Hong Kong Jockey Prediction

LI Yunli, HONG Seoyoung
20659584, 20658322
HKUST MSBD 5013 TPZG Team

1 Exploratory data analysis

The explanatory analysis is the processing of observing and understanding of collected data from various angles. In other words, it is the process of intuitively looking at the data in a graph or statistical way before we started to train the model. This step is important for a better understanding of what the data represents and discovering the potential problems in data by examining the distribution of the data. Moreover, through the exploratory data analysis by looking at various angles, this will allow us to modify the strategy of building a predictive model. Exploratory data analysis can be done in four steps as follow: setting the target variable, observing the individual attributes values of the data.

1.1 Target variable

Our project is aiming to predict which horse will win in the first place and which horses will be placed in third place. So, our target is predicting '*Ind_win*' and '*Ind_pla*', which are the indicators of the winner of the race.

1.2 Observe the individual attribute values of the data

By looking at the graph for the categorical feature, we have noticed that racecourse is only two categories, which might need to be converted as 0 and 1 for making this value available to train the model while track feature has

10 levels. Also, the most game placed in 1200m with 34439 number of the game and most of the horses are age 4 to 5.

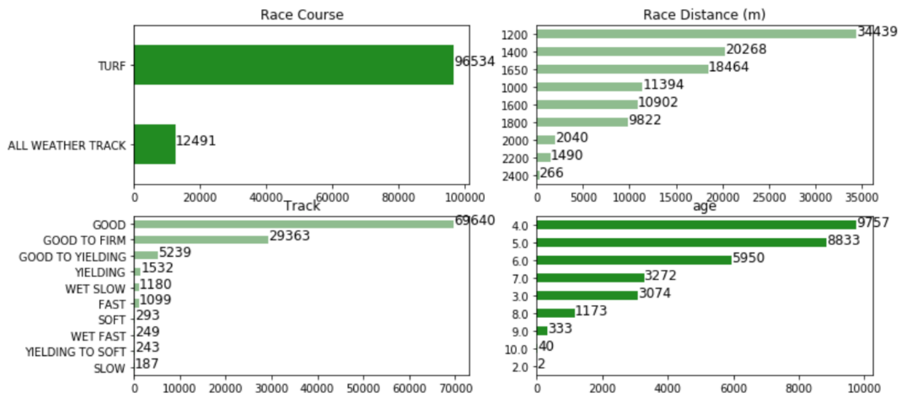


Figure 1: Histogram for Categorical value

In terms of continuous value, we used boxplot to get statistical value by categories. With the boxplot, there are few missing values and the outlier exist in ‘*horseweight*’, that need to be handled because those missing value might affect the machine’s performance. we also can observe median and every for each categories expressed in visually in the box plot.

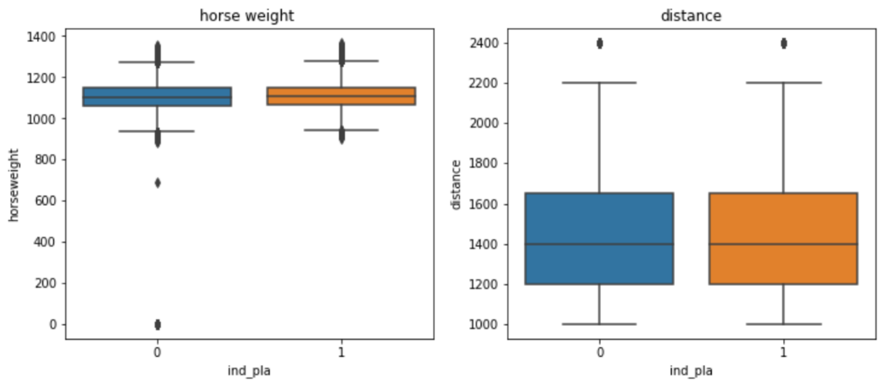


Figure 2: Boxplot for Numerical value

1.3 Relationship between features

By making a correlation table between all 45 features, there is a highly related feature. Those feature '*distance*' and '*rfinishm*' have the greatest value with 0.99775, since naturally the more distance has, the longer time horse need. Features '*rm*1' to '*rm*6' are also highly related as we can expected because those features represent each section's finish time in one game. P4, p6, and distance have a high correlation value as well. There is also a negative correlation. Except those features with correlation value over 0.5, we decided to filter the value over 0.2 for training our model, choosing 10 features, such as *age*, *tname*, *jname*, *rating*, *track*, *bardraw*, *besttime*, *going*, *win_t*5, *place_t*5, *ind_win*, *ind_pla*. So, here is correlation matrix for the features we have chosen to make all the variable to be independent so that the model performs better.

