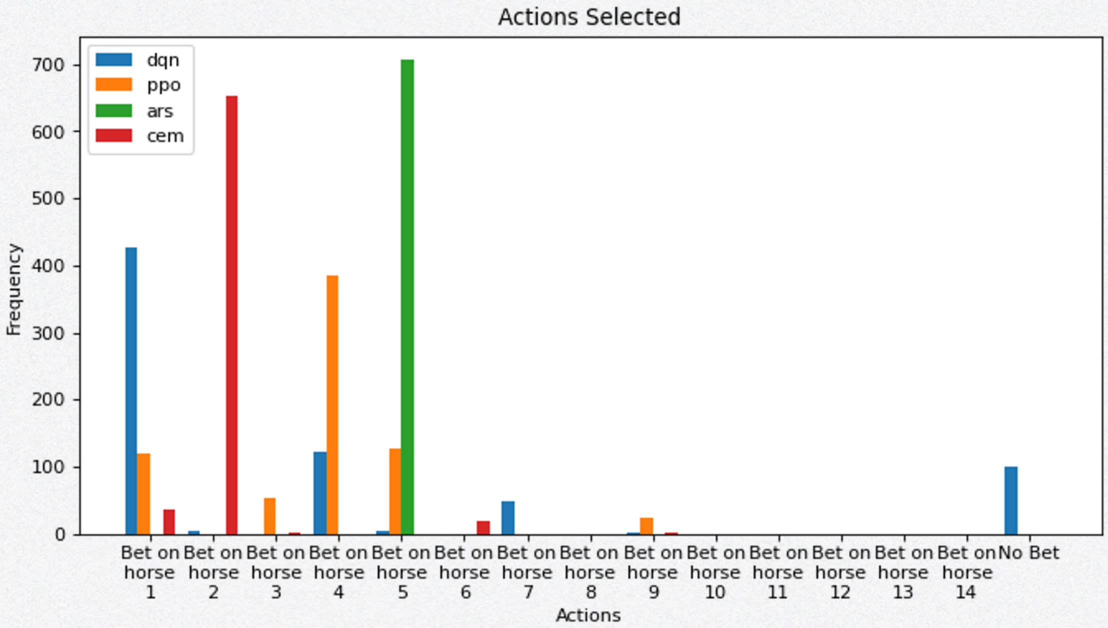


Win Rate (Type 2)



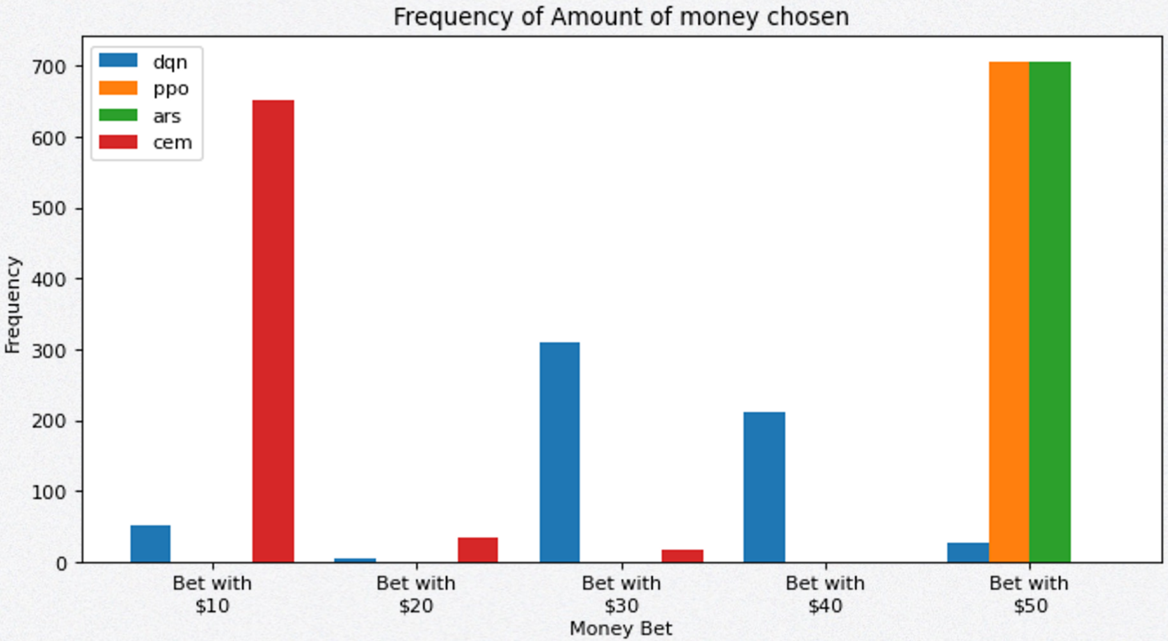
- **CEM** has **highest** win rate
- **ARS** has **highest** loss rate
- **Majority** of selection are **betting**

