THE CHINESE UNIVERSITY OF HONG KONG

FINAL YEAR PROJECT REPORT

Predicting Horse Racing Result with Machine Learning

Author:
Yide LIU

Supervisor:
Prof. Michael LYU

A final report submitted in fulfillment of the requirements for the final year project

in the

LYU 1703
Faculty of Engineering
Department of Computer Science and Engineering

May 21, 2018

Declaration of Authorship

I, Yide LIU, declare that this work titled, "Predicting Horse Racing Result with Machine Learning" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the project is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

“No wife can endure a gambling husband; unless he is a steady winner.”

Thomas Dewar

THE CHINESE UNIVERSITY OF HONG KONG

Abstract

Faculty of Engineering

Department of Computer Science and Engineering

BSc degree in Computer Science

Predicting Horse Racing Result with Machine Learning

by Yide LIU

Neural networks with a large number of parameters are very powerful machine learning systems. While neural networks has already been applied to many sophisticated real-world problems, its power in predicting horse racing results has yet not fully explored. Horse racing prediction is to predict the finishing time and ranking of horses in a race. Horse racing has been one of the most intriguing sports and entertainment in Hong Kong owing to betting nature and the uncertainty of racing. Here we explore the possibilities of the neural network in predicting the final result and examined several deep learning techniques concerning the problem. We constructed an augmented racing record dataset, horse dataset and weather dataset and designed four approaches: sequence to sequence for sets, multi-layer perceptron with normalization and rank network. We showed that the most of the neural network approach can make accurate prediction in finishing time. Furthermore, our models outperforms the traditional betting system such as win-odds betting and performance-based betting in terms of in the horse ranking and final results.

Acknowledgements

We would like to express my special thanks of gratitude to our supervisor Professor Michael R. Lyu as well as our advisor Mr. Edward Yau who gave us the golden opportunity to do this wonderful project on the topic 'Predicting Horse Racing Result with Machine Learning', which also helped us in doing a lot of research and we came to know about so many new things and we are really thankful to them.

Secondly, we would also like to thank our friends who helped us a lot in finalizing this project within the limited time frame.

Contents

List of Figures

List of Tables

Chapter 1

Overview

This final year project focuses on predicting horse racing result with deep learning. Throughout this report we will demonstrate the work done during the first semester. This chapter offers a brief overview to this final year project and introduction to the topic. Moreover, it provides a glance of the related work and previous approaches on the horse racing predictions. It then introduces the difficulties in predicting horse racing results. Lastly it shows the objective of this project which is to accurately predict the final racing results and make promising net gain.

1.1 Introduction

Neural networks with number of non-linear hidden layers is proved to be highly expressive to learn complicated relationship between their inputs and outputs (Srivastava et al., 1989). Pragmatically, neural networks are shown to present its learning power in machine learning and becoming the dominant approach for many problems (Srivastava et al., 2014). Although they are introduced first in late 1950s (Widrow and Hoff, 1959), the increasing functionality of computer hardware and the use of graphics processing unit (GPU) enables the training processing recently. In the latest research in visual object recognition and then following in other research fields, neural networks are designed to goes deeper in layers(He et al., 2015) and a term called "Deep Learning", i.e. training a deep neural network with multitudinous layers,is appearing often in both the academic and the public society.

However, when researching on a new field of studies, traditional approach begins studying networks expressive power from very few layers (Huang et al., 2016). In visual object recognition, for example, begins with a primitive network called LeNet (LeCun et al., 1998) consisted of 5 layers and recent study of Highway Network (Srivastava, Greff, and Schmidhuber, 2015) and Residual Networks (He et al., 2015) surpass 100 layers. Latest research in neural networks in this year shows that very deep networks with exceeding 1000 layers have been studied and employed (He et al., 2016). While it helps to go deeper in network structure, the study of neural networks requires researcher to start from a very beginning of smaller version. These approaches are accord with the nature of neural networks: the exceeding number of neurons and parameters, the large carnality of hyper-parameter space, the appropriate score function and insidious structure issues. Since training a deep neural network takes days and months to run (Vanhoucke, Senior, and Mao, 2010), it is reasonable to train network with simple structure in order to accelerate the research progress.

Conventionally, neural networks are developed in steps to target sets of well-known academic problems. Neural networks are fully explored in those classical problems: text classification (Kim, 2014; Zhang, Zhao, and LeCun, 2015), and sentimental analysis (Santos and Gatti, 2014; Ouyang et al., 2015) in natural language processing, pattern recognition and object detection (Ren et al., 2015; Szegedy, Toshev, and Erhan, 2016) in computer vision, auto-encoding (Lange and Riedmiller, 2010) and noisy-encoding (Graves, Mohamed, and Hinton, 2013) in information theory. In spite of the promising performance in classical problems, the power neural network in other real-world problems is still under exploration. The complexity and unclear relationship makes it difficult to express the relationships.

One of these undetermined real-world problems is horse racing. The prediction of horse racing result has been a popular research topics in recent years. However, the research in this fields make little progress over these years. Few paper is published in academic domain after the prediction problem is firstly introduced in 2008. Two similar studies reviewed the power of neural network using different optimization techniques and performed finishing time predictions base on previous horses racing records (Snyder, 1978b; Williams and Li, 2008). Last year, LYU1603 (Tung and Hei, 2016) worked with two different approaches: binary classification on winning and logistics regression on horse finishing time. Their models realized positive net gains only with a threshold over 95% on betting confidence. Those studies provides different approaches to interpret horse racing problems but in contrast reveals a lack of understanding in horse racing predictions.

The horse racing events, while they are commonly considered a special kind of game, follows similar characteristics shared with stock market predictions where futures performances are related to precious and current performances to some extent. On the other hand, unlike games of perfect information such as GO[1] and PENTAGO[2] (Méhat and Cazenave, 2011), the optimal value function, which determines the outcome of a game, is not well-defined (Silver et al., 2016). While the horse racing prediction problem being a mixture of imperfect information and stochastic randomness (Snyder, 1978a), previous naive approaches fails to capture the critical information and produce few promising results. To the best of our knowledge, current horse racing prediction is limited and the results are under satisfaction.

In this final year project, we scrutinize features of horse racing events and predict horse racing results directly through finishing time. The rest of this report is organized as follows: Chapter2 illustrates how first-hand data is collected and structured. Moreover it provides prudent statistical analysis on related features and data standardization. In Chapter3-5, we review all approaches we tried during this term, including sequence to sequence for sets (Vinyals, Bengio, and Kudlur, 2015), deep neural network with normalization and rank network. In the last chapter, we conclude the accomplishment achieved in this term and give a brief future outlook.

1.2 Background

Horse racing is a sports to run horses at speed. Horse racing is not only a professional sports but also of a beloved entertainment of betting in Hong Kong. Every season, hundreds of races are held respectively in Shatin and Happy Valley racecourses at different tracks and distance. In each race, 8-14 horses runs in a row for the fastest and various bet types are created for entertainment on the result of the races.

Horse racing events are managed by the Hong Kong Jockey Club (HKJC). HKJC is a non-profit organization to formulate and develop horse racing, sporting and betting entertainment in Hong Kong. Moreover, it is the largest taxpayer and community benefactor in Hong Kong. It holds a government-granted monopoly in providing pari-mutuel betting on horse racing. In the history of horse racing in Hong Kong, the HKJC plays a essential role in promotion and regulation and combines the betting entertainment into this sports. "With strict rule enforcement, betting fairness

[1]https://en.wikipedia.org/wiki/Go_(game)
[2]https://en.wikipedia.org/wiki/Pentago

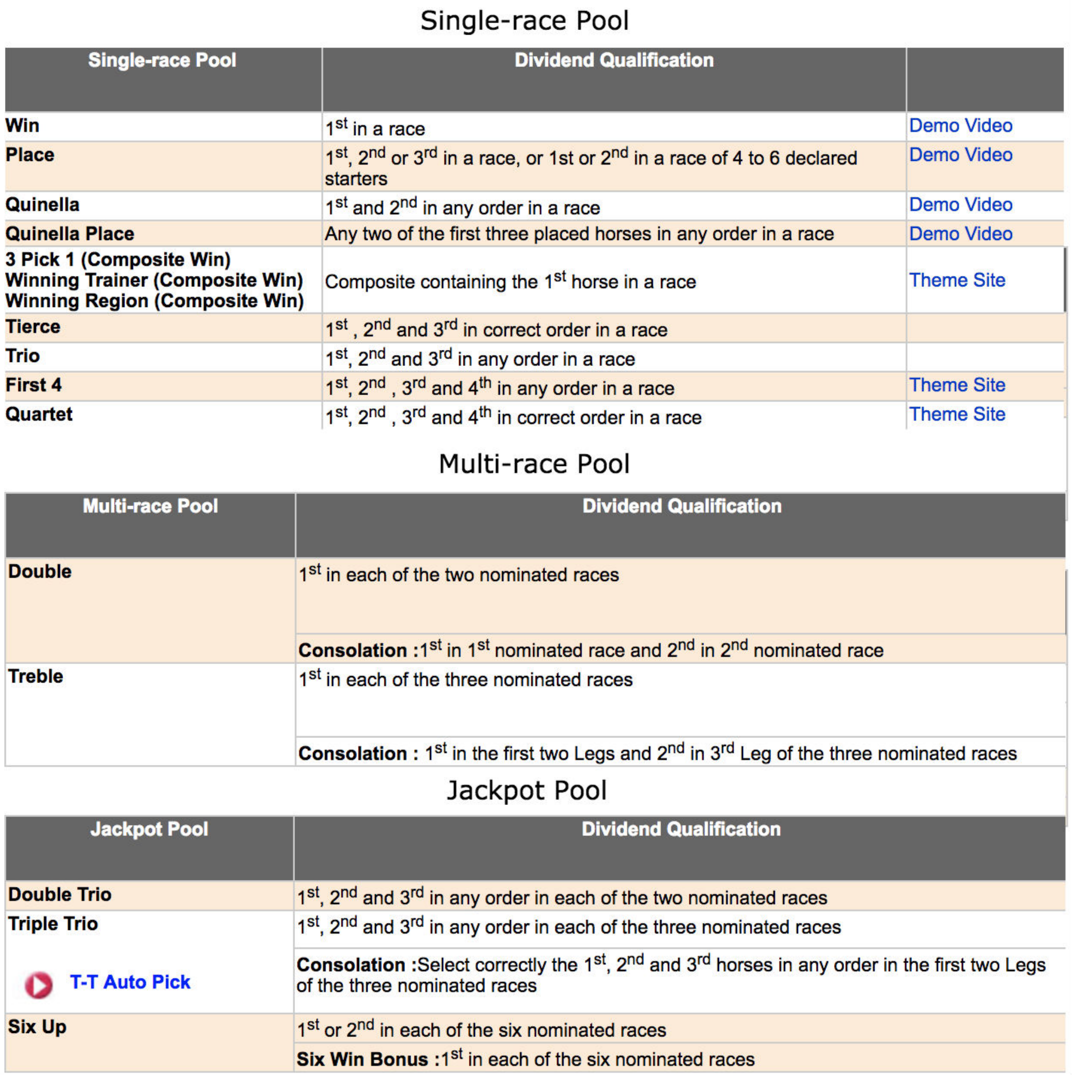
and transparency, the HKJC has taken the Hong Kong racing to a world-class standard and also earned itself an enviable global reputation as a leading horse racing organization."

1.2.1 Pari-mutuel betting

Betting is the most fascinating attraction of horse racing by the nature of pari-mutuel betting system. Pari-mutuel betting is a betting system in which the stake of a particular bet type is placed together in a pool, and the returns are calculated based on the pool among all winning bets (Riess, 1991).
Dividend is divided by the number of winning combinations of a particular pool. Winners shares the percentage of pool payout proportional to their betting stakes and taxes are reducted from the dividend in a particular ratio.

1.2.2 Types of bets

There are multiple types of bets of a single race as well as multiple races. The following figures from the HKJC website provides an explanation of each bet type.



Single-race Pool

Single-race Pool	Dividend Qualification	
Win	1st in a race	Demo Video
Place	1st, 2nd or 3rd in a race, or 1st or 2nd in a race of 4 to 6 declared starters	Demo Video
Quinella	1st and 2nd in any order in a race	Demo Video
Quinella Place	Any two of the first three placed horses in any order in a race	Demo Video
3 Pick 1 (Composite Win) **Winning Trainer (Composite Win)** **Winning Region (Composite Win)**	Composite containing the 1st horse in a race	Theme Site
Tierce	1st , 2nd and 3rd in correct order in a race	
Trio	1st, 2nd and 3rd in any order in a race	
First 4	1st, 2nd , 3rd and 4th in any order in a race	Theme Site
Quartet	1st, 2nd , 3rd and 4th in correct order in a race	Theme Site

Multi-race Pool

Multi-race Pool	Dividend Qualification
Double	1st in each of the two nominated races
	Consolation :1st in 1st nominated race and 2nd in 2nd nominated race
Treble	1st in each of the three nominated races
	Consolation : 1st in the first two Legs and 2nd in 3rd Leg of the three nominated races

Jackpot Pool

Jackpot Pool	Dividend Qualification
Double Trio	1st, 2nd and 3rd in any order in each of the two nominated races
Triple Trio	1st, 2nd and 3rd in any order in each of the three nominated races
T-T Auto Pick	**Consolation :**Select correctly the 1st, 2nd and 3rd horses in any order in the first two Legs of the three nominated races
Six Up	1st or 2nd in each of the six nominated races
	Six Win Bonus :1st in each of the six nominated races

Types of bets

1.3 Methodology

In last semester, we showed that the inconsistency between finishing time and final ranking of horses. Moreover, the pari-mutuel betting system strengthen the difficulties to have positive net gain. Here, we improved our research on the finishing time of horses and furthered this idea to learn the relative ranking of horses. Moreover, we devised strategies for game selection and only bet on horses with high confidence.

It is worth mentioning that it is still an open topic to model horse racing results and different approaches cannot avoid the deficiency in predictions and betting. In this section, we provides pros and cons for most common approaches in other researches.

1.3.1 Regression on finishing time

One way to deal with this problem is to build a regression model. In this project, we train a supervised neural network regressor on the finishing time and then ranks each horse base on the calculated predicted time. The model takes individual records of race information into account and then learn the mapping from the feature space to the finishing time. The unreasonably distributed finishing time reported in the last year is resolved with normalization and would be discussed in chapter 3. This approach is proven to be a suitable way to to study the horse racing prediction.

1.3.2 Binary classification on winning

The second to solve the problem is to predict whether a horse will win or not. However, directly doing binary classification of win or lose is unfeasible because the data will be imbalanced. The target class, "win" (or equally "1"), is under-represented with only less than 10% horses. In this case, problems arises due to asymmetric costs of misclassifying elements of different classes(He and Garcia, 2009). It would cause problems for normal classifier to learn the correct win distribution.

1.3.3 Multi-class classification on ranking

The third way to predict the horse racing result is to directly predict the ranking of each horse. However, it is difficult and unclear to interpret the classification result and the associated likelihood. The example below demonstrates the vagueness: If we look at the table horizontally, and we can derive the correct results 1^{st}, 2^{nd} and 3^{rd}. However, we can also look in this way by column and we may derive the wrong results of 1^{st}, 3^{rd} and 2^{nd}.