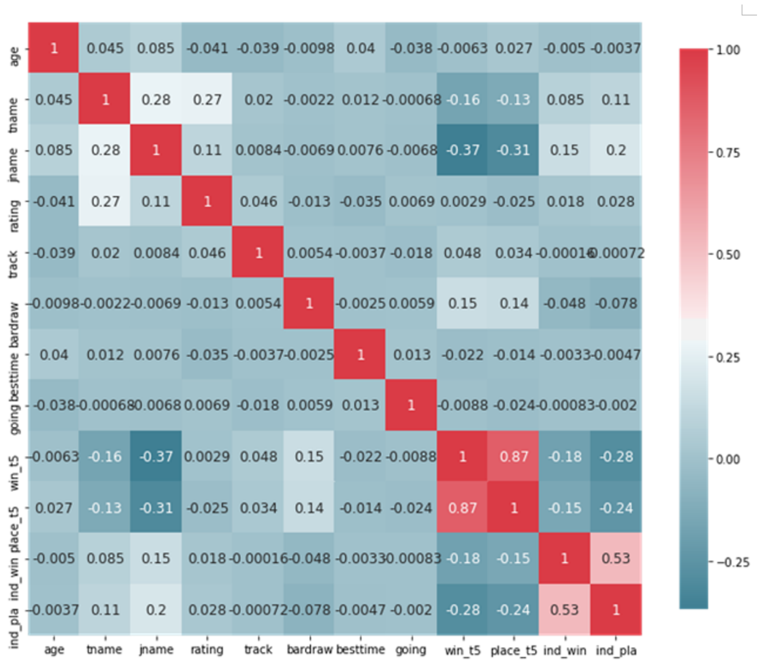


Figure 3: Boxplot for Numerical value

2 Idea and rationality

The purpose of this project is to predict the winning horse using the information in the dataset. However, using all the feature in the dataset will cause the overfitting so we have studied about the domain knowledge for feature and the model.

2.1 Feature Selection Rationality

As we mentioned above, we have selected a feature that independent each other by using the correlation relationship. In this section, we will look through those features rationality and model rationality.
Firstly, the racehorses age range is from 2 to 10. Four-year-old horses and five-year-old horses accounted for the largest number at 9757 and 8833. In general, the younger the racehorse, the higher the ranking.
Given the feature trainer and jockey, these are also important for predicting which horse will win because experienced jockey will be able to recover quickly from the unexpected situation such as a wide draw, a knock, or being blocked during the race. Besides, the ability of the jockey is the difference in handling during a race, which is critical for improving the chances of winning. Also,the trainer's performance is different depending on which track the horse will run. Even though the overall trainer's ranking is low, on each one's favorite track they could have good performance.
In general, if the horse starts the race from the inner circle, that might have been a competitive advantage. However, there is a room that we have to consider because it can be different depending on the horse's running style and the horse's favorite track and distance.

2.2 Model Selection Rationality

In real word problem, the classic prediction methods, such as linear regression, is not suitable to describe the complex relationship between the features. Such difficulties can be solved by applying Artificial Neural Networks(ANNs). ANN is an adaptive system which means that the parameter in the model can be changed during the operation toward the best prediction results by putting layers in between input and output. Even though ANN has the limitation because of the randomness of choosing the first value to operate the model, Artificial Neural Network is worthy to apply in this project expecting further improvements in the future.So, we decide to apply the Artificial Neural Networks model in our project.

3 Feature engineering

Data preprocessing is a very important step to produce meaningful results for analyzing and building a model in data-driven work. Raw data contain a lot of noise, unstable and missing value. Therefore, such data can cause serious error in the predictive model. So, this step cannot be treated as minor step. In this project, data preprocessing conducted in 4 steps which are dropping the variables, treating missing value, one-hot encoding and changing the data scale.