Actions Selected (Type 2)



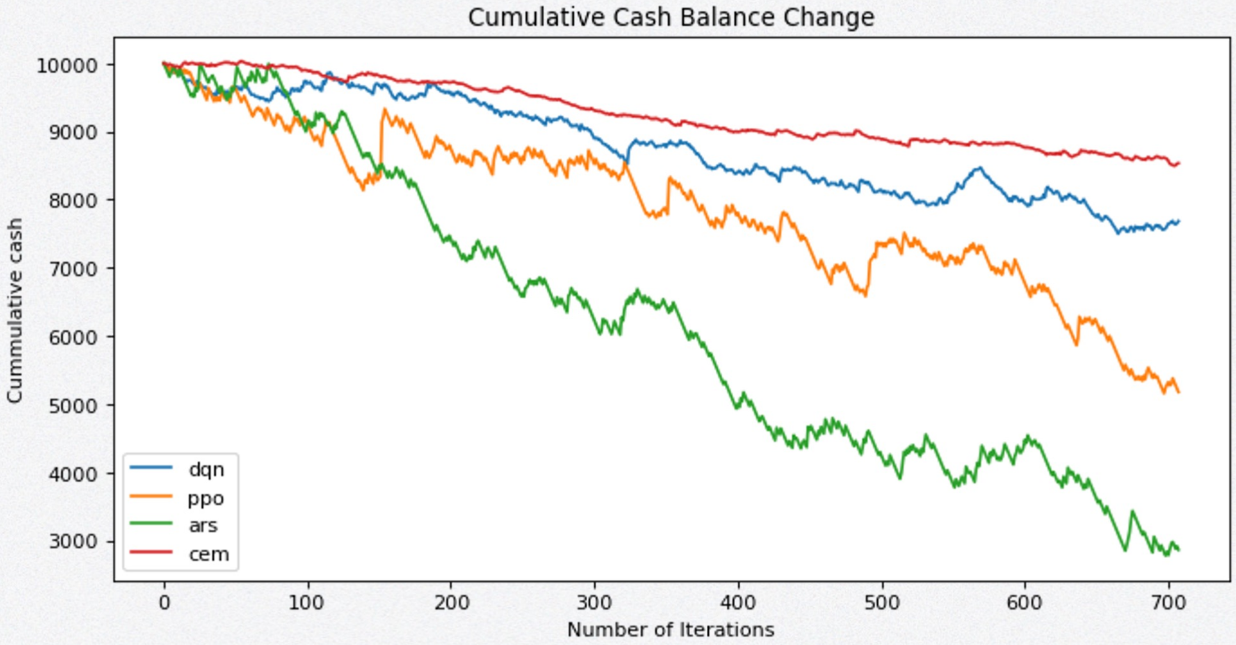
- Only **ARS** bet on **single option**
- **DQN, PPO & CEM bet safer than before**

Money Actions Selected (Type 2)



- PPO & ARS only bet $50
- CEM mostly bet with $10
- DQN bets with different amount

Profitability (Type 2)



- **All losing money**
- **CEM** perform **better** than others
- **PPO & ARS** has great loss