TABLE 1.1: Example: vagueness when doing classification on ranking

Horse/Place	1^{st}	2^{nd}	3^{rd}
#1	60%	40%	20%
#2	30%	60%	50%
#3	50%	40%	60%

1.4 Objective

In this project, we restrict our discussion on "win" and "place" bets, which is the most effective way to reflect the model efficiency and explore the relationship between betting gain and prediction accuracy. In other bet types, especially on multiple horses or multiple races, the probability to bet on all horses sharply lessen and the gain-accuracy relation is weak. In this regard, our objective is to create a model to predict horse racing results and beat public intelligence baseline in net gain.

The following table revisits the "win" and "place" bets and defines the prediction accuracy of each type ($Accuracy_{win}$, $Accuracy_{place}$).

TABLE 1.2: WIN and PLACE bet revisit

Bet Type	**Dividend Qualification**	**Accuracy Definition**
win	1^{st} in a race	Correct win bets out of all bets
place	1^{st}, 2^{nd}, 3^{rd} in a race	Correct place bets out of all bets

To distinguish the prediction accuracy of finishing time and $Accuracy_{win}$, $Accuracy_{place}$, we use both mean-square-error (MSE) and mean-absolute error (MAE) to measure the accuracy of finishing time prediction. Then we define $Accuracy$ to be associated with the model outputs only.

Chapter 2

Data Preparation

Recent research in various areas has shown that neural networks works well on different fields attributed to its flexibility of in learning complex patterns. However, this flexibility may lead to serious over-fitting issue and performance degradation in small datasets in the case of memerization effects (Arpit et al., 2017) (e.g. more parameters than training examples) when a model "memorizes" all training examples.

In general, neural network is prone to work better on large-scale and complete datasets especially when the model works on raw features. Yet, dataset constructed by LYU1603 only involves limited features and out-of-date records. Therefore, instead of adapting our models to take in data by LYU1603, we built 2 large datasets in last semester: racing records and raceday weather. In this term, we upgraded our racing records dataset and imputed the missing values from HKJC website. In addition, we collected and constructed the horse information dataset to help distinguish each horse and model horse strength. Our datasets contain all records originating from 2011 and the program automatically collects the newest race information.

Before feeding the data into the neural networks, we applied several techniques to process the data and improved model performance with feature engineering tricks. To choose and generate representative features, we stepped into the racecourse and experienced the game during last month. The feature engineering process involves expert knowledge and makes it simpler to design the model.

This chapter first illustrates the approach to collecting data and then describes the datasets and corresponding database structures. After that, it provides careful analysis on significant features taken in our model. The last section shows the preprocessing steps applied in this research.

2.1 Data Collection

Historical data are provided online by several commercial companies[1] and HKJC official website. A possible way to constructed our database is to purchase from such companies. However, due to the financial conditions and the quality of these sources, we switched to obtain the data by ourselves from the web.

Base on the characteristics on three datasets, we design three handy crawlers accordingly from: HKJC offical and Timeanddate. Historical data from 2011 to up-to-date data are collected and maintained in MySQL server. Our crawling system can automatically collect the latest racing, horse and weather information. It allows the user to predict the result of future races.

2.2 Data Description

2.2.1 Racing Record

The horse racing record dataset contains all racing data from 2011. Each row in the dataset represents a record keeping information of a certain horse in a selected race. The dataset contains 63459 records from 5029 races taken place in Hong Kong. The following table describes the useful features that directly crawled from HKJC website.

During the winter break, we upgraded the dataset to crawl complete information of the trainer and jockey, adding 367 trainers/jockeys information into the dataset. This augmentation is proven to help the model discover the correlation between horse and trainers/jockeys, which leads to improvement of the final result.

[1]hkhorsedb

TABLE 2.1: Useful racing features collected from HKJC website

Feature	Description	Types	Values
raceyear	Race year	Record Index	-
racemonth	Race month	Record Index	-
raceday	Race day	Record Index	-
raceid	Unique id of a race	Record Index	-
location	Location where a race take place	Categorical	ST, HV
class	Class of horses meaning strength of a horse	Categorical	1 to 5
distance	Distance	Categorical	1000, 1200, 1400, 1600, 1650, 1800, 2000, 2200
course	Track	Categorical	*Over 7 distinct values
going	Track condition	Categorical	* Over 10 distinct values
raceno	Race number in a race day, abstraction of race time	Categorical	1 to 8
horseno	Number assigned by HKJC to a horse	Categorical	1 to 4-14
horseid[a]	Unique identifier of a horse	Categorical	*Over 1000 distinct values
jockeycode[a]	Unique identifier of a jockey	Categorical	*Over 50 distinct values
trainercode[a]	Unique identifier of a trainer	Categorical	*Over 30 distinct values
draw	Draw of a horse	Categorical	1 to 4-14
actualweight	Weight of gears carried by a horse	Real Value	-
horseweight	Weight of a horse	Real Value	-
winodds	"WIN" Odds of a horse	Real Value	1-00
place_odd	"PLACE" Odds of a horse	Real Value	1-99
recordid	Unique id of a record	Record Index	-
place	Place of a horse **LABEL**	Categorical	1 to 4-14
finishtime	Finishing time of a horse **LABEL**	Real Value	-

Features generated after a race is abandoned since they cannot be collected before a race
Complete Categorical values are listed in Appendix

Apart from features directly crawled from HKJC website, we extracted advantageous features imitating professional tips. Base on the initial measured data, we derived 2 informative features, old place and weight difference, that explore the trend between two consecutive races of a horse. Moreover, we explicitly created a feature called "dn" to indicate whether the race is taken place during daytime or nighttime. The following table provides descriptions for these features. Last but not least, we merged the racecourse and location features into a single feature "lcg" to indicate different racecourse conditions in different location.

In comparisons with LYU1603, the extracted data is brainstormed and added by us originally.

TABLE 2.2: Extracted racing features

Feature	**Description**	**Types**	**Values**
origin	The place where the horse was born	Categorical	
dn	Day or Night	Categorical	D, N
old_place	Place of a horse in the last race	Categorical	1 to 4-14
weightdiff	Weight difference of a horse since the last race	Real Value	-

2.2.2 Horse

The horse dataset is retrieved from the official HKJC website. It contains a total of 3795 horses that ran in the races from 2011 to now. The dataset records age, gender and lineage of horses which helps the model to distinguish and compare different horses. The following table describes the horse characteristics to take into account.

TABLE 2.3: Horse features

Feature	Description	Types	Values
origin	Place of birth	Categorical	*Over 15 distinct nations
age	Horse age	Numerical	3-10
color	Horse color	Categorical	*Over 7 distinct colors
sex	Horse sex	Categorical	'Colt', 'Filly', 'Gelding', 'Horse', 'Mare', 'Rig'
sire	Father	Categorical	*Over 729 distinct horses
dam	Mother	Categorical	*Over 3514 distinct horses
dam's sire	Maternal grandfather	Categorical	*Over 951 distinct horses
horseid[a]	Unique identifier of a horse	Categorical	*Over 1000 distinct values

Features generated after a race is abandoned since they cannot be collected before a race
Complete Categorical values are listed in Appendix

2.2.3 Weather

We obtained weather dataset from Timeanddate. The dataset is obtained from historical data recorded in two observatory located near Shatin and Happy Valley racecourses, containing 5029 weather information on every race in Horse Racing Record database. Each row of weather dataset indexed by the racetime. The following table illustrates information contained in the weather datasets.

The weather data is first taken into model inputs in this research and statistical analysis is conducted in following section.

TABLE 2.4: Weather features

Feature	Description	Types	Values
raceyear	Race year	Record Index	-
racemonth	Race month	Record Index	-
raceday	Race day	Record Index	-
raceno	Race number in a race day	Record Index	1 to 8
location	Location where a race take place	Record Index	ST, HV
temperature	Temperature when a race start	Real Value	-
weather	Weather condition when a race start	Categorical	*Over 28 distinct values
wind_speed	Wind seed when a race start	Real Value	-
wind_direction	Wind direction	Categorical	*16 combination of diretions
humidity	Humidity when a race start	Real Value	-
moon	Moon phase of race day	Real Value	0-28[a]

[a] cycle of moon phase is 28 Complete Categorical values are listed in Appendix

2.3 Data Analysis

2.3.1 Racing Features

In this section, we will carefully examine the relationship between features in horse racing dataset and horse performance.

Class

Class is a strutured feature manually created by HKJC by horse rating. There are 5 classes in addition to Group and Griffin races. In handicap races, the standard upper rating limit for each class will be as follows:

TABLE 2.5: Class and rating standard in handicap races

Race Class	Standard Upper Rating Limit
1	120
2	100
3	80
4	60
5	40

The following figure from HKJC website illustrates the complete class system of HKJC races. It is worth mentioning that our project only focus on handicapped races among class 1 to 5.

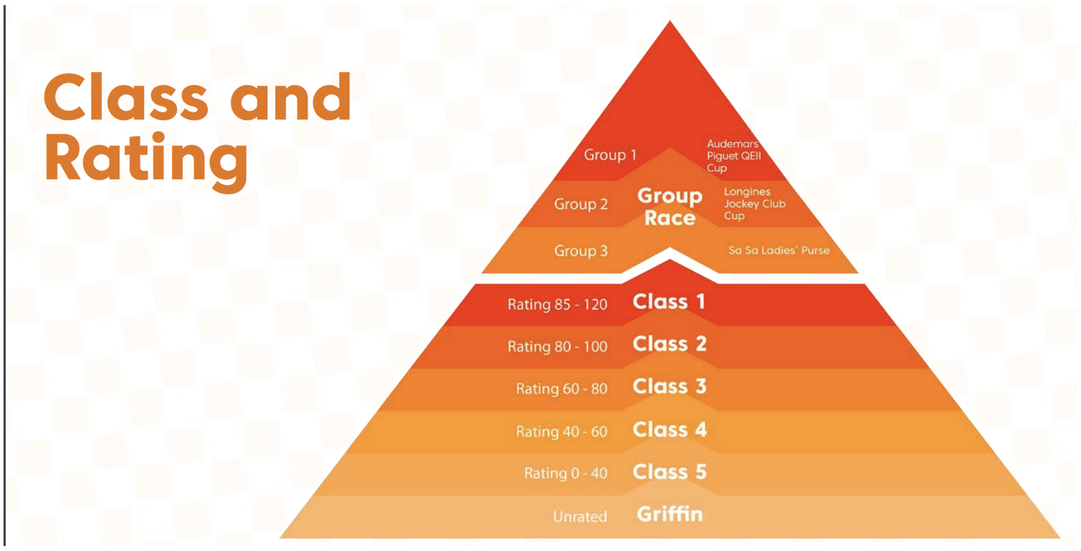


FIGURE 2.1: Complete class system

Horse rating is calculated by HKJC base on recent performance of a horse, however, due to the limitation of historical data the rating of a horse in race time is not recorded. One approach is to intimidate a horse power standard to train our models. Yet last year project has shown that ELO system only has little contribution to training performance. In result, we only use class to do anticipation.

The following table illustrates the correlations between class and horse performance. When finishing time is normalized by distance, we can observe a trend that the higher class of a horse, the quicker it finishes. Moreover, our experiments showed that the model would have better accuracy in predicting the final ranking of horses in class 1-2 than class 3-5, which shows that the upper class horses have stabler and predictable performance.

TABLE 2.6: Class Correlation Matrix

	finishtime	**place**	**class**
finishtime	1.000000	0.480295	0.328883
place	0.480295	1.000000	0.017722
class	0.328883	0.017722	1.000000

*Finishtime is normalized by distance to represent horse performances.

Win odds

Win odds reflects the public intelligence on the expectation of horses in a single races. In Pari-mutuel betting system, the larger the win pool of a horse, the lower of win odds of a horse. The correlation between both finishtime and place shows that public intelligence is wise enough to predict horses.

Although following public intelligence improves the model predictions, the betting system by design reversely makes betting difficult: betting on the lowest rate will result in negative net gain in statistical expectation. Therefore, enhancing model prediction is necessary and challenging.

TABLE 2.7: Winodds Correlation Matrix

	finishtime	**place**	**win odds**
finishtime	1.000000	0.480295	0.227527
place	0.480295	1.000000	0.472042
win odds	0.227527	0.472042	1.000000

*Finishtime is normalized by distance to represent horse performances.

Weight

Most races are handicaps and more weights are added to the stronger runners in order to equalize the chance of winning. The allocated weight is determined by the horse recent performance (proportional to the horse rating assigned by HKJC).

In our research, we inspect relationships between performance and weight-related features. There are three features in total: carried weight, horse weight and actual weight. Carried weight is the weight of gear attached to a horse in handicap race. Horse weight is weight of the horse itself. Actual weight is a extracted feature which is the sum of carried weight and horse weight. Actual weight represents the real weight of a horse.

The following correlation matrix portrays linearity between all weight features.

TABLE 2.8: Weight Correlation Matrix

	finishtime	**place**	**carried weight**	**horse weight**	**actual weight**
finishtime	1.000000	0.480295	0.005194	-0.057819	-0.056780
place	0.480295	1.000000	-0.099052	-0.030117	-0.039850
carried weight	0.005194	-0.099052	1.000000	0.037630	0.138260
horse weight	-0.057819	-0.030117	0.037630	1.000000	0.994897
actual weight	-0.056780	-0.039850	0.138260	0.994897	1.000000

*Finishtime is normalized by distance to represent horse performances.

The table illustrates that neither of weight features is closely related to finishing time or place. The statistics is convincing since the handicapped rules are designed to adjust the horse power. Yet we take a deeper look into these features trying to understand whether such features will help in predictions. We randomly select two horses and analyze the correlation between weight features and their performances.

TABLE 2.9: Weight Correlation Matrix of "A003"

	finishtime	**place**	**carried weight**	**horse weight**	**actual weight**
finishtime	1.000000	0.629141	0.702231	-0.725612	-0.574200
place	0.629141	1.000000	0.190517	-0.555372	-0.595973
carried weight	0.702231	0.190517	1.000000	-0.641935	-0.337157
horse weight	-0.725612	-0.555372	-0.641935	1.000000	0.938297
actual weight	-0.574200	-0.595973	-0.337157	0.938297	1.000000

*Finishtime is normalized by distance to represent horse performances.

TABLE 2.10: Weight Correlation Matrix of "L169"

	finishtime	**place**	**carried weight**	**horse weight**	**actual weight**
finishtime	1.000000	0.244387	0.160243	-0.027932	0.031567
place	0.244387	1.000000	0.038202	0.149508	0.147911
carried weight	0.160243	0.038202	1.000000	0.105043	0.448122
horse weight	-0.027932	0.149508	0.105043	1.000000	0.936099
actual weight	0.031567	0.147911	0.448122	0.936099	1.000000

*Finishtime is normalized by distance to represent horse performances.

We can conclude from the above table that weight features are working differently with horse performance of individual horses. The horse "A003" may suffer from obesity in this case and losing weight help it perform in a large scale. On the contrary, the horse "L169" may be of well health and weight features have minor influence on its performance.
Another possible explanation for the above matrices is that the horse "L169" is in a less competitive class while the horse "A003" competes with strong opponents and the change in weights influence "A003's" performance greatly.

Weight difference

Weight difference is the difference of horse weight from the last race. When the weight difference is large, it indicates that the horse is growing fat and lose competitiveness. The feature to some extend reflects the health conditions of a horse and in this part we try to identify the relationship.

TABLE 2.11: Weight difference Correlation Matrix

	finishtime	**place**	**weight difference**
finishtime	1.000000	0.480295	0.061861
place	0.480295	1.000000	0.073577
weight difference	0.061861	0.073577	1.000000

*Finishtime is normalized by distance to represent horse performances.