3.1 Deletion

By checking the information about the dataset, we have noticed that there are many null values especially in $rm1$ to $rm6$, $p1$ to $p6$, $m1$ to $m6$ and $d1$ to $d6$. All nan value for those features replaced to '0' at the first. However, because the number of the missing value is greater than the number of observations and highly related each other, we consider those features might have not been informative for predicting which horse will win in a game. So, those columns are dropped in this project.

3.2 Missing value

For inferring the missing value, mean, median and mode are the most common way of the averaging method. *besttime*'s missing value replaced the mean of the *besttime*'s mean. There are also a few missing values on *jname* and *tname* which are a string type of data. Those features are quantified by the definition we made which will explain in the next section. Obviously, if there is no jockey's information and trainer's information, there is no clue for how we quantify the missing value, so we filled with 0. Because if there is no record for a previous game. It means that they are newbies that have no experience, it can be a reasonable replacement.

3.3 One hot encoding

because the machine only can use the numerical data for training, the categorical feature needed to apply one-hot encoding, which is often used for converting into binary vectors. If categorical values that have no relationship between them, those can be converted as 0 or 1 as the rule we made. For instance, *track* feature has two values which are $turf$ and $all\ weather\ track$. These two values need converting if the feature has $turf$, replaced It as 1, otherwise replace it as 0. On the other hand, if the relationship between variable exists, the feature can be coded integer value depending on the level. For example, *going* has the level. Depending on the track condition, the observation varies from fast to yielding to soft. So, this feature can be coded 1 to 10 without losing the relationship between data.

3.4 Change the scale

The data format of *besttime* is not suitable to train the model because it is a string type. To change the string to a numerical value, we made a function for *besttime*, which splits the string by spliters and calculate in seconds units.

3.5 define a new feature

As mentioned above, *jname* and *tname* are a string type that needs to be converted into a numerical value. The initial idea of how we quantify this value is measuring the jockey and trainer's experience. We define experience as the ratio of the total number of observations to the number of participation games of that jockey or trainer. Those who have no information, considering as newbie, gave 0 value, which means no experience before.

4 Prediction methods & results

Methods we have submitted are listed as follow:

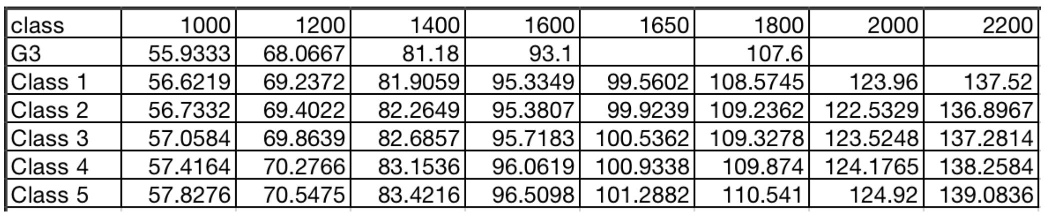
Index	Method	Submitted on
1	Criteria table	Week 0(Trial submission)
2	Score calculation	Week 1
3	Neural Network version 1.0	Week 2
4	Neural Network version 2.0	Week 3 - 5
5	Neural Network version 3.0	Week 6

Table 1: Prediction methods

4.1 Criteria table

Recap: Setting up a criteria table and comparing each horse's latest performance with it.
Submitted on: Week 0 - 30/09/19

Dividing the data by race class and distance and filtering out only the top 3 place horses, we created a criteria table `BestAvgTable.csv` on finish time which indicates the best record of each class on different distance.



class	1000	1200	1400	1600	1650	1800	2000	2200
G3	55.9333	68.0667	81.18	93.1		107.6		
Class 1	56.6219	69.2372	81.9059	95.3349	99.5602	108.5745	123.96	137.52
Class 2	56.7332	69.4022	82.2649	95.3807	99.9239	109.2362	122.5329	136.8967
Class 3	57.0584	69.8639	82.6857	95.7183	100.5362	109.3278	123.5248	137.2814
Class 4	57.4164	70.2766	83.1536	96.0619	100.9338	109.874	124.1765	138.2584
Class 5	57.8276	70.5475	83.4216	96.5098	101.2882	110.541	124.92	139.0836

Figure 4: BestAvgTable.csv

And based on this table, we collected the racing horse data(which is an unnecessary step and we have not fixed it until the third submission) and compared their latest 3 average performances on the same distance and class, subtracting the corresponding criteria number and storing them in a list called `prediction`.