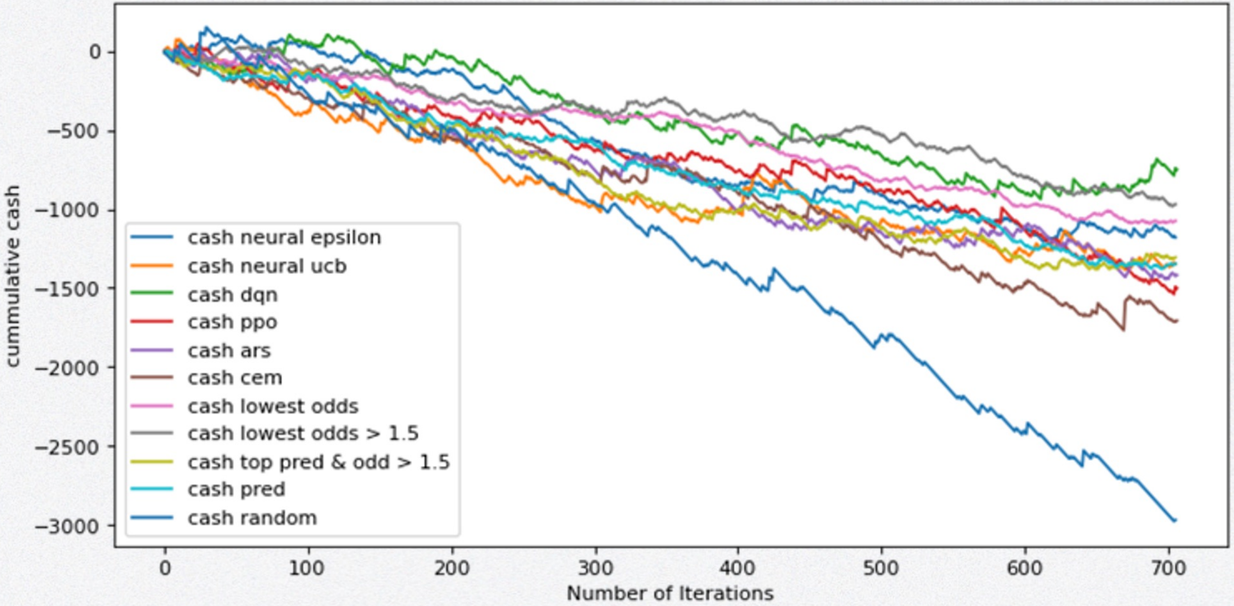
Overall Comparison

Optimal Actions Counts

	Optimal	Sub-optimal	Non-optimal
Neural Epsilon	**53 (7.50%)**	**243 (34.37%)**	**411 (58.13%)**
Neural UCB	65 (9.19%)	117 (16.55%)	525 (74.26%)
DQN	59 (8.35%)	205 (29.00%)	443 (62.66%)
PPO	63 (8.91%)	197 (27.86%)	447 (63.22%)
ARS	**89 (12.59%)**	127 (17.96%)	491 (69.45%)
CEM	59 (8.35%)	**84 (11.88%)**	**564 (79.77%)**

- Optimal:
  - Place: **top 3**
  - Reward: **top 3**
- Sub-optimal:
  - Place: **top 3**
- Non-optimal:
  - otherwise

Overall Comparison (Bet 1 option)



- **Lowest odd > 1.5** outperform other
- **DQN** perform the best among all algorithms

Model Selection

Why Model Selection

Model selection: selecting best suitable model at each time step

- No guarantee that a particular algorithm consistently performs well
- The best performing algorithm might be different over time
- Combining power of different algorithms

How?

We again use bandit algorithm (**EXP3**)

Why EXP3

EXP3 (**E**xponential-weight algorithm for **E**xploration and **E**xploitation)

- Adversarial Bandit (no assumption to make it work)
- Only update belief by reward (we don't use contextual since it would be just trying to approximate other algorithms)
- sensitive to reward changes(exponential)

Suitable when the behavior of algorithms might constantly changing

Procedure

EXP3 picks one algorithm at a time and bet according to the decision of the chosen algorithm

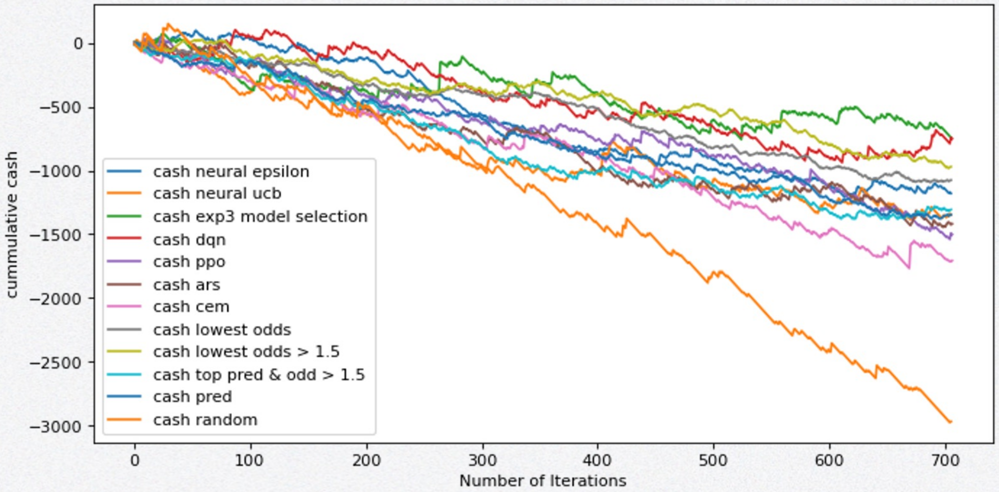
Action Set

- Algorithms include DQN, PPO, ARS, CEM, neural bandit, neural UCB.
- All run on the simplest setting (bet on 1 horse at a time with $10 bet)