Combined with the analysis in the preceding sections, we believe that individual weight difference influences the performance in different ways even though it shows no relation with the overall performance.

Old place

Old place can be strictly defined as the place of a horse in its previous game. Consistent performance relies heavily on the strength of the horse nature. Even though horse racing is affected by a number of factors (for example, running with better opponents), a horse with an excellent place in the previous race tends to runs consistently as shown in the following correlation matrix. We claim that the nature of horse is invariant to minor environment changes in most cases.

TABLE 2.12: Old place Correlation Matrix

	finishtime	**place**	**old place**
finishtime	1.000000	0.480295	0.144424
place	0.480295	1.000000	0.236155
old place	0.144424	0.236155	1.000000

*Finishtime is normalized by distance to represent horse performances.

2.3.2 Horse

Horse features such as horse breed provides a clue to help determine which horse will have better performance. Statistically speaking, a horse have similar performance and race results to its sire (Father) and dam (Mother). For example, some sires are known to produce horses that run better in short distance while others produce horses that stand out in longer distance. However, when judged from outside, the characters of various horse breeds are expressed by some critical features: sex, color and age. In China, a certain kind of horse with special color, ferghana horse (汗血寶馬), is believed to be the most precious and strongest horse among all breeds. Similarly, in analysis of horse features, it is shown that horses with the color Roan run faster in all weather track and have the best chance of winning.

Color

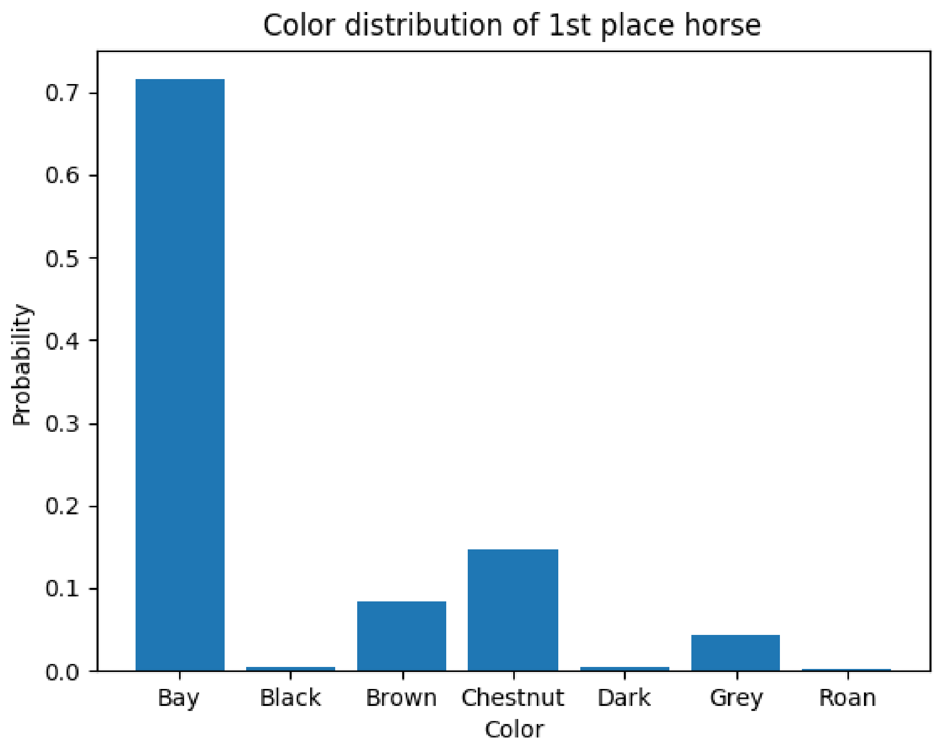


FIGURE 2.2: 1st place distribution over horse colors

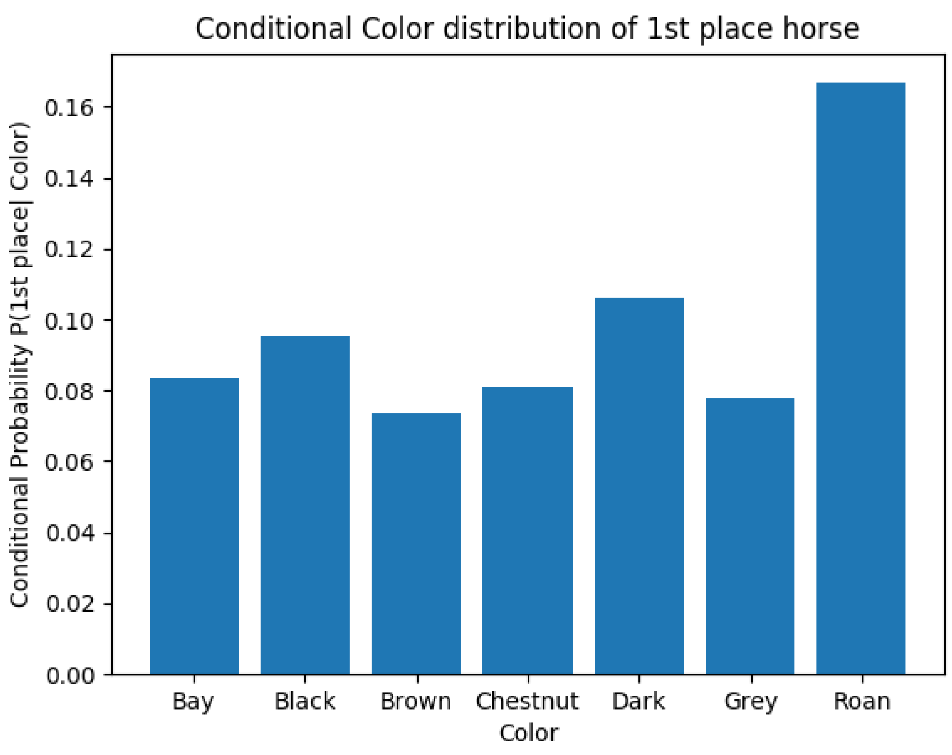


FIGURE 2.3: 1st place possibility over horse colors

The figures illustrate the 1st place distribution and likelihood. The above figures show that most of the races are won by the horse of color Ray following by Chestnut, Brown and Grey. Other horses with different colors have fewer time to win the game, taking a small percentage less than 5%. However, the 1st place distribution can be affected by the prior $P(Colors)$.

To identify the relationship between horse colors and horse strength, we explore the 1st place likelihood over horse colors. As shown in the second figures, the likelihood distributes differently from the previous analysis. Although the Roan horses win the least amount of races, they are the most competent breed with over 16% of likelihood to win. The large difference between 0.1% winning times and 16% of likelihood indicates that the Roan horses are rarely raced yet more probable to win. In a result, we can conclude the color is a critical feature to infer the winner of a race.

Sex

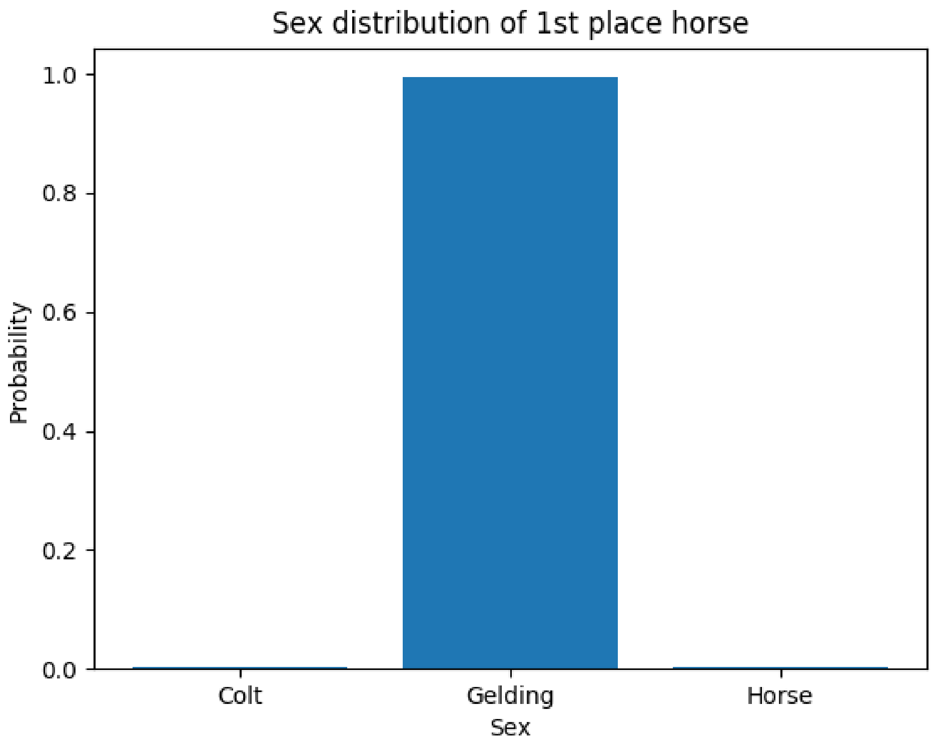


FIGURE 2.4: 1st place distribution over horse sexes

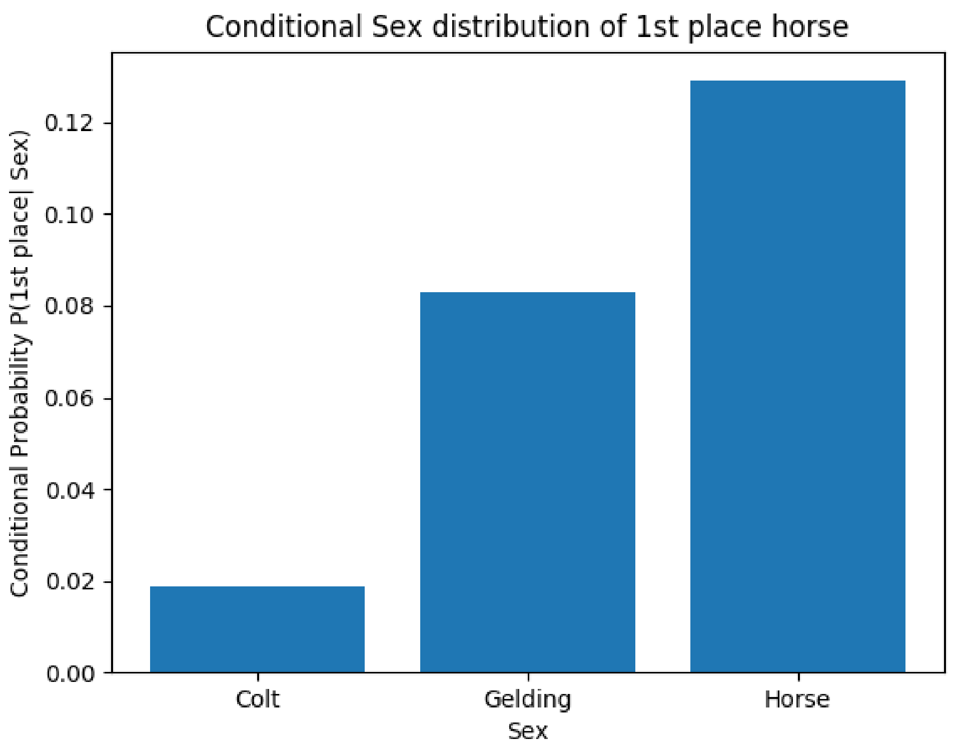


FIGURE 2.5: 1st place possibility over horse sexes

Similarly, most of races are won by Gelding sex, taking around 98% of all races. Yet the most likely kind of horse to win is the Horse sex.

Age

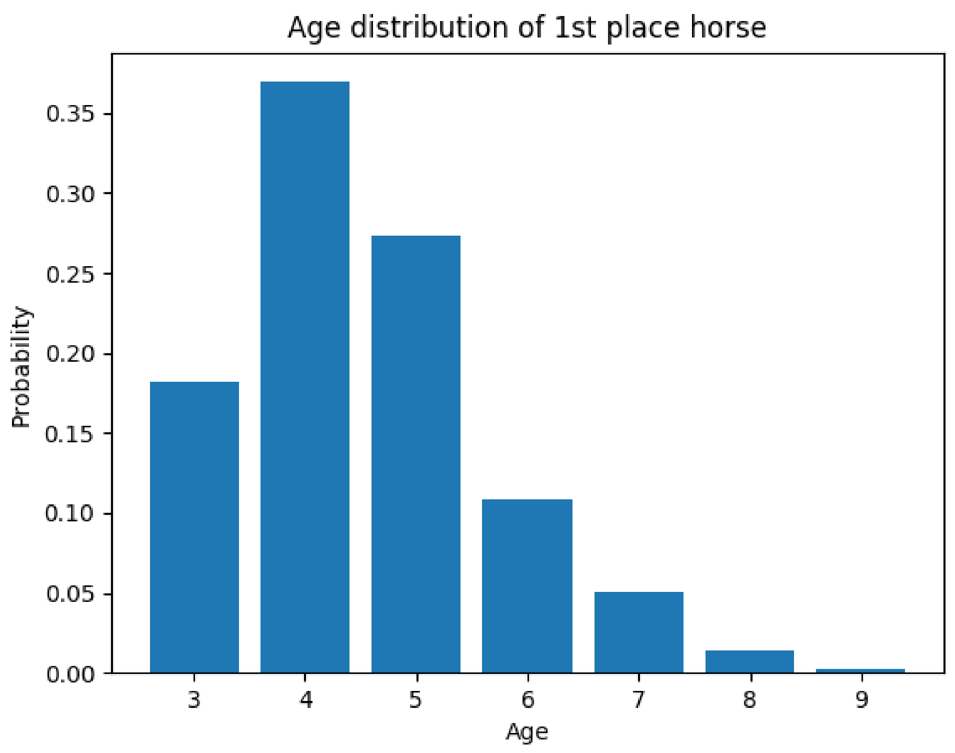


FIGURE 2.6: 1st place distribution over horse ages

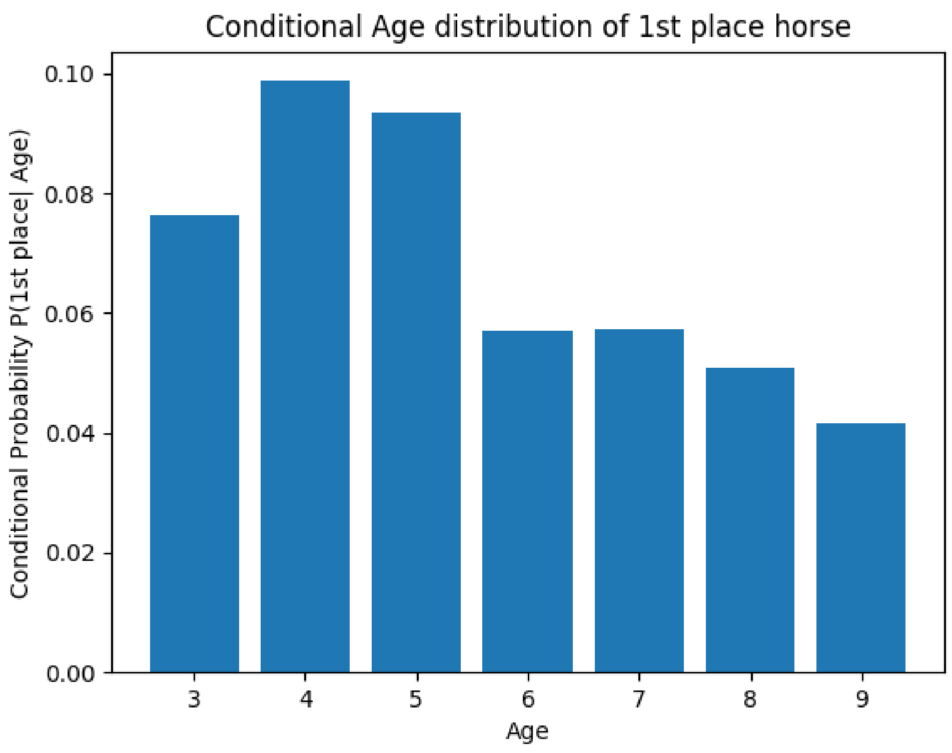


FIGURE 2.7: 1st place possibility over horse ages

Horse age are mostly correlated with performance in terms of health, durability and explosiveness. It be observed that the horses are in their zenith in age of 4 and 5. When the horses gets older, their performance and strength decline rapidly along with the age. To maximum the likelihood to win, it is reasonable to bet on horses in their teens.

