Department of Computer Science and Engineering,

The Chinese University of Hong Kong

Final Year Project, Final Report Term 2

Predicting handicap result of Soccer using betting odds via Machine Learning

Supervised by

Prof. Michael R. Lyu

Written by

Chan Cheong, 1155100189

Sun Ka Ho, 1155098418

Abstract

Machine learning is increasing popular for solving different kind of problems nowadays, myriads of machine learning methods have been developed. Some problems are fascinating but difficult to solve, for instance, trying to obtain a positive return from betting.

The goal of this project is to use different machine learning techniques and models to predict the handicap result for soccer games. The data features include odds from different bookmakers, past 10 encounter record of 2 teams, FIFA estimated player scores and FIFA estimated team scores. We found that the initial odds released by bookmakers have significant contribution to the model performance. Various ways of feature engineering techniques were used to enhance our dataset and several models were built on top of it. By picking the strength of each model, we conducted a model that perform well in most of the league. We also developed a software which can predict the handicap result for soccer games in real-time.

Acknowledgements

We would like to thank our supervisor and adviser, Professor Michael R. Lyu and Mr. Edward Yau for all their advice and guidance. Their constant supports had been our motivation for the project.

Table of Contents

1. Introduction

1.1 Overview

This project focuses on utilizing machine learning to predict the handicap result of soccer games and make a profitable return. This report describes the process of the work done during the entire academic year, there are two phrases in this project, phrase one for semester one and phrase two for semester two. The entire workflow of the project including data collection, data selection, feature engineering, modelling, evaluation and prediction software.

1.2 Motivation

Nowadays, soccer games are extremely popular around the world. Among hundreds of countries, there are their own national team and leagues. In FIFA20 database, there are more than seven hundred soccer clubs stored in database. Both member of our team are fascinated in soccer games although we never play soccer game in reality. Due to the popularity of it, many commercial businesses related to soccer grow continuously. Currently, soccer game is commercialized because of its popularity so there are more and more businesses related to soccer grow greatly, such as the computer games, figures and T-shirt. Among them, one of the sectors has the largest growth is gambling. Thus, it is easy to see that there are many sponsor logos of bookmakers in different soccer matches.

Betting odds are some quantitative numbers generated by bookmakers that are used to maximize their profit. Under this premise, the betting odds should contain some latent meanings. The assumption of the project is that bookmakers will be very meticulous on calculating the betting odds hence it is reasonable for treating them as factors in the machine learning models. In recent year, machine learning especially deep learning is becoming prevalent and popular. Thus, this project will try to use a machine learning approach to predict soccer game results.

Since soccer games are commercialized, the problem of match-fixing (or conflicts of interest) emerged. Sometimes the betting odds for a strong team are higher than the weak team in a match which is strange. As a result, the strong team was defeated by the weak team in that match. Thus, it is reasonable to suspected that if there is any match-fixing in the popular leagues. The hypothesis is that betting odds might be able to tell such information. Therefore, betting odds will be mainly used to model the game. Eventually, a profitable model is preferred.

1.3 Objective

The final goal of this project is to build a machine learning model which can predict the soccer results profitably using odds. The entire project was divided into two phrases, the first phrase was to set up the project pipeline and the second phrase was methods exploration and model optimization.

In semester one (first phrase), the main focus was on data cleaning and modelling.

i) All useful data will be collected and cleaned.
ii) Finalize the model pipeline and choose the baseline model.
iii) Different models for comparison will be deployed.

In semester two (second phrase), the main focus was on optimization and deployment.

i) Explore different data augmentation and modelling techniques.
ii) Model optimization with fine tuning network structure.
iii) Real-time prediction program.

There are some different settings in the two phrases. In the first term, we focused on the odds and players features. However, in the second term, we did some experiment and found that by just using the initial release odds, it provides a huge improvement on our models.

1.4 Project Workflow

First, data were collected from three different websites[1] by using Scrapy and Selenium. Second, data will be merged into JSON format where each JSON file stored all the matches on a daily basis. In addition, table-like CSV files which contain the useful raw features were created. These CSV files were inputted into the pre-processing pipeline. Some feature engineering was performed under this state. The cleaned dataset will then pass to the modelling state where varying models were trained and tuned. Scikit-learn [1]and TensorFlow [2] are two of the libraries often used in the entire project. Eventually, the model results will go through the evaluation process, graphs and metrics will be generated. A prediction program is developed using the finalized model and betting strategy.

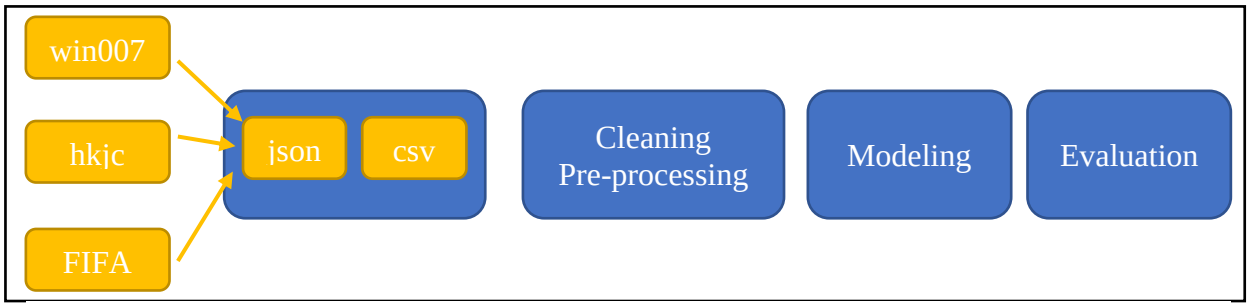

Figure 1. Project Workflow

[1] http://live.win007.com/, https://g10oal.com/ and https://www.fifaindex.com/

1.5 Glossary

Terms	Explanations
Betting category	Various types of gambling games.
Betting margin	$\left(\sum_{s=odds} \frac{1}{s}\right) - 1$ A type of commission fees that is included in the odds so that the expected value of the return is always less than 0. For example, if both of the handicap odds for the home team and away team are 1.85. Betting margin = $\frac{1}{1.85} + \frac{1}{1.85} - 1 \approx 8\%$
Goal miss	A term that describes if that shoot is missed.
Handicap	Refer the Section 1.6 (Definition of Handicap)
Handicap Result	The goal results after calculated handicap
Line	A value set by the bookmakers as the limitation of a betting category
League	A general term for the type of a match, including both leagues and cups
Lineup	An arrangement of the players in a team during a match
Season of a match	For league, there is a part of the year in which the soccer matches will be played

1.6 Definition of Handicap

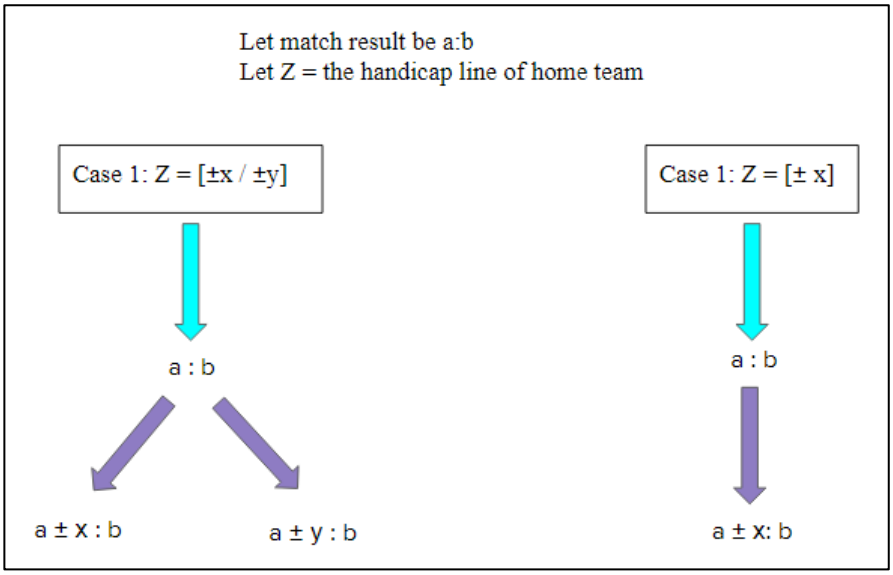

Figure 2. Illustration of handicap

Handicap is a betting category that makes the match being fairer and evener because the goal difference of the stronger team is required to be higher than a specific number to win the bet. In Figure 2, you can find that there are 2 types of handicaps which are [± x / ± y] and [± x]. Then, the result (a : b) after calculating the handicap will be (a ± x : b ± y) or (a ± x : b).

There are 5 possible results in handicap which are winning all the money, winning half of the money, getting back the original money (draw), losing half of the money and losing all the money.

An example will be illustrated. Let assume one will bet on the home team to win, there are 5 possible handicap results which are as the following:

handicap result	Meaning of the class	line of handicap of home team	Example of goal results under handicap
2	winning all the money	[-1]	3:1 -> 2:1
1	winning half of the money	[+0/0.5]	1:1 -> 1:1/1.5:1
0	getting back original money	[-1]	2:1 -> 1:1
-1	losing half of the money	[-2/-2.5]	3:1 -> 1:1/0.5:1
-2	losing all the money	[-1/-1.5]	0:0 -> -1:0/-1.5:0

Table 1. Example of 5 possible handicap results

The first row of the Table 1 is an example of winning all the money. If the final score of the match is 3:1 and handicap for the home team is [-1]. The goal result under handicap will be 2:1 (3-1:1 = 2:1), which is the home team score is greater than the away team score so the final result is winning all the money based on the assumption of buying handicap of home team.

The second row shows an example of winning half of the money. If the final score of the match is 1:1 and the handicap for the home team is [+0/0.5]. The goal result under handicap will split into two cases which are 1:1 and 1.5:1 because 1:1 split into two cases which are 1+0:1 and 1+0.5:1. There is only one case (1.5:1) where the home team score is greater than the away team score. Therefore, the result is only winning half of the money.

The third case will be getting back all the money. This can only happen when the true handicap result of the match is 0. In this case, no matter which team we betted, we can get back all the money.

In handicap, users are only available to buy the home team and away team before they already covered all the possible cases. Thus, after getting the prediction from our model, we can only choose to buy home/away team or not bet for the match. In other words, we will bet the home team in handicap if the predicted result is positive (class 1 and 2) whereas we will bet the away team in handicap if the predicted result is negative(class -1 and -2). Otherwise, we will not bet for the match (class 0).

1.7 Past Research

Predicting the soccer result is a popular research topic but only few of research focus on predicting soccer result using odds. Although not many people tried to use odds for predicting soccer result, one of the previous final year project [3] had shown it would be possible to do so. From the report, they raised a betting strategy using Kelly Formula [4] which an extremely probability theory relating to investment and gambling. Moreover, they designed a LSTM model for predicting the horse racing result using odds. LSTM is a very popular deep learning for handling time series problem. Since our datasets are also time-related; therefore, we would like to implement such approach in this project but with different neural network architecture as the odds type is not as same as theirs.

There is a similar research [5] about predicting the soccer result based on the odds. Although the main target in their research was home, away and draw odds (HAD) which is different to the handicap odds, the direction of their work inspired us. For example, they raised that the relationship of odds from bookmakers and the match result can be analysed by the Friedman test and he Kruskal-Wallis test. Thus, Kruskal-Wallis test will be used in this project to determine the dependency of the handicap odds and handicap result.

2. Methodology

In this project, various machine learning techniques will be applied to model the handicap results. One of our hypotheses for the entire project is that odds imply a lot of hidden useful information. It is because odds are carefully calculated by the bookmakers with their own formulae and interest. Hence, odds will be the main focus in this project.

Each bookie calculate and release their betting odds for each match before it started. It is common that these betting odds vary over time in order to maximise the bookie's profit. Thus, all changes of the handicap odds that is *5 minutes before the match start* are collected as it is vital to not use the future data to do prediction.

Since there are many different soccer leagues (including the leagues and cups, as defined in glossary) around the world, the relationship of the features and handicap results from different leagues might not be same. Therefore, we proposed a by-league model and all-league model, this former one is essentially trained on a league basis (we expected the model can learn information league by league) and the latter one is trained with one-hot labels on the league (we expected the model can learn the general signal of all leagues).

The type of handicap results will be used as the label (y) because it has the least average betting margin (as stated in glossary) compare to others. In other words, handicap is the betting category that is most likely to have a profitable return. In addition, the odds of the handicap are relatively evenly distributed in which the lowest odd is about 0.62. In betting policy, it is better than other betting categories whose lowest odd could be 0.1.

There are 5 possible classes in handicap result, introduced in Section 1.6. Thus, a supervised learning using classification approach will be focused on this project. The predicted handicap result is based on the assumption of buying the home team's handicap so that if the model predicted that betting in home team will results in losing money, the strategy for buying the away team will be considered. 80% of the data will be split into training set and the rest be the testing set chronologically.

3. Dataset

Scrapy and Selenium will be mainly used to crawl the desired data from various soccer related websites[2]. A JSON-like files will be used to store all the information.

3.1 Weather and Temperature

The weather and temperature of matches are collected because they will affect the performance of the players. For example, the number of goals will be smaller in a raining data as the players cannot run fast.

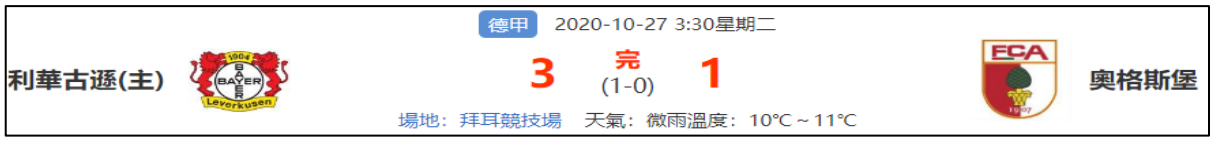

Figure 3. Weather and temperature of a match

3.2 Odds

3.2.1 Handicap Odd

Handicap odd is a value that represents the percentage of money that one can earn if one correctly bet on that winning team. Normally, odds will be low if the team is very likely to win that match. Moreover, when more people choose to bet on that team, the odds for that team will decrease accordingly.

In this project, handicap odds from different bookmakers are collected, in which companies like HKJCHKJC (Hong Kong Jockey Club), Crown, Bet365 and Macau slot will be focused. Handicap odds are collected to predict the handicap results as they have a direct relationship with the handicap results which will be proven in the later Section 4.8.1.

Match No.	Teams (Home vs Away)	Expected Stop Selling Time	Odds Home	Odds Away
Tuesday Matches				
TUE 3	Lokomotiv Moscow[+2.5/+3] vs Bayern Munich[-2.5/-3]	28/10 01:55	1.76	2.11
TUE 4	Shakhtar Donetsk[+1/+1.5] vs Inter Milan[-1/-1.5]	28/10 01:55	1.98	1.86
TUE 8	Barnsley[0/-0.5] vs QPR[0/+0.5]	28/10 03:00	2.02	1.82

Figure 4. Handicap odds of HKJC

[2] www.hkjc.com, www.win007.com, www.fifaindex.com

博彩公司	多盤口	初盤			終盤		
		主隊	盤口	客隊	主隊	盤口	客隊
澳門	−	0.81	半球/一球	0.89	1.02	半球/一球	0.68
	盤口2	0.65	半球	1.05	0.74	半球	0.96
Crown	−	0.95	半球/一球	0.87	0.91	半球	0.98
	盤口2	0.68	半球	1.28	0.58	平手/半球	1.47
	盤口3	1.21	半球/一球	0.72	1.23	半球/一球	0.71
	盤口4	1.88	一球	0.41	1.88	一球	0.41
Bet365	−	1.00	半球/一球	0.85	0.77	半球	1.02
	盤口2	0.60	平手/半球	1.30	0.55	平手/半球	1.38
	盤口3	1.15	半球/一球	0.68	1.05	半球/一球	0.75

Figure 5. Handicap odds of non-HKJC bookmakers

3.2.2 HAD Odd

HAD odd represents the odds of home team winning the match, away team winning the match and draw respectively. It is different to the handicap odd as there is not a handicap line. Thus, the team that having a larger goal will win the match and draw if neither. The reason that collecting HAD odds will be stated in section 1117.3.

Match No.	Teams (Home vs Away)	Expected Stop Selling Time	In Play Betting	Odds Home	Draw	Away
Thursday Matches						
THU 1	Al Duhail vs Al Shorta SC	16/04 01:45		1.53	3.60	5.20
THU 2	Rotherham vs Coventry	16/04 02:00		2.40	3.05	2.65
THU 3	Manchester Utd vs CF Granada	16/04 03:00		1.30	4.40	7.80
THU 4	Slavia Prague vs Arsenal	16/04 03:00		4.05	3.45	1.70

Figure 6. HAD odds of HKJC

所有公司	主勝	和	客勝	主勝率	和率	客勝率	返還率
bet 365(英國)	2.50	3.30	2.90	38.17	28.92	32.91	95.43
威廉希爾(英國)	2.50	3.25	2.90	38.00	29.23	32.76	95.01
立博(英國)	2.55	3.20	2.80	36.93	29.43	33.64	94.18
bet-at-home(馬爾他)	2.48	3.05	2.85	37.27	30.30	32.43	92.42
易勝博(安提瓜和巴布達)	2.43	3.20	2.80	38.06	28.90	33.03	92.49
Interwetten(塞浦路斯)	2.50	3.10	2.85	37.26	30.05	32.69	93.16
10BET(英國)	2.50	3.25	2.90	38.00	29.23	32.76	95.01
12BET(菲律賓)	2.53	3.15	2.98	37.71	30.28	32.01	95.39

Figure 7. HAD odds of Win007

3.3 The Lineup of Team

Lineup depicts the distribution of team players in each match. It usually composited by 3 to 4 integers, each integer represents the number of players at the position of the team's side of the soccer pitch. For example, according to Figure 8, the lineup of the home team (left-hand side) is 4-4-2 which represents that there are 2 sets of 4 players are placed in front of the goalkeeper and the remaining 2 players are placed in front of them.

The lineup of a team will be released 30 minutes before the match starts. In a soccer match, the lineups of the two teams are significant for the match result as each lineup has its own advantages and disadvantages. Thus, this information is collected.

Figure 8. Lineups of home team and away team in a match

3.4 League

There are many leagues in the world. However, this project only focuses on the leagues and cups belong to matches offered by HKJC.

Figure 9. List of the countries whose leagues are supported by HKJC

Figure 10. List of the type whose cups are supported by HKJC

3.5 Recent 10 matches

The meaning of recent 10 matches is the past 10 finished matches before the target match date. For example, if we are predicting a (target) match which will start on 23-11-2020, the recent 10 matches are the past 10 finished matches before 23-11-2020.

3.5.1 Recent 10 Matches of Home/Away Team

Past 10 finished matches of the home/away team before the target match date.

利華古遜 近 10 場 同主 歐霸盃 德甲 德國盃 球會友誼

類型	日期	主場	比分(半場)	角球	客場	皇冠 終盤 主	盤口	客	平均歐賠 終盤 主	和	客	全場 勝負	讓球	大小
歐霸盃	20-10-23	利華古遜	6-2 (2-1)	5-2	奈斯	1.07	一/球半	0.81	1.50	4.48	5.95	勝	贏	大
德甲	20-10-17	緬恩斯	0-1 (0-1)	6-2	利華古遜	0.94	*一球	0.96	5.22	4.39	1.58	勝	走	小
德甲	20-10-03	史特加	1-1 (0-1)	7-12	利華古遜	0.92	*平/半	0.98	3.13	3.76	2.17	平	輸	小
德甲	20-09-26	利華古遜	1-1 (1-1)	4-1	RB萊比	0.98	*平/半	0.92	3.08	3.68	2.21	平	贏	小
德甲	20-09-20	沃爾夫斯	0-0 (0-0)	4-4	利華古遜	0.88	*平/半	1.03	3.02	3.53	2.30	平	輸	小
德國盃	20-09-13	諸德施泰(中)	0-7 (0-6)	0-7	利華古遜	0.98	*四球半	0.84	43.72	18.45	1.02	勝	贏	大
球會友誼	20-09-04	利華古遜	1-1 (1-1)		安德列治	0.84	球半	0.98	1.47	4.25	5.49	平	輸	小
歐霸盃	20-08-11	國際米蘭(中)	2-1 (2-1)	3-6	利華古遜	0.85	平/半	1.04	2.10	3.55	3.41	負	輸	大
歐霸盃	20-08-07	利華古遜	1-0 (0-0)	7-4	格拉斯哥	0.86	半/一	1.04	1.66	4.27	4.70	勝	贏	小
德國盃	20-07-05	利華古遜(中)	2-4 (0-2)	3-5	拜仁慕尼	1.01	*球半	0.89	8.09	5.72	1.32	負	輸	大

Figure 11. Example of recent 10 matches of home team for a match on 02-11-2020

3.5.2 Recent 10 Encounters

Past 10 finished matches where the participated teams are as same as the target match. Reason that we collect such information is it was proven that data of encounters is useful [6].

對賽往績

近 10 場 主客相同 德甲 球會友誼 德國盃

類型	日期	主場	比分(半場)	角球	客場	皇冠 終盤 主	盤口	客	平均歐賠 終盤 主	和	客	全場 勝負	讓球	大小
德甲	20-02-23	利華古遜	2-0 (1-0)	2-3	奧格斯堡	0.84	一球	1.07	1.50	4.63	5.92	勝	贏	小
德甲	19-09-28	奧格斯堡	0-3 (0-1)	1-3	利華古遜	0.94	*半/一	0.97	4.39	4.27	1.70	勝	贏	小
德甲	19-04-27	奧格斯堡	1-4 (1-1)	3-3	利華古遜	1.06	*半/一	0.85	4.55	4.25	1.68	勝	贏	大
德甲	18-12-08	利華古遜	1-0 (0-0)	4-2	奧格斯堡	0.95	半/一	0.96	1.70	4.02	4.63	勝	贏	小
德甲	18-03-31	利華古遜	0-0 (0-0)	5-1	奧格斯堡	1.06	一/球半	0.85	1.48	4.44	6.53	平	輸	小
德甲	17-11-04	奧格斯堡	1-1 (0-0)	6-3	利華古遜	1.03	*平/半	0.88	3.22	3.63	2.13	平	輸	小
德甲	17-02-18	奧格斯堡	1-3 (0-2)	2-4	利華古遜	0.98	*平/半	0.93	3.27	3.26	2.26	勝	贏	大
德甲	16-09-22	利華古遜	0-0 (0-0)	6-3	奧格斯堡	0.98	一/球半	0.93	1.46	4.47	6.79	平	輸	小
德甲	16-03-05	奧格斯堡	3-3 (2-0)	0-5	利華古遜	1.12	平手	0.81	2.92	3.27	2.43	平	走	大
德甲	15-10-04	利華古遜	1-1 (1-1)	9-1	奧格斯堡	0.88	一/球半	1.04	1.38	4.69	7.87	平	輸	小

Figure 12. Example of recent 10 encounters for 利華古遜 vs 奧格斯堡 on 02-11-2020

3.6 Statistics of the Season of The Match

This statistic result is based on the past matches in the target season before the target match date. According to the Figure 13, we can see that there is a lot of data can be found from the statistic table.

聯賽積分排名

[德甲-12]利華古遜										
全場	賽	勝	平	負	得	失	淨	得分	排名	勝率
總	4	1	3	0	3	2	1	6	12	25.0%
主	1	0	1	0	1	1	0	1	14	0.0%
客	3	1	2	0	2	1	1	5	3	33.3%
近6	4	1	3	0	3	2	1	6		25.0%
半場	賽	勝	平	負	得	失	淨	得分	排名	勝率
總	4	2	2	0	3	1	2	8	4	50.0%
主	1	0	1	0	1	1	0	1	12	0.0%
客	3	2	1	0	2	0	2	7	2	66.7%
近6	4	2	2	0	3	1	2	8		50.0%

[德甲-9]奧格斯堡										
全場	賽	勝	平	負	得	失	淨	得分	排名	勝率
總	4	2	1	1	5	3	2	7	9	50.0%
主	2	1	0	1	2	2	0	3	9	50.0%
客	2	1	1	0	3	1	2	4	5	50.0%
近6	4	2	1	1	5	3	2	7		50.0%
半場	賽	勝	平	負	得	失	淨	得分	排名	勝率
總	4	2	1	1	2	1	1	7	8	50.0%
主	2	1	0	1	1	1	0	3	9	50.0%
客	2	1	1	0	1	0	1	4	9	50.0%
近6	4	2	1	1	2	1	1	7		50.0%

Figure 13. Example of a statistic table for a match in Win007

Figure 13 contains a lot of information and their meanings are as below:

賽: total matches,

勝: won matches, 平: drew matches, 負: lost matches

得: number of goals scored, 失: number of goals missed

淨: number of goals scored - number of goals missed

排名: ranking among the teams in same league,

勝率: winning rate

3.7 Player Information

The information and metrics of players were collected from 2 popular websites, FIFAIndex and win007. These may be crucial for the match result because the information can estimate the power of a team in a match. For example, a player with a preferred right foot will perform better plays on the right-wing.

3.7.1 Win007 Player Information

Win007 is a popular and comprehensive website containing lots of soccer data. The collected player information is similar to the Figure 14. The preferred foot and estimated value are expected to be significant features. However, we will mainly focus on the estimated value as the preferred foot is not a numeric number and we are still finding a handling process for it. In win007, the estimated value is calculated by different features such as the current salary of the player and the performance of the player in current season.