Then we sorted the prediction list as as descending order. The bigger the number is, the better the horse performance is compared with the winning record. So we found the biggest element, marking the winprob as 1; And we found the the top 3 horses, marking the plaprob as 1. We did not manage to add betting strategy for this simple predictive version.

Problems: We did not use the correct submitting format. And the process required lots of manual sections which undermined the efficiency and accuracy of prediction. No betting strategy.

4.2 Score calculation

Recap: Preprocessing the training data, extracting key features and setting up thresholds for each one. Adding up every feature's score and ranking the scores.
Submitted on: Week 1 - 7/10/19

In the phase, we mainly focused on exploring the potential useful features for our model. After comparing the correlation, we selected some key features as follow:

Features	Description
tname	Trainer
jname	Jockey
age	Age of the horse
hid	Horse ID
Rating	Horse Rating
Ratechg	Rating change of the horse from previous race
Horseweight	Horse's weight
exweight	The handicapped weight carried by the horse
Besttime	The best finishing time of the horse on the race with same venue, distance and track

Table 2: Selected features

For each features mentioned above, we set up different thresholds in terms of different classes and if the figure is equal or greater than the corresponding threshold, we mark it as 1. In the end of the program, we added up every label and got scores for each horse who would participate in the games. The bigger the score, the better chance the horse is going to win. Thus, by sorting the scores in descending order, we marked the *winprob* of the horse with greatest score as 1, otherwise 0; and the *plaprob* of the horse with the top 3 scores as 1, otherwise 0.

Problems: We still did not use the correct submitting format. The the setting of thresholds was intuitive and naive. As expected, the prediction result was not ideal. And also, no betting strategy.

4.3 Neural Network v1.0(Tensorflow v1)

Recap: splitting the training data in terms of different classes and preprocessing each class, transforming the selected parameters as numerical data for training.
Submitted on: Week 2 - 14/10/19

Features	Description
tname	Trainer
jname	Jockey
hid	Horse ID
Ratechg	Rating change of the horse from previous race
Horseweight	Horse's weight
horseweightchg	The horse's weight change from previous race
datediff	The date difference between the previous match and current match of the horse
Besttime	The best finishing time of the horse on the race with same venue, distance and track

Table 3: Training parameters on version 1.0

The neural network has 3 layers:

- **Input layer:** 10 parameters as listed above. All the input have been transformed to numerical value;

- **Hidden layer:** 7 nodes. We applied Relu as the activation function and after comparing the training outcomes with different nodes, 7 gave the fairly better performance within our trials. We also normalized each input in the hope of reducing the training loss;
- **Output layer:** Sigmoid as activation function, which returns the probability of each input. We used *ind_win* as training label, setting the threshold as 0.4.

Problems: We still did not use the correct submitting format, but it was a big step towards the right track. Because we had only trained one model of place winner, the *winprob* results were not accurate. However, the *plaprob* outcomes were not ideal either. Since neural network is like a black box, we did not know what's going on under the hood. After modifying parameter several times, the outcomes did not improve much. In addition, the predicted y were polarized. In some sense, it reduced the loss effectively because many *winprob* and *plaprob* would be predicted as 0, which is in line with the fact. However, the accuracy was still a problem. Moreover, we did not split the training and testing set. No betting strategy.

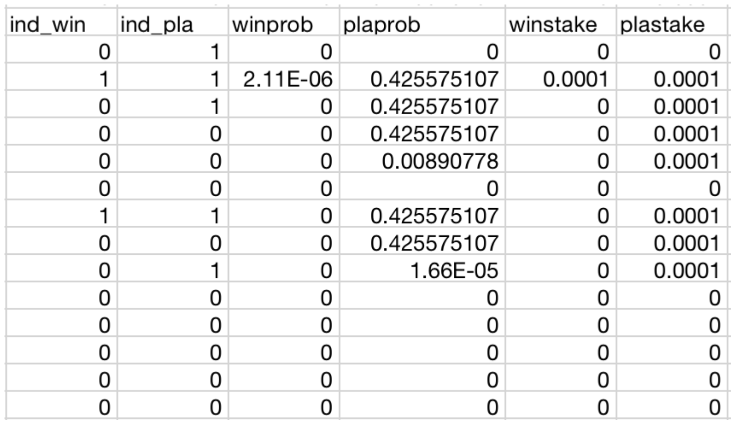
4.4 Neural Network v2.0(Tensorflow v1)