2.3.3 Weather

Weather conditions contribute to horse racing finishing time. As shown in last semester, weather features are useful in both predicting the finishing time and final ranking. The table below from term 1 report shows the model can successfully minimize the finishing time prediction error and in the meantime improve the bet accuracy.

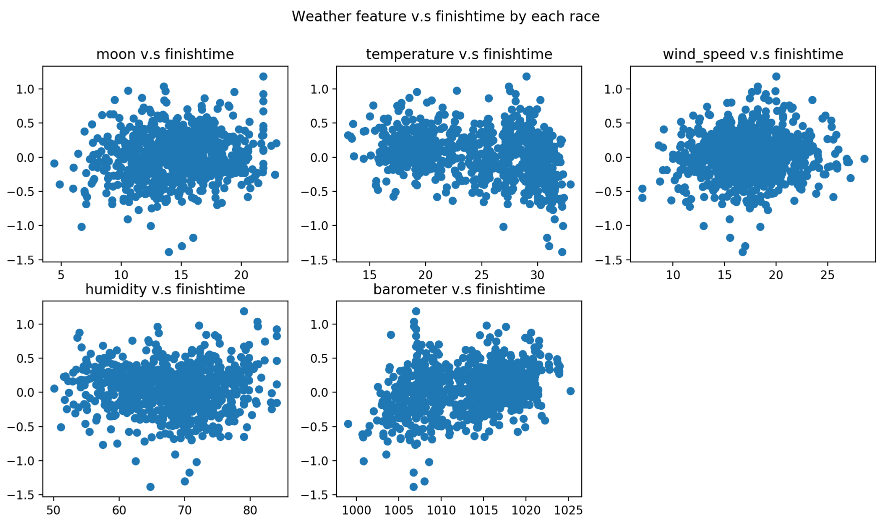
The table showed that using weather features in the model is necessary as it not only helps to reduce MAE loss but also increase the accuracy in the final ranking (and betting). The expressive power of neural network learns the relationship between horse and weather and is able to tell the best horse in different environment.

TABLE 2.13: Model performance with/without odds or weather

Models	Model 00	Model 01	Model 10	Model11
Loss	515.2	461.2	556.4	417.7
$Accuracy_{win}$	0.08367	0.07029	0.08090	0.10742
$Accuracy_{place}$	0.08753	0.10031	0.09063	0.09461
Net gain(WIN)	-1087	-991	-1378	-568

* Binary digits represents with/without odds or weather respectively. For example, 00 means without odds and weather and 01 means without odds and with weather

When it comes to overall performance of the horses, average finishing time also varies in different weather conditions. In common sense, horses tends to be longer in raining days than in sunny days. Moreover, horses are prone to run faster under warmer temperature. Empirical results on average finishing time against different weather conditions show that common sense proves out to have high credibility.



*Finishtime is normalized by distance to represent horse performances.

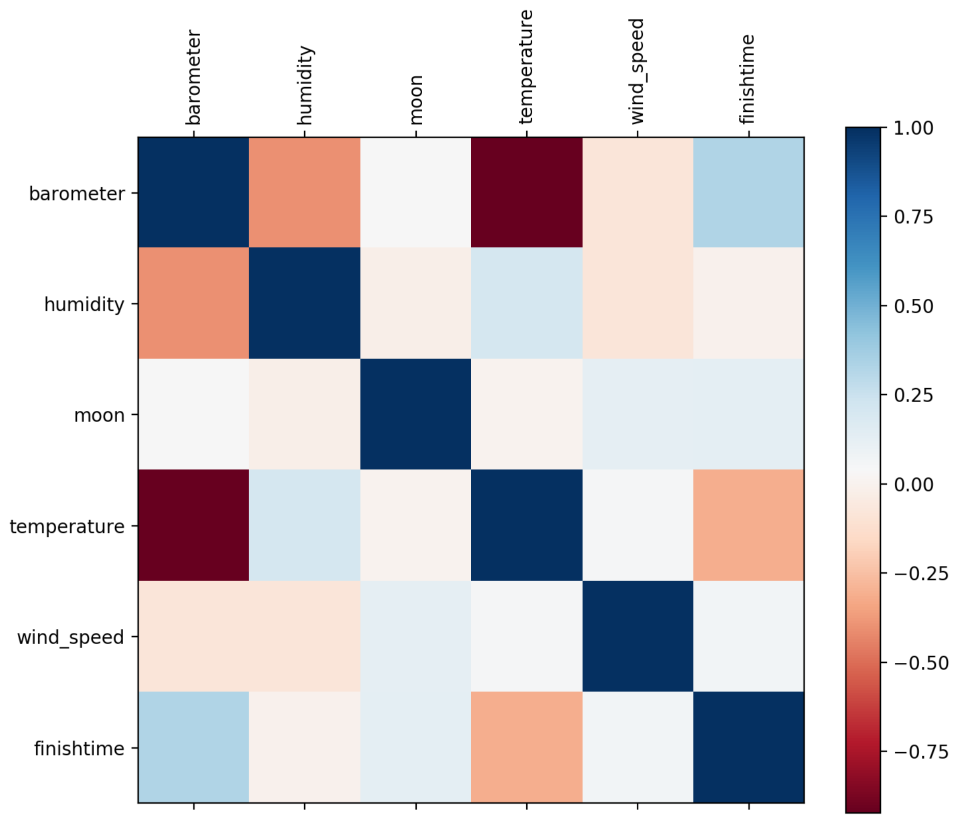
FIGURE 2.8: Average finishing time against different weather conditions

One possible explanation for the fluctuation in finish time is that, a change in weather regardless of its form, may have subtle influence on horse nature.

On the one hand, the weather can indirectly determine horses performances by slightly changing the course environment. For example, the condition of a race track plays an important role in the performance of horses in a race. A slight flutuation in humidity and raining can make a great impact in track surface density, porosity, compaction and moisture. In a result, the horse tends to run in a different speed.

On the other hand, the rise in humidity and temperature can affect the health condition and further emotions of horses. Horses are astute on their surroundings. When the environment changes quickly, they can easily get agitated and flee away. In general belief, horses themselves are responsible for the minor change of environment and the results differs naturally under different weather.

The following figure shows the correlations between the horses finishing and weather conditions collected in our research. The finishing time is shown to have strong correlations with some of weather features, such as temperature and humidity. However, relationship between other features is wanting to be discovered. These features indirectly influence the horse performance with a nonlinear relation to individual horse performance.



*Finishtime is normalized by distance to represent horse performances.

FIGURE 2.9: Average finishing time correlation with weather

2.4 Data Preprocessing

In this project, we joined the three datasets base on the unique race and horse identifier and worked with a large-scale dataset with over 50 features. We then split the datasets into two training and test sets. The training set contains race records from 2011 to the end of 2017 and the test set contains records after 2017. Furthermore, we are constantly making prediction on the latest race and simulate to bet on the selected horses.

2.4.1 Normalization

In this research, we perform *z-score* normalization on real value data columns in our datasets. The mean and standard deviation is calculated base on training set and then apply to test set to avoid information leak.

Normalization is a common requirement of for machine learning tasks in order to make training less sensitive to the scale of features(Sola and Sevilla, 1997). The most commonly used is $z - score$ normalization that scales individual samples to have unit norm, i.e. given the real value data matrix X, where the rows represent the individual records and the columns represent the features, the normalization transform the matrix into $X_{normalized}$, such that

$$X_{normalized,j} = \frac{X_j - mean(X_j)}{std(X_j)}$$

where X_j is the j^{th} column of matrix X.

2.4.2 Embedding Network

Categorical data are challenging to train and always mask valuable information in a dataset. It is crucial to represent the data correctly in order to locate most useful features in the dataset and downgrade the model performance. Multiple encoding scheme is available across the market. One of the most simple one is through one-hot encoding in which data is encoded into a group of bits among which the legal combinations of values are only those with a single high (1) bit and all the others low (0) (Harris and Harris, 2010). However, one-hot encoding suffers from its high cardinally and the feature space can grow exponentially making it unfeasible to train.

In this semester, instead of using handcrafted encoding schemes (such as one-hot encoding), we employ a small network for each categorical data. The embedding network map the categorical values to embedding vectors of assigned dimension. The architecture of the embedding network are as followed. Given the categorical input z of dictionary size N and the output dimension M, the network employs a simple linear transformation over the entire dictionary d:

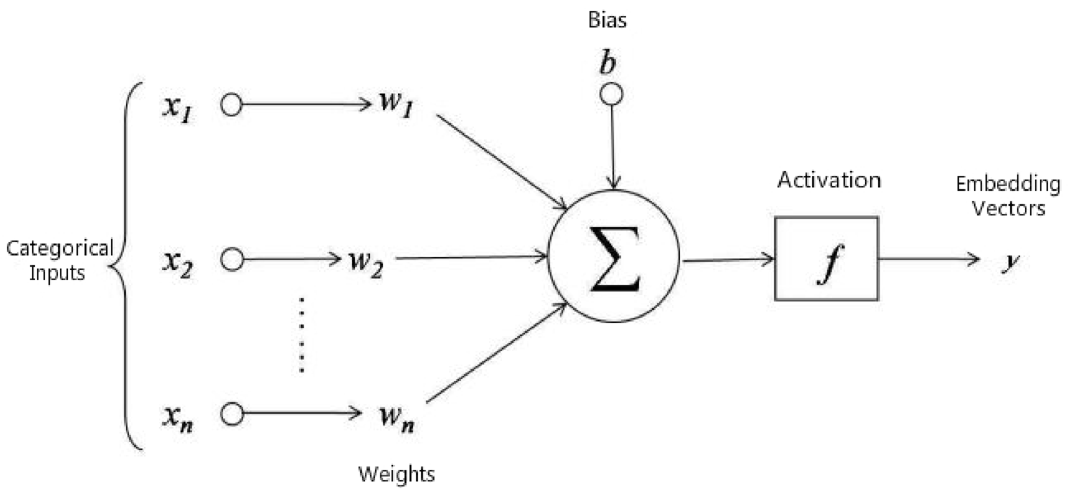


FIGURE 2.10: Embedding Network Architecture

Then the embedding network output a slice of dictionary base on the input by index look-up. The error only propagates onto the selected slice and updates that particular weights for the feature.

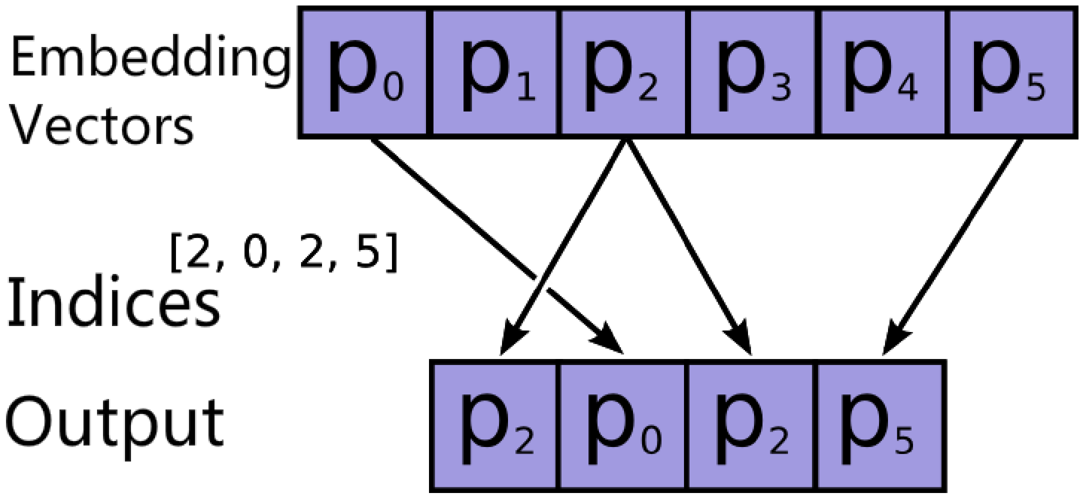


FIGURE 2.11: Embedding Network Output

Compared with human-made encoding schemes, this type of network only uses around thousands of parameters which can be well-trained within 10 iterations. The time needed for training is dramatically reduced because of the decreased feature space dimension. Moreover, the embedding network automatically learns to encode semantic meaning of the features during training. Overall speaking, the embedding network is a great balance point between performance and training time.

This network also enables us to select confident races to bet. By using embedding network, we are transforming the features into matrix representations where certain properties and similarities can be calculated by notation of distance. Different from feature columns approach in last semester, we have more access to the embedding network since we are able to extract the embedding of each feature instance. After we obtain the embedding, we calculate the Euclidean distances between the environment factors to discover the similarity between different races. We can then apply nearest neighbour/clustering techniques for final selection.

Chapter 3

Sequence to sequence for sets

As stated in the last term report, the deep neural network model predicts the horse racing results individually and the output finishing time is inconsistent within a race. Moreover, we observed that the model makes better bet decision in races with consistent finishing time. Therefore, a nature approach is to divide the racing records by races and work with different set of records race by race.

Recently, variable sized ordered inputs/outputs, namely sequences, have become the first option in supervised learning thanks to recurrent neural network (Sutskever, Vinyals, and Le, 2014). Many complex tasks that takes a sequence of observations can be tackled with the sequence-to-sequence (seq2seq) framework. Starting from 2014, the sequence-to-sequence framework are shown to be efficient in solving sequential data of variable input length. The framework has a potential in solving machine translation (Sutskever, Vinyals, and Le, 2014), Part-of-Speech parsing (Vinyals, Fortunato, and Jaitly, 2015), image captioning (Donahue et al., 2015) and other sequential problems.

However, horse racing and many other real-world problems that have variable sized inputs/outputs should not be interpreted as sequence but set. In 2015, (Vinyals, Bengio, and Kudlur, 2015) proposed a variant of seq2seq framework to address the set input and set output separately. Vinyals's experiments showed that the modification to the seq2seq framework works efficiently on set input problems with the power of recurrent neural network. Similarly, the ordering of horse number is randomly generated in horse racing, which can be viewed as a set input. In this regard, we researched the possibility to apply the framework on the horse racing prediction and designed several experiments to test the model.

In this chapter, we first introduce the mathematics underpinning for the seq2seq (for set) framework and then present our implementation of set-to-sequence (set2seq) framework. We showed that the set2seq model is unsuitable for the horse racing predictions with several prepared experiments and results.