Figure 14. Example of a player information in win007

3.7.2 FIFA Player Information

FIFA is an extremely popular soccer simulation video game. It simulates the real soccer world vividly as all its soccer data is updated frequently. Since the update of the FIFA player information is too frequent, it is complicated to use the latest FIFA data before the start of a match. Thus, the FIFA player information in the beginning of a year will be collected. Moreover, we will focus on the OVR and POT which represent the ability and the potential of the player respectively. These 2 numbers are mainly based on the salary and the recent performance, this is also the reason that the update of FIFA data is so frequent. Also, the preferred positions which are the positions of the players in a lineup are collected as it might be useful while combining with the lineup feature. The column "Hits" represent the total number of shooting of the players.

		OVR	POT	Name	Preferred Positions	Age	Hits	
		94	94	Lionel Messi	CF ST RW	31	354	
		94	94	Cristiano Ronaldo	LW ST	33	352	

Figure 15. Example of a player information in FIFA from FIFAIndex

3.8 FIFA Soccer Team Score

OVR, a FIFA soccer team score which is calculated by a lot of features such as the recent performance and ranking of the team. Moreover, the ability of a newly joined player is one of the factor affecting OVR. Similar to the FIFA player score, frequent updates happen on FIFA soccer team score. In order to facilitate the calculation, we will collect the annual FIFA soccer team score only and uses the latest annual score prior to that match. Usually this score is released at the end of each year. For instance, a match played on 12-05-2019 will be using FIFA soccer team score generated on 31-12-2018 (latest). Among the FIFA soccer team score, we will use the ATT, MID and DEF which represent the attacking rating, midfield rating and defensive rating respectively. Although we do not know the exact formula of these 3 data, it should be calculated by the number of goals, number of miss, the quality of attack and the quality of defend etc.

	Name	League	ATT	MID	DEF	OVR	Team Rating
	Liverpool	Premier League	89	84	86	85	★★★★★
	Manchester City	Premier League	86	86	83	85	★★★★★

Figure 16. Example of a team information in FIFA from FIFAIndex

3.9 Overview

All the above data will be combined into JSON format and an organized version will be store in CSV. The CSV contains data as follows:

Indices are Match Time, Home Team, Away Team and League

Data:

- home_away_label: whether it is a home game or not
- HKJC_[home/away]_odds_[first/last]
- Offshore_[home/away]_odds_[first/last]: using handicap of HKJC
- Offshore_[home/away]_odds_[first/last]: using handicap of offshore bookmakers
- [home/away/homeVSaway]_past_ten_records:
    - Total_number_of_Goal_scored
    - Total_number_of_Goal_against
    - Total_number_of_Goal_difference
    - Average_total_number_of_Goal_scored_per_match
    - [winning/losing/draw]_rate
- Player_[weight/height/age/value/hit/power/potential_power]
- Team_[ATT/MID/DEF/power/rating]
- Weather
- humidity
- Lineup
- HKJC_handicap_results: This is our target label (Y)

In the final stage of the project, our dataset contains data from July 2017 to Dec 2020.

3.10 Evaluation Graph

We want to simulate the real-world situation when performing evaluations. Therefore, an arbitrary amount ($200 in our case) will be invested in each match. The profit or loss will be completely based on the model's suggestion. If the model predicts correctly, the profit for that match will be $\$200 \times (odds)$. In contrast, if the model chooses to bet on the wrong team, it will lose $200. Notice, $200 will be returned if the ground truth is "Draw". In other words, no gain or loss on that match. Only the testing set (20% of data) is used on evaluation. Due to the COVID19, most of the matches from April and June are cancelled. Therefore, a flat line is shown on the evaluation plots.

4. Feature Engineering

After collecting the targeted information from various websites, feature engineering will be applied after cleaning the dataset. First of all, a table-like dataset is generated from the JSON files, in which each row represents one match and each column represents one feature (Table). Afterward, feature engineering techniques were performed on the table. Eventually, this table of data will be passed to the modelling procedure.

matchTime	homeTeamName	awayTeamName	leagueName	hkjc_hdc_home_first	hkjc_hdc_away_first	hkjc_hdc_home_last	hkjc_hdc_away_last
2018-09-01 17:00:00	FC橫濱	京都不死鳥	日職乙	1.19	0.70	1.16	0.72
	水戶蜀葵	松本山雅	日職乙	0.80	1.05	0.82	1.02
2018-09-01 17:30:00	橫濱水手	柏雷素爾	日職聯	0.79	1.06	0.84	1.00
2018-09-01 18:00:00	大阪飛腳	川崎前鋒	日職聯	1.00	0.84	1.03	0.81
2018-09-01 18:30:00	尚州尚武	全南天龍	韓K聯	1.07	0.78	1.11	0.75
...	...	...	...	...	...	...	...
2020-07-31 08:30:00	山度士	邦迪比達	巴聖錦標	0.91	0.85	0.88	0.88
2020-07-31 17:30:00	西悉尼流浪者	威靈頓鳳凰	澳洲甲	0.98	0.79	1.02	0.76
2020-08-01 03:10:00	巴黎聖日門	里昂	法聯盃	0.81	0.96	0.82	1.00
2020-08-01 07:30:00	奧蘭多城	洛杉磯FC	美職業	0.75	1.03	0.77	1.00
2020-08-01 08:30:00	普埃布拉	藍十字	墨西聯	0.84	0.92	0.80	0.96

Table 2. Example of the data table

4.1 HKJC Time Relative Odds

For HKJC, the earliest and latest odds will be used. For the non-HKJC, instead of using all the handicap odds from non-HKJC bookmakers, the latest available odds before *the HKJC easiest odd* and before *the HKJC latest odd* are used with the same handicap line as the HKJC one. These data are named as [bookmaker]_hdc_hkjc[First/Last]_[home/away]. We tried to use the first and last odds offered by non-HKJC bookies but the performance was worse than using odds which are relative to HKJC's first and last released time.

#	時間	主勝	盤口	客勝	同盤場數	主盤勝率*
#7	10-28 02:47	2.02	[0]	1.81	9	11.11%
#6	10-27 22:31	2.05	[0]	1.78	15	20.00%
#5	10-27 22:28	2.09	[0]	1.75	8	25.00%
#4	10-27 22:28	2.07	[0]	1.77	13	15.38%
#3	10-27 21:38	2.03	[0]	1.80	19	26.32%
#2	10-27 21:12	1.99	[0]	1.83	10	30.00%
#1	10-27 11:41	1.96	[0]	1.86	14	28.57%

Figure 17. Example of the HKJC handicap odds

時間	比分	克魯	盤口	林肯城	變化時間	狀態
		0.74	受讓平手/半球	**0.96**	10-27 21:40	即
		0.90	平手	**0.80**	10-27 21:39	即
		0.84	平手	**0.86**	10-27 17:02	即
		0.77	平手	**0.93**	10-26 21:59	即

Figure 18. Example of the non-HKJC handicap odds

Moreover, since the odds will change from time-to-time, the handicap line may change if the odd is too small. Thus, while calculating the time relative odds, all bookmakers should use the same handicap line because it is necessary to keep a same criterion during comparison. An example is illustrated with the help of figures stated above (Figure 17), the handicap line of HKJC is "[0]" (Figure 17) which is same as the Chinese word "平手" (Figure 18). For non-HKJC bookies, the odds in the blue rectangle (Figure 18) will be used as it is the latest odds before the earliest odd of HKJC at 10-27 11:41. Similarly, the odds in the green rectangle will be used as it is the latest odds before the latest odd of HKJC at 10-28 02:27 with same handicap line "[0]".

4.2 Players and Team Scores

There are 11 players on each soccer team. All the net worth, power and potential power of the 11 players will be averaged into different indicators respectively. For example, we will take an average net worth of 11 players (columns) and transform them into one feature (column), ***average net worth***. These averaged data are used to represent the ability of players for one team.

In addition, team scores will be used also. Similarly, the power, attack score, midfield score and defence score for each team will be considered as features of the model too.

4.3 Encounters of Each Match

Encounter records for each match which the past matches of the same 2 teams are also taken into consideration. For example, Team A competes with Team B on 01/08/2020. The closest ten matches' results prior to 01/08/2020 constitute the winning, losing and draw rate, number of missed goals, scored goals for both teams. Such data will be considered in three dimensions, Team A alone, Team B alone and TeamA_TeamB. Currently, the data related to TeamA_TeamB is used in order to prevent too many features is in the model which might affect the performance of the model.

4.4 One Hot Encoding on League Name

League name is the name of the association of soccer matches, it is mainly based on the name of a country or continent. There is an assumption that the league of a match has a strong relationship with the odds. Since it is a string data, it cannot use as a feature directly. We tried 2 approaches to handle the league name, changing it into categorical data and doing one-hot encoding on it.

4.4.1 Categorical League

There are many different types of leagues so we used an unique number to represent each leagues.

hkjc_leagueName	西班牙甲組聯賽	阿根廷甲組聯賽	比利時甲組聯賽	挪威超級聯賽	意大利甲組聯賽	西班牙甲組聯賽	法國甲組聯賽	巴西甲組聯賽	巴西甲組聯賽	阿根廷甲組聯賽	...	日本職業聯賽	日本職業聯賽	南韓職業聯賽	日本職業聯賽	日本職業聯賽	日本職業聯賽	蘇格蘭超級聯賽	瑞典超級聯賽	瑞典超級聯賽	蘇格蘭超級聯賽
hkjc_leagueName_ID	68	35	87	75	89	68	10	5	5	35	...	46	46	28	46	46	46	38	70	70	38

Table 3. Example of the categorical league

Since we assume the league has a high relationship with odds, we evaluate the performance by using odds and the categorical league only. After passing these features to the modelling pipeline, Figure 19 is obtained but most of the models gave a bad performance.

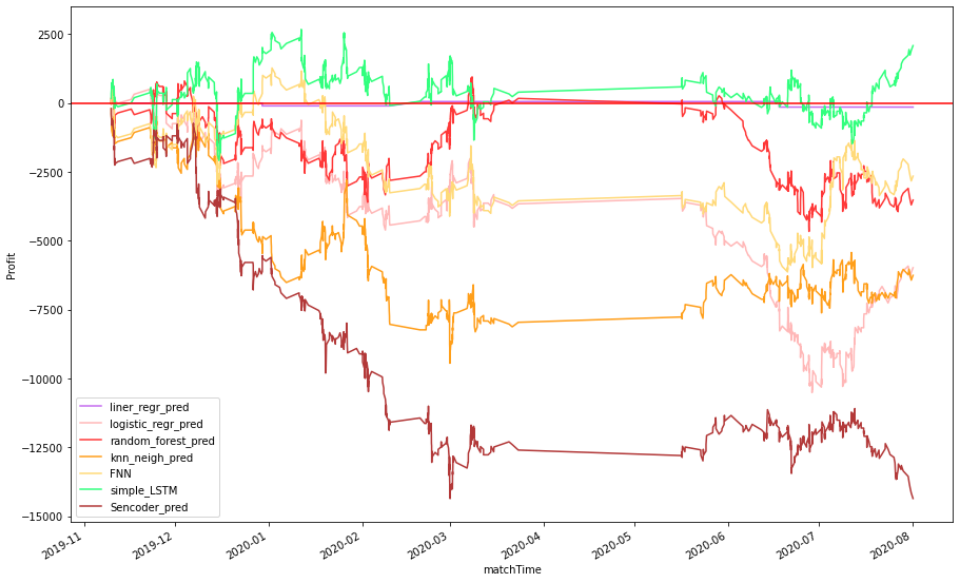

Figure 19. Example of the categorical league

4.4.2 One Hot League

According to the text book written by Anand Deshpande and Manish Kumar [7], one hot encoding can expressively represent the categorical data in a better way. Thus, we tried to the league name using one hot encoding.

...	league_西盃	league_西超杯	league_超霸盃	league_阿根廷盃	league_阿甲	league_阿超盃	league_非洲盃	league_非洲預選	league_韓K聯	league_韓足總
...	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
...	0	0	0	0	0	0	0	0	0	0
...	0	0	0	0	1	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
...	0	0	0	0	0	0	0	0	0	0

Table 4. Example of the categorical league

Similarly, we pass the dataset which contains one hot feature and odds to the modelling pipeline. According to Figure 20, all models except random forest model got a better performance than categorical league. It proves that the model performs better using one hot feature compared to the categorical feature. Therefore, one-hot encoded league is used as one of the features for our final model.

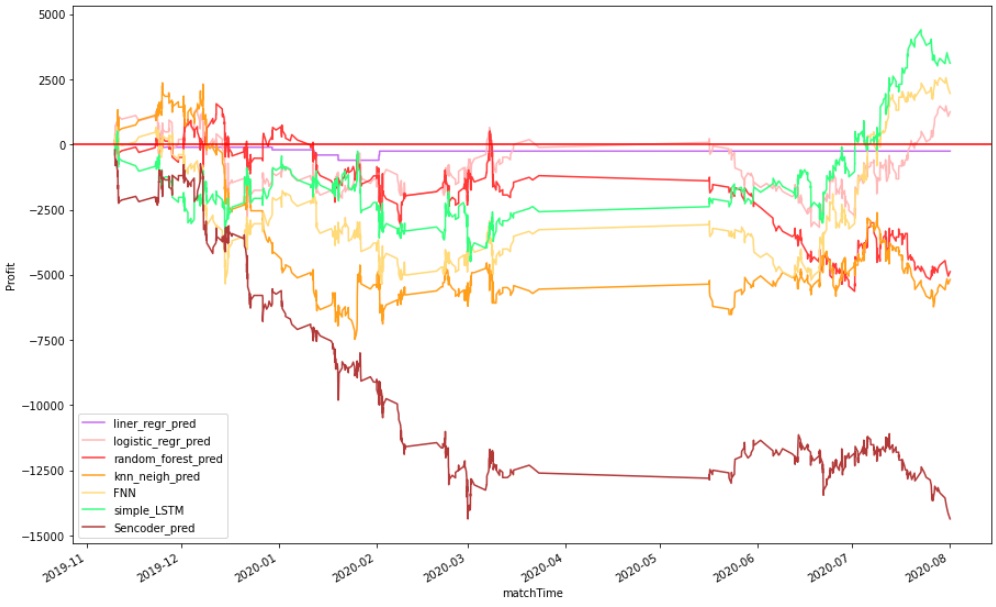

Figure 20. Example of the categorical league

4.5 Recent 10 Matches

Since the recent 10 matches data is not a numeric number, specific treatment is needed to turn them into some values so that it is suitable for modelling. For both home/away teams and encounter matches, the same method will be used to process the recent 10 matches.

對賽往績

近 10 場 主客相同 德甲 球會友誼 德國盃

類型	日期	主場	比分(半場)	角球	客場	皇冠 終盤			平均歐賠 終盤			全場		
						主	盤口	客	主	和	客	勝負	讓球	大小
德甲	20-02-23	利華古遜	2-0 (1-0)	2-3	奧格斯堡	0.84	一球	1.07	1.50	4.63	5.92	勝	贏	小
德甲	19-09-28	奧格斯堡	0-3 (0-1)	1-3	利華古遜	0.94	*半/一	0.97	4.39	4.27	1.70	勝	贏	小
德甲	19-04-27	奧格斯堡	1-4 (1-1)	3-3	利華古遜	1.06	*半/一	0.85	4.55	4.25	1.68	勝	贏	大
德甲	18-12-08	利華古遜	1-0 (0-0)	4-2	奧格斯堡	0.95	半/一	0.96	1.70	4.02	4.63	勝	贏	小
德甲	18-03-31	利華古遜	0-0 (0-0)	5-1	奧格斯堡	1.06	一/球半	0.85	1.48	4.44	6.53	平	輸	小
德甲	17-11-04	奧格斯堡	1-1 (0-0)	6-3	利華古遜	1.03	*平/半	0.88	3.22	3.63	2.13	平	輸	小
德甲	17-02-18	奧格斯堡	1-3 (0-2)	2-4	利華古遜	0.98	*平/半	0.93	3.27	3.26	2.26	勝	贏	大
德甲	16-09-22	利華古遜	0-0 (0-0)	6-3	奧格斯堡	0.98	一/球半	0.93	1.46	4.47	6.79	平	輸	小
德甲	16-03-05	奧格斯堡	3-3 (2-0)	0-5	利華古遜	1.12	平手	0.81	2.92	3.27	2.43	平	走	大
德甲	15-10-04	利華古遜	1-1 (1-1)	9-1	奧格斯堡	0.88	一/球半	1.04	1.38	4.69	7.87	平	輸	小

Figure 21. Example of recent 10 encounter matches

For every recent 10 matches, there are match results and handicap line columns. First, we need to determine the target team. For example, when calculating the recent 10 matches of away team, the target team will be the away team. Otherwise, the target team will be the home team even in encounter matches. Then, we will calculate the handicap result for each match in the recent 10 matches. For example, according to Figure 21, since this is the table of recent 10 encounter matches, the target team should be Bayer Leverkusen (利華古遜) which is the home team of the match. For the match in the first row, we can find that the result(or goal difference) is 2-0 and the handicap line is -1(一球). Thus, the handicap result will be 2-1-0 = 1. Similarly, we will do the same calculation for all 10 matches and sum them up. Finally, we will get a numeric number which will be one of the features for our model.

The reason for calculating in this way is that handicap results can represent the performance of the team in one match under a fair condition. As mentioned in Section 1.6, the handicap line is a number that can make the match to become fairer and evener. In other words, the probability of one team winning a match under the handicap line is close to 0.5 if draw is not considered. Thus, the handicap line is the estimated performance. Therefore, the handicap result can represent the difference between the estimated performance and real performance of a team.

In project phrase two, we present a new way to interpret this type of data, it will be mentioned in Section 5.5.4.

4.6 Team Names Merging

Since most of the team data are collected from 2 websites which are win007 and FIFAIndex, and each of them has their own data format, a merging process should be implemented on them. In the beginning, the merging process compared the Traditional Chinese team names from both websites because the English team names are not provided by win007. However, the merging performance is not what we expected as many team names are different between the two websites. For example, "卡山魯賓" in FIFAIndex is equivalent to "魯賓卡山" in win007. This behaviour can impede the merging process badly, as a results, we might loss a lot of useful data.

After a thoughtful consideration, we decided to create a team mapping table which contains all the team name from the 2 websites. We searched the team names one by one and there are 1803 teams in total. An example is shown on Table 5.

Index	Win007	FIFAIndex
15	哈特斯菲爾德	哈德斯菲爾德
31	阿克寧頓	艾寧頓
39	AFC 溫布頓	AFC 溫布頓
54	禾夫斯堡	沃爾夫斯堡
55	弗賴堡	費雷堡
59	史浩克 04	史浩克零四
63	雲達不來梅	雲達不萊梅
64	杜塞爾多夫	杜斯多夫
65	柏達邦	柏德博恩
67	奧斯納貝克	奧斯納布克
68	海登咸	海登海默
81	布倫瑞克	布倫斯維克
82	卡爾斯魯厄	卡斯魯厄

Table 5. A part from the team mapping table

After applying the team mapping table strategy, the team merging problem is solved and the number of records increased 60%.

4.7 Player Names Merging

Similarly, a merging process should be implemented on player data as we used data from two different websites. In the beginning, the merging process compared the English name of the players. However, our experiment result shows that a player can have a different English in win007 and FIFAIndex so it is difficult to combine them directly. Nonetheless, it is impossible to create a player mapping table similar to the team merging solution because the number of players is a hundred times more than the number of teams. After trying many methods, we decided to use set operations and searching method to alleviate the problem.

When the merging program compares player names from different websites, a set method will be applied, which is turning the player names from 2 websites into 2 sets. For example, “Erling Braut Haaland” will turn into [“Erling”, “Braut”, “Haaland”]. Then, the comparing process will check if the set from one website is the subset of the other set. If it is a subset of the other set, the program will further compare the team names according to the team mapping table to make sure if two names refer to the same person. On the other hand, if it is not a subset of the other set, a searching method will be applied which will search for a another name that refer to the same player from Transfermarkt[3] website. Then, using the set method again to compare the new sets.

After applying set method and searching method, although the name difference problem is not completely solved, the number of merged player data increased a lot.

[3] https://www.transfermarkt.com/

4.8 Data Analysis

In other to have a better understanding of our dataset. Some exploratory data analysis (EDA) is done to find out the relationship between variables through statistical approach.

4.8.1 Association between Odds and Handicap Result

In order to prove that the direction of our project is workable, we need to find out the association of odds and the handicap result. Since odd is a continuous variable and the handicap result is a non-dichotomous categorical variable, Pearson's Correlation Coefficient is not a suitable test to find out the association between them because it is used to find the relationship between 2 continuous variables [8]. Instead, we used Kruskal-Wallis H Test which is a nonparametric test that can prove the association between a continuous variable and a non-dichotomous categorical variable [9].

First, since we want to avoid the influence of the league on odds, we selected a league with a large number of matches and store all the corresponding records into a csv. As Kruskal-Wallis H Test cannot prove the association between 2 odds features (continuous variables) and handicap results (categorical variable), we introduced a new variable (hkjc_diff) using the HKJC initialled handicap odd of home team to minus the HKJC initialled handicap odd of away team. Moreover, we assumed that the significant level to reject the null hypothesis is 0.05. The null hypothesis of Kruskal-Wallis H Test is the two variables are independent. Afterwards, R is used to calculate the test and the result is shown in Figure 22.

```
        Kruskal-Wallis rank sum test

data:  hkjc_diff by hkjc_hdc_results
Kruskal-Wallis chi-squared = 40.591, df = 4, p-value = 3.266e-08
```

Figure 22. Kruskal-Wallis H Test between initiated odds and handicap result

According to Table 6, since the p-value is less than the significant level 0.05 by a lot, there is sufficient evidence to prove that there is an association between the odds and the handicap result. Similarly, we performed the same test on all bookmakers' odds and the result was as below table.

	p-value	
	Initiated odd	Last odd
HKJC	3.27E-08	0.01329
Bet365	5.85E-05	0.01096
Crown	0.0009125	0.003302
Macau	5.04E-06	0.01617

Table 6. Result of Kruskal-Wallis H Test

According to the table, there is sufficient evidence to prove the association between the odds and handicap result as all the p-values are less than 0.05. Also, there is an interesting finding is that all initiated odds have a larger significant association with handicap result than last odds in all bookmakers. Therefore, according to the result from the Kruskal-Wallis H Test, it is possible to predict the handicap result using odds.

4.8.2 The Number of Groups in Handicap Result

After proving the direction of our project is workable, we need to determine the number of classes of handicap result (label y). Originally, we set 5 classes for the handicap results. However, the actions of winning all the money and winning half of the money are both considered to bet on the home team handicap. A similar concept applies to betting on the away team. Thus, we want to find that if it is possible to combine the winning/losing half money and all money.

To find the differences between groups, Pairwise Wilcoxon Rank Sum Tests, which is a statistical test to check if the 2 groups are different or not based on the median [10], is used. Since taking all bookmakers and odds to perform the test is a little bit messy, we used the HKJC initialled handicap odds as an example instead.

```
        Pairwise comparisons using Wilcoxon rank sum test

data:  hkjc_diff and football_data$hkjc_hdc_results

   -2      -1      0       1
-1 0.00028 -       -       -
0  0.01833 0.60497 -       -
1  0.60497 0.00139 0.02506 -
2  2.6e-06 0.62821 0.62821 0.00040

P value adjustment method: BH
```

Figure 23. Pairwise Wilcoxon Rank Sum Tests between the groups in handicap result

According to the Figure 23, there is a sufficient difference between the winning/losing half money and all money. Thus, we will not combine them in this project. In other words, 5 classes will be used as the labels in the entire project.

4.8.3 HKJC Odds Compare with Other Bookmakers

The goal of this project is to conduct a model and betting strategy in order to have the maximum profit return when betting on HKJC odds. This leads to one interesting question that is "how will the profit be, if one bet on other bookmakers". In this section, we will try to answer this question by simulating the behaviour of betting via other bookmakers. To do so, we run our modelling pipeline with the default setting (The default setting is formulated at the end of project phrase 1). The only different is that the odds from other bookmakers will be used for evaluation (calculating profit) instead of HKJC odds. We selected three other popular bookmakers for comparison, Bet365, crown and Macau Slot. Table depicts the profit earned at the end using different models and the odds of different bookmakers under the same modelling pipeline. In other words, the prediction for each model is the same. It is interesting to see that Crown generally provides favourable odds for customer according to our dataset. On the other hand, the odds of HKJC are the worst for customer, the betting margin is almost the highest across all bookmakers. This imply that if one wants to achieve a higher profit when betting in soccer game, one should consider the bookmaker, Crown. In this project, we will stick to betting on HKJC, because it is the project goal and it is the most challenging one. If this project can eventually achieve a good profit under HKJC odds, it can even achieve a higher profit using other bookmakers.