Reward

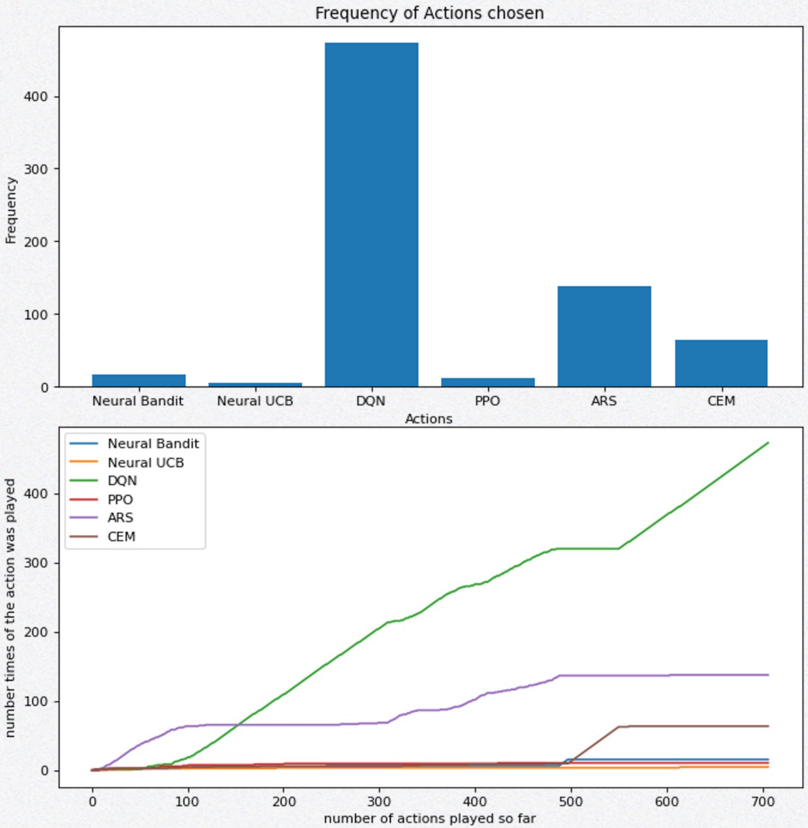
- Reward of selected algorithm by betting on its decided horse

Result



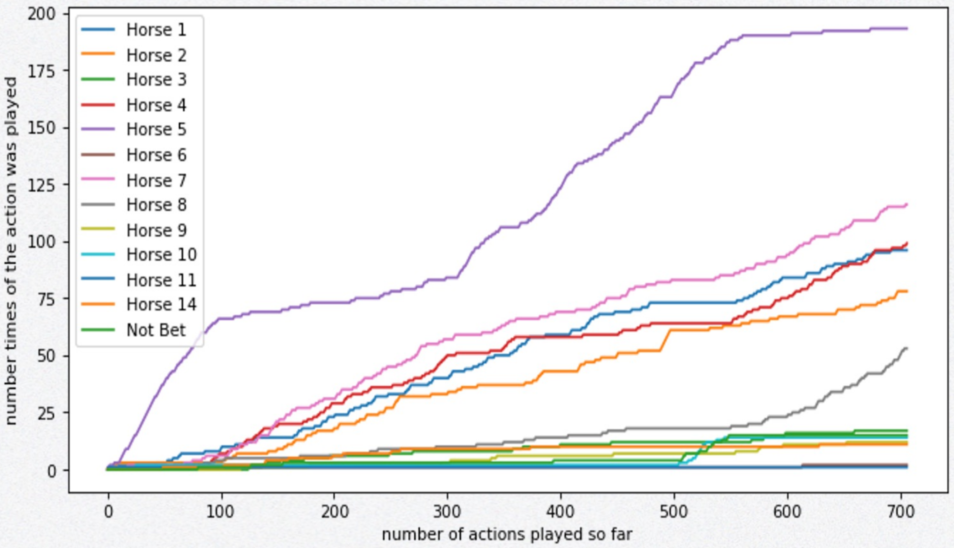
Using EXP3 to do model selection **outperforms** any single algorithm!

Comparing algorithms by EXP3



- DQN most selected overtime and follow by ARS and CEM
- Bandit is less selected which shows its weaker performance compared to others

Observing betting strategy from EXP3



- Horse 5 is selected the most. And followed by 7, 11, 4, 2
- Not bet is seldomly chosen
- Almost all are not safe options but EXP3 doesn't lose much at the end

5

Conclusion

Conclusion

- Horse racing prediction model
  - Enhanced interpretability of random forest
    - Showed how features affect the results
  - Acceptable betting strategy
    - Based on error range
    - Reduced loss without missing a lot profits
- Horse betting strategies
  - Bandit algorithms
    - Not flexible (variable bet, unaware of state like cash balance, hardly profit for negative expected return)

    -> better use other algorithms
    - But can be used in other scenarios
    - Shown good performance in model selection
  - Other algorithms
    - Comparable accuracy and profits to the bandits

6

Q&A Section

The End
Thanks!