3.1 Sequence to sequence (for sets) framework

Similar to vanilla RNN, the sequence to sequence (for sets) framework works on sequences of input and output data and learns to maximize the conditional probability. Yet the sequence to sequence framework uses a encoder/decoder approach, where the input is read completely using an encoder, which is either a RNN or LSTM . The final state of the encoder is then fed to a decoder LSTM whose purpose is to produce the target sequence, one token at a time (Vinyals, Bengio, and Kudlur, 2015). The framework works generally with variable size input and output data becasuse the final state contains the sequential information of the input which helps to maximize the conditional probability.

Formally, given a sequence pair (X, Y), where $X = (x_1, x_2, ..., x_s)$ is an input sequence and $Y = (y_1, y_2, ..., y_t)$ is an output sequence of possibly different length. The sequence to sequence framework learns the conditional probability $P(X|Y)$ using the chain rule to decompose:

$$P(Y|X) = \prod_{t=1}^{T} P(y_t|y_1, y_2, ..., x_1, x_2, ..., x_s) \tag{3.1}$$

To enable the sequence to sequence framework to work with set input, (Vinyals, Bengio, and Kudlur, 2015) proposed a modification associate 'memory' to LSTM encoder with content base attention. Attention mechanism is an associated memory on the framework which enable the network to review all the information from the input set. This mechanism introduces a context vector between the encoder and decoder. At each output step, the mechanism takes in all encoder output and computes the probability distribution of the source with the softmax function. By using attention mechanism, the framework can access global information and view the input sequence as a set.

3.2 Implementation

The model we built for predicting the horse racing results includes three main modules: read, process and output. The read module is responsible for reading the input, the process module is for encoding the records set input and the output module is to write the output sequence. The model designed is shown as follows.

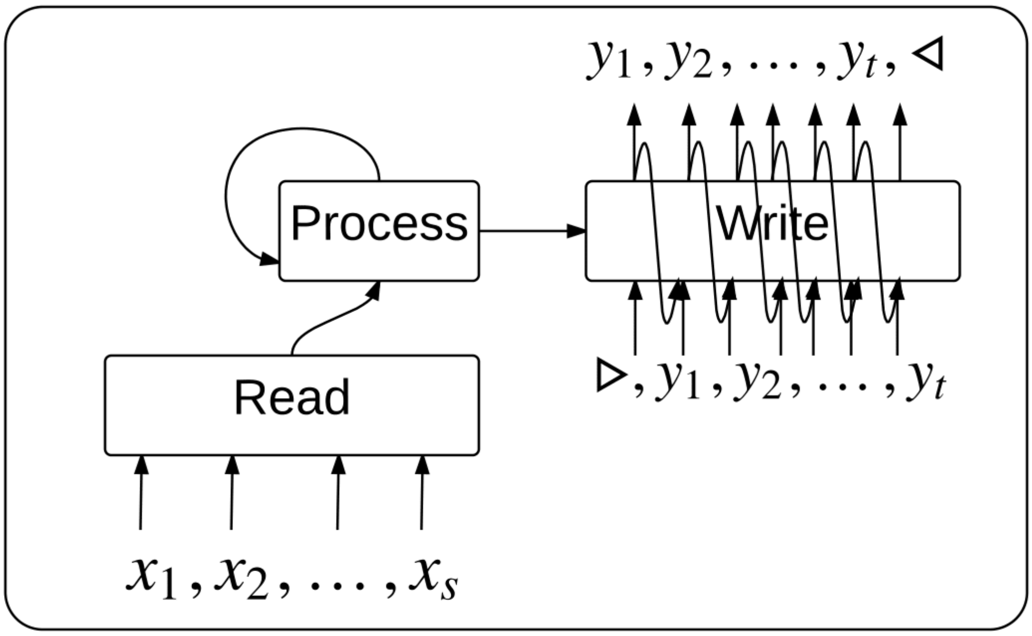

$y_1, y_2, \dots, y_t, \triangleleft$
Process
Write
Read
$\triangleright, y_1, y_2, \dots, y_t$
$x_1, x_2, \dots, x_s$

FIGURE 3.1: Model Architecture of set2seq framework (Vinyals, Bengio, and Kudlur, 2015)

3.2.1 Read module

Read module consists of a group of small embedding networks (see 2) to map each feature into a memory vector and concatenate all features into a huge m_i.

3.2.2 Process module

Process module is bidirectional LSTM network with attention mechanism. Similar to (Vinyals, Bengio, and Kudlur, 2015), the LSTM has no input or outputs and updates the set information in each hidden states. We claim that the last hidden state h_T is invariant to ordering of the input and then plug it in the output module.

3.2.3 Output module

Output module is an LSTM pointer network (Vinyals, Fortunato, and Jaitly, 2015) with the initial states as h_T. Instead of generative model, the pointer network use a Softmax function over all the input set and select the best possible horse as the output (i.e. the horse with highest likelihood). The pointer network uses a soft pointer base on the learned likelihood and may select duplicate members as a result.

3.3 Experiment and Result

To apply sequence to sequence for set framework, we designed two experiments to explore the feasibility on the horse racing predictions. The first experiment is aimed to predict rankings for all horses in a race. During the data processing stage, we first grouped the races by their unique identifier *raceid*. After grouping the races, we sorted the input records by the horse number in order to alleviate any influence from the ordering. We used the set2seq architecture described in the former sections where the read and process modules encode the unordered race records input and the output module uses pointer network to write the horse indices from fastest to slowest. The input dataset is in the shape of [batch size, race number, horse number]. The output sequence dataset is in the shape of [batch size, race number, 1] where the last dimension is *numpy.argsort(rankings)* denoting the sorted rankings of horses within a race. For simplicity, we only considered races of a fixed number of horses.

However, the output contains duplicated indices of horses. This incident aroused since the pointer network only uses the soft pointer to output the best horse base on the likelihood. Unfortunately, the set2seq framework is not suitable for ranking the horses.

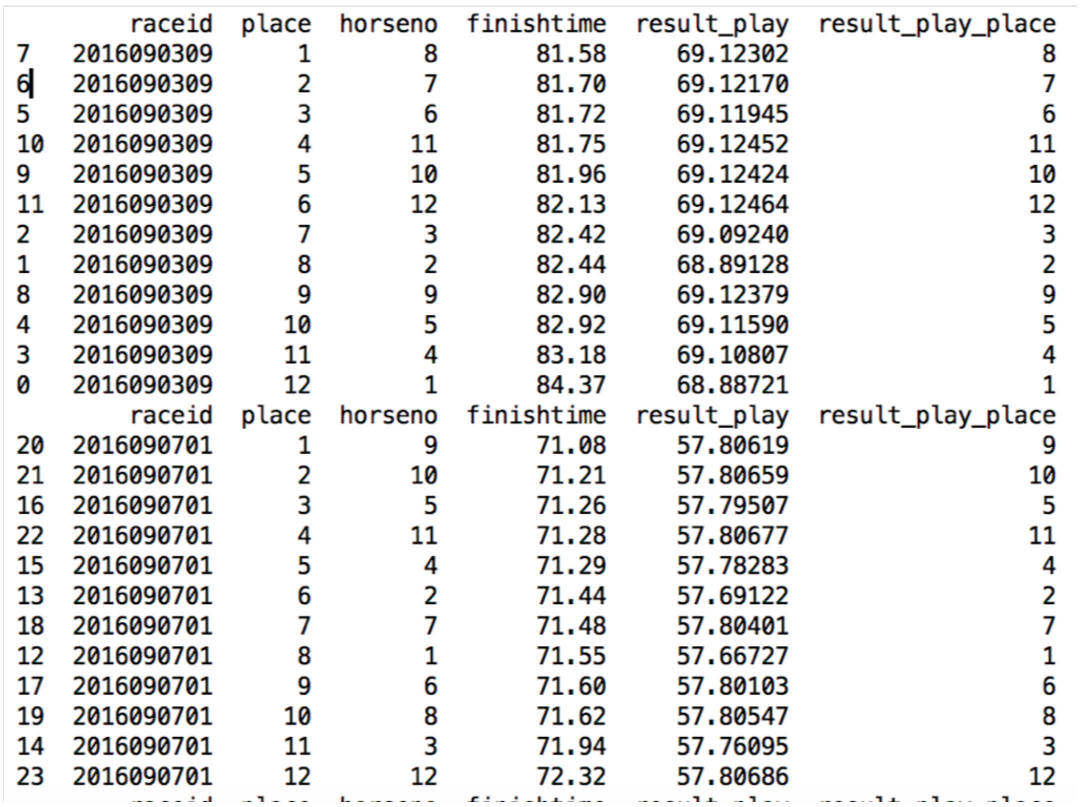
The second experiment is to predict the finishing time horses in a race. In this experiment, we made some modification to the framework mentioned above. Instead of applying pointer network in the output module, we employed a simple LSTM to perform regression on the horses finishing data. At the end, we sorted the output finishing time and bet on the best horses predicted.

The result of the experiment is shown in the following table. As we observed, the performance of set2seq framework is no better than random bet. In order words, the framework seems to learn nothing during training and fail to rank the horses.

Models	Random Bet	Odds Based	Set2seq Finishing Time
$Accuracy_{win}$	0.08333 (1/12)	0.27325	0.08577
$Accuracy_{place}$	0.25 (3/12)	0.55813	0.2846
Net gain	-	-996.0(WIN)/-1044.1(PLACE)	-2116.5 (WIN)/-1329.9(PLACE)
Return/Bet	-	-0.19(WIN)/-0.2023(PLACE)	-0.41 (WIN)/-0.26(PLACE)

TABLE 3.1: Set2seq Ranking Performance

The following figure is taken from the output. Note that when we fed the data into the model the input set is manually ordered by horse number. The figure showed that the framework does not learned from the high dimensional feature vectors. It outputs the finishing time following the horse number. In other words, the set2seq framework fails to encode the input records as a set.



```
        raceid  place  horseno  finishtime  result_play  result_play_place
7   2016090309      1        8       81.58     69.12302                  8
6   2016090309      2        7       81.70     69.12170                  7
5   2016090309      3        6       81.72     69.11945                  6
10  2016090309      4       11       81.75     69.12452                 11
9   2016090309      5       10       81.96     69.12424                 10
11  2016090309      6       12       82.13     69.12464                 12
2   2016090309      7        3       82.42     69.09240                  3
1   2016090309      8        2       82.44     68.89128                  2
8   2016090309      9        9       82.90     69.12379                  9
4   2016090309     10        5       82.92     69.11590                  5
3   2016090309     11        4       83.18     69.10807                  4
0   2016090309     12        1       84.37     68.88721                  1
        raceid  place  horseno  finishtime  result_play  result_play_place
20  2016090701      1        9       71.08     57.80619                  9
21  2016090701      2       10       71.21     57.80659                 10
16  2016090701      3        5       71.26     57.79507                  5
22  2016090701      4       11       71.28     57.80677                 11
15  2016090701      5        4       71.29     57.78283                  4
13  2016090701      6        2       71.44     57.69122                  2
18  2016090701      7        7       71.48     57.80401                  7
12  2016090701      8        1       71.55     57.66727                  1
17  2016090701      9        6       71.60     57.80103                  6
19  2016090701     10        8       71.62     57.80547                  8
14  2016090701     11        3       71.94     57.76095                  3
23  2016090701     12       12       72.32     57.80686                 12
```

FIGURE 3.2: Set2seq output snapshot

Chapter 4

Deep neural network with normalization

In last Chapter, we explored the feasibility to apply set2seq framework on horse racing problem, longing to solve predict the racing results race by race. Unfortunately, the set2seq framework, which uses attention mechanism for extra network memory, fails to encode the set of racing records and processes the data as a sequence. To continue the idea and make consistent predictions with each race, we returned to the deep neural network for finishing time prediction, proposing a normalization technique to learn the racely time distribution. Apart from pre-processing the data before feeding into the network, we employ normalization layers inside the network to normalize the intermediate layer output.

In this chapter, we first introduce the normalization layer and its working principle. Then, we showed that in-network normalization minimized the MSE and MAE loss by more than 100% and ultimately improves the bet accuracy. The result verified the observation in the last report that the model have better performance on races with consistent finishing time prediction.

4.1 Batch Normalization Layer

Batch Normalization is a technique to provide any layer in the neural network with input of zero mean and unit variance via mini-batching (Ioffe and Szegedy, 2015). It is well known that the neural network converge faster and have better performance when we pre-processed the inputs and make them zero mean and unit variance (LeCun et al., 1998). However, during training the distribution of the network layer output/activation is changing due to the parameter change. The distribution gradually deviating from zero mean,unit variance and decorrelated conditions, which is called by internal co-variate shift. As shown in the following figure, the internal co-variate shift effect is extraneous when multiple layers stack up and the output distribution can be also be deviated. In the case of horse racing prediction, we observed in last semester that the output finishing time distribution of test racing records are deviated from the ground truth. While the model are decreasing the loss in training set, it never converges on the test set and the MSE loss fluctuate around 500 due to the co-variate shift. By using normalization the input of each layer, we would remove the ill effect of internal co-variate shift and ultimately learn the racely finishing time distribution.