	HKJC	bet365	crown	macau
odds_base	-1545942.0	-1490916.0	-1398971.5	-1341249.0
liner_regr_pred	161197.0	152678.0	233626.8	130850.0
logistic_regr_pred	389457.0	402119.0	591805.9	240487.0
random_forest_pred	-303154.0	-274929.0	-105251.5	67629.0
knn_neigh_pred	-447322.0	-422072.0	-244543.7	-469395.0
FNN	-864712.0	-834567.0	-621208.0	-598008.0
Sencoder_pred	364493.0	397609.0	588198.8	299317.0
expected_value	-1468570.0	-1468570.0	-1468570.0	-1468570.0

Table 7. Profit obtained comparison

4.9 Pre-processing

All of the data mentioned above will be passed to the pre-processing and modelling process. Normalisation of the dataset will be performed to ensure each factor is in a comparable scale. For example, according to Figure 24, the distribution of the player score and odds are on a very different scale, the standard deviation of player's power rating is a lot larger than HKJC odds. This might harm some of the models if two variables are in very different range and scale, the one with larger value might be dominated by the model. Hence, normalizing the dataset before any modelling is preferred so that each feature will have a zero mean and standard deviation equal to 1.

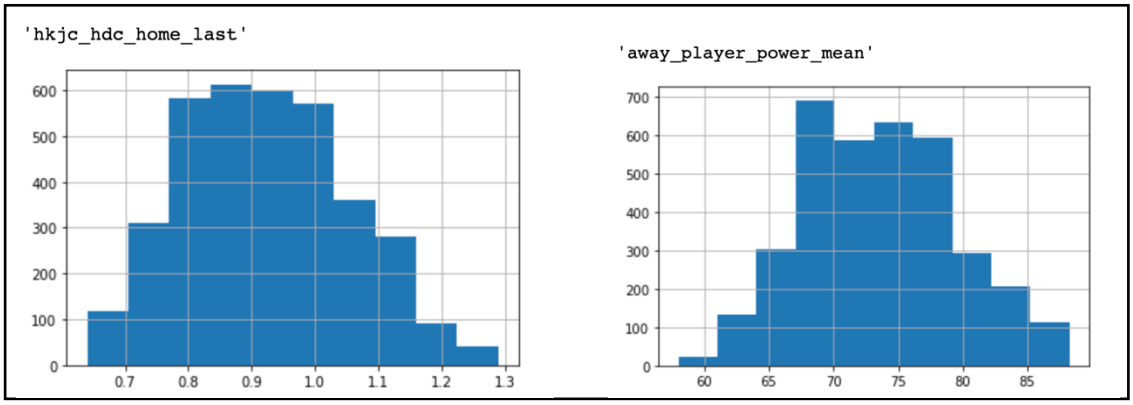

Figure 24. Distribution of two HKJC odds and player score

The histograms below (Figure 24) show the difference between before and after normalization. Each colour represents one feature. The raw data contained features with difference distributions where some were mean at 0 some were mean at 70. A uniform and standard version can be obtained after performing normalization.

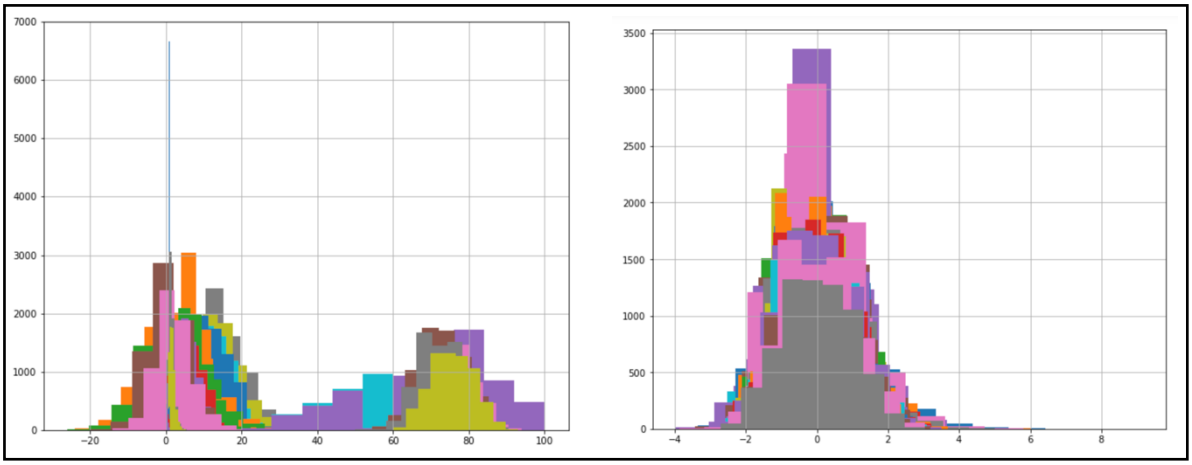

Figure 25. Normalization Results

4.9.1 Imputation

Due to the fact that our datasets were crawled from different websites and the data format used on different websites is different. The problem that leads to missing data happens easily. For instance, the whole match record will be dropped in the combining process if the data are not merged successful due to the different data formatting stated in Section 4.6 and 4.7. After our EDA, we found that these problems mainly happen on the player and team score. Missing values is problematic when the dataset is passed to modelling state. Hence, imputation is needed in advance.

Imputation is an approach to fill in the missing value by using some heuristics from the dataset. There are many intuitively reasonable methods for imputation, the popular and simple ones are forward/ backward fill and fill by mean/ mode/ median. In our dataset, we will mainly focus on player and team score, other features like *the total number of shooting* did not make sense for one to fill in. It is because this type of features can vary a lot from match to match, performing imputation on these features might bring more errors to our model. In contrast, the player and team information are relatively stable and can be inferred by some records on the existing dataset. Since the player score in a team usually similar in different matches so it is reasonable to use the past/future player scores to estimate the missing player score. Hence, it is not a big problem for us to use future player score as a heuristic to calculate the past missing value.

K nearest neighbours (KNN) imputer was used in the project with k = 2. It is an imputation method which uses an NAN compatible version of Euclidean Distance (or l-2 norm) to choose the closest k nearest neighbours for imputation. First, the distance matrix will be calculated for each data point. The formula is shown as below. Notice: *number of present coordinates* is the number of values which is not null in the coordinate of it in both data point.

$$dist_{i,j} = \sqrt{\frac{\#\ of\ coordinates}{\#\ of\ present\ coordinates}} \times l2_{norm}(i,j) \colon \forall i \forall j \text{ [11]} \qquad \text{Eq. 1}$$

For example, the missing data is at point ***i***, the closest point ***a*** and ***b*** will be used to calculate point ***i***. The further away the data point from ***i***, the less contribution to point ***i***. A mathematically formula is shown below:

$$i_j = a_j \times \left(1 - \frac{dist_{i,a}}{dist_{i,a}+dist_{i,b}}\right) + b_j \times \left(1 - \frac{dist_{i,b}}{dist_{i,a}+dist_{i,b}}\right) \colon \forall (missing\ j) \qquad \text{Eq. 2}$$

Here ***j*** is the coordinate of that missing data point.

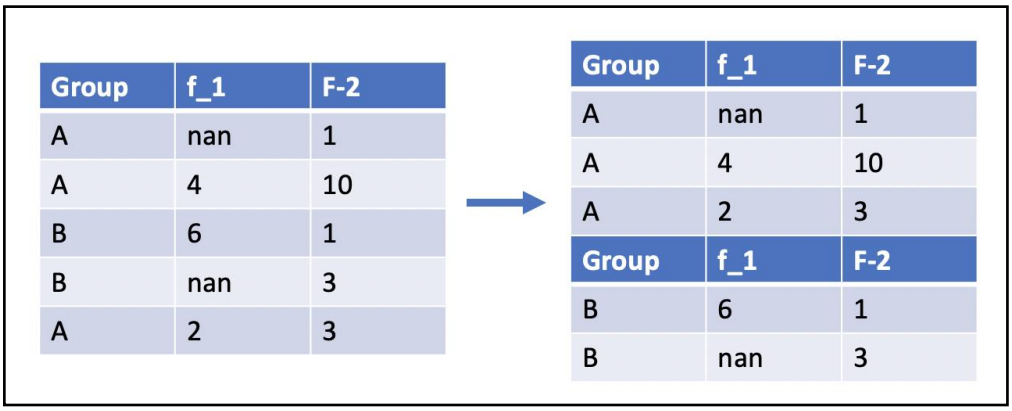

Figure 26. Illustration of separate by [team, league]

In particular, we will separate the dataset into home and away set because they are different teams. For each home and away set, we further separate the dataset by *[team name, league]* (the Group in Figure 26) because we assume even that the same team will perform differently in different league. KNN imputer will be applied at this level to fill in all the NAN values. One extreme case for the score of a player is that all the values in the separated dataset are NAN. In that case, we will replace those NAN by FIFA team's score.

4.9.2 Principal Components Analysis

Since there are many different types of features in our data, some of them might be linearly correlated, this might affect the performance and convergent of some models. In order to tackle the problem of correlated features, Principal Components Analysis (PCA) [12], a dimension reduction approach is used. PCA is an unsupervised method which calculate the eigenvalues and eigenvectors of the covariance matrix. Normalised data is preferred when calculating the covariance matrix so we always normalize our dataset. Since the covariance matrix is always symmetric so each of the decomposed eigenvectors will be orthogonal and therefore uncorrelated. Thus, performing PCA can help us to combine linearly dependent features so that a set of uncorrelated features can be obtained. In fact, the new feature set generated by PCA is a set of covariance matrix's eigenvectors.

A technique for PCA, proportion of variance explained (POV), is used to alter the dimension to be reduced instead of setting a hard threshold for it. When performing eigenvector decomposition on the covariance matrix, each eigenvector will have its own eigenvalue. This eigenvalue tells us that how much the variance can be explained by its eigenvector. If one uses all the eigenvectors, 100% of the variances can be explained but then it will not be a dimension reduction. Hence, we want to select a set of eigenvectors which can explain at least 95% of the variance. In this case, the dimension can be reduced and most of the original information of the input features can be obtained. One caveat of dimension reduction is that the reduced features become uninterpretable.

4.9.3 Autoencoder

Besides using the simple statistic approach to do the dimension reduction, an unsupervised neural network approach is tried. Autoencoder [13], a dimension reduction technique based on neural network, in which the input layer and the output layer are set to be the same. In order words, a neural network is predicting an output which is exactly same as the input. Thus, only the inner layer of the network structure is interested. The part starting from the input layer to the inner desired layer, it is called encoder or those layers are called encoded layers. While the part started from the inner interested layer to the output layer is called decoder or those layers are called decoded layer.

The strength of Autoencoder is that it is able to turn the input data into a reduced manner and being able to transform it back. Hence, the inner layer of the network is a compressed representation (latent factors) of the input layer with fewer dimensions. This inner desired layer will be used as the input features and passed into the modelling process. In our project, since we used other extra features like player rating, team rating. The autoencoder can help us to combine these features with the odds-type features, in which we assume to be most important, in a non-linear way.

In this project, a single layer autoencoder was constructed. A multi-layer autoencoder (Section 4.9.4) and an autoencoder with supervised learning (Section 5.5.3) was also developed. The autoencoder will compress the input features into a new set of features which dimension is 70% (in our case) of its original. For example, 70 factors will be able to represent the original 100 features after applying autoencoder. All the encoded layers and decoded layers will have a ReLU activation function except the last one. We have tried to use a ReLU in the last layer but the autoencoder failed to reconstruct its original features. It is mainly because the dataset we are using, contains negative values. In this case, the popular activation functions for autoencoder (sigmoid and ReLU) are not suitable in our use case. ReLU returns output which is larger or equal to zero whereas sigmoid yields value from zero to one. However, our dataset contains values range from negative infinite to positive infinite. Thus, a linear activation function was used in order to reconstruct the input data. An illustration of our autoencoder is depicted on Figure 27.

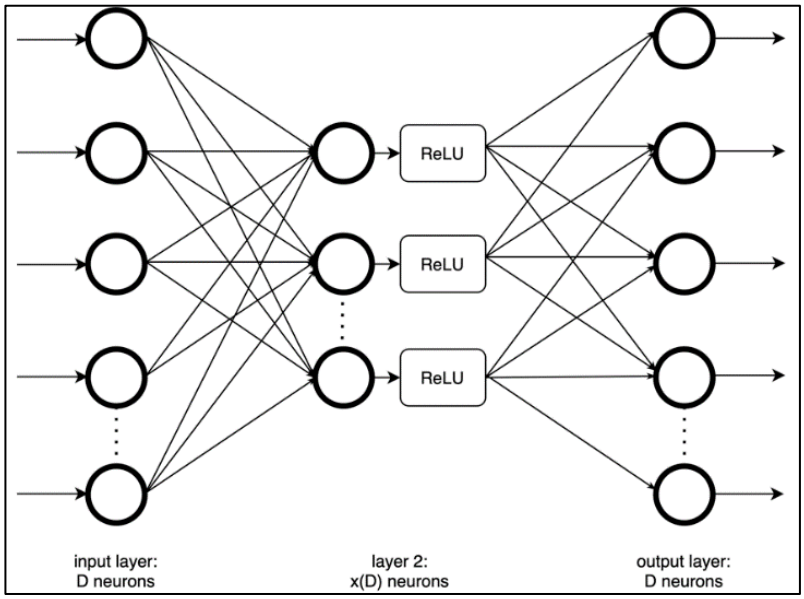

Figure 27. Architecture of Autoencoder

4.9.4 Stacked, Deep Autoencoder

A multi-layer approach on autoencoder which is called Stacked Autoencoder is also tried. The main difference between single-layer and multi-layer is that Stacked Autoencoder has more layers therefore it is able to perform more complex tasks. There are actually two methods for training a Stacked Autoencoder, the first way is literally stacking multiple autoencoder together as mentioned in [14]. However, instead of using the label as the fine-tuning step. At this stage, we just want to focus on dimension reduction; therefore, we will stop at the deepest inner hidden layer and use this hidden layer as the input to other models. The Stack Autoencoder with supervised fine-tuning will be introduced in Section 5.5.3.

The first way is a greedy layer-wise pretraining methods (Stacked Autoencoder) which trains the first hidden layer of the first autoencoder. Next, using the previous hidden layer obtained from the first autoencoder to train another autoencoder's hidden layer. Finally, repeating this process until the desire number of layers is reached. In other words, n single-layer autoencoders are needed to train in this approach. The other way is to train all the layers at once (Deep Autoencoder), setting up one universal loss function (MSE in our case) and connecting all the layers together. The advantages of the first approach is that each hidden layer contains a lot of information encoded because it is trained greedy and by layer whereas the second method is more sensitive to the initial weights of the deep neural network which may have a stronger regularizing effect to the model [15].

In this project, both methods were tried. The architecture of the two methods is very similar in our case, both have 3 hidden layers and using ReLU as the activation function of the encoded layers. Besides the way to train the network, one major difference is that all of the activation function in decoded layers are linear in greedy layer-wise methods whereas only the last activation function is linear in the deep learning approach. Figure 28 shows their difference.

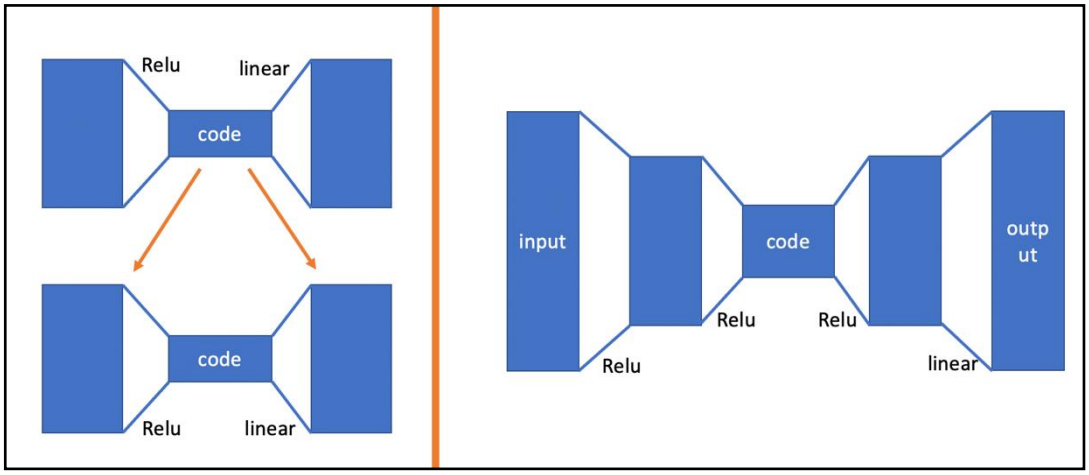

Figure 28. Illustration of Stacked Autoencoder and Deep Autoencoder

4.10 Data Augmentation

The entire project is focused on the soccer game that is available on HKJC, where the number of games available are limited, and people have stopped playing soccer game for few months due to COVID-19 outbreak. As a result, the data size of the project is not very large, thus, data augmentation is performed to enlarge the size of our dataset. In this project, we tried two popular ways to do data augmentation, the first one (Bootstrap sampling) is a statistical way and the other (GAN) is a deep learning way.

4.10.1 Bootstrap Aggregating

Bootstrap aggregating [16] (or bagging) is one of the ensemble strategy that takes in multi predictors and return a single prediction. Bootstrap sampling is a technique to re-sample data with replacement from the original dataset, resulting some data may occur more than one times or did not exist in the new dataset at all. In our case, we set the number of sample as 1.4 times of our original dataset to mimic the behaviour of having a larger dataset. In addition we repeat this step 5 times to reduce the bias. In other words, there are 5 new datasets obtained, where all of them are 1.4 times larger than original one. Each of the 5 new datasets will be passed to modelling phrase and 5 predictions will be generated. Bootstrap aggregating is a strategy that can turn these five predictions into one. Mathematically, we want $y = f(X)$ where X is a vector that contains the 5 predictions and y is the final prediction.

In our pipeline, the first 80% of the data will used for bootstrap aggregating and the remaining 20% will be used for final prediction. For example, if our original dataset has 100 records, the first 80 records will be passed to bootstrap. As a results, 5 datasets (112 records each, because 1.4 times larger) are passed to modelling stage. We will ensemble the 5 predictors, that are generated from the modelling stage, into one predictor. Eventually, we will use the ensemble predictor to do prediction on the last 20 records from the original dataset.

In the applied Predictive Modelling book [17], the authors suggested to predict on the “out-of-bag” data. Notice: “out-of-bag” data refer to those which are in the original dataset but not in the newly generated bootstrap dataset. However, we did not directly adopt the mentioned strategy due to the natural of our dataset and problem. Since we are predicting on the last 20% of our dataset chronologically and the “out-of-bag” data are located randomly among the entire dataset, resulting not much of the “out-of-bag” data are located in the last 20%. Hence, We will directly use the last 20% of bootstrap dataset to perform evaluation and cross-validation. The experiment results for this data augmentation methods can be found in Section 7.4.3.

4.10.2 Generative Adversarial Nets

Instead of using a statistical approach to perform data augmentation, one of the popular generative model is considered. Generative Adversarial Nets (GAN) is widely used in image generation, we are inspired by this idea and would like to have some experiment on whether we can generate data in this way so to increase the size of our dataset.

GAN

GAN [18] is an implicit generative model which do not require explicitly stating the density function. It contains two neural network models, a discriminator and a generator. A discriminator is used to tell if the input data is a real data (in our case, any data point in our dataset) or a fake data (any data that is not draw from our dataset). A generator try to generate some data and it's goal is to fool the discriminator with the generated data. A minimax game training approach is used to build a GAN. Firstly, random noise is generated as fake data and input to discriminator. The discriminator would like to maximize the probability of correctly identifying the real data. Afterwards, random noise will be inputted to generator and it will try to return an output (generated data) such that the probability of discriminator correctly identifying the real data is minimized. These two processes keep repeating until the loss the minimized.

CTGAN

Since our dataset is a tabular instead of images, some columns (features) might be categorical and GAN is used to transform random noise into continuous data so the discrete features needed to be tackle in a special way. Conditional GAN (CTGAN) [19] [20] is used to solve this problem. The difference between CTGAN and GAN is that CTGAN will sample the row (data point) according to the discrete features. For example, if we have two categorical column $\boldsymbol{D_1}$ and $\boldsymbol{D_2}$ where $\boldsymbol{D_1}$ is range from 1 to 3 and $\boldsymbol{D_2}$ is either 1 or 2. We will randomly select a column between $\boldsymbol{D_1}$ and $\boldsymbol{D_2}$, lets says $\boldsymbol{D_2}$ is selected. Afterwards, we will randomly select a categorical value from $\boldsymbol{D_2}$ and assume the selected value is 1. Since every categorical columns can be represent in one-hot vector. A ***cond*** vector, which is the combined one-host mask ($\boldsymbol{m_i}$) of the column $\boldsymbol{D_i}$ over all column $\boldsymbol{D}$, will be [0,0,0,1,0] in the above example. It is because we selected $\boldsymbol{D_2}^* = 1$ so one-hot mask for $\boldsymbol{D_1}$ and $\boldsymbol{D_2}$ are [0,0,0] ($\boldsymbol{m_1}$) and [1,0] ($\boldsymbol{m_2}$) whereas ***cond*** is the concatenation of $\boldsymbol{m_1}$ and $\boldsymbol{m_2}$. Generator take random noise as input and try to minimize the probability of discriminator to have a correct prediction with extra cross-entropy loss on $\boldsymbol{m_i}^*$ and $\hat{\boldsymbol{m}}_i^*$ where $\boldsymbol{m_i}^*$ is the inputted ***cond*** and $\hat{\boldsymbol{m}}_i^*$ is the generated ***cond***. The discriminator is exactly the same as normal GAN. Figure 29 illustrate the entire training process of CTGAN.

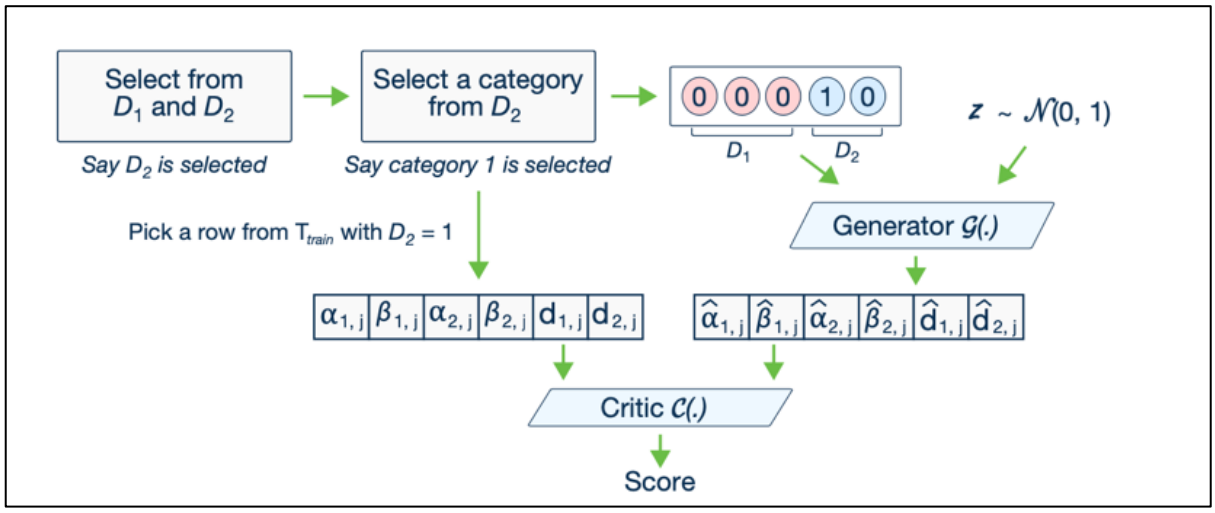

Figure 29. CTGAN procedure [19]

Bookie Prediction

Since odds are one of the major feature in our dataset, for the same set of odds from two data points, it is possible for the final results (y) might be different. We want to model this behaviour so that we can have a more stable model. One assumption is that bookmakers tends to publish a lower odds for a team if they expect that team to win at the end. Thus, we apply this concept to the initial odds released by bookie and generate a new variable ***bookie_expected***. ***bookie_expected*** will be assigned to 1 if the lower odds team win eventually, else, it is assigned to be 0. The physical meaning is to mark down if the bookie did a correct prediction on that march. Once the odds are released, it keeps changing according to the supply and demand in the market. We can interpret it as a lower odds imply more people believe that team will win eventually. Hence, a new feature (***customer_expected***) is created for this behaviour, the protocol of generating this feature is same as ***bookie_expected***. To further analysis these two newly created columns, we can create a new column ***bookie_customer_expected*** such that if bookmakers and customer belief are the same, the value is set to 1 and 0 otherwise. The three newly created columns are a non-linear transformation of the results (y), hence they will only be used for GAN generator and will be discarded before passing to modelling phrase.

Test-like Data

From the further analysis on the Cluster-then-Predict model in semester 2 (Section 7.1), we suspect that the distribution of the dataset vary over time. It makes sense since the strength of soccer team or player change over time. We make a hypothesis base on this assumption, which is the data at time t is more related to the data at time $t +/- i$ than $t +/- j$ where $i < j$. Because of this, we build another discriminator on top of CTGAN and use it to discriminate the testing data and training data. In our pipeline, training data are the first 80% of data and testing data are the remaining 20% in a chronological order. In other words, the testing data is newer than the training data. We want the generated data to look like testing data instead of training data. Same as the ***bookie prediction*** procedure, we only apply the ***test-like data*** concept on the first 80% of the data. Within this 80% data, we do a train test split again so that we have three dataset. For example, for a dataset that have 100 data point, we split it into three sets such that they have 64, 16, 20 data points respectively. In ***test-like data*** case, we apply test like CTGAN on the first two datasets such that the generator would generate data that look like the one on the second dataset, meanwhile, the discriminator will try to discriminate the data between the first and second dataset.