Recap: minor modifications of version1.0 but with the correct submitting format finally.
Submitted on: Week 3 - 5(21/10/19 – 04/11/19)

Features	Description
tname	Trainer
jname	Jockey
bardraw	Draw
rating	Horse rating
track	Race track (TURF, AWT)
going	Race track condition
win_t5	Win odds of the horse 5 minutes before the race
place_t5	Place odds of the horse 5 minutes before the race
besttime	The best finishing time of the horse on the race with same venue, distance and track

Table 4: Training parameters on version 2.3(week 5)

We had been mistaken the *hid* as *Bardraw*, which functions as a key factor in matches. After replacing it, the RMSE decreased a little. And we corrected the repetitive training problem that led to polarized outcome. However, due to unknown reason, the predicted results decreased sharply, approaching 0.



ind_win	ind_pla	winprob	plaprob	winstake	plastake
0	1	0	0	0	0
1	1	2.11E-06	0.425575107	0.0001	0.0001
0	1	0	0.425575107	0	0.0001
0	0	0	0.425575107	0	0.0001
0	0	0	0.00890778	0	0.0001
0	0	0	0	0	0
1	1	0	0.425575107	0	0.0001
0	0	0	0.425575107	0	0.0001
0	1	0	1.66E-05	0	0.0001
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Figure 5: Predition outcomes appraching 0 examples

Moreover, in week 5 we started to add betting strategy. We set a fix ratio of $default\ stake$ as 1/10000. Because of the cloesed-zero characteristics of our output, if $winprob$ is greater than zero, we place our bet. The same as place betting. In fact, the performance of this version is the best since we achieved our highest ranking since we got into the leaderboard. However, we think the model did not perform well, because the accuracy did not improve much and the decreasement of RMSE is due to the cloesed-zero trait of results. I find it intriguing since it presents as a problem for our model but also a cure for our performance.

Problems: Outputs were cloesed to zero. Naive betting strategy.

4.4.1 Results

TPZG			
Week	3	4	5
Rank	15	14	18
AverageRMSEwin	0.2875	0.2853	0.2783
AverageRMSEplace	0.4432	0.4126	0.4036
FinalWealth(Real odds)	1	1	0.9954
FinalWealth(Fair odds)	1	1	0.9987
RMSEwin_Rank	18	19	21
RMSEplace_Rank	17	13	11
Bankroll_Rank	10.5	10.5	16
Average_Rank	15.1667	14.1667	16

Table 5: Leaderboard results on week 3-5

TPZG			
Week	3	4	5
AverageRMSEwin	0.3159	0.2895	0.2927
AverageRMSEplace	0.5195	0.4532	0.4333
MeanRetPerDollar(Real odds)			0.049
TotalProfit(Real odds)	0	0	0.001
FinalWealth(Real odds)	1	1	1.001
MeanRetPerDollar(Fair odds)			0.2796
TotalProfit(Fair odds)	0	0	0.0055
FinalWealth(Fair odds)	1	1	1.0055
No.Horses	0	0	127
No.Games	0	0	18

Table 6: Backtesting results on week 3-5

4.5 Neural Network v3.0(Tensorflow v2)

Recap: changing tensorflow version from 1 to 2 and applying a keras model.
Submitted on: Week 6 - 11/11/19

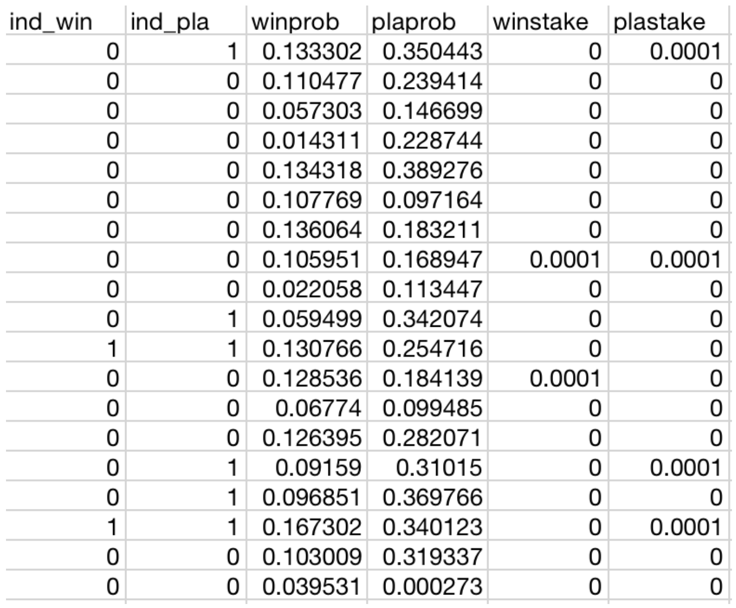
The main structure of the network remained the same. Some minor adjustments such as changing the hidden layer nodes to 15, adding cross validation and with *adam* optimizer and *mean_squared_error* as loss function. This version fixed the closed-zero results problem.