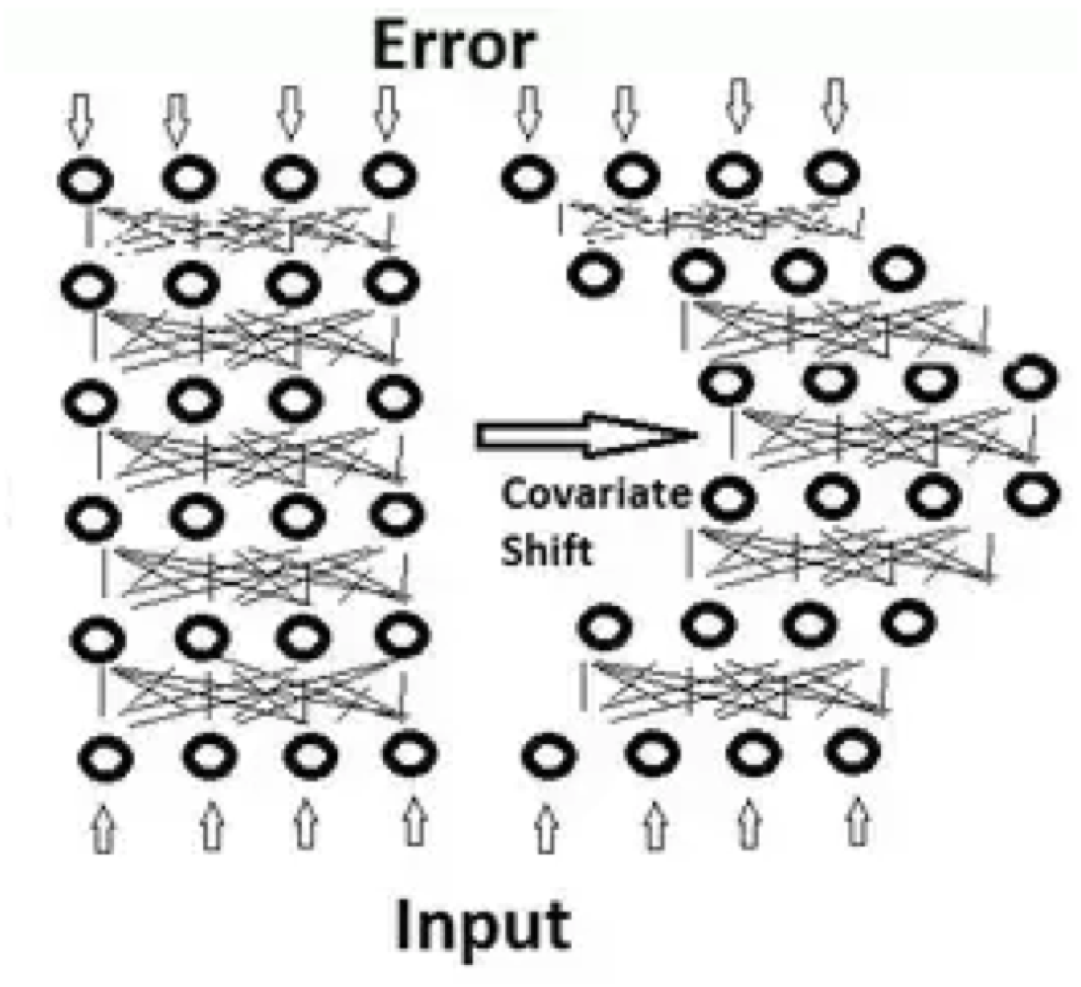
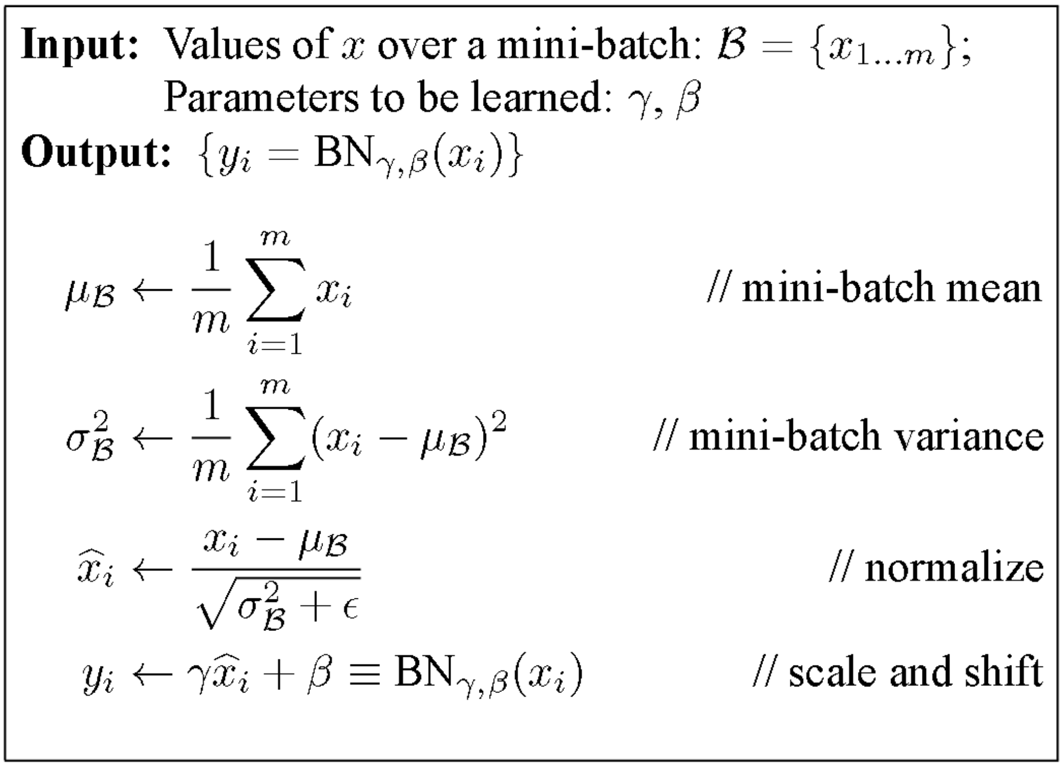


FIGURE 4.1: Visualization of Co-variate shift

The algorithm (Ioffe and Szegedy, 2015) is presented in the following figures, where ϵ is a constant added to the mini-batch variance for numerical stability.



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

FIGURE 4.2: Batch Normalization algorithm, applied to activation x over a mini-batch

4.2 Model design

We modified our deep neural network regression model in the last semester. We first used a group of small networks to embed each features onto memory vector of length 50. The model then concatenates all features into a large input vector and feed into two dense layers. We also insert two normalization blocks consisting of a dropout layer and batch normalization layer. The dropout layer has probability of 0.5. We implemented the model using Keras learning framework, which offers a high-level API for fast prototyping. The following figure gives an illustration.

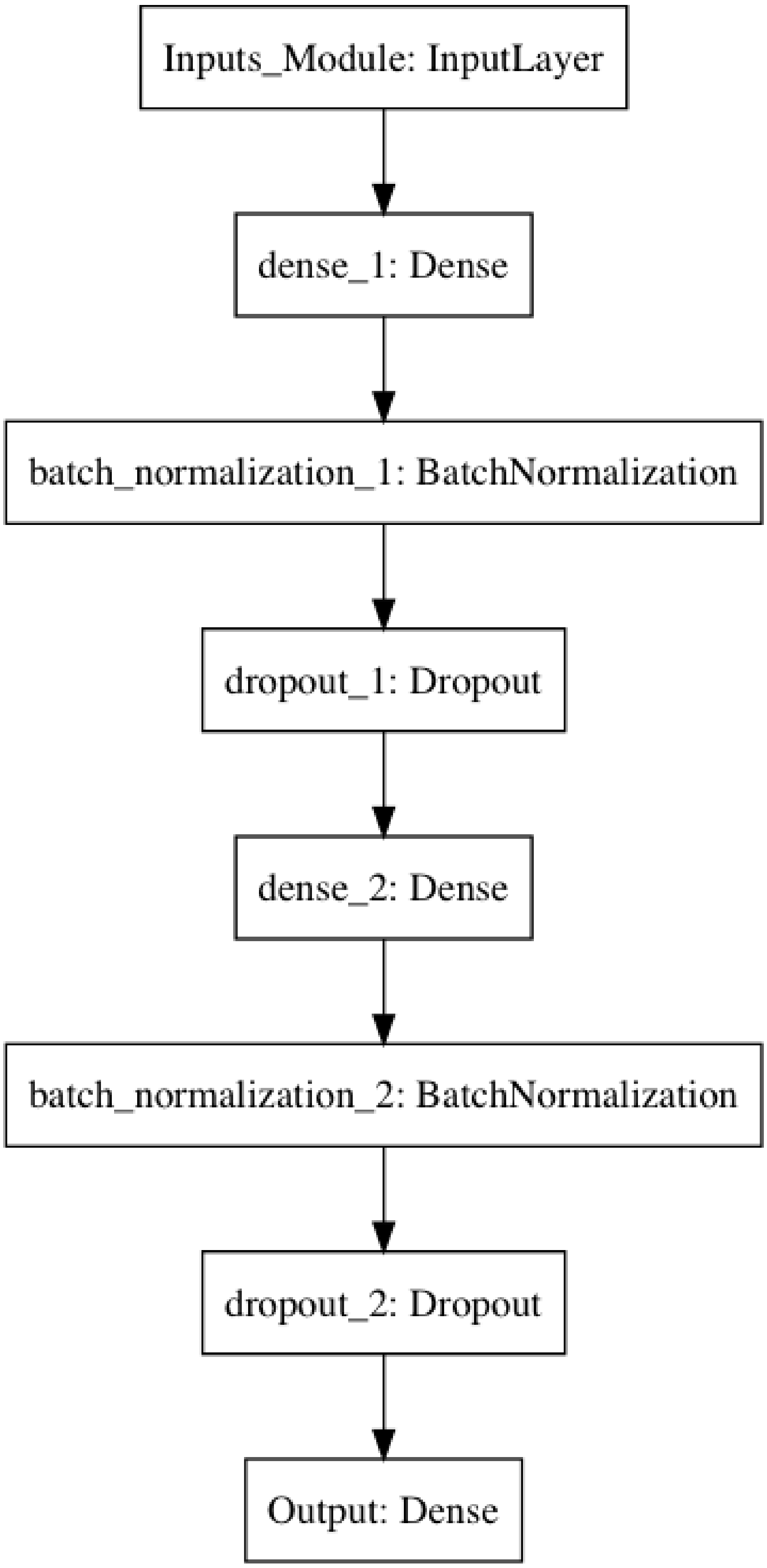

Inputs_Module: InputLayer
dense_1: Dense
batch_normalization_1: BatchNormalization
dropout_1: Dropout
dense_2: Dense
batch_normalization_2: BatchNormalization
dropout_2: Dropout
Output: Dense

FIGURE 4.3: Deep neural network with normalization

The input module is described in the Chapter 2 which is 30 embedding networks learning the semantics of input features. The output of the small networks are concatenated into a vector of length 1500 that contains race, weather and horse information. Then we use 2 dense-bn blocks to learn the non-linearity. We also insert 2 dropout layers in between the dense-bn blocks to regularize. The dropout probability is set to 0.5. The following table illustrates the configurations for the bn model.

Layer	Output Shape	Parameters Numbers
Inputs Module	-	537M
Concatenate	(Batch Size, 1500)	0
dense 1	(Batch Size, 32)	384M
batch normalization 1	(Batch Size, 256)	1024
dense 2	(Batch Size, 256)	65792
batch normalization 2	(None, 256)	1024
Output	(Batch Size, 1)	257
Total	-	921M

TABLE 4.1: Model BN Configurations

The kernel of weights are initialized with a Gaussian distribution. The mean and standard deviation used is (0, 0.02).

4.3 Experiment and Result

Following the standard in last semester, we are using the three criteria to evaluate the model:
1. Loss: MAE, MSE of between the ground truth and finishing time prediction.
2. Accuracy: accuracy of win and place bet of all races.
3. Net gain: overall net gain over the test dataset and the ratio of return over bet.

The following table shows the performance of the model. Comparisons are also included across traditional models and last year model. We can easily tell that the loss is greatly decreased by more than 100% using the batch normalization model. Moreover, the model achieves the same level as public intelligence in betting horses. The deep neural network model is shown to be capable to select the most excellent horse, however, the nature of regression methodology limits the model to learn the final ranking for most horses.

Models	Random Bet	Odds Based	Model Old	Model BN
MSE	-	-	417.7	3.68
MAE	-	-	18.43	1.42
$Accuracy_{win}$	0.08333 (≈1/12)	0.27325	0.1074	0.2440
$Accuracy_{place}$	0.25 (≈3/12)	0.55813	0.3142	0.4886
Net gain	-	-1754.0(WIN)/-1792.0(PLACE)	-568(WIN)/-1285(PLACE)	-1284.0(WIN)/-1221.4(PLACE)
Return/Bet	-	-0.21(WIN)/-0.22(PLACE)	-	-0.15(WIN)/-0.15(PLACE)

TABLE 4.2: Model BN Performance

The following two figures show that the min-max finishing time prediction by race from the old and BN model. We can see from the following figures that the min-max finishing time distribution is distributed reasonably as real racing games. The min-max finishing time is now clustered around 2 seconds and upper bounded by 6 seconds. Moreover, the last figure illustrates the connection between min-max finishing time and bet accuracy and net gain directly.

Base on the statistics, we claim that how consistent the finishing time is predicted is positively correlated with how well the model make decision.