5. Modelling

5.1 Metrics

Two types of metrics were used. The first one is 5-class-accuracy which is equal to number of matches predicted correctly divided by the total number of matches.

The other metric is the accumulated profit over time. This is the most important metric if one want to have a profitable result. However, in this project, the 5-class-accuracy will be the main focus and used in hyperparameter tuning because accuracy can guarantee the profit in a more general way. Comparing with the F1 score, the accuracy focuses more on the true positive so it will be more suitable for this project as only true positive can guarantee to avoid a loss (since we will not bet when the true positive is class 0). In other words, higher accuracy will have a larger probability to win the future matches.

5.2 Hyperparameter Tuning

Every machine learning model has its own set of hyperparameters that work the best for the desired problem. For instance, number of neighbours considered in K nearest neighbour, activation function used in neural networks. It is difficult for one to explain why this set of parameters work in this case but not the others, especially in deep learning where it is a black box inside. Therefore, for each model used in this project, we perform hyperparameter tuning to choose the best set of parameters which is suitable for the dataset. In other words, a set of hyperparameters will be inputted manually during the training process. If one want to tune two set of parameters by giving two lists which length are X and Y. The total number of combinations will be XY and hence the algorithm will need to run XY times. Since the goal of term 1 was to set up a pipeline so we used grid search to perform hyperparameter tunning as to prevent the randomness. On the other hands, random search was the main focus in term 2 because the goal was to do optimization. Although more computation power is needed for random search, it tends to give a better parameter set if we increase the sample size from the parameter set. In our case, each neural network model will be passed to a random search with 100 random parameters so to increase the chance of finding the best parameters of our problem.

Since searching best hyperparameters on the training set might easily lead to overfitting which will affect the performance on the testing set. As a result, 5-fold cross validation was adopted, each fold will have 20% of the training data and one of the folds will be used as the validation set. For each set of hyperparameters, 5 training and testing are performed and the

average of the 5-class-accuracy-metrics (Section 5.1) is obtained. Finally, the set which has the higher average metrics will be the final candidate.

20%	20%	20%	20%	20%
Validating	Training	Training	Training	Training
Training	Validating	Training	Training	Training
Training	Training	Validating	Training	Training
Training	Training	Training	Validating	Training
Training	Training	Training	Training	Validating

Table 8. 5-fold cross validation

5.3 Benchmark Models

Serval benchmark models were developed. The main purpose of having such models is to compare the performance of our models. Two strategical betting methods and one statistical metric were constructed.

5.3.1 Odds-based Prediction Model, Strategical Betting

A simple random betting strategy will be betting according the odds. The rationale of this is bookmakers will lower the odds for the team which they think will win (This assumption is also used in Section 4.10.2 test-like data). Therefore, a counter-strategy on this was adopted. In other words, we will bet on home team if the handicap odd of home team is lower and vice versa.

5.3.2 Naïve Betting Model, Strategical Betting

Besides randomly betting on each match, one may use extra information to help them make a better prediction. For example, when betting on Newcastle versus Chelsea, one might use the past winning rate for each team to help them make the decision on whether to bet on the home team or away team. Hence, such a strategy was adopted.

The model will check if the winning rate of home team is larger than the away team by at least 0.3. If the statement is true, the model will bet on the home team. If vice versa, it will bet on the away team. Otherwise, it will choose to not bet on this match. A threshold of 0.3 was set for a conservative betting.

```
if home_team.winning_rate > (away_team.winning_rate + 0.3):
    bet_home()
elif away_team.winning_rate > (home_team.winning_rate + 0.3):
    bet_away()
else:
    no_bet()
```

Figure 30. Pseudocode for navie model

5.3.3 Expected Value Model, Statistical Metric

Since the bookmakers will ensure the gamblers to have a negative expected return, this negative expected value was calculated. In other words, the average profit after randomly guessing for many times should converge to the expected value. Indeed, if any models perform better than the Expected Value curve, the models will be considered to be good. It is because it outperforms the random guess. Notice: the expected value curve will always be decreasing (having a negative return).

$$EV = \frac{(invest_{money} * home_{odd} + invest_{money} * away_{odd})}{2} \quad \text{Eq. 3}$$

According to the blog post written by Smarkets [21], the expected value is a measurement that to calculate the money that can be got by the bettor. However, the expected value in gambling in soccer is always negative because it is not always be a fair game. Suppose that there is a match that home team has 50% winning the match and away team also. It is intuitive that one should get 0 money if he/she bets both teams winning the match. However, it is not true in reality because there is betting margin hided in the odds. Therefore, the bookmakers can always earn money from the gambling.

5.4 Statistical Models

Statistical models are the models that applied statistical analysis, like mathematical representation. Most of the statistical models predicted the further result based on the inferred relationship between the data. Thus, it is similar to the neural network. However, neural network can have better inference about the relationship among the data due to the non-linearity [22]. Notwithstanding this, statistical models still have a pretty good performance.

5.4.1 Linear Regression

Linear regression is an approach to model the linear relationship between the response and the explanatory variable. Since it can only represent the linear relationship, it is seldom to use for classification, especially the multiclass classification. Thus, the basic requirement of other models should get a better performance than linear model. In our case, we will round the predicted output to nearest integer to mimic the behaviours of classification. It turns out to be one of the most conservative models compare to all others that we have developed. As there is more than one feature in our model, multiple linear regression model is used. The formula of linear regression is $Y = X\beta + e$ where Y is the response, X is the explanatory variable and e is the error. Since it is a multiple linear model

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix},\ X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ 1 & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix},\ \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} and\ e = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix} \qquad \text{Eq. 4}$$

where n is the number of matches and p is the number of features.

Moreover, we tried to apply elastic net regularization into the linear regression. Nonetheless, all coefficients become zero and finally only the intercept is remained. Thus, linear regression without regularization will be used in the following evaluation.

5.4.2 Logistic Regression

Logistic regression is an approach to model the probability of a binary dependent variable. Thus, what we applied for our project is multinomial logistic regression which is similar to a composition of sets of binary logistic regression. Hence, the formula of the probability of $\boldsymbol{i^{th}}$ match belonging to class $\boldsymbol{k}$ is

$$P(Y_i = k) = 1 - \sum_{j \in K \setminus \{k\}} P(Y_i = j) \text{ where } \boldsymbol{K} \text{ is the set of classes.} \quad \text{Eq. 5}$$

If $\boldsymbol{k}$ is the last class,

$$P(Y_i = k) = 1 - \sum_{j \in K \setminus \{k\}} P(Y_i = k) e^{\beta_j X_i} \text{ where } \boldsymbol{X} \text{ is the feature scalar} \quad \text{Eq. 6}$$

$$= \frac{1}{1 + \sum_{j \in K \setminus \{k\}} e^{\beta_j X_i}}$$

If $\boldsymbol{k}$ is not the last class,

$$P(Y_i = k) = \frac{e^{\beta_k X_i}}{1 + \sum_{j \in K \setminus \{k\}} e^{\beta_j X_i}} \quad \text{Eq. 7}$$

Moreover, a hyperparameter tuning is applied on logistic regression so the best strength of inverse of regularization which is the penalty to avoid the model be overfitted. Thus, $c = \frac{1}{\lambda}$ will be added into the cost function. Therefore, the following evaluation of logistic model will adopt this parameter.

During project phrase 2, while we are having experiment on CTGAN, we found that the logistic regression yield a good performance on one league, hence, we selected it as one of the candidate of our final best model. This selected model is named as logistic 2.0. On the other hands, the logistic on our default pipeline is selected also and named as logistic 2.1.

5.4.3 Random Forest

Decision tree is a very popular machine learning technique, thanks to its robustness and interpretability. It uses a greedy and divide & conquer approach to build a tree of rules for the dataset. First, it tries to create a rule from a set of one or multiple features then choose the set which has the lowest impurity for splitting. Then, one or more internal nodes are created and the same procedure happens on each internal node until all of the nodes become leaf node. A leaf node is a node which contains data from one class or it has an impurity of zero. The impurity can be measured by entropy or gini. $gini = 1 - \sum_i^k {p_i}^2$ and $entropy = -\sum_i^k p_i log_2(p_i)$. [23] The lower the value (for both functions), the lower the impurity. The two functions are actually very similar; therefore, we will put them into the hyperparameter tuning process.

We have also passed the maximum tree depth to hyperparameter tuning. Decision tree can be overfitted easily if we let it to grow forever. Hence, setting a depth limit can stop the tree from memorise the training set and perform worse in the testing set.

Random forest is an ensemble of decision tree, it consists of a collection of trees which are trained with different parts of dataset (either in observations or features). For the classification problem, the label with the majority vote of trees is the final output. Random forest can help to prevent overfitting as the output is determined by multiple trees.

5.4.4 XGBoost

The full name of XGBoost is eXtreme Gradient Boosting which is a gradient boosting decision tree. Boosting is one of the ensemble methods that a strong classifier is generated via combining the effort of several weak classifiers. Comparing to the decision tree, all the weak classifiers are dependent because the weak classifier will add weightings to the data before passing it to the next weak classifier. The value of the weighting in proportion to the classification error so the misclassified data will be emphasized by the next weak classifier. Thus, the next weak classifier can always improve the weakness part of the last weak classifier greatly. Gradient boosting is extremely similar to boosting, in which the difference is that minimizing the lose function via adding weak classifiers in gradient descent is the objective of the former [24].

The loss function of XGBoost called the objective function which this equation is as below.

$$\mathcal{L}^{(t)} = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) * w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \qquad \text{Eq. 8}$$

g_i: Gradient (First order derivative $\partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$)

h_i: Hessian (Second order derivative $\partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$)

T: The number of leaves

w_j: weight/leaf score

λ: L2 regularization

γ: Complexity of adding a new node

The gain after splitting

$$= \frac{1}{2}[score\ of\ left\ subtree + score\ of\ right\ subtree - score\ of\ not\ spliting\ ]$$

$$= \frac{1}{2}\left[\frac{\sum_{i\in I_j} g_i{}^2{}_{Left}}{\sum_{i\in I_j} h_i{}_{Left} + \lambda} + \frac{\sum_{i\in I_j} g_i{}^2{}_{Right}}{\sum_{i\in I_j} h_i{}_{Right} + \lambda} - \frac{\left(\sum_{i\in I_j} g_i{}_{Left} + \sum_{i\in I_j} g_i{}_{Right}\right)^2}{\sum_{i\in I_j} h_i{}_{Left} - \sum_{i\in I_j} h_i{}_{Right} + \lambda}\right] - \gamma \quad \text{Eq. 9}$$

According to the above equation, it is clear that the weighting of XGBoost is influenced by the maximum depth of tree, the L1 and L2 regularization so they will be the tunning parameters in hyperparameter tuning to reduce the probability of overfitting.

5.4.5 K-nearest Neighbour

K-nearest neighbour (KNN) [25] is a nonparametric method in which no assumption on the dataset is needed, for instance, the dataset does not need to follow a certain type of distribution. It is also a lazy learning technique in which no model was built for prediction. In fact, we have used this technique in Section 4.9.1 when we are discussing about the imputation. The concept of KNN classifier is very similar to KNN imputer. Since all the missing values are handled before modelling so we can directly use Euclidean Distance or any other distance metric without handling with the NAN.

Consider $\boldsymbol{x}$ as an unlabelled input data and $\boldsymbol{k}$ as the parameter. $d_k(x)$ is the distance from $\boldsymbol{x}$ to the $\boldsymbol{k}$-nearest neighbour. The label of $\boldsymbol{x}$ will be determined by points which have $d_a(x) : a \in [1, k]$. In uniform distance weighting, label which has the majority of votes from these points will be the label of a new unlabelled data. In inversed-distance weighting, each vote has its weight of $\frac{1}{d}$, data point which is closer to the new input, the more weight it has. Hence the label of the new input is determined by the weighted vote. In this project, two weighing schemes and the optimal value for $\boldsymbol{k}$ will be pass to hyperparameter tuning.

5.5 Neural Network Models

Neural Network, also known as Artificial Neural Networks (ANNs). It is a statistical model which mimics the activities happen in human brain.

ANNs, as shown in its name. It involves the collection of neurons (nodes or units), each neuron represents a mathematical operation. Each neuron can either receive or send a signal (number) to other neuron. Two neurons can transmit signal via a directed edge which has a weight assigned on it. During the transmission, after a signal left its sender, the signal will multiply with weight and finally arrive at the destination. Usually, a neuron will receive many signals at each time, all these signals will be added together and pass to an activation function. As a result, a new single signal is formed and eventually send to another neuron. Usually a neural network consists of one input layer, multiple hidden layers and an output layer. Each layer contains a group of neurons, the structure of the ANN depends on how the neurons are linked by edges. Records of a dataset will be taken as the input to ANN and the outputs are the representations that we want the ANN to learn. Each time when inputs are fed into the ANN, the weight of each edge will be updated until the ANN is able to generate the prediction correctly.

5.5.1 Feedforward Neural Network

Feedforward Neural Network (FNN) is a special type of ANN which the signals flow from input to output in a forward direction only. Its goal is to approximate a function $y = f(x)$ where when the input is $\boldsymbol{x}$ and output is $\boldsymbol{y}$ [26]. Since this project is a 5 classes classification problem. The output layer of the FNN will contains 5 neurons where each neuron represents one class (handicap results). Hence, a softmax activation function is used on the last layer so that each neuron will return a value between 0 to 1 and most importantly, the sum of these 5 numbers is equal to 1. In other words, each of the neurons in the output later represent the probability of each class.

There are some refinements on our FNN over time. Mainly, one was built in term 1 and we name it FNN 1.0. The updated one was developed in term 2 with intensive fine tuning, we name it FNN 2.0.

FNN 1.0

In the architecture for our FNN 1.0, it consists of 3 hidden layers in total. The number of neurons in the hidden layers was set in a dynamic way. A list of three number $\boldsymbol{x = [x_1, x_2, x_3]}$ determine the number of neurons in each hidden layer where $\boldsymbol{x_i}$ is in $\boldsymbol{(0, 1)}$. For example, a $\boldsymbol{D}$ dimension of vector and the $\boldsymbol{x=[0.8, 0.5, 0.3]}$ is inputted into the FNN, the number of neuron in first hidden layer is equal to $0.8D$, the second and third are $0.5D$ and $0.3D$ respectively. If $x_i D$ is not an integer, a round-up operation will be applied on it. The optimal list of three numbers ($\boldsymbol{x}$) will be searched by hyperparameter tuning, where a set of lists will be inputted and the optimal one will be used. On the other hand, the activation functions used in the hidden layer are all ***sigmoid*** function. We have tried different variations of activation function and the best one is to apply ***sigmoid*** function on all the layers. Figure 31 visualizes the overall structure of the FNN in our project phrase 1.

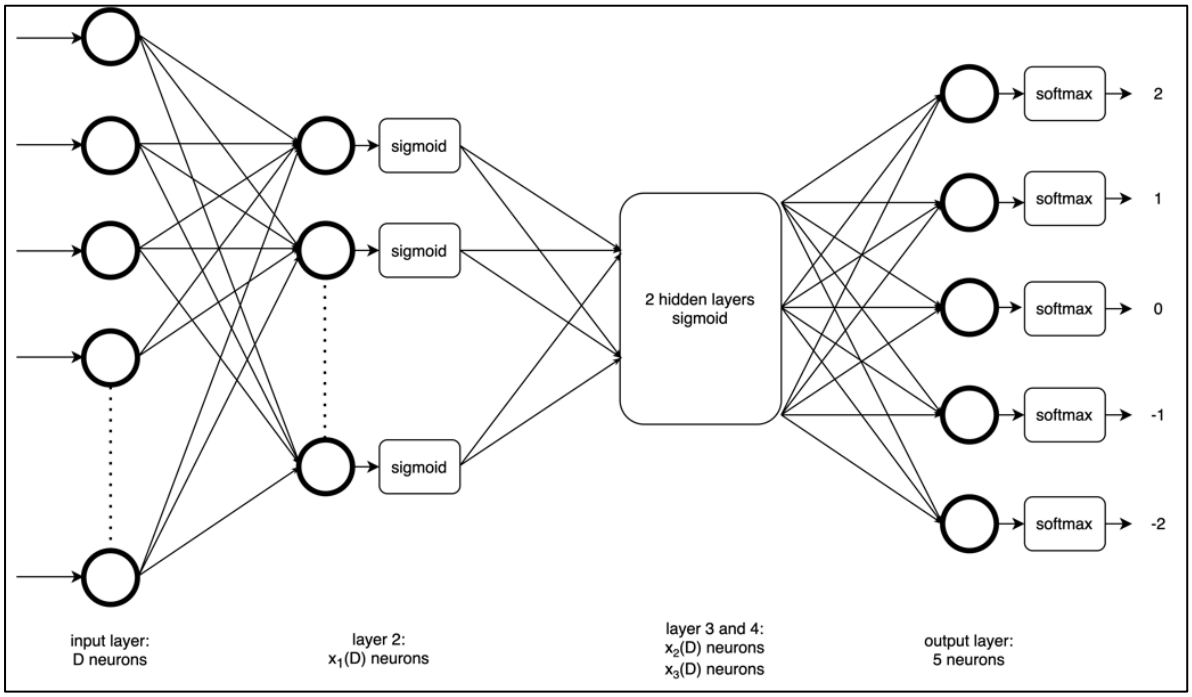

Figure 31. Network structure of the Feed Forward Neural Network

FNN 2.0

In semester 2, we decided to re-work on our FNN because there are more data available and we want improve the predictability of our model. Firstly, we relax the constrain on the number of layers in the FNN. It contains an arbitrary number (***m***) of layers, in other words, list of number of neurons will be $\boldsymbol{x = [x_1, x_2, \ldots, x_m]}$. This list ***x*** is added into the hyperparameter set and pass to the random research scheme. Another problem we observed in FNN 1.0 is that the network is not wide and not deep, causing the network is not good at both memorization and generalization [27]. Hence, we relax the constraint on the value of $\boldsymbol{x_i}$. Since the network structure will be passed to hyperparameter tuning and cross-validation, we do not worry about the final parameter set will end up will a large $\boldsymbol{x_i}$ because a model that is extremely good at memorization (overfit model) will be filter out on the cross-validation stage.

Since we allow a deeper network structure, the vanishing gradient problem might happen and needed to be tackle. Vanishing gradient occurs when the backward flow gradients become zero. We used ***sigmoid*** as the activation function in our FNN 1.0, if the value is too small or too larger, the derivative of ***sigmoid*** function will be close to zero and causing vanishing gradient problem. Thus, ***ReLU*** activation function will be adopted in FNN 2.0. We aware that the initial stage of the weight might lead to different local optimal. Hence, we take that into account on hyperparameter tuning. The weights are initialized in normal distribution with zero mean and ***sd*** standard deviation, ***sd*** is added to hyperparameter set and sampled from a uniform

distribution from ***1e-5*** to ***1e-1***. Exponential Decay Learning scheme was adopted with learning rate ***lr*** and decay rate at 0.95, where ***lr*** is a hyperparameter and sampled from uniform distribution from ***1e-6*** to ***1e-3***. Figure 32 illustrates the network structure of FNN 2.0.

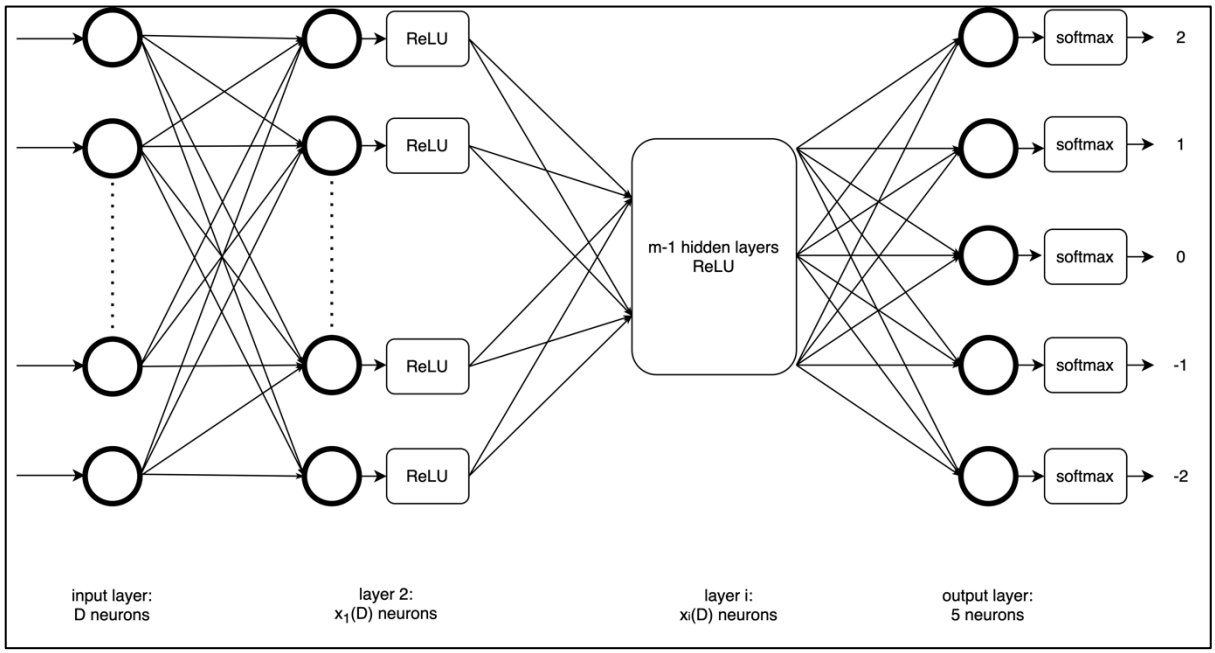

Figure 32. Network structure of the Feed Forward Neural Network 2.0

Regularization

Although the capability of the neural network models is very strong, overfitting happens easily in such model. In this project, three ways to prevent overfitting has tested. The first one is early stop [28], when the loss remains steady for 8 epochs, the training algorithm will stop immediately to prevent further reach to a local minimum. Figure 33 depicts the difference between adopting an early stop. It is obvious that without the early stop, the model overfit on the training data easily.

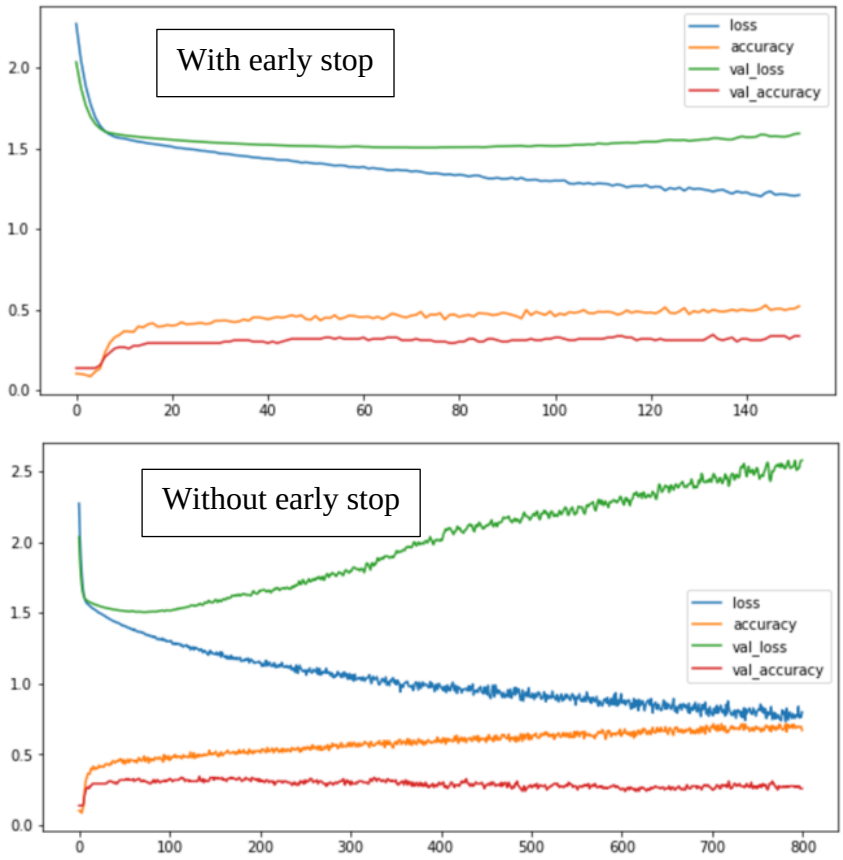

Figure 33. With early stop VS without early stop

The second approach is to use dropout layer [29], this approach mimics the behaviour of ensemble model by turning off some of the neurons randomly in each epoch. We can assign ***p*** (***p*** is 0.9 in our case) to units in each layer, where ***p*** is the probability of retaining neurons in that layer (Notice: ***rate*** in Keras is the probability of dropping out that neuron, which is the opposite from the original paper). In each training epoch, some neurons will be turned off randomly, so each of training process is like training a different neural network. The dropout layer is removed during the prediction process, meaning that all neuron is turn on. If one neuron is kept during the training process with ***p***, the output of that neuron will be scaled down by multiplying ***p*** during the testing process. This ensure the output of prediction is same as the

expected output when training (Because ***1-p*** of the neurons are down) [29]. Figure 34 shows the different between using dropout layer. One interesting finding is that the early stop is triggered when a dropout layer is used but not in the other case. Since we introduce a new way to do regularization and we want to keep our model as simple as possible, thus, dropout technique was discarded in project phrase 2.