Since the closed-zero problem had been fixed, we could not adopt the same betting strategy as the previous one, which would definitely result in bankruptcy. We set a *mthresh* of 4, and if the product of *winprob* times *win_t*5 is greater than *mthresh*, we place our bet; For place betting, we applied the same methodology, adding a *mthresh_pla* of 1.5. If the product of *plaprob* times *pla_t*5 is greater than *mthresh_pla*, we place our bet.

Problems: Outputs were prone to low value, with *winprob* not larger than 0.3 and *plaprob* not larger than 0.5. Naive betting strategy.

4.5.1 Results

Since the leaderboard outcome of week 6 submission has not been revealed, we can not know the improvement of our latest model in general. Thus, only the backtesting result of `HR201911W2.csv` can be shown as below.



ind_win	ind_pla	winprob	plaprob	winstake	plastake
0	1	0.133302	0.350443	0	0.0001
0	0	0.110477	0.239414	0	0
0	0	0.057303	0.146699	0	0
0	0	0.014311	0.228744	0	0
0	0	0.134318	0.389276	0	0
0	0	0.107769	0.097164	0	0
0	0	0.136064	0.183211	0	0
0	0	0.105951	0.168947	0.0001	0.0001
0	0	0.022058	0.113447	0	0
0	1	0.059499	0.342074	0	0
1	1	0.130766	0.254716	0	0
0	0	0.128536	0.184139	0.0001	0
0	0	0.06774	0.099485	0	0
0	0	0.126395	0.282071	0	0
0	1	0.09159	0.31015	0	0.0001
0	1	0.096851	0.369766	0	0
1	1	0.167302	0.340123	0	0.0001
0	0	0.103009	0.319337	0	0
0	0	0.039531	0.000273	0	0

Figure 6: Result examples

TPZG	
Week	6
AverageRMSEwin	0.2782
AverageRMSEplace	0.4075
MeanRetPerDollar(Real odds)	1.2676
TotalProfit(Real odds)	0.0065
FinalWealth(Real odds)	1.0065
MeanRetPerDollar(Fair odds)	1.7668
TotalProfit(Fair odds)	0.0091
FinalWealth(Fair odds)	1.0091
No.Horses	49
No.Games	16

Table 7: Backtesting results on week 6

5 Summary and discussion

This project is very helpful as it is the first project that leads us into the world of data science. We both were not familiar with Hong Kong Jockey Competition and it took us a long time to figure out how to deal with the data. At the beginning of our work, we struggled relentlessly with feature engineering due to the lack of knowledge in this area. In a later stage, properly implementing and debugging neural network models became our main task.

Besides the methods mentioned above, we have also tried all other supervised models we have learned throughout this semester such as linear regression model, logistic regression model, random forest model and so on. However, the overall performances of these models were not as good as our NN models. The reason behind this might fall on the data preprocessing step. We did not redesign the the engineered features and some of them might not be the effective forms for different training models. Additionally, we might not have applied the proper optimizers on different models. After a significant amount of trials and comparisons, we stuck with NN and made great efforts on optimizing it.

All in all, this project offers a good opportunity for us to practice data technologies, requiring lots of hands-on learning and exploration of the data. Many simplified assumptions were made throughout this project in the interest of time and scope, much more work is needed to further resolve many of the detailed problems that will have huge impacts on the performance results. Execution costs is also something that we have thus far ignored, which might have a negative impact on the final results. Further investigation of Jockey matches, incorporating comprehensive understanding of different models and skillfully applying them in terms of different circumstances will be our future work.

[0]Archived submissions: `https://github.com/Lizyll/MSBD5013`