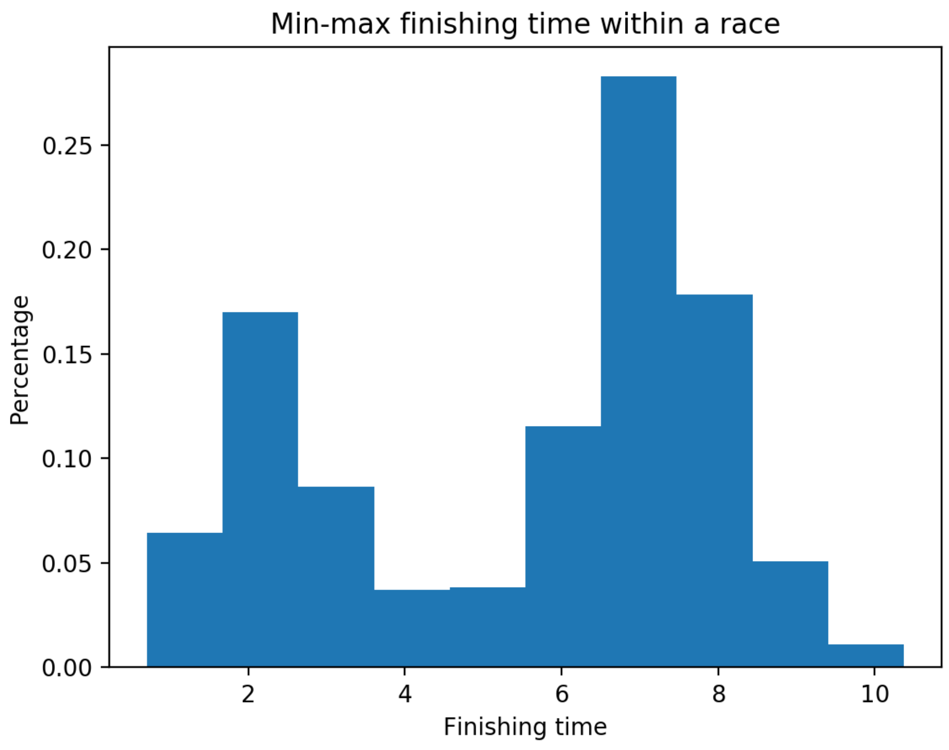


FIGURE 4.4: Min-max finishing time by races of old model

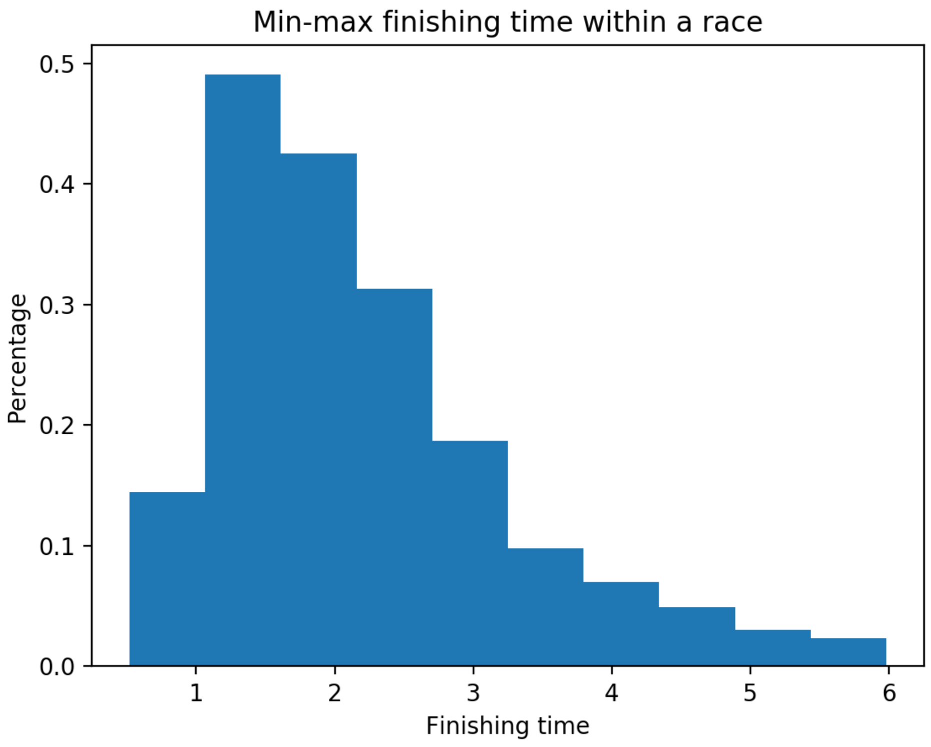


FIGURE 4.5: Min-max finishing time by races of BN model

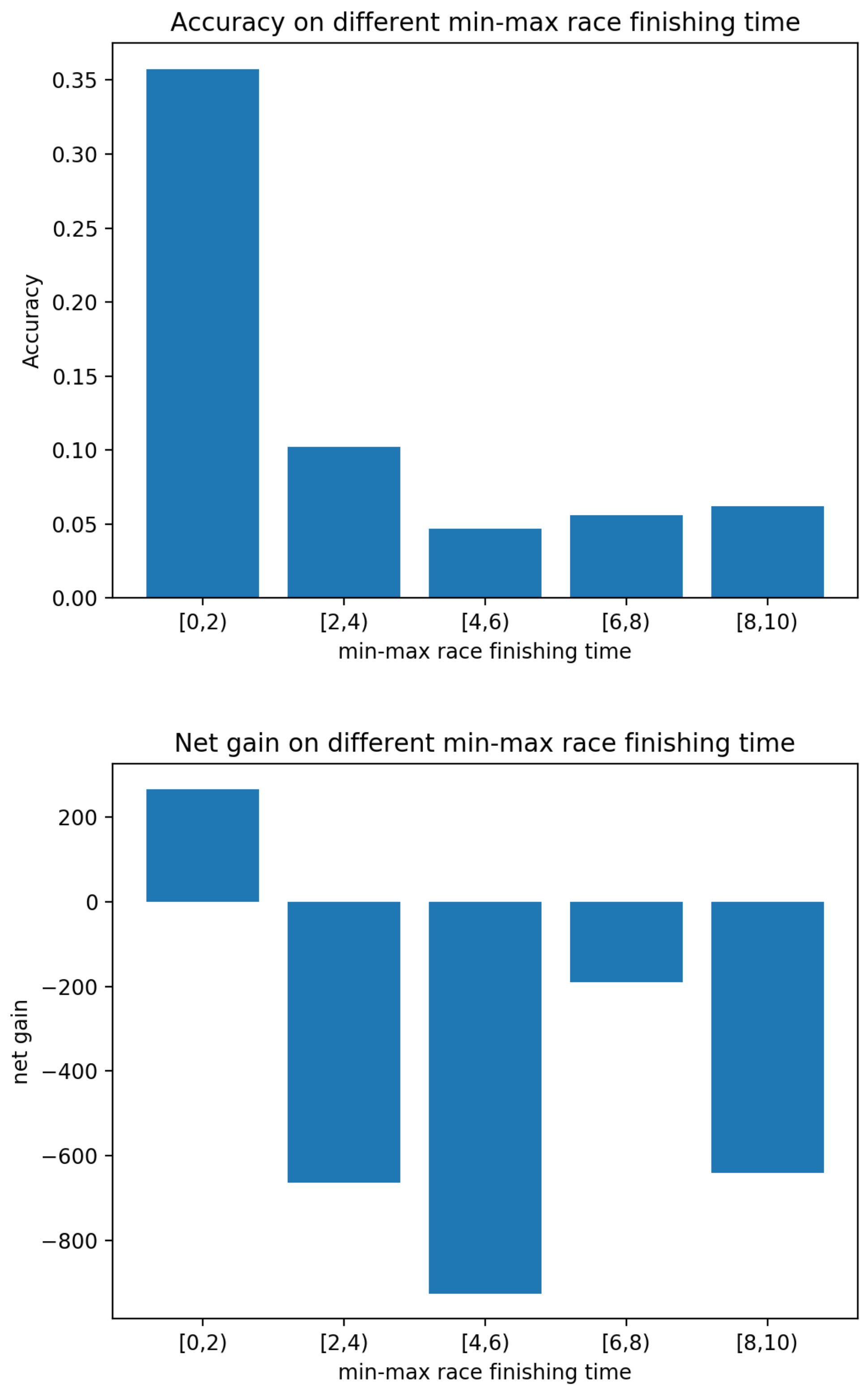


FIGURE 4.6: Relation between Min-max finishing time and accuracy and net gain

Chapter 5

Rank network

In Chapter4, we introduce the batch normalization layer which boost the network performance on both finishing time regression and final result. However, the regression methodology is designed to capture the general trends in finishing time. The BN model focuses on a single horse at a time in the loss function. It is trained individually on a single record to predict how fast a horse will run in the race. The final ranking is obtained from sorting. Although the BN model predicts the finishing time accurately within a race, the predicted time for records is independent from each other. The BN model is insufficient to capture the interrelation between horses or understand the minor performance fluctuations.

To address the weakness, in this Chapter, we present a ultimate rank network targeting on the ranking among horses, proposing a differential rank function on top of BN model. This pair approach target on a pair of race records in the loss function. It aims to compute the optimal ordering for the pair directly and ultimately derives the ranking of all horses. We first introduce the network design. Then we present the experiments result, showing that the network outperforms all traditional models and BN model.

5.1 Rank network

The rank network learns the ranking results by learning the relative ranking between pairs of horses (Cao et al., 2007; Burges et al., 2005). Given a pair of horses/records $[x_i, x_j]$ and the finishing time difference ε_{ij}, we use the BN model $f : \mathcal{R}^d \mapsto \mathcal{R}$ such that the rank order is specified by the finishing time prediction. Specifically, $f(x_i) < f(x_j)$ means that the model asserts x_i is faster than x_j, denoted as $x_i < x_j$

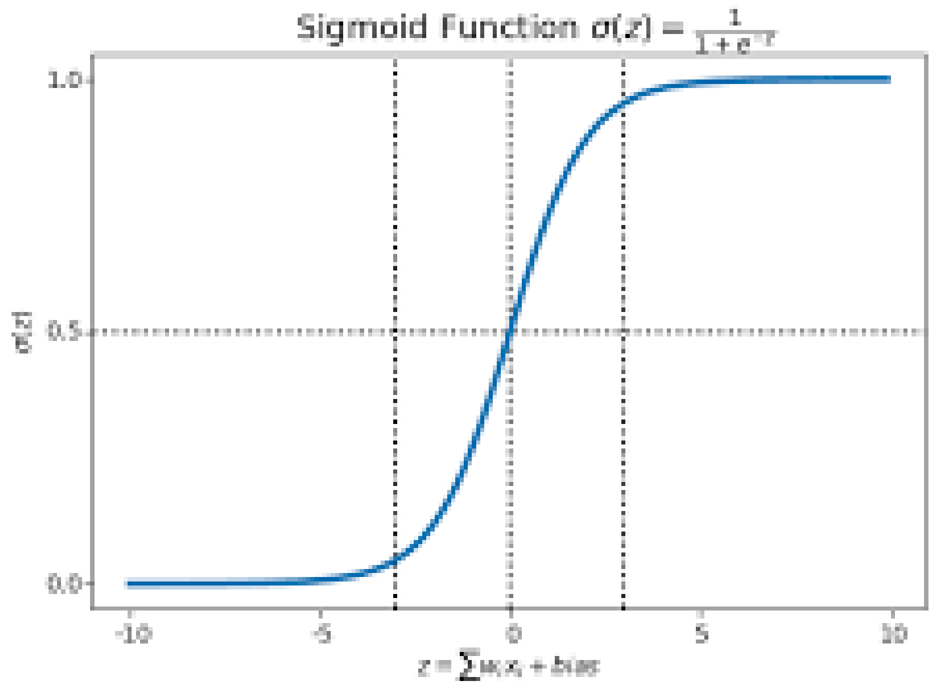


FIGURE 5.1: Logistic/Sigmoid function

To model the probability of $x_1 < x_2$, we use the logistic function on the input $f(x_i) < f(x_j)$. For simplicity, we define $z_{ij} = f(x_i) - f(x_j)$ to be the difference. Formally, the model learns the probability that x_i is faster than x_j during training, which is the following:

$$P_{ij} = \frac{e^{z}_{ij}}{1 + e^{z}_{ij}} \tag{5.1}$$

We define the loss to be the binary cross entropy between the prediction P_ij and the ground truth $\hat{P}_{ij}$. In the experiment, we approximate $\hat{P}_{ij}$ with the finishing time difference ε_{ij}. The cost is defined as follows:

$$\Delta_{ij} \equiv \varepsilon_{ij} * -log(sigmoid(P_{ij})) + (1 - \varepsilon_{ij}) * -log(1 - sigmoid(P_{ij})) \tag{5.2}$$

The following figure illustrate model architecture.

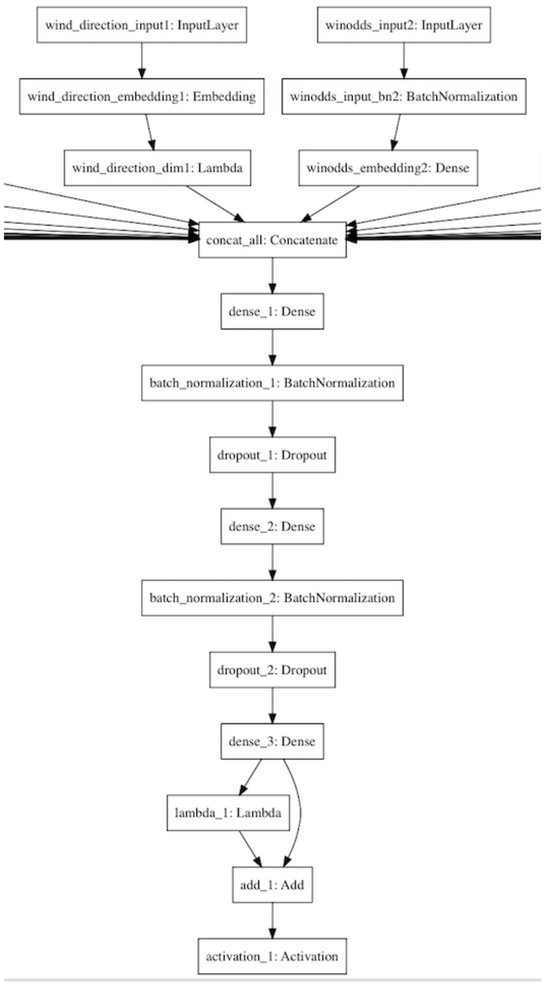

wind_direction_input1: InputLayer
winodds_input2: InputLayer
wind_direction_embedding1: Embedding
winodds_input_bn2: BatchNormalization
wind_direction_dim1: Lambda
winodds_embedding2: Dense
concat_all: Concatenate
dense_1: Dense
batch_normalization_1: BatchNormalization
dropout_1: Dropout
dense_2: Dense
batch_normalization_2: BatchNormalization
dropout_2: Dropout
dense_3: Dense
lambda_1: Lambda
add_1: Add
activation_1: Activation

FIGURE 5.2: Rank Model

Notice that the model may result in a lack of confidence. For example in the case $x_1 < x_2$, $x_2 < x_3$, $x_3 < x_1$, we cannot interpret which assign the ranking to any horse. To avoid this issue, we do not to output the probability in final prediction, instead we simply take the output of f and sort the value accordingly.

5.2 Experiment and Result

In the experiment, we used the same training and test set and evaluation criteria as the former chapters. The results are shown in the following tables. The rank model reach 30% accuracy in win and outperforms model BN by 25%.

Interestingly, the rank model has higher accuracy to bet on races in class 1 and 2 where the horses are the most competitive and have stable performance. The tables

Models	Random Bet	Odds Based	Model Old	Model BN	Rank Model
$Accuracy_{win}$	0.08333 (≈1/12)	0.27325	0.1074	0.2440	0.3047
$Accuracy_{place}$	0.25 (≈3/12)	0.55813	0.3142	0.4886	0.5215
Net gain	-	-1754.0(WIN)/-1792.0(PLACE)	-568(WIN)/-1285(PLACE)	-1284.0(WIN)/-1221.4(PLACE)	181.5(WIN)/-124.5(PLACE)
Return/Bet	-	-0.21(WIN)/-0.22(PLACE)	-	-0.15(WIN)/-0.15(PLACE)	0.1745(WIN)/-0.1197(PLACE)

TABLE 5.1: Rank Model Performance

below shows that the $Accuracy_{win}$ and $Accuracy_{place}$ of class 1 and 2 races are three times over the rest of the classes.

Class	1	2	3	4	5
$Accuracy_{win}$	0.5625	0.4090	0.2355	0.1904	0.2457
$Accuracy_{place}$	0.625	0.6363	0.5181	0.4920	0.5084
Net gain (WIN)	78.0	103.5	-918.5	-1216.0	-133.5
Return/Bet (WIN)	0.43	0.12	-0.33	-0.38	-0.11
Net gain (PLACE)	-27.9,	-96.6,	-706.3,	-829.2,	-187.5
Return/Bet (PLACE)	-0.17	-0.11	-0.25	-0.26	-0.15

TABLE 5.2: Rank Model Performance Across All Classes

To help understand the results in table, we offer some empirical analysis in the figures comparing the net gain across all classes.