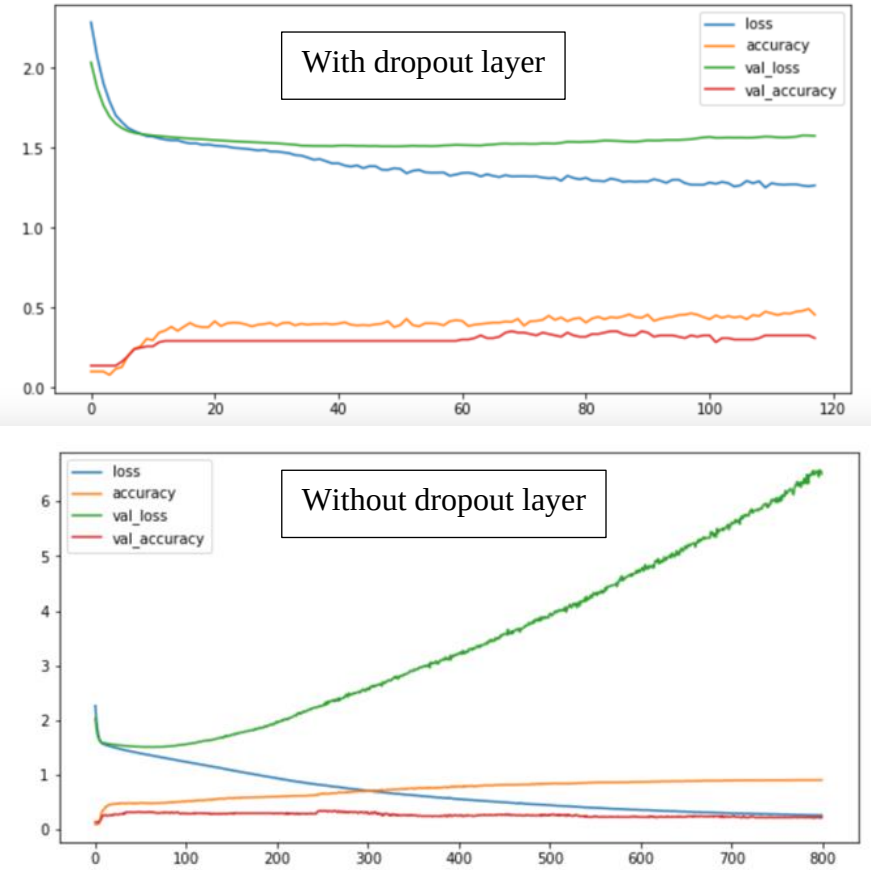

Figure 34. Using dropout vs without dropout

Due to the fact that ***ReLU*** was mainly used in project phrase 2, if all of the inputs to a neuron's ReLU are negative, all neuron's output will be zero and dead ***ReLU*** problem occurs. Batch Normalization [30] helps to tackle this problem by normalizing across each data in each mini-batch before entering activation function. Moreover, when all the mini-batch data are in a standard scale, it benefit the training step and produce a more stable weight and converge faster. Since for the one fixed data point, it can appear in different mini-batch with different data points randomly. Hence, the parameter of Batch Normalization (mean, standard derivation) will be different, it provides some regularization effect. Since there are disputes on whether Batch Normalization should be applied before or after activation function. In this project, we will follow the original paper approach, written by Sergey Ioffe and Christian Szegedy, to use Batch Normalization before activation function.

5.5.2 Long Short-term Memory (LSTM)

Long short-term memory is a special type of Recurrent Neural Network (RNN) which is a neural network that can handle time series data well. In a RNN model, the input hidden state $\boldsymbol{h_i}$ of a node $\boldsymbol{n_i}$ are the output from last node $\boldsymbol{n_{i-1}}$ and the new data $\boldsymbol{x_i}$. Relatively, the output data are not only the output $\boldsymbol{y_i}$, but also the output hidden state $\boldsymbol{h_{i+1}}$ which is one of the input data of node $\boldsymbol{n_{i+1}}$. Thus, this kind of neural networks structure helps RNN handle the time series a lot.

In RNN model, if the entire sequences of data are too large, the gradient will be vanished and exploded. Thus, LSTM is raised to relieve those problems [31]. The implementation of LSTM model is adding one more input and output cell state $\boldsymbol{c_i}$. In overview, the cell state $\boldsymbol{c_i}$ change slowly whereas the hidden state $\boldsymbol{h_i}$ changes quickly.

In the architecture for our LSTM, it consists of 4 hidden layers in total and the number of neurons in the hidden layers was set in a dynamic way too. In the input layer, the number of neurons is as same is the number of features. For the hidden layers, If the number of features is $\boldsymbol{N_f}$, the number of neurons is **round-up($\boldsymbol{N_f}$*(4 – i + 1)*0.2)** in the $\boldsymbol{i^{th}}$ hidden layer where 4 is the total number of hidden layers. In the output layer, the number of neurons is 5 which is the total number of classification classes for this project. For the activation function, according to the result of hyperparameter tuning, the best activation functions in all neurons are sigmoid function so the overview of the model is as below. Figure 35 depicts the architecture of our LSTM model.

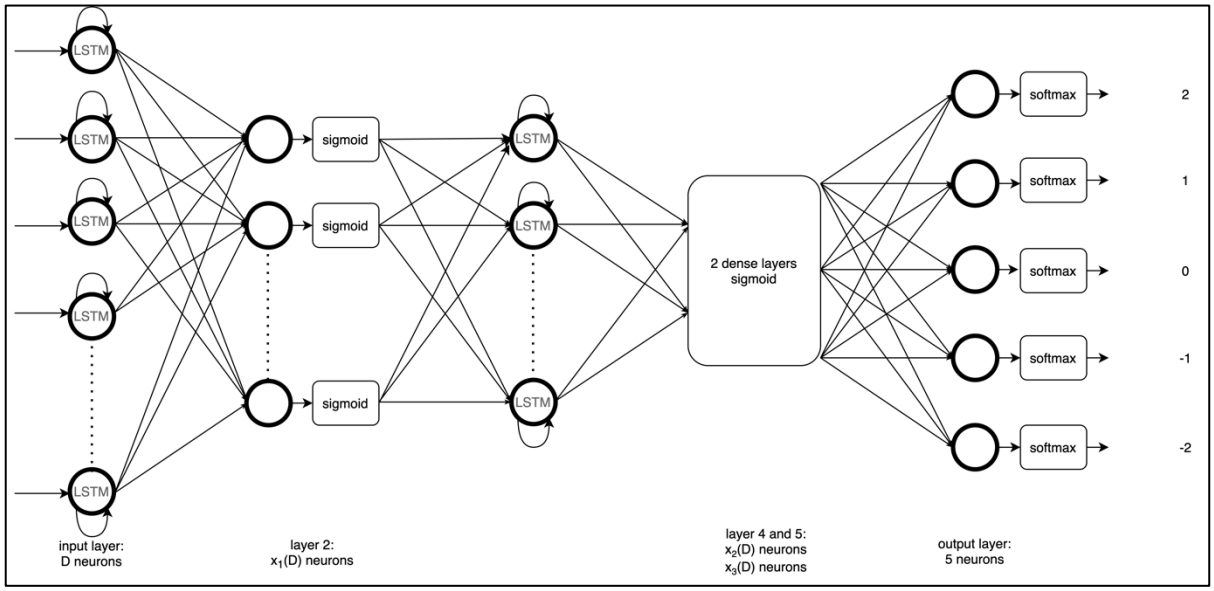

Figure 35. Structure of the LSTM Model version 1.0

Since the GAN dataset was used, hyperparameter tunning for the architecture of LSTM model was processing again. Also, we added different hidden layers in the tunning process. In

the validation processing, the profit of LSTM model improved more than 300% and shown in section 7.6.

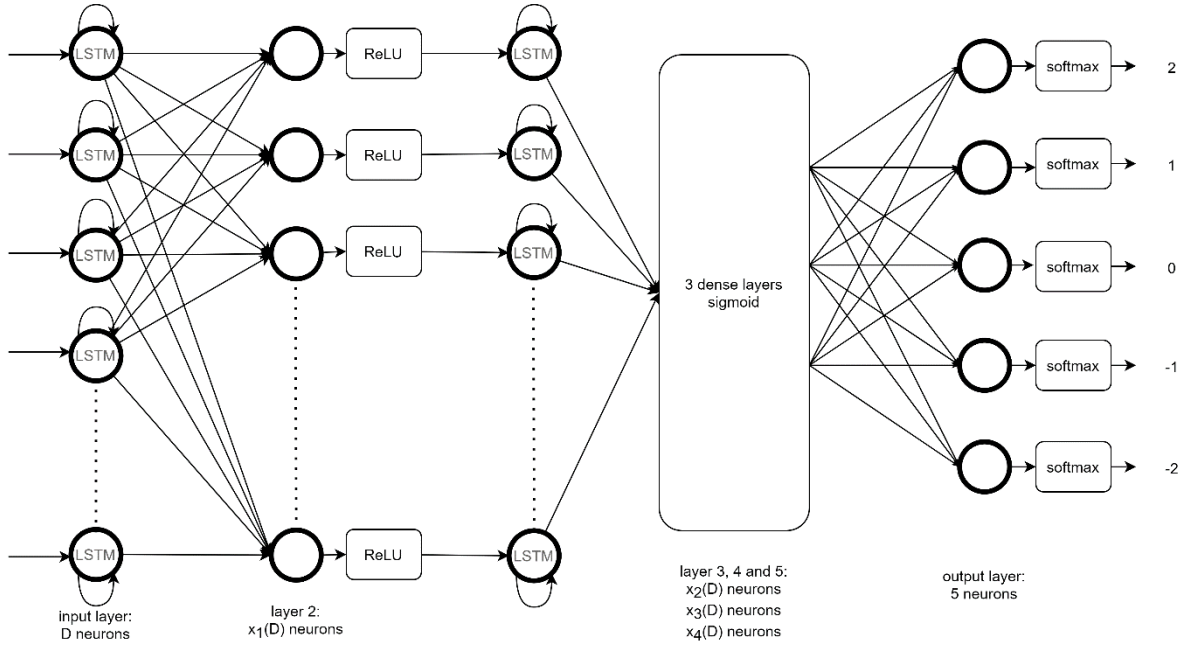

Figure 36. Structure of the LSTM Model version 2.0

5.5.3 Stacked Autoencoder with Supervised Fine Tuning (Sencoder)

In the Section 4.9.4, we tried to perform dimension reduction via Stacked Autoencoder. Another usage of Stack Autoencoder is to use the inner layers to do classification directly [32]. The structure used in this project is a Stacked Autoencoder with 4 hidden layers. In order words, 4 autoencoders were trained in a greedy layer-wise approach. A popular activation function, ReLU, is used in the hidden layer of all the 4 encoded layers. On the other hand, a linear activation function is used on the output layer because our dataset contains negative values which are less than -1. A linear activation function gave an unbounded output in this case.

Firstly, input the data into the first autoencoder and the weight of the hidden layer (AKA: hidden_layer_W_1) is obtained. Then, we use hidden_layer_1 as the input to the second autoencoder and we will obtain the hidden_layer_W_2. This process is repeated to the third and fourth autoencoder until we get the hidden_layer_W_4. Afterward, the four hidden_layer_W are connected and we embed an extra layer which contains 5 neurons at the end. At last, we will fine-tune this Stacked neural network using a supervised method. In this case, the handicap results will be the output layer using softmax activation function. A diagram which visualizes the whole process is given in Figure 37.

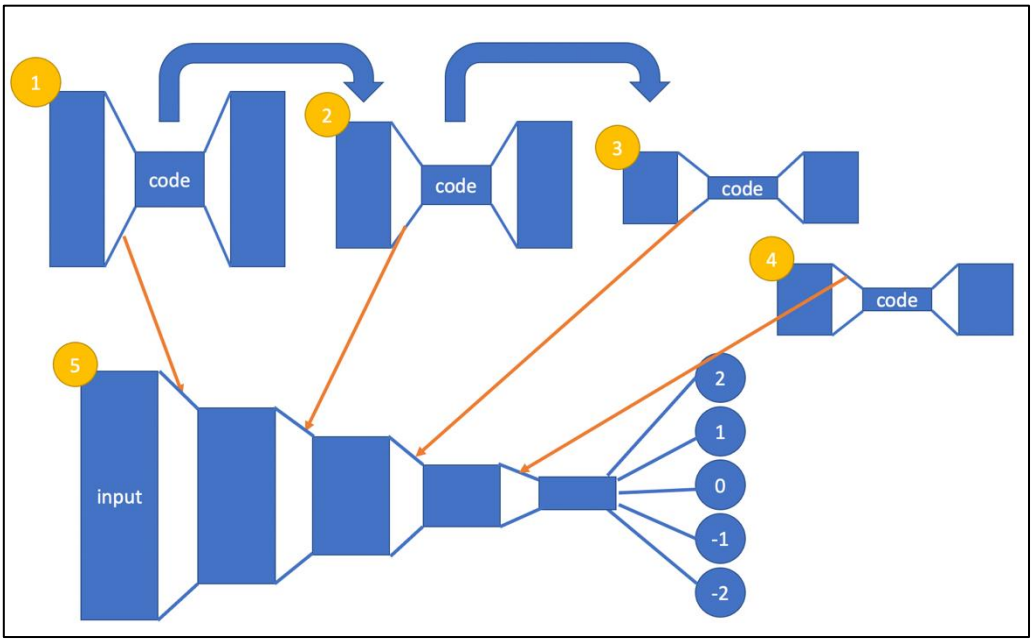

Figure 37. Visualization of Stacked Autoencoder with supervised method

Although the entire network structure of the Stacked Autoencoder (supervised) look very similar to our simple Feedforwards Neural Network, the way to train them makes it has a significant difference. The Feedforwards Neural Network is neural network which contains 4 hidden layers which does not necessarily have to be working on dimension reduction. In contrast, the second last layer in the Stacked Autoencoder (supervised) is actually a compressed representation of the input features, prediction will be done directly on these compressed features. We will only use this model when we are not using any dimension reduction on the pre-processing. It is because we do not want to do the dimension reduction twice as the error will easily be built up in this case.

We observe that the Stacked Autoencoder (supervised) was not performing very well in project phrase one. We believe one of the reasons is that it is preforming prediction immediately after the dimension reduction task which is not complex equal for the model to do prediction. In other words, in the fine-tuning state, the learning (gradient update) process is limited by the fact that there is only one layer for training. Hence, by adding more layers might tackle this problem. We decided to add three extra fully connected layers at the end (right after the encoding task) in order to give more computation power to the model. The last layer is still containing five neurons (representing five classes) with softmax loss. Moreover, we also adopted a different training strategy, inspired by Transfer Learning [33], for the Stacked Autoencoder (supervised). Firstly, the encoded part is trained in a greedy layer-wise approach

which is exactly the same as above (term 1). Afterwards, we freeze the weights of the encoded layers and adopt Exponential Decay Learning scheme to train the newly added layers. Exponential Decay Learning helps to reduce the learning rate from time to time exponentially, this helps to stabilize the gradient descent process from preventing bouncing between local optimal. In our case, the learning rate will be decreased by 5% for every 20 epochs. In addition, Early Stop was also used to prevent overfitting. After the network's weights being converged, we unfreeze all the encoded layers and perform an Adam gradient descent algorithm again on the entire network with a very low learning rate (1e-7 in our case). We do not want to destroy the functionality of dimension reduction at the encoded layers, and at the same time, we want the encoded part and the prediction part to cooperate with each other smoothly. Therefore, we choose a very low learning rate and train the network again. Figure 38 depicts the network structure of the project phrase 2 Stacked Autoencoder (supervised), we name it as Sencoder 2.0.

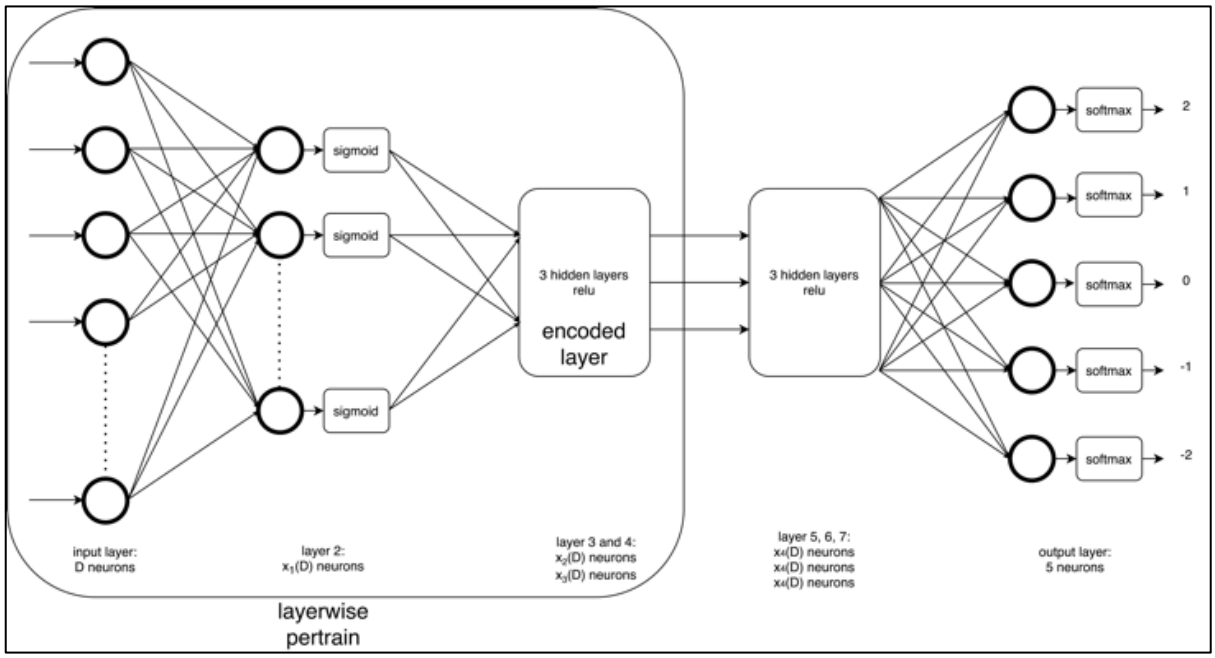

Figure 38. Network Structure of the newly Stacked Autoencoder (supervised)

5.5.4 Convolutional Neural Network

In Section 4.5, we tried to convert the recent past 10 records table in a numerical number and act as a feature of the dataset. In this section, we present a new way to represent these features (recent past 10 records table). In the encounter dataset (Section 4.3), we have three feature data which are Team A alone, Team B alone and TeamA_TeamB. An example of TeamA_TeamB feature is shown in Figure 39. We pre-process these features and treat the features mentioned above as pictures (Table). Since each feature (or picture) has the same dimension, we can further combine them into one picture with three different channels. In other words, three matrix with dimension ***(H, W)*** are combined to a 3D matrix ***(H,W,3)***.

Figure 39. Example of TeamA_TeamB past ten records

	主场	比分(半场)	客场	主	盘口	客	主	和	客	胜负	让球	大小
0	0	1.5	1	0.88	0.00	1.02	2.05	3.40	3.50	-1	-1	1
1	1	0.5	0	0.83	0.00	1.07	1.61	4.00	5.25	1	1	0
2	0	-1.0	1	1.03	0.00	0.87	2.30	3.40	3.00	1	1	1
3	0	-1.5	1	1.10	0.00	0.78	4.75	4.20	1.55	1	1	1
4	1	1.5	0	0.88	-0.50	1.02	1.83	3.50	4.33	1	1	1
5	0	-1.0	1	0.95	-2.25	0.90	12.00	8.00	1.12	1	-1	1
6	0	0.0	1	0.86	0.00	1.04	1.36	5.00	7.50	0	1	0
7	1	-1.0	0	1.11	0.00	0.79	2.25	3.40	3.10	-1	-1	1
8	0	0.0	0	0.81	0.00	1.09	3.00	3.20	2.37	0	-1	0
9	1	-0.5	0	1.00	0.00	0.90	2.15	3.40	3.30	-1	-1	0

Table 9. Example of processed TeamA_TeamB past ten records

After the pre-processing stage, we generate one picture for each data point. Convolutional Neural Network (CNN) [34] is used to process the picture-like data. CNN is a very popular deep learning method used to deal with pictures in both industry and academia. The main idea of CNN is to use different kind of filters to perform operation in a picture. In our case, we fit the picture into CNN and the remaining features into a fully-connection feedforward neural network, afterwards, we concatenate these two network structures and append a few fully connected layer and afterwards a softmax layer at the end to do the prediction. Zero-centre pre-processing is also applied on the picture-like data. In particular, we take the mean of all the picture data and subtract all the picture data from it. We did not perform normalization afterwards in order to follow the image pre-processing convention [35].

Figure 40 illustrates the network structure of CNN used in our project. Same as other neural network models mentioned in this project, hyperparameter tuning will be applied to the CNN model also. There are three Conv2D layers in our CNN network structure, each filters size are 3 by 3 with stride 1 and one dimensional zero padding was performed on the four sides, so to retain the same dimension in the output layers. The number of filters in each layer is represent as a list of number, $x_1 = [x_{11}, x_{12}, x_{13}]$. Meanwhile, there are six fully connected feedforward layers in the CNN model structure. The number of neurons in these layers is represented by using another list, $x_2 = [x_{21}, \ldots, x_{26}]$. These two lists are then passed to hyperparameter tunning process. The usage of these two lists is similar to the FNN model (Section 5.5.1). In our CNN architecture, there is a max pooling layer right after every Conv2D layer. The max pooling filters are configured as size 2 by 2 with stride 1, it does not perform any zero padding. All the activation function used in the hidden layers are ReLU, the reason is similar to our FNN model which is to prevent vanishing gradient problem.

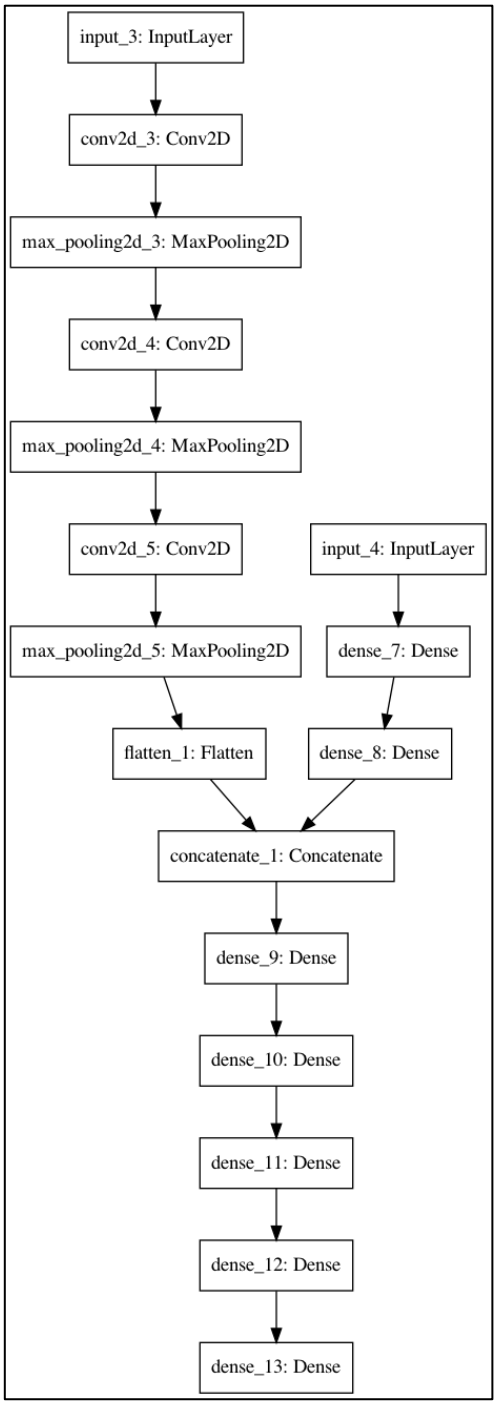

Figure 40. Network Structure of CNN

5.6 Cluster-then-Predict Model

Although the main focus of the project is choosing between a by-league model, an all-league model or even forming our best betting strategy and model, we also tried a way to define the "league" by ourselves. This idea is inspired by a paper written by Rishabh Soni and K. James Mathai [36]. In their paper, they cluster the twitter with similar words and model on it. The rationale of this is to from clusters which are similar and this might help during the modelling stage. In our case, we first combine all the league and obtain one dataset. Afterward, a K-mean clustering is performed in this dataset and the aim is to separate them by our "league". Eventually, put these clustered datasets into our modelling pipeline.

In this project, Elbow Method [37] was used to determine the best number of cluster ***k***. Elbow Method utilize the distortion metric to calculate a score for different value of ***k*** and select the best ***k***. Mathematically, $distortion = \sum_i (X_i{}^2 - centre_{X_i}{}^2)$ where ***X*** is the data point and $\boldsymbol{centre_X}$ is the cluster's centre of ***X***. Obvious when k increase, distortion decrease. According to Figure 41, the "elbow" occurred when ***k*** equal to 9 in our dataset. Hence, nine cluster will be used for K-means algorithm.

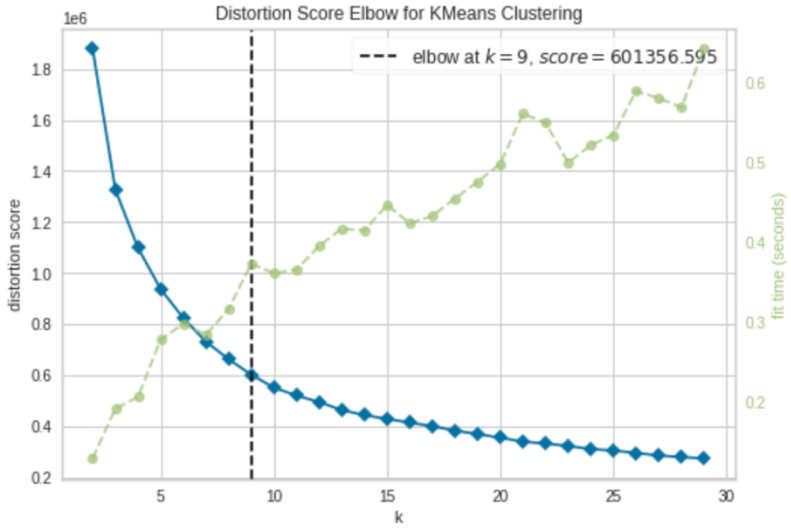

Figure 41. Elbow Method plot

5.7 Ensemble (Aggregating)

During our experiments, we found that different models yield different results under the same setting. The results are not consistent in on each league (in by-league and all-league model), this is harmful when we are choosing the team to bet since it might not be obvious that which model is the robust and reliable one. Although some models might seem promising when we are evaluating them on our testing data, there is no way to guarantee the good results will still occur in the future unknown data. Therefore, if one wants to be conservative on betting, one might not want to just rely on one or two models. One method to tackle this problem and have a stable prediction is to adopted ensemble learning, we used stacking approach in our case. Ensemble (aggregating) is a technique which combine the results from different models and generate one results. Mathematically, ensemble can be expressed as $y = f(X)$, where y is a scalar and X is a vector which contains different predictions from different models. In this project, we also tried bootstrap aggregating (bagging) in Section 4.10.1. The main difference is that aggregating takes multiple predictors under the same dataset whereas bagging takes the same predictor in different dataset.

There are two approaches for doing ensemble in this project. The first approach is ***mean***, it treats each model prediction fairly. In our case, the mathematical formula is $y = \frac{1}{N}\sum_i x_i$ where $\boldsymbol{x_i}$ is any integer range from -2 to 2. Since we will bet on home team whenever y is larger than 0. ***mean*** has the same effect as ***majority vote*** and it is better to represent in numerical values.

The second approach is ***weighted majority vote***, it is a more aggressive approach compare to ***mean***, because under ***weighted majority vote***, one model might have a high weight and dominate the ensemble result .The mathematical formula is $y = \frac{1}{N}\sum_i x_i * w_i$ where $\boldsymbol{w_i}$ is the weight for each prediction, $\boldsymbol{w_i}$ is the normalized cross-validation score (higher is better) in our case. The second approach is created mainly because we observed some models might perform very badly so we want to decrease their contribution in the ensemble decision.

The experiment results will be discussed in Section 7.2.

6. Evaluation, Phrase 1

In order to present the entire project results and workflow in a more clear way, we will divide the evaluation part for term 1 and term 2 in two separate sections. The metrics (Section 5.1) and evaluation methodology (Introduction in this Section) are same in both terms. In this section, we will discuss about term 1 procedure and all the corresponding experiment results. Section 7 will mainly focus on term 2 results.

We separate the training and testing in chronological order, meaning the latest 20% of the matches are those in the testing set. Splitting in this way can prevent data leakage as we are not using the future data to predict the past data.

The first goal to achieve in the entire project is to excess the expected value benchmark because it means the models are doing better than the bookmakers. Usually odds-based benchmark model performs better than the expected value; therefore, the second aim is to perform better than the odds-based. Last, we want the model to have a positive return at the end. Ideally, we want to obtain a plot that the curves have an upward trend (earning money in each bet). Notice during the evaluation, three curves will be the main focus, the light-yellow one is the FNN, the light green one is the LSTM and the dark green one is the odds-based benchmark model. At the end of the entire project, a real-time prediction model will be built base on the performance of each model in the testing data (Section 8).

The goal in project phrase 1 (term 1) is to formulate the project pipeline. Thus, we will first choose the best way to do the prediction among three methods, which are Cluster-Then-Predict, all-league and by-league. Second, we will try various pre-processing techniques on it to finalize our best model. At last, some feature selection methods will be applied to see if it can bring an uplift to our best model. All the decision we made in Section 6 will be used as the default project pipeline and carry to project phrase 2.

When we are doing the evaluation. All the statistical models, neural network models and 2 of our benchmark models will be used. We decided not to include the naïve based benchmark model because we found that its performance was very bad, even worser than the expected value. Hence, in order to plot the graphs in a clearer way, it will not be included.

6.1 Cluster-then-Predict Model, Phrase 1

In this part, we will evaluate the results and comment on the Cluster-then-Predict Model. By using the elbow method, ***k*** was decided to be 9. The data size of each cluster is as follows. The number of records in each cluster is distributed quite evenly (around 700), except for the first and second clusters. It contains varying league within each cluster, there is no special pattern on which league will belong to which cluster.

Cluster	Number of Records
1	1373
2	314
3	952
4	464
5	830
6	594
7	729
8	872
9	607

Table 10. Dataset for each cluster

We will evaluate each cluster's performance by passing the entire dataset into our modelling pipeline. Overall, the neural network models were not performing exceptionally in each cluster. Perhaps it is because using a linear approach (K-means) to form the clusters does not necessarily help when we model in a non-linear way. Most of the models perform better than the expected return curve which is good news for us.

The best models for the 9 clusters are odds-based benchmark, FNN, FNN, Random Forest, Random Forest, LSTM, KNN, FNN and KNN respectively. We are surprised to see a benchmark model outperformed all the others in the first cluster. Perhaps the k-means can really help forming meaningful cluster. The main purpose of creating such cluster-then-predict model is to find out which model perform the best in each cluster. In the second semester, we will keep monitoring this behaviour by adding the data from Aug-2020 to Dec-2020 (Section 7.1). We want to see if the best models in each cluster now the best in the new dataset are still or not. If that is the case, we might be able to find a way to obtain a profitable return. For example, the odds-based benchmark was the best model in the first cluster (Figure 42), we suspect that k-means cluster these matches in one group. In that case, we can achieve a profitable return by using odds-based benchmark model in the future.

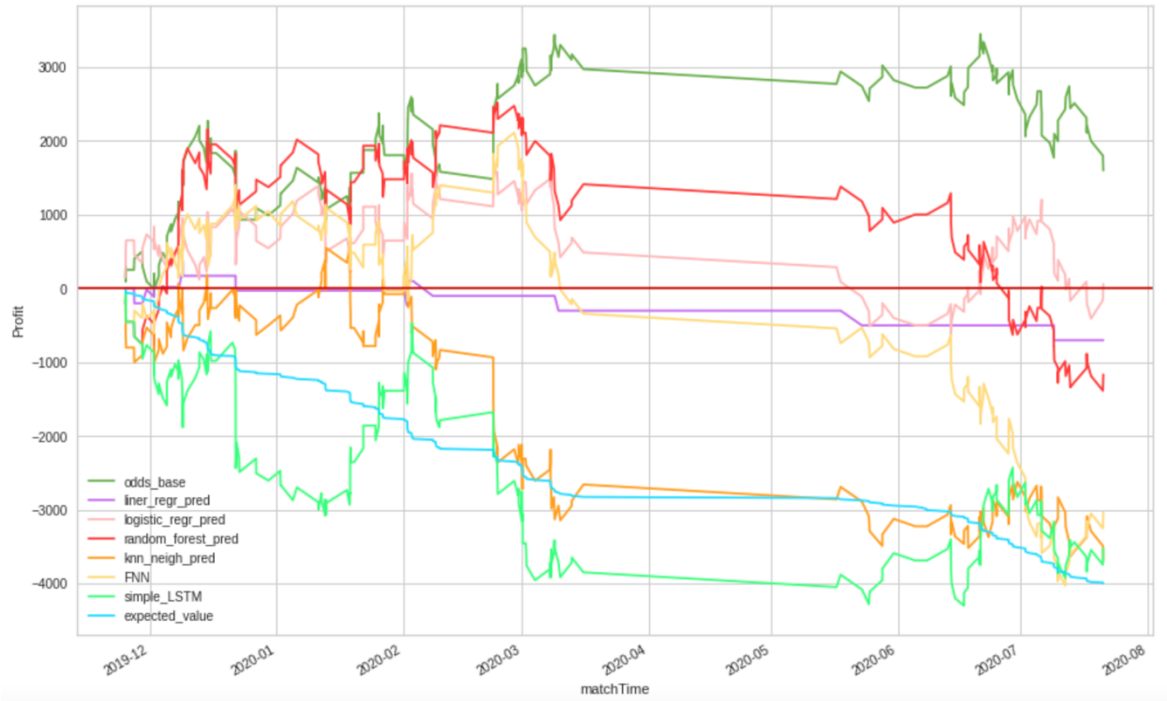

Figure 42. K-means cluster 1

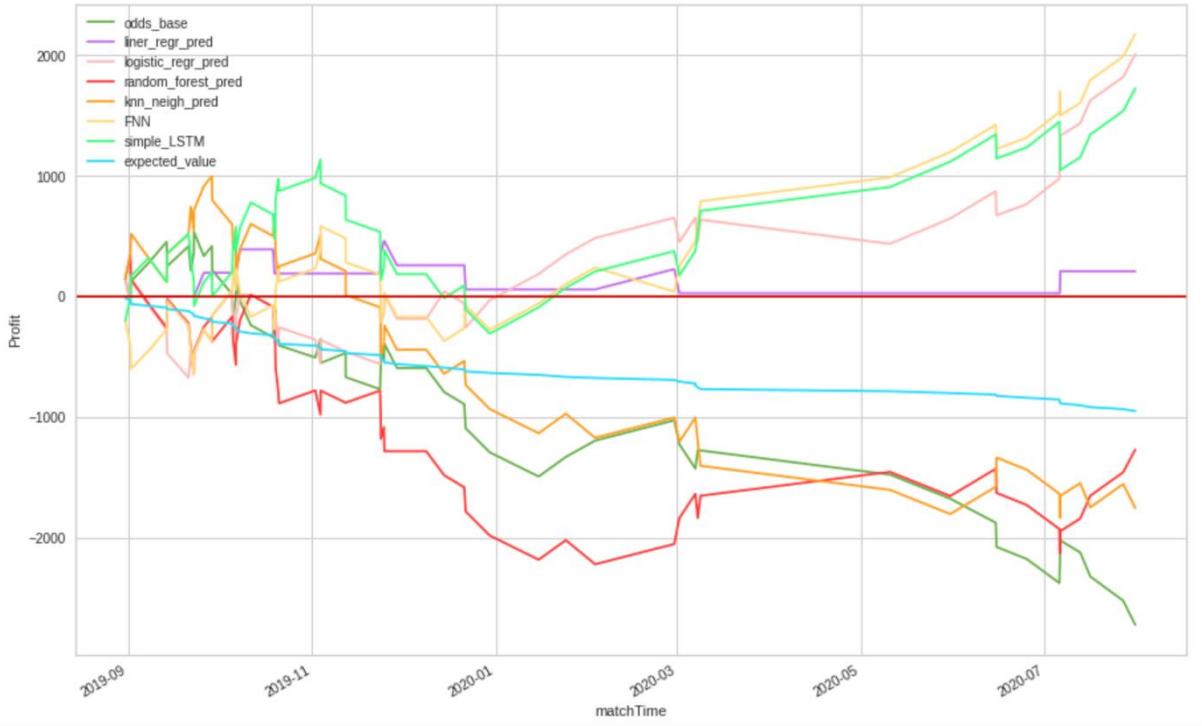

Figure 43. K-means cluster 2

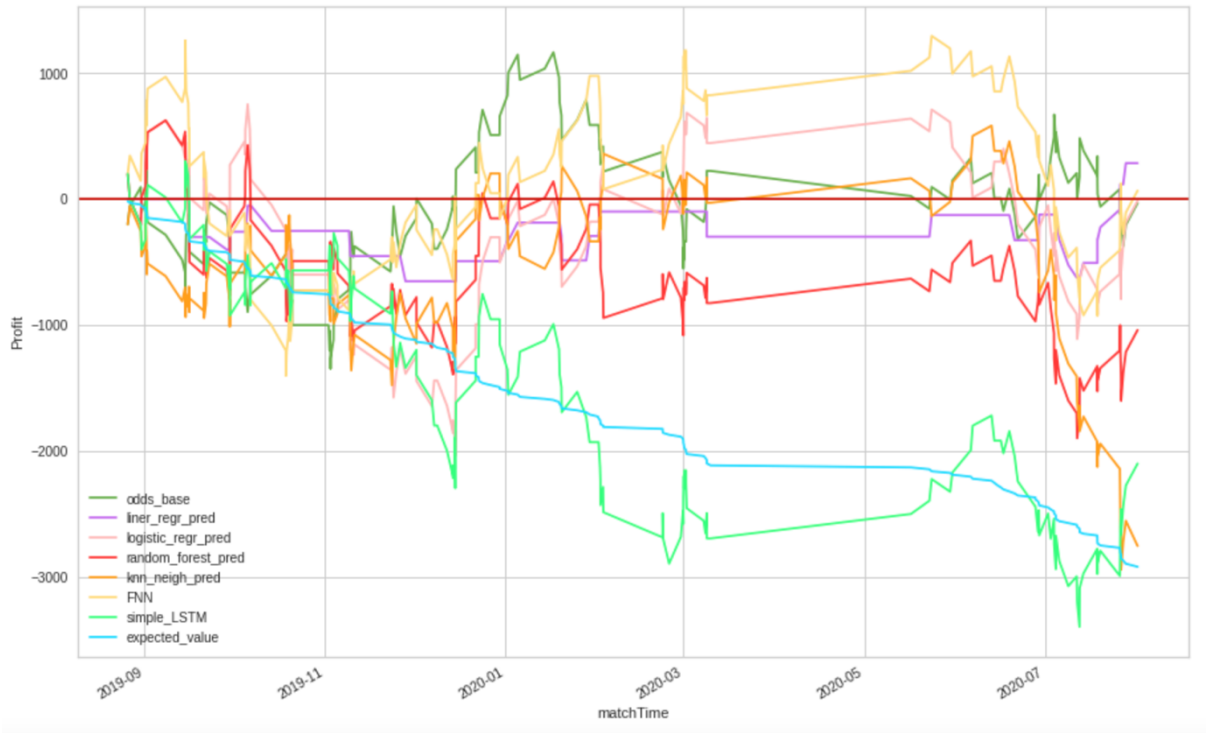

Figure 44. K-means cluster 3

Figure 45. K-means cluster 4

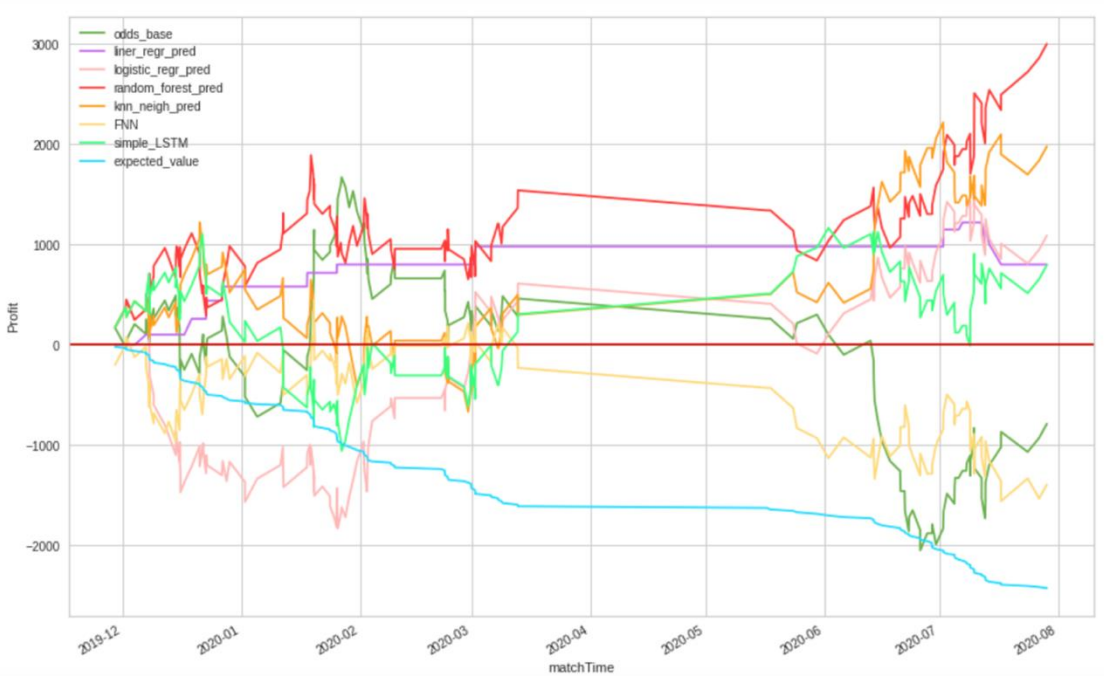

Figure 46. K-means cluster 5

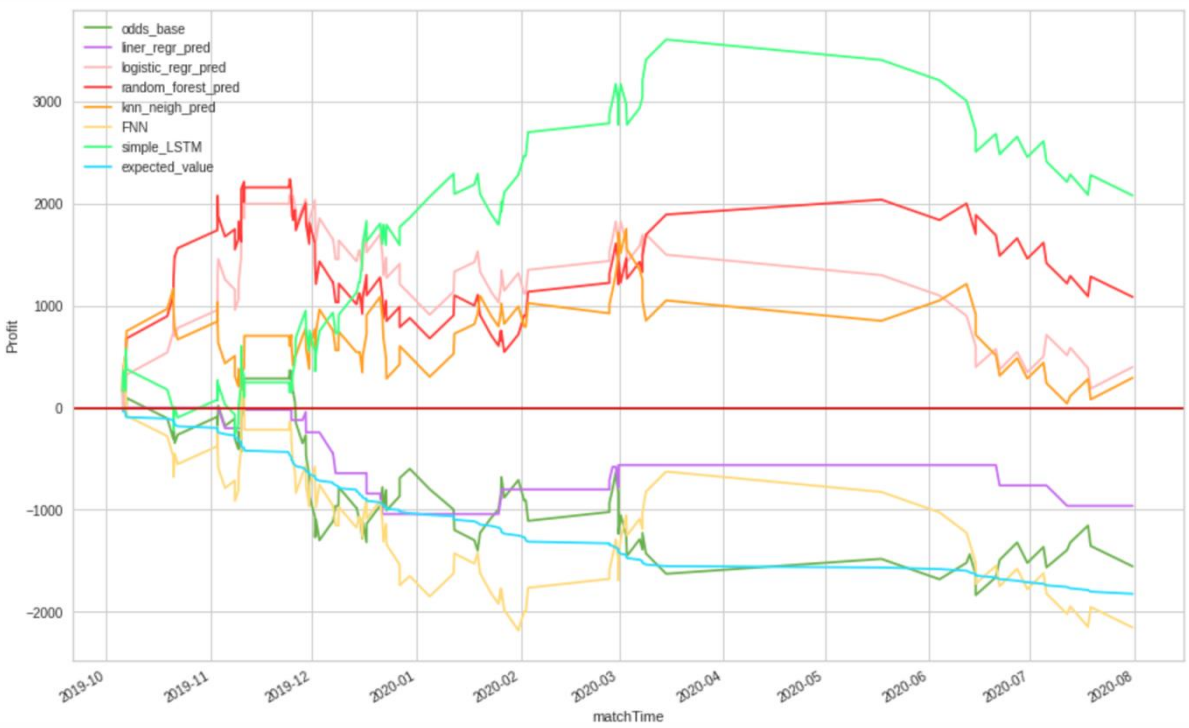

Figure 47. K-means cluster 6

Figure 48. K-means cluster 7

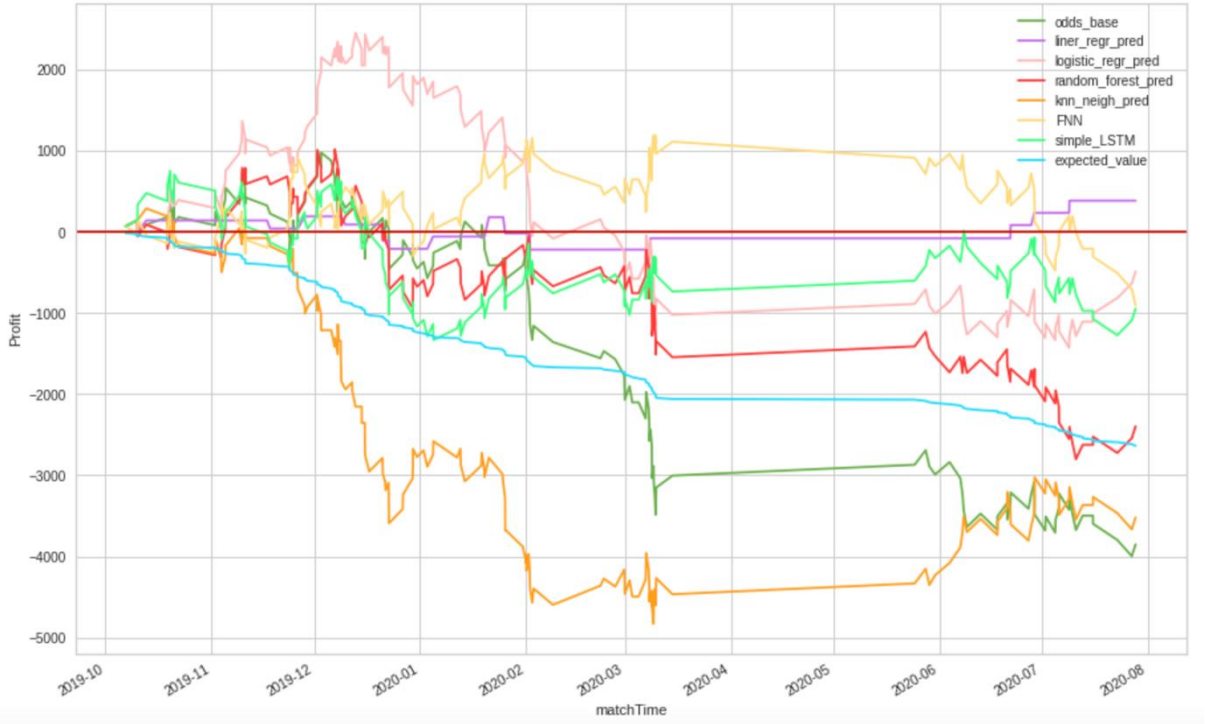

Figure 49. K-means cluster 8

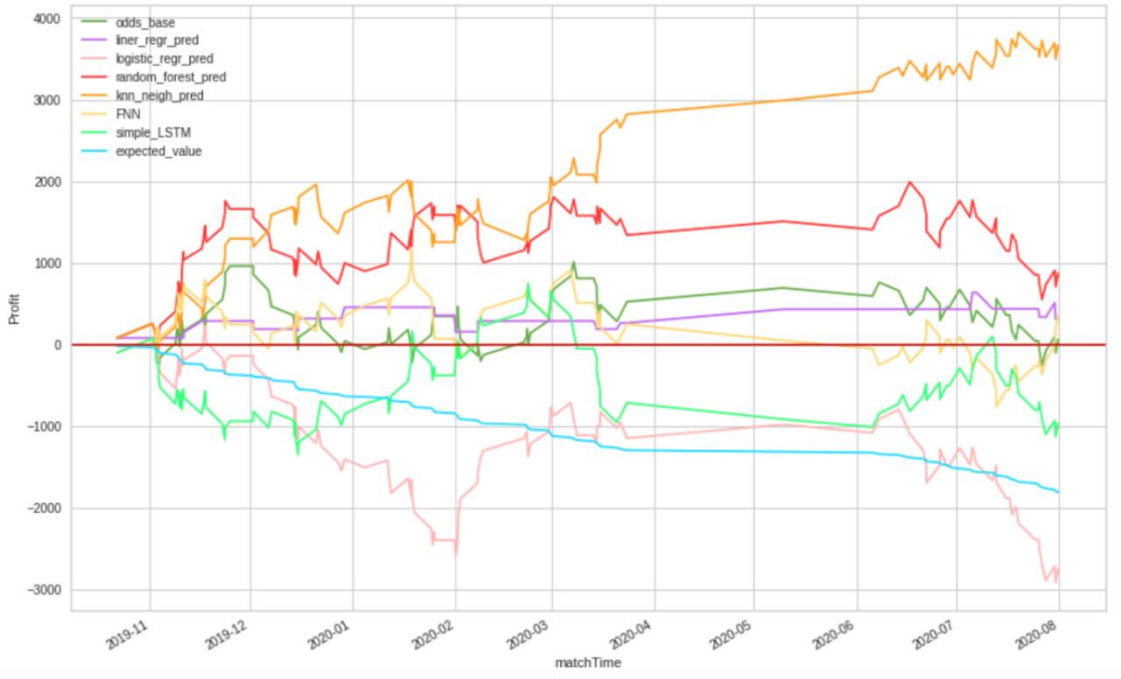

Figure 50. K-means cluster 9

Besides looking from cluster-level, we would also want to know how is the overall performance be so that it is easier for us to compare with the other two approaches. Figure 51 depicts the results when we combine all the predictions in each cluster into one and plot the profit accordingly. Overall, all of the models perform better than our two selected benchmark models. Three of the statistical models actually gave a positive return at the end. However, the LSTM performed badly in this section.