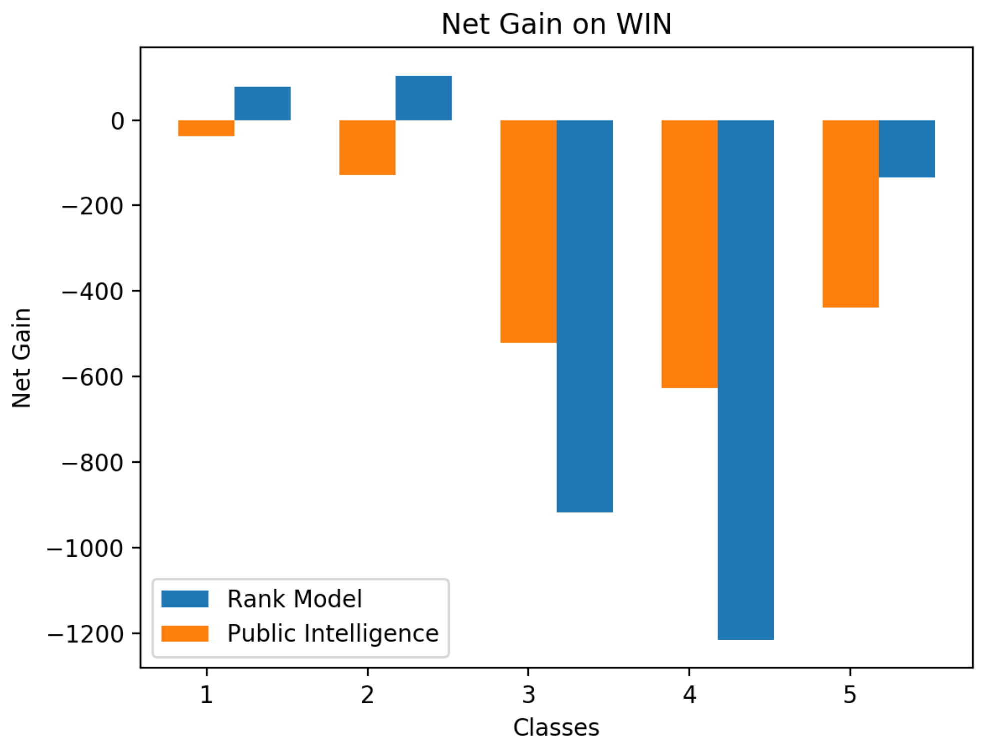


FIGURE 5.3: Net Gain on WIN Across All Classes

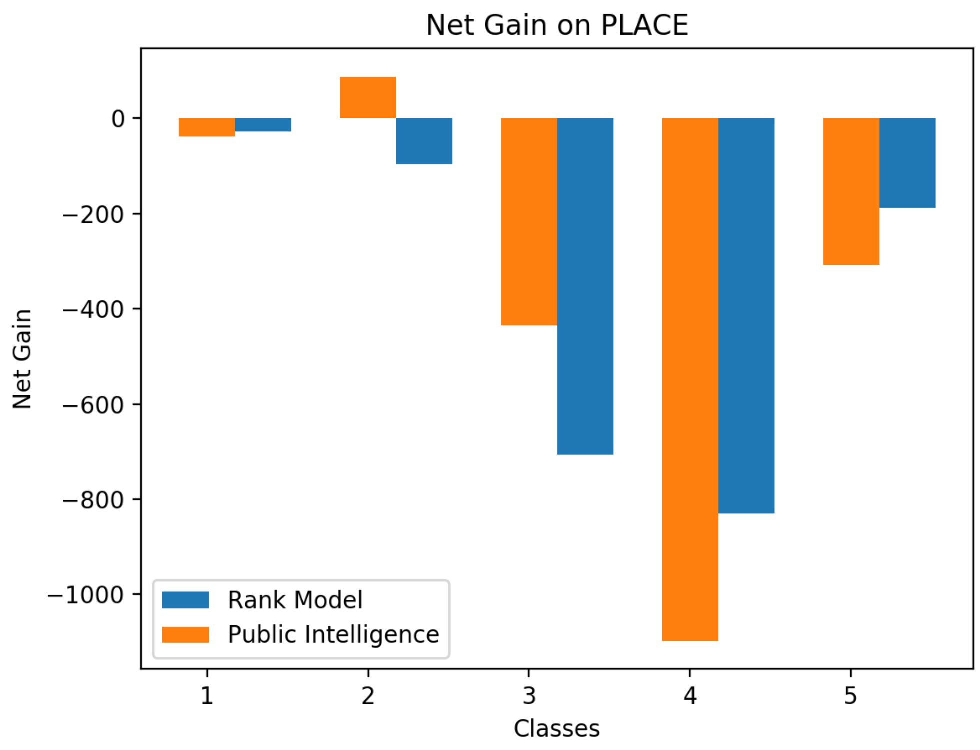


FIGURE 5.4: Net Gain on PLACE Across All Classes

When only betting on class 1 and 2 races, the rank model surpass public intelligence and constantly make promising net gain. To discuss the results, we claim that the rank model is an effective method to learn the ranking of horses. This is a strong indication that the performance of the horse should be rank against each other rather than simply by regression. The following table shows that the rank model double the accuracy of model BN when the classes are fixed on class 1 and 2.

Models	Random Bet	Odds Based	Model BN	Rank Model
$Accuracy_{win}$	0.08333 (≈1/12)	0.4326	0.2692	0.4857
$Accuracy_{place}$	0.25 (≈3/12)	0.6346	0.5011	0.6359
Net gain	-	-167.4(WIN)/49.0(PLACE)	-120.0(WIN)/-216.5(PLACE)	181.5(WIN)/-24.5(PLACE)
Return/Bet	-	-0.16(WIN)/0.04(PLACE)	-0.11(WIN)/-0.21(PLACE)	0.1745(WIN)/-0.02(PLACE)

TABLE 5.3: Rank Model Performance on Class 1 and 2

5.3 Overall Result

To present our ultimate results, we bet on confident races in class 1 and 2 by polling. We train a set of rank models with different random Gaussian initialization. The confidence threshold is fixed on 50%, meaning that bet will only made on races with more than half agreement across all models.

The result shown in the following figure indicates our model defeats public intelligence in betting. Furthermore, our model maintains positive net gain on the entire rece season.

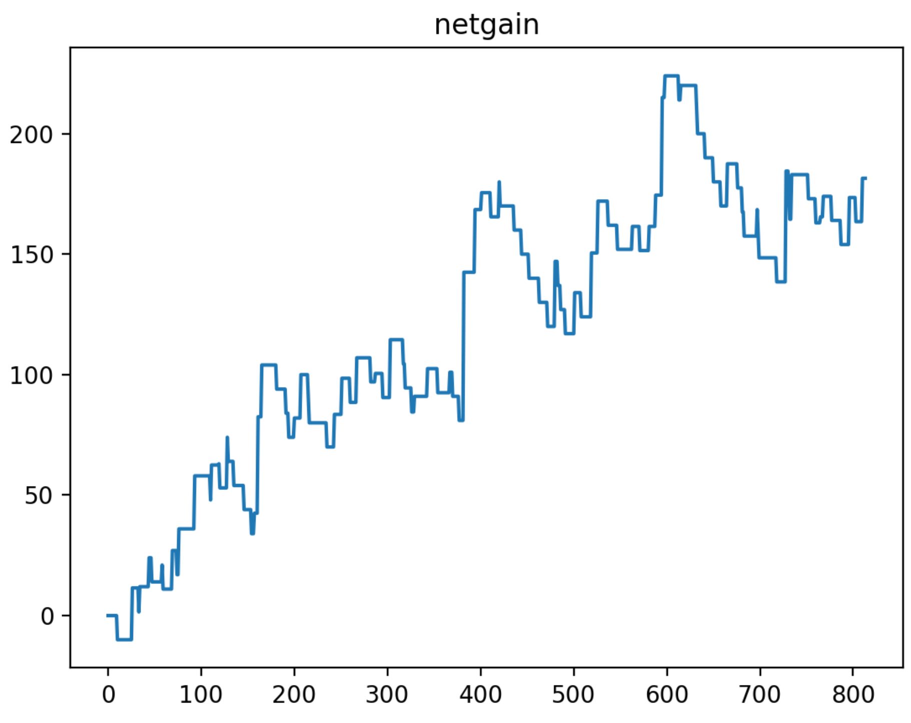


FIGURE 5.5: Final Results

Chapter 6

Conclusion

This report offers an empirical study on the horse racing predictions. We prepared all relevant information on horse racing and construct a large dataset to approach the problem. We explored and compared three different methods to make predictions: sequence to sequence for sets, multi-layer perceptron with normalization and rank network. Analysis shows that the batch normalization technique helps the model to make consistent prediction which simultaneously improve the model performance on betting More importantly, the rank network is proved to be an effective method to learn the ultimate ranking of horses. Our model outperforms public intelligence and make profits in the long run.

In the future, we hope to apply pattern matching to extract the most confident races to bet. It will be interesting examine the embedding of feature and analyze similarity between races.

Appendix A

Development Environment

This project mainly uses Keras as the devepment framework. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Keras allows easy and fast prototyping through its user friendliness. Specifically, Keras framework has the following guiding principles:

1. User friendliness. Keras is an API designed for human beings, not machines. It puts user experience front and center. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear and actionable feedback upon user error.

2. Modularity. A model is understood as a sequence or a graph of standalone, fully-configurable modules that can be plugged together with as little restrictions as possible. In particular, neural layers, cost functions, optimizers, initialization schemes, activation functions, regularization schemes are all standalone modules that you can combine to create new models.

3. Easy extensibility. New modules are simple to add (as new classes and functions), and existing modules provide ample examples. To be able to easily create new modules allows for total expressiveness, making Keras suitable for advanced research.

4. Work with Python. No separate models configuration files in a declarative format. Models are described in Python code, which is compact, easier to debug, and allows for ease of extensibility.

Appendix B

Complement Information of Dataset

B.1 Racing Record

TABLE B.1: Useful racing features collected from HKJC website

Feature	**Description**	**Types**	**Values**
raceyear	Race year	Record Index	-
racemonth	Race month	Record Index	-
raceday	Race day	Record Index	-
raceid	Unique id of a race	Record Index	-
location	Location where a race take place	Categorical	ST, HV
class	Class of horses meaning strength of a horse	Categorical	1 to 5
distance	Distance	Categorical	1000, 1200, 1400, 1600, 1650, 1800, 2000, 2200
course	Track	Categorical	’A’, ’A+3’, ’AWT’, ’B’, ’B+2’, ’C’, ’C+3’
going	Track condition	Categorical	’FAST’, ’GOOD’, ’GOOD TO FIRM’, ’GOOD TO YIELDING’, ’SLOW’, ’SOFT’, ’WET FAST’, ’WET SLOW’, ’YIELDING’, ’YIELDING TO SOFT’
	Race number in a race day,		

B.2 Horse

TABLE B.2: Horse features

Feature	Description	Types	Values
origin	Place of birth	Categorical	'ARG', 'AUS', 'BRZ', 'CAN', 'FR', 'GB', 'GER', 'GR', 'IRE', 'ITY', 'JPN', 'NZ', 'SAF', 'SPA', 'USA'
age	Horse age	Numerical	3-10
color	Horse color	Categorical	'Bay', 'Black', 'Brown', 'Chest-nut', 'Dark', 'Grey', 'Roan'
sex	Horse sex	Categorical	'Colt', 'Filly', 'Gelding', 'Horse', 'Mare', 'Rig'
sire	Father	Categorical	729 unique horses
dam	Mother	Categorical	3514 unique horses
dam's sire	Maternal grandfather	Categorical	951 unique horses
horseid	Unique identifier of a horse	Categorical	*Over 1000 distinct values

B.3 Weather

TABLE B.3: Weather features

Feature	Description	Types	Values
raceyear	Race year	Record Index	-
racemonth	Race month	Record Index	-
raceday	Race day	Record Index	-
raceno	Race number in a race day	Record Index	1 to 8
location	Location where a race take place	Record Index	ST, HV
temperature	Temperature when a race start	Real Value	-
weather	Weather condition when a race start	Categorical	'Clear', 'Drizzle Fog', 'Fog', 'Haze', 'Light rain Broken clouds', 'Light rain Fog', 'Light rain Partly cloudy', 'Light rain Passing clouds', 'Lots of rain Passing clouds', 'Partly cloudy', 'Passing clouds', 'Rain Passing clouds', 'Rain showers Fog', 'Rain showers Partly cloudy', 'Rain showers Passing clouds', 'Scattered clouds', 'Scattered showers Passing clouds', 'Sprinkles Fog', 'Sprinkles Partly cloudy', 'Sprinkles Passing clouds', 'Sprin-

Bibliography

Arpit, Devansh et al. (2017). “A closer look at memorization in deep networks”. In: *arXiv preprint arXiv:1706.05394*.

Burges, Chris et al. (2005). “Learning to rank using gradient descent”. In: *Proceedings of the 22nd international conference on Machine learning*. ACM, pp. 89–96.

Cao, Zhe et al. (2007). “Learning to rank: from pairwise approach to listwise approach”. In: *Proceedings of the 24th international conference on Machine learning*. ACM, pp. 129–136.

Donahue, Jeffrey et al. (2015). “Long-term recurrent convolutional networks for visual recognition and description”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2625–2634.

Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton (2013). “Speech recognition with deep recurrent neural networks”. In: *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, pp. 6645–6649.

Harris, David and Sarah Harris (2010). *Digital design and computer architecture*. Morgan Kaufmann.

He, Haibo and Edwardo A Garcia (2009). “Learning from imbalanced data”. In: *IEEE Transactions on knowledge and data engineering* 21.9, pp. 1263–1284.

He, Kaiming et al. (2015). “Deep Residual Learning for Image Recognition”. In: pp. 1 –12. URL: https://arxiv.org/pdf/1512.03385.pdf.

– (2016). “Identity mappings in deep residual networks”. In: *European Conference on Computer Vision*. Springer, pp. 630–645.

Huang, Gao et al. (2016). “Densely Connected Convolutional Networks”. In: URL: https://arxiv.org/pdf/1608.06993.pdf.

Ioffe, Sergey and Christian Szegedy (2015). “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167*.

Kim, Yoon (2014). “Convolutional Neural Networks for Sentence Classification”. In: URL: http://www.aclweb.org/anthology/D14-1181.

Lange, Sascha and Martin Riedmiller (2010). “Deep Auto-Encoder Neural Networks in Reinforcement Learning”. In: URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5596468.

LeCun, Yann et al. (1998). "Gradient-Based Learning Applied to Document Recognition". In: *PROC. OF THE IEEE*, pp. 1 –46. URL: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf.

Méhat, Jean and Tristan Cazenave (2011). "A parallel general game player". In: *KI-künstliche Intelligenz* 25.1, pp. 43–47.

Ouyang, Xi et al. (2015). "Sentiment Analysis Using Convolutional Neural Network". In: URL: http://ieeexplore.ieee.org/document/7363395/.

Ren, Shaoqing et al. (2015). "Faster R-CNN: Towards Real-Time Object Detection-with Region Proposal Networks". In: URL: https://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf.

Riess, Steven A (1991). *City games: The evolution of American urban society and the rise of sports*. Vol. 114. University of Illinois Press.

Santos, Cicero Nogueira dos and Maira Gatti (2014). "Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts". In: URL: http://www.aclweb.org/anthology/C14-1008.

Silver, David et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *Nature* 529.7587, pp. 484–489.

Snyder, Wayne (1978a). "Decision-making with risk and uncertainty: The case of horse racing". In: *The American Journal of Psychology*, pp. 201–209.

Snyder, Wayne W (1978b). "Horse racing: Testing the efficient markets model". In: *The Journal of finance* 33.4, pp. 1109–1118.

Sola, J and J Sevilla (1997). "Importance of input data normalization for the application of neural networks to complex industrial problems". In: *IEEE Transactions on Nuclear Science* 44.3, pp. 1464–1468.

Srivastava, Nitish et al. (1989). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Neural Networks* 2.15, pp. 359 –366. URL: https://www.cs.cmu.edu/~epxing/Class/10715/reading/Kornick_et_al.pdf.

– (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.15, pp. 1929 –1958. URL: https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf.

Srivastava, Rupesh Kumar, Klaus Greff, and Jurgen Schmidhuber (2015). "Deep Residual Learning for Image Recognition". In: pp. 1 –6. URL: https://arxiv.org/abs/1505.00387.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*, pp. 3104–3112.

Szegedy, Christian, Alexander Toshev, and Dumitru Erhan (2016). “Deep Neural Networks for Object Detection”. In: URL: https://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detection.pdf.

Tung, Cheng Tsz and Lau Ming Hei (2016). “Predicting Horse Racing Result Using TensorFlow”. In: URL: http://www.cse.cuhk.edu.hk/lyu/_media/students/lyu1603_term_1_report.pdf?id=students%3Afyp&cache=cache.

Vanhoucke, Vincent, Andrew Senior, and Mark Z. Mao (2010). “Improving the speed of neural networks on CPUs”. In: URL: https://static.googleusercontent.com/media/research.google.com/zh-CN//pubs/archive/37631.pdf.

Vinyals, Oriol, Samy Bengio, and Manjunath Kudlur (2015). “Order matters: Sequence to sequence for sets”. In: *arXiv preprint arXiv:1511.06391*.

Vinyals, Oriol, Meire Fortunato, and Navdeep Jaitly (2015). “Pointer networks”. In: *Advances in Neural Information Processing Systems*, pp. 2692–2700.

Widrow, Bernard and Marcian Hoff (1959). “Adaptive Switching Circuits”. In: URL: http://www-isl.stanford.edu/~widrow/papers/c1960adaptiveswitching.pdf.

Williams, Janett and Yan Li (2008). “A case study using neural networks algorithms: horse racing predictions in Jamaica”. In: *Proceedings of the International Conference on Artificial Intelligence (ICAI 2008)*. CSREA Press, pp. 16–22.

Zhang, Xiang, Junbo Zhao, and Yann LeCun (2015). “Character-level Convolutional Networks for Text Classification”. In: URL: https://arxiv.org/pdf/1509.01626.pdf.