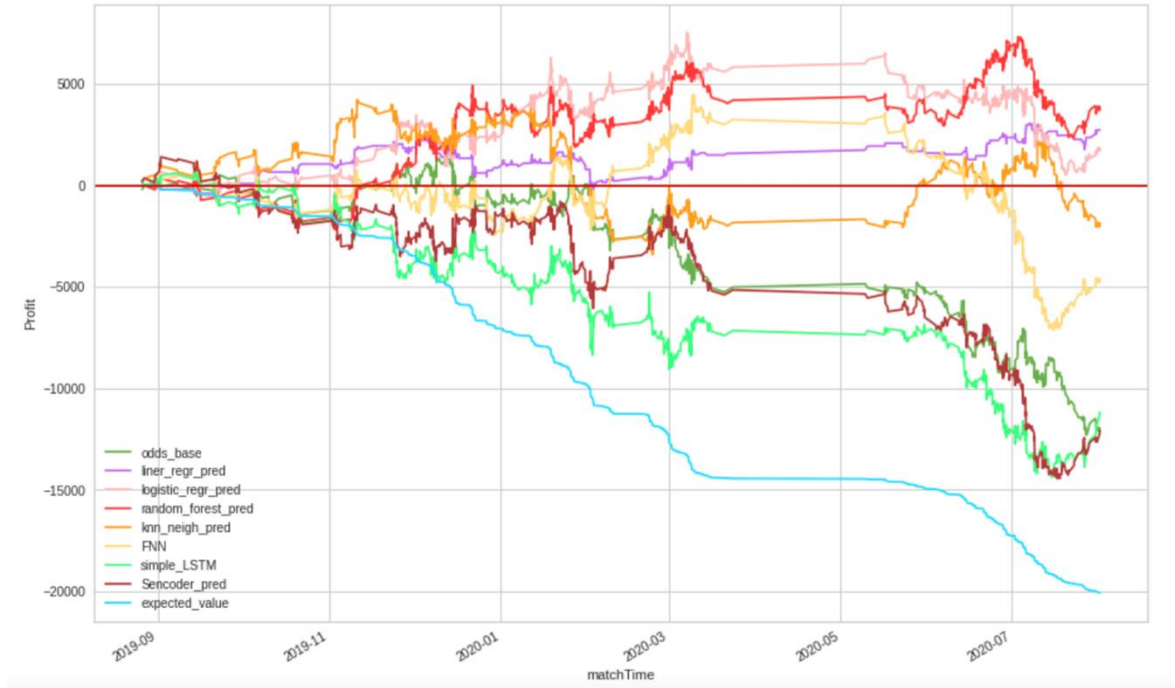

Figure 51. Overall k-means profit

6.2 By-league Model

Besides using k-means to separate them, we would like to separate the dataset by league and model on each of them. We believe the bookmakers will have a different standard on calculating the odds for different leagues. Therefore, separating them should help in the modelling procedure. We will train on the league which has more than 200 records because if a league has too little data, it might affect the performance. Figure 52 illustrates the selected league and the number of data point in each league. Eighty percent of data in each league were trained and the remaining were testing, Figure 53 depicts the combined testing performance.

```
{'挪超': 215,
 '日職聯': 232,
 '德乙': 472,
 '美職業': 253,
 '法甲': 383,
 '英冠': 463,
 '歐冠盃': 214,
 '西甲': 691,
 '德甲': 520,
 '英超': 738,
 '荷甲': 250,
 '意甲': 594,
 '澳洲甲': 252}
```

Figure 52. League to train for by-league and all-league Model

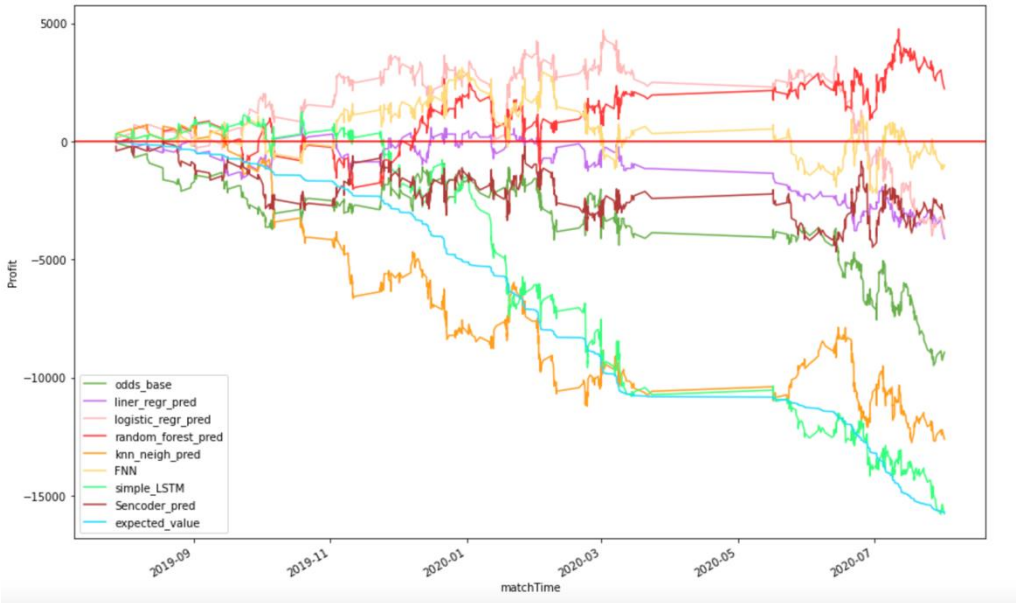

Figure 53. By-league Model Results

6.3 All-league Model

All-league is the last candidate of our best model. It uses all the data to train regardless of whether that data point is belonging to which league. Since the odds from different league might probably come from different distribution, putting them into one variable might bring difficulties to our models. Especially on those leagues which has only a few numbers of matches. They might decrease the consistency of the dataset. Therefore, when we train this model, we only select the league that has more than 200 records (Figure 52). In this model, 80% of combined and selected were trained and the remaining were tested. Figure 54 shows the all-league performance.

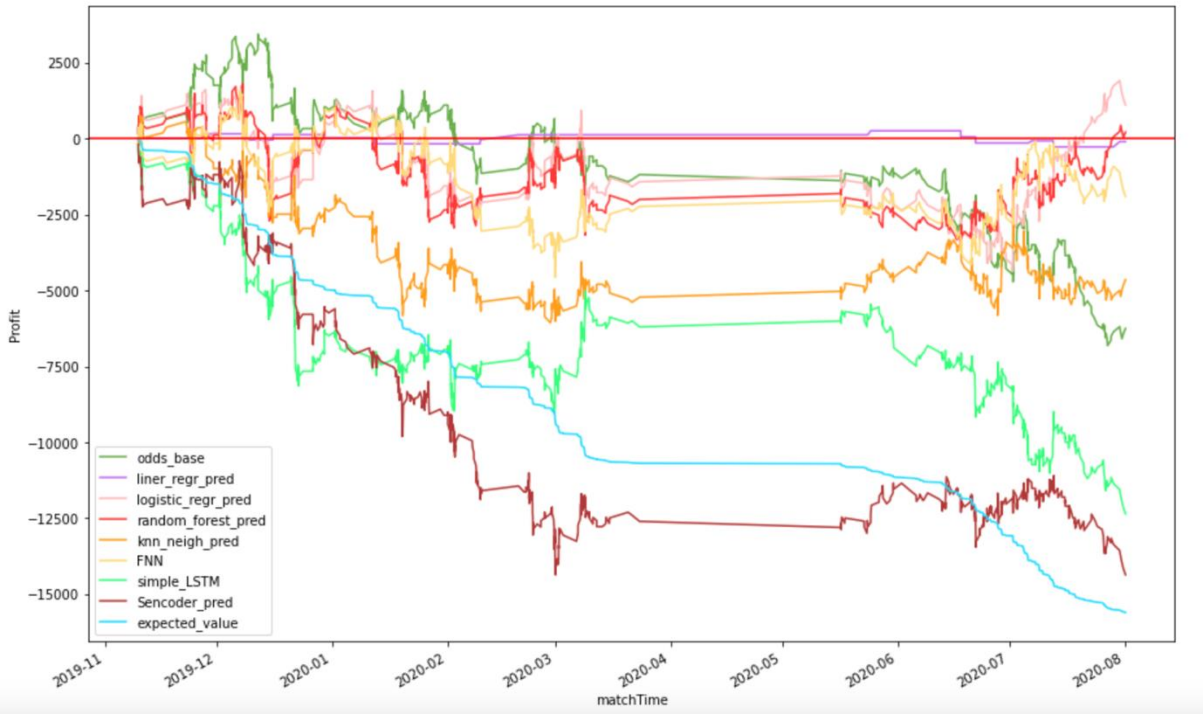

Figure 54. All-league Model Results

6.4 Best Model

We have already obtained the results for 3 approaches and would like to select one of them to perform other experiments or improvements on it. Overall, LSTM has the worst performance in all 3 candidates whereas the other neural network. It is strange that LSTM model got a deficient performance in all the arrangements of leagues. Thus, here is a hypothesis that since not all features are time-series data, the performance of LSTM will be influenced by the non-time-series data a lot and an evaluation will be done with it later (Section 6.7). The results also depict the difference between FNN and supervised stacked autoencoder (Sencoder). Although, their architectures were very similar, FNN clearly outperform Sencoder in every scenario. In addition, the performance of Sencoder is as worse as LSTM. We suspected that it is because prediction was done immediately after the encoded layer whereas FNN has more layers so it had more capability. In project phrase two, we will try to increase the number of layers for Sencoder. We are expecting a performance similar to FNN. On the other hand, most of the models in Cluster-Then-Predict Model and by-league model outperform the odds-based benchmark model.

Table 11 depicts the overall profit gain on each candidate. At this stage, the Sencoder will not be the main factor on selecting the best model because it has some strange behaviour in the individual. For example, it has a same profit curve (predictions are exactly the same) no matter which league we are predicting. Hence, the other two neural network models will be focused.

Although the all-league does not have the best performance in statistical model, it is still better than the performance of by-league significantly. Since the objective in term 2 is neural network model and optimization, so the all-league model is the most suitable one for term 2 tasks and it will be the main focus in project phrase 2.

	Expected Value	Odds based	Linear Regression	Logistic Regression	Random forest	KNN	FNN	LSTM	Sencoder
Cluster-then-Predict	-20000	-12000	2500	1500	3750	-2000	-4950	-11000	-12000
by-league	-16000	-9000	-4250	-4250	2250	-12500	-750	-16000	-3000
all-league	-16000	-6250	-50	1250	100	-4500	-2000	-12500	-14000

Table 11. Comparison of 3 approaches

6.5 Dimension reduction

In this section, PCA and 3 types of autoencoders, and their performance will be discussed. The input feature for this section are as follows.

```
Index(['home_away_label', 'hkjc_hdc_home_first', 'hkjc_hdc_away_first',
       'hkjc_hdc_home_last', 'hkjc_hdc_away_last', 'bet365_hdc_hkjcFirst_home',
       'bet365_hdc_hkjcFirst_away', 'crown_hdc_hkjcFirst_home',
       'crown_hdc_hkjcFirst_away', 'macau_hdc_hkjcFirst_home',
       'macau_hdc_hkjcFirst_away', 'bet365_hdc_hkjcLast_home',
       'bet365_hdc_hkjcLast_away', 'crown_hdc_hkjcLast_home',
       'crown_hdc_hkjcLast_away', 'macau_hdc_hkjcLast_home',
       'macau_hdc_hkjcLast_away', '總進球_away', '總進球_home', '總失球_away',
       '總失球_home', '淨勝球_away', '淨勝球_home', '場均進球_away', '場均進球_home', '勝率_away',
       '勝率_home', '平率_away', '平率_home', '負率_away', '負率_home', '同主客進_away',
       '同主客進_home', '同主客失_away', '同主客失_home', '同主客淨勝_away', '同主客淨勝_home',
       '同主客均進_away', '同主客均進_home', '同主客勝_away', '同主客勝_home', '同主客平_away',
       '同主客平_home', '同主客負_away', '同主客負_home', 'fifa_team_home_score_ATT',
       'fifa_team_home_score_MID', 'fifa_team_home_score_DEF',
       'fifa_team_home_score_能力', 'fifa_team_home_score_球隊評分',
       'fifa_team_away_score_ATT', 'fifa_team_away_score_MID',
       'fifa_team_away_score_DEF', 'fifa_team_away_score_能力',
       'fifa_team_away_score_球隊評分', 'home_player_power_mean',
       'home_player_hidden_power_mean', 'away_player_power_mean',
       'away_player_hidden_power_mean'],
      dtype='object')
```

Figure 55. Inputted features for Section 6.5

First, the main focus will be on the single layer Autoencoder. We evaluate the goodness of the autoencoder by whether it is able to reconstruct the input dataset rather than the final profit gain. It is because the original purpose for building the autoencoder is to do dimension reduction. Mean Square Error was used for the evaluation metric, its mathematical formula is $MSE = \frac{1}{N}\sum_i (y - y_{pred})^2$. In other words, we will measure the MSE between the original input and reconstructed input, we then choose the best dimension. One intuition is that the more neurons in the hidden layer, the better reconstruction ability it has. It is because containing more units means the neural network is able to perform more complex computation. Hence, we will not barely select the percentage which gave the lowest MSE. Figure 56 depicts the MSE metric for each of the dimensions in Autoencoder with one layer. Since Autoencoder is a neural network model which involved random initialization of weights so the plot is generated by running 10 times. The values in the x-axis mean the percentage; for example, if we have 100 features and 0.7 means the hidden layer contains 70 neurons. As we expected, 90% results the lowest MSE score. However, we choose 0.7 in this case because overall we want to select the smallest dimension but still be able to reconstruct most of the inputted data.

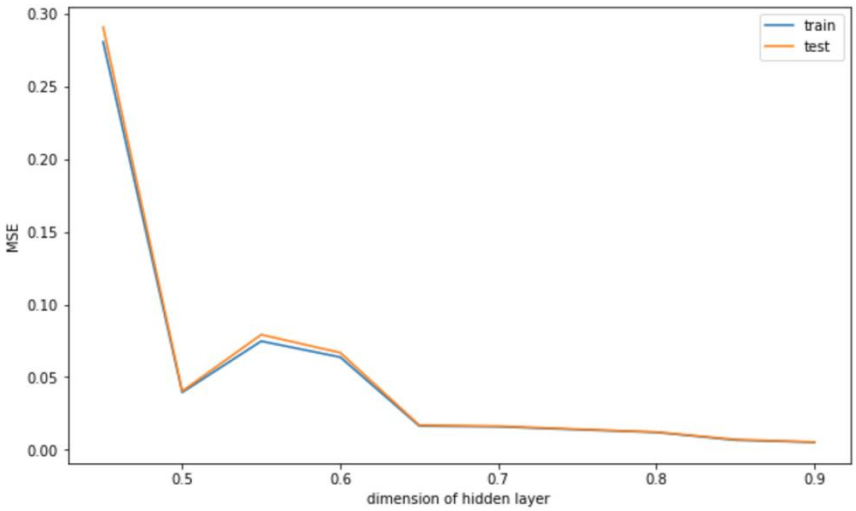

Figure 56. Autoencoder MSE

Moreover, we evaluated the two approaches on our 2-layer autoencoder, the greedy layer wise pretrain and the deep learning. Once again, the plots are generated by running under the same configuration 10 times.

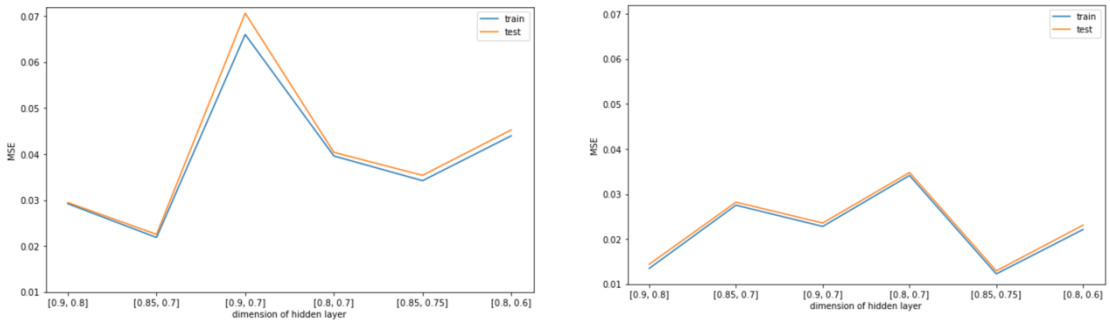

Figure 57. MSE on Deep Autoencoder and Stacked Autoencoder

According to Figure 57, we can see that overall Stacked Autoencoder performs slightly better than Deep Autoencoder. The x-axis is a list of two values each represent the dimension of encoded layers. For example, when the input dataset contains 100 features, [0.9, 0.8] means the first and second encoded layers contain 90 and 80 features respectively. In other words, the structure of the network is 100->90->80->90->100. On the same set of hyperparameters, the MSE of Stack Autoencoder fluctuates between 0.035 to 0.015 whereas Deep Autoencoder has a larger fluctuation and the set [0.9, 0.7] performs the worst. It is because during training the layer wise, the first layer contains more information than the one in the deep learning approach. The deep learning approach needs to find the optimal weights for all of the layers which might limit the optimization process. Overall, the set [0.85, 0.75] will be selected for both Autoencoders. However, we will see that in the later section during the evaluation, deep

Autoencoder actually performs better in term of profit metric. Perhaps it is because a deep learning approach to encode the input data is more suitable for the deep learning models.

Here, we will discuss about the performance of PCA in the 3 selected league (La Liga, Premier League and EFL Championship) and all-league. Since we are applying POV on PCA, we would like to look have a look at how many variables are being selected under each POV.

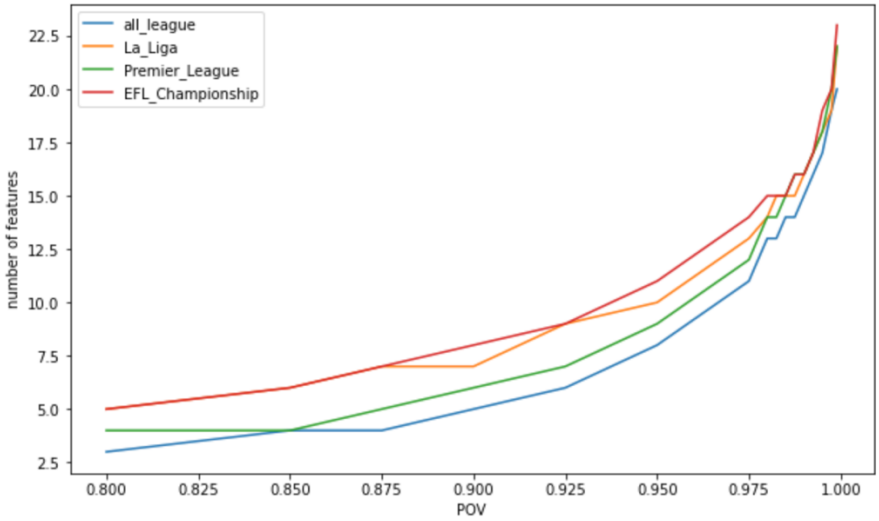

Figure 58. Number of features selected, PCA_POV

Figure 58 shows the number of features selected in each league from 0.8 to 0.999 POV. It is clear that the all-league dataset has the least features set on all POV. We suspected that it is because it contains the largest dataset and therefore PCA has enough information to separate the data. Indeed, we checked the number of factors selected by PCA during training (80% of data) and the number tends to increase in all of the league.

One interesting finding is that when we try POV equal to 1. The average number of features selected in the 4 type of leagues is 53 whereas the inputted dimension is 59. Indeed, when we check for the explained variance for the 4 leagues, on average around 6 of the variables gave an explained variance which is close to zero. In other words, 6 of the features in the dataset are closely correlated. Hence, they are dropped by PCA.

We will now perform PCA with different POV (mainly focus on POV equal to 0.999 and 0.95) on modelling to have a look at their performance. 4 statistical models, FNN and benchmark models will be used as the evaluation. We will use the all-league approach as the base because it will be our main focus in semester 2.

Figure 59. PCA with POV=0.999

Figure 60. PCA with POV=0.95

Figure 59 and Figure 60 illustrate the difference in performance on POV=0.999 and on POV=0.95. Overall, 0.999POV performs better than 0.95POV. It is because 0.999POV has more features, which means it contains more information. Although, there are twice more

features retained in 0.999POV than 0.95POV, since the extra features have a low explained variance so 0.999POV did not bring a drastic improvement compare to 0.95POV. Our models did perform better by applying PCA (especially on 0.999POV) because PCA helps to generate a set of uncorrelated features which are better for modelling. Notice: we can rarely see the linear regression curve because linear regression gave up doing the prediction and results in a flat line most of the time.

Next, we will come back to the two multi-layer autoencoders performance. Statistical models, two neural network models and two benchmark models will be used on evaluation. The deep autoencoder approach had a slightly better overall performance. Three models were below our odds-based benchmark model in the stacked autoencoder approach. If we compare autoencoder results with Figure 54; indeed, it did give a slightly better result as more models have a return that higher than the odds-based benchmark model across the whole time period. Surprisingly, neural network models gave completely different performance results. FNN performed good in stacked autoencoder but very bad in deep autoencoder and vice versa. For statistical models, their performance is similar except KNN. It proved both approaches can really use fewer dimensions to represent the original dataset. Although we found that stacked autoencoder should be able to reconstruct better than deep autoencoder, the profit metric shows deep autoencoder gave a better result. In fact, the difference in reconstruct ability was very small, there is just around 1% difference on average.

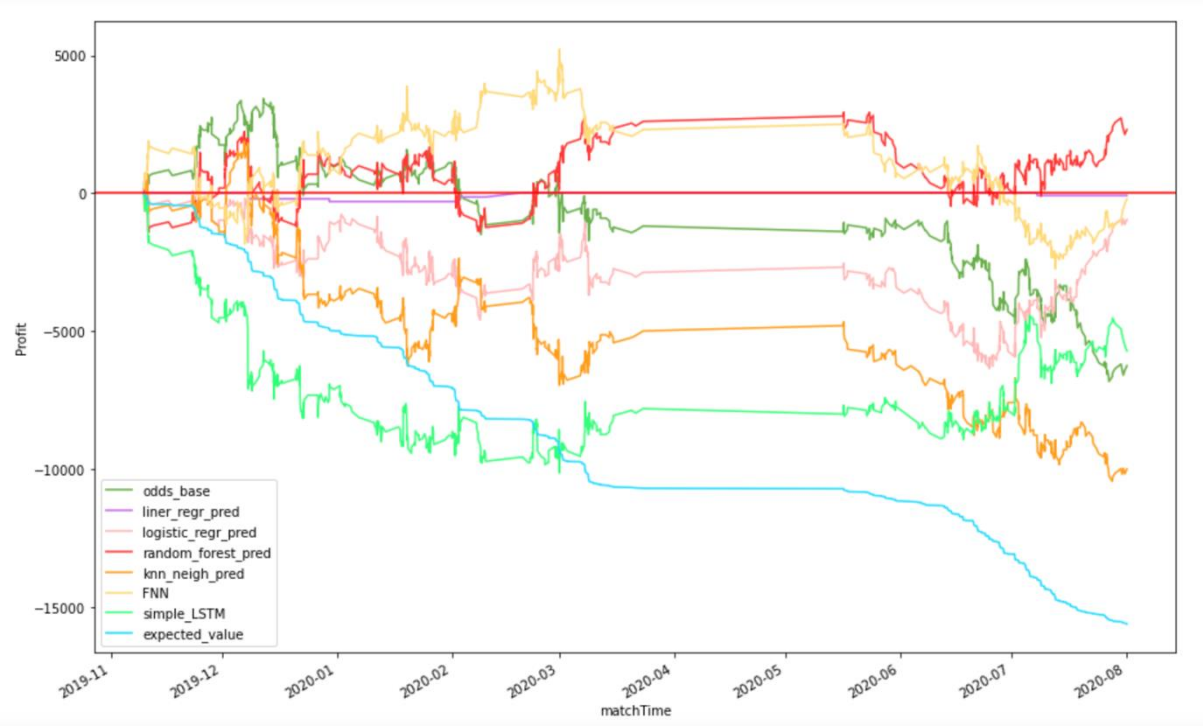

Figure 61. Stacked Autoencoder Result

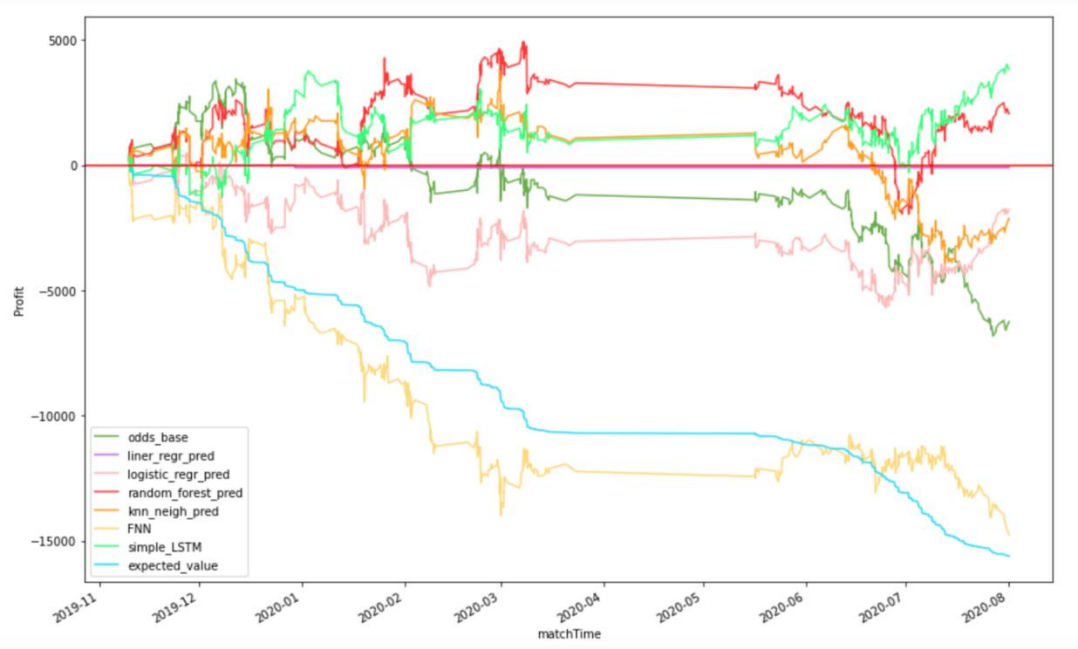

Figure 62. Deep Autoencoder Results

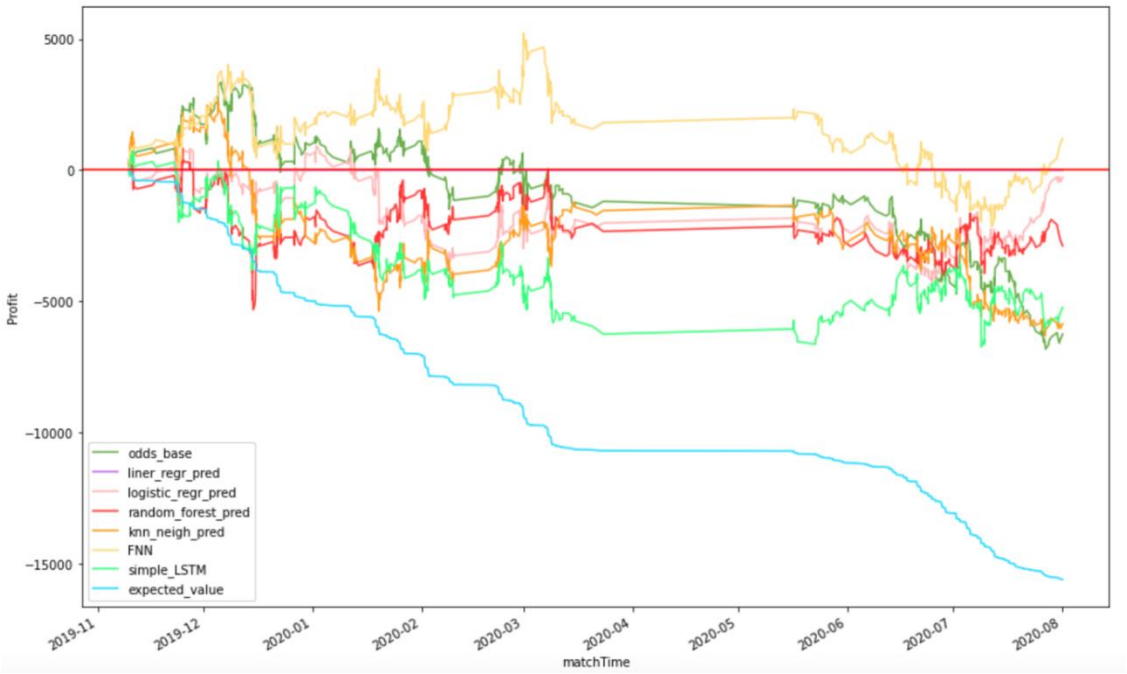

Figure 63. Autoencoder Results

Last but not least, Figure 63 shows the performance if we use a one-layer autoencoder. Overall, the result is in between the two multi-layer autoencoders approach. The result is closest to the all-league model as the MSE of the autoencoder is the lowest among all other autoencoders. However, since multi-layer neural networks usually have a high capability to perform a more complex task. Thus, we will mainly focus on stacked/deep autoencoder in the future experiments.

Table 12 shows an overall comparison of all type of dimension reduction approaches. As mentioned above, overall performance of deep autoencoder (-$1975 on average) is slightly better than stacked autoencoder (-$2525 on average). Notice: the average was taken within machine learning models only, the benchmark models were ignored. In addition, dimension reduction really helped to mix the odds features and other features so that the performance improved generally.

(all-league)	Expected Value	Odds based	Linear Regr	Logistic Regr	Random forest	KNN	FNN	LSTM	AVG
Base	-16000	-6250	-50	1250	100	-4500	-2000	-12500	-2950
PCA, POV=0.95	-16000	-6250	0	1250	-1000	-10000	-5000	-3500	-3042
PCA, POV=0.999	-16000	-6250	-50	2000	-5750	-4500	-1300	100	-1583
Stacked Autoencoder	-16000	-6250	-50	-1000	2000	-10000	-100	-6000	-2525
Deep Autoencoer	-16000	-6250	-50	-1800	2000	-2000	-14000	4000	-1975
Autoencoder	-16000	-6250	0	-100	-3000	-6000	1200	-5100	-2167

Table 12. Comparison of dimension reduction

6.6 Feature Selection

Since we applied a lot of simple models in this project, it is time consuming if we do feature selection for all models and all leagues. Thus, feature selection is only done for linear regression and logistic regression in AIC for the matches in the La Liga (西甲) because it contains the second largest number of matches and is the most famous league around the world.

6.6.1 Akaike Information Criterion (AIC)

Akaike Information Criterion is a popular metric to evaluate the performance of fitting of a model. The formula of AIC is $AIC = 2 * p - 2\ln(L)$ where L is the maximized point from the likelihood function of the model and p is the number of features for current model. For example, in regression model, the formula of AIC is as below:

$$AIC = 2p - 2\left\{-\frac{n}{2}\left[ln(2\pi) + 1 + \ln\left(\frac{RSS_i}{n}\right)\right]\right\}$$

Therefore, we can use AIC to compare the models with different features and find out the best model. According to the formula above, AIC is the indicator of the difference between the features and fitted features so the smaller AIC represents a better model [38].

When selecting features using AIC, both forward selection and backward selection needed to be considered. Generally, forward selection means to start the feature selection with an intercept model and add one more feature in every step. On the other hand, the backward selection means to start the feature selection with a model containing full features and remove one feature in every step. Thus, the forward selection and backward selection may have different output models so the best model should be generated after comparing the AIC of the final models from forward selection and backward selection.

AIC for Linear Regression

Forward Selection

According the result below, the linear regression model from forward feature selection should contain those 8 features.

```
Step:  AIC=161.3
fd$hkjc_hdc_results ~ fd$home_full_勝率_總 + fd$away_full_失_近6 +
    fd$fifa_team_home_score_ATT + fd$homeVsAwayPassTenRecord_converted +
    fd$crown_hdc_home_first + fd$同主客净勝_home + fd$bet365_hdc_initialFirst_away +
    fd$bet365_hdc_initialFirst_home

                                    Df Sum of Sq    RSS    AIC
<none>                                           390.52 161.30
```

Figure 64. AIC result using forward feature selection for linear regression model in La Liga

Backward Selection

According the result below, the linear regression model from backward feature selection should contain those 96 features.

```
Step:  AIC=106.14
fd$hkjc_hdc_results ~ fd$fifa_team_home_score_ATT + fd$fifa_team_home_score_MID +
    fd$fifa_team_home_score_DEF + fd$fifa_team_home_score_能力 +
    fd$fifa_team_home_score_球隊評分 + fd$fifa_team_away_score_ATT +
    fd$fifa_team_away_score_MID + fd$fifa_team_away_score_能力 +
    fd$hkjc_hdc_home_first + fd$hkjc_hdc_away_first + fd$hkjc_hdc_home_last +
    fd$hkjc_hdc_away_last + fd$bet365_hdc_home_first + fd$bet365_hdc_away_first +
    fd$bet365_hdc_away_last + fd$bet365_hdc_initialFirst_away +
    fd$bet365_hdc_initialLast_home + fd$bet365_hdc_initialLast_away +
    fd$crown_hdc_home_first + fd$crown_hdc_away_first + fd$crown_hdc_home_last +
    fd$crown_hdc_away_last + fd$crown_hdc_initialFirst_away +
    fd$crown_hdc_initialLast_home + fd$crown_hdc_initialLast_away +
    fd$macau_hdc_initialFirst_home + fd$bet365_hdc_hkjcFirst_home +
    fd$bet365_hdc_hkjcFirst_away + fd$crown_hdc_hkjcFirst_home +
    fd$crown_hdc_hkjcFirst_away + fd$bet365_hdc_hkjcLast_away +
    fd$crown_hdc_hkjcLast_home + fd$crown_hdc_hkjcLast_away +
    fd$macau_hdc_hkjcLast_home + fd$homePassTenRecord_converted +
    fd$總進球_away + fd$總進球_home + fd$總失球_home + fd$場均進球_away +
    fd$場均進球_home + fd$勝率_away + fd$勝率_home + fd$平率_away +
    fd$同主客進_away + fd$同主客進_home + fd$同主客失_away +
    fd$同主客失_home + fd$同主客均進_away + fd$同主客均進_home +
    fd$同主客勝_away + fd$同主客勝_home + fd$同主客平_away +
    fd$同主客負_away + fd$home_full_賽_主 + fd$home_full_賽_客 +
    fd$home_full_賽_近6 + fd$home_full_勝_主 + fd$home_full_勝_客 +
    fd$home_full_平_主 + fd$home_full_平_客 + fd$home_full_平_近6 +
    fd$home_full_得_客 + fd$home_full_得_近6 + fd$home_full_失_主 +
    fd$home_full_失_客 + fd$home_full_失_近6 + fd$home_full_排名_主 +
    fd$home_full_排名_總 + fd$home_full_勝率_主 + fd$home_full_勝率_總 +
    fd$away_full_賽_主 + fd$away_full_賽_客 + fd$away_full_賽_近6 +
    fd$away_full_勝_主 + fd$away_full_勝_近6 + fd$away_full_平_客 +
    fd$away_full_平_近6 + fd$away_full_得_主 + fd$away_full_失_主 +
    fd$away_full_失_客 + fd$away_full_失_近6 + fd$away_full_排名_主 +
    fd$away_full_排名_客 + fd$away_full_排名_總 + fd$away_full_勝率_主 +
    fd$away_full_勝率_客 + fd$home_player_weight_mean + fd$away_player_weight_mean +
    fd$home_player_value_mean + fd$home_player_age_mean + fd$away_player_age_mean +
    fd$home_player_hit_mean + fd$away_player_hit_mean + fd$home_player_power_mean +
    fd$away_player_power_mean + fd$away_player_hidden_power_mean

                                    Df Sum of Sq     RSS    AIC
<none>                                           87.047 106.14
```

Figure 65. AIC result for backward feature selection in La Liga

Best Linear Regression Model

Since the AIC for the output model of forward feature selection is 161.3 and the AIC for the output model of backward feature selection is 106.14. Therefore, the best linear regression model is the model from backward feature selection. Then, we used those features from the best model to evaluate all the model.

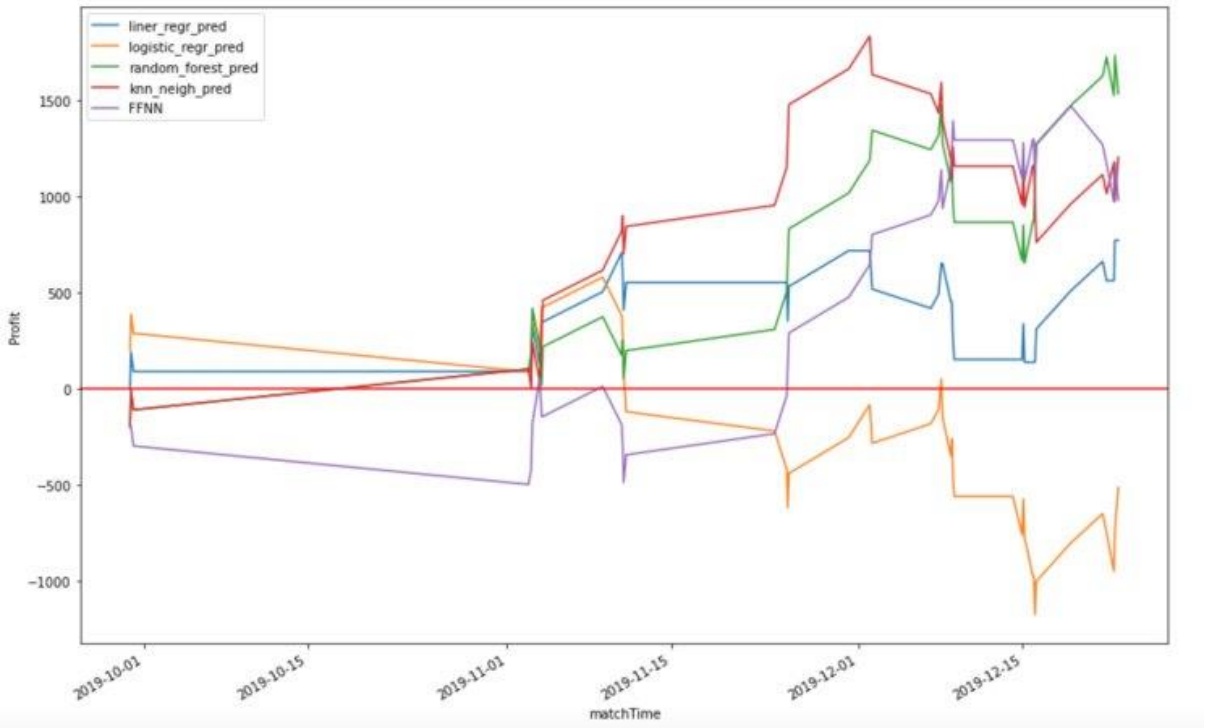

Figure 66. Predicted profit using selected features from best linear regression model

While using the selected features from best linear regression model, most of the models got a positive profit.

AIC for Logistic Regression

Forward Selection

According the result below, the logistic regression model from forward feature selection should contain those 6 features.

```
Step:  AIC=412.21
fd$hkjc_hdc_results ~ fd$away_full_勝率_客 + fd$home_full_負_總 +
    fd$home_full_勝率_總 + fd$away_full_平_近6 + fd$away_full_得分_近6 +
    fd$同主客均進_home
```

Figure 67. AIC result using forward feature selection for logistic regression model in La Liga

Backward Selection

According the result below, the linear regression model from backward feature selection should contain those 225 features.

```
Step:  AIC=1008
fd$hkjc_hdc_results ~ fd$home_away_label + fd$fifa_team_home_score_ATT +
    fd$fifa_team_home_score_MID + fd$fifa_team_home_score_DEF +
    fd$fifa_team_home_score_能力 + fd$fifa_team_home_score_球隊評分 +
    fd$fifa_team_away_score_ATT + fd$fifa_team_away_score_MID +
    fd$fifa_team_away_score_DEF + fd$fifa_team_away_score_能力 +
    fd$fifa_team_away_score_球隊評分 + fd$hkjc_hdc_home_first +
    fd$hkjc_hdc_away_first + fd$hkjc_hdc_home_last + fd$hkjc_hdc_away_last +
    fd$bet365_hdc_home_first + fd$bet365_hdc_away_first + fd$bet365_hdc_home_last +
    fd$bet365_hdc_away_last + fd$bet365_hdc_initialFirst_home +
    fd$bet365_hdc_initialFirst_away + fd$bet365_hdc_initialLast_home +
    fd$bet365_hdc_initialLast_away + fd$crown_hdc_home_first +
    fd$crown_hdc_away_first + fd$crown_hdc_home_last + fd$crown_hdc_away_last +
    fd$crown_hdc_initialFirst_home + fd$crown_hdc_initialFirst_away +
    fd$crown_hdc_initialLast_home + fd$crown_hdc_initialLast_away +
    fd$macau_hdc_home_first + fd$macau_hdc_away_first + fd$macau_hdc_home_last +
    fd$macau_hdc_away_last + fd$macau_hdc_initialFirst_home +
    fd$macau_hdc_initialFirst_away + fd$macau_hdc_initialLast_home +
    fd$macau_hdc_initialLast_away + fd$bet365_hdc_hkjcFirst_home +
    fd$bet365_hdc_hkjcFirst_away + fd$crown_hdc_hkjcFirst_home +
    fd$crown_hdc_hkjcFirst_away + fd$macau_hdc_hkjcFirst_home +
    fd$macau_hdc_hkjcFirst_away + fd$bet365_hdc_hkjcLast_home +
    fd$bet365_hdc_hkjcLast_away + fd$crown_hdc_hkjcLast_home +
    fd$crown_hdc_hkjcLast_away + fd$macau_hdc_hkjcLast_home +
    fd$macau_hdc_hkjcLast_away + fd$homePassTenRecord_converted +
    fd$awayPassTenRecord_converted + fd$homeVsAwayPassTenRecord_converted +
    fd$總進球_away + fd$總進球_home + fd$總失球_away + fd$總失球_home +
    fd$淨勝球_away + fd$淨勝球_home + fd$場均進球_away + fd$場均進球_home +
    fd$勝率_away + fd$勝率_home + fd$平率_away + fd$平率_home +
    fd$負率_away + fd$負率_home + fd$同主客進_away + fd$同主客進_home +
    fd$同主客失_away + fd$同主客失_home + fd$同主客淨勝_away +
    fd$同主客淨勝_home + fd$同主客均進_away + fd$同主客均進_home +
    fd$同主客勝_away + fd$同主客勝_home + fd$同主客平_away +
    fd$同主客平_home + fd$同主客負_away + fd$同主客負_home +
    fd$home_full_賽_主 + fd$home_full_賽_客 + fd$home_full_賽_總 +
    fd$home_full_賽_近6 + fd$home_full_勝_主 + fd$home_full_勝_客 +
    fd$home_full_勝_總 + fd$home_full_勝_近6 + fd$home_full_平_主 +
    fd$home_full_平_客 + fd$home_full_平_總 + fd$home_full_平_近6 +
    fd$home_full_負_主 + fd$home_full_負_客 + fd$home_full_負_總 +
    fd$home_full_負_近6 + fd$home_full_得_主 + fd$home_full_得_客 +
    fd$home_full_得_總 + fd$home_full_得_近6 + fd$home_full_失_主 +
    fd$home_full_失_客 + fd$home_full_失_總 + fd$home_full_失_近6 +
    fd$home_full_淨_主 + fd$home_full_淨_客 + fd$home_full_淨_總 +
    fd$home_full_淨_近6 + fd$home_full_得分_主 + fd$home_full_得分_客 +
    fd$home_full_得分_總 + fd$home_full_得分_近6 + fd$home_full_排名_主 +
    fd$home_full_排名_客 + fd$home_full_排名_總 + fd$home_full_排名_近6 +
    fd$home_full_勝率_主 + fd$home_full_勝率_客 + fd$home_full_勝率_總 +
    fd$home_full_勝率_近6 + fd$away_full_賽_主 + fd$away_full_賽_客 +
    fd$away_full_賽_總 + fd$away_full_賽_近6 + fd$away_full_勝_主 +
    fd$away_full_勝_客 + fd$away_full_勝_總 + fd$away_full_勝_近6 +
    fd$away_full_平_主 + fd$away_full_平_客 + fd$away_full_平_總 +
    fd$away_full_平_近6 + fd$away_full_負_主 + fd$away_full_負_客 +
    fd$away_full_負_總 + fd$away_full_負_近6 + fd$away_full_得_主 +
    fd$away_full_得_客 + fd$away_full_得_總 + fd$away_full_得_近6 +
```

Figure 68. AIC result using backward feature selection for logistic regression model in La Liga

Best Logistic Regression Model

Since the AIC for the output model of forward feature selection is 412.21 and the AIC for the output model of backward feature selection is 1008, Therefore, the best logistic regression model is the model from forward feature selection. Then, we used those features from the best model to evaluate all the model.

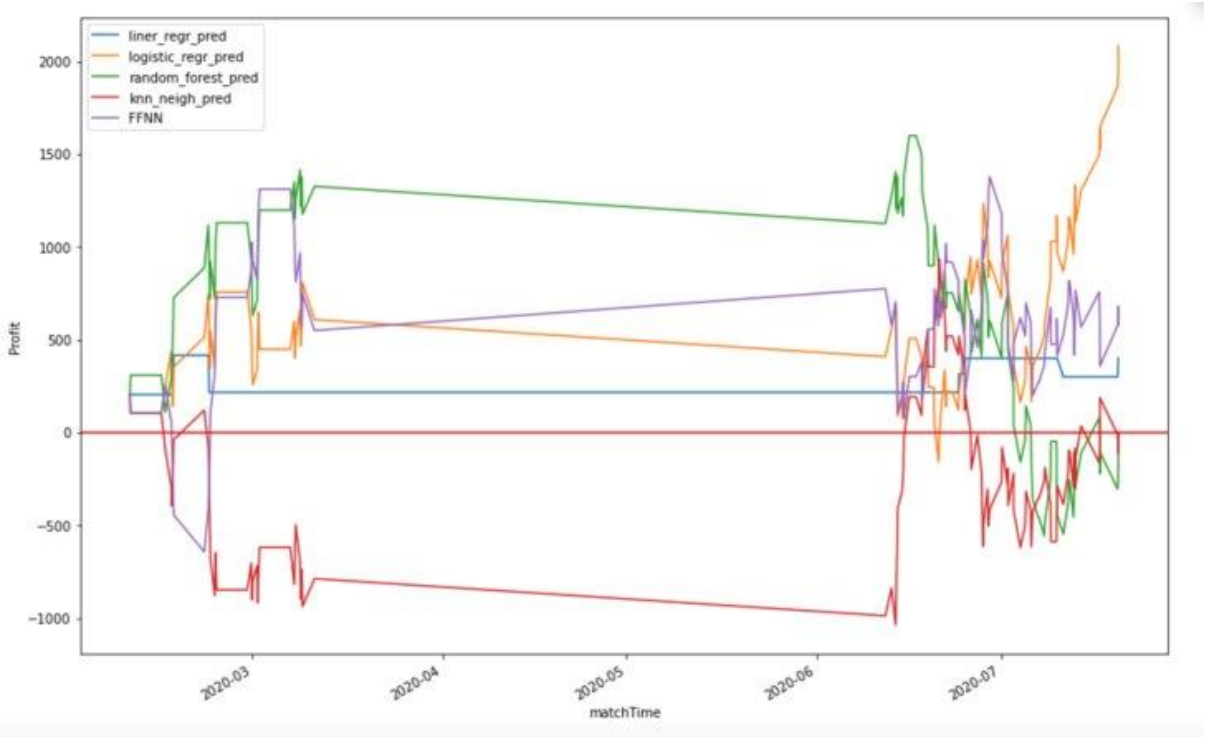

Figure 69. Predicted profit using selected features from best logistic regression model

While using the selected features from best logistic regression model, most of the models got a positive profit, specifically the logistic regression model. It is because the feature selection procedure is based on a logistic model.

6.6.2 Recursive Feature Elimination

It is a backward selection algorithm which helps to choose the best set of features that give the highest performance on a given model [39]. In our case, the Support Vector Machine model with linear kernel will be used for recursive feature elimination (RFE) as mentioned in this paper [40].

The procedure of the SVM-RFE algorithm is to train the model with all the input variables, the ranking of each feature will then be evaluated and the least significant feature will be removed. This process is repeated until the number of desired features is reached. The ranking of the features is determined by the magnitude of its w_i and the feature with the smallest ${w_i}^2$ will be eliminated. For example, after training a 5 class SVM, a [10 by n_features] weights matrix (w) is obtained. The feature importance can be calculated from this w, by taking the square of w and perform a column-wise sum. As a result a dimension of [n_features] vector is obtained and the feature which has the smallest w_i will be eliminated. Notice: since RFE determined feature importance by the coefficient; therefore the paper [40] has to use a linear kernel must be used in this case.

Since the number of features to be removed is a parameter which is needed to be tuned. Hence, 5-fold cross validation is applied here to search for n_features from 1 to N, where N is the total number of input features. In our case, we input a dataset which contain 59 features and 12 of them are selected by RFE. It is interesting that all the selected features are odds related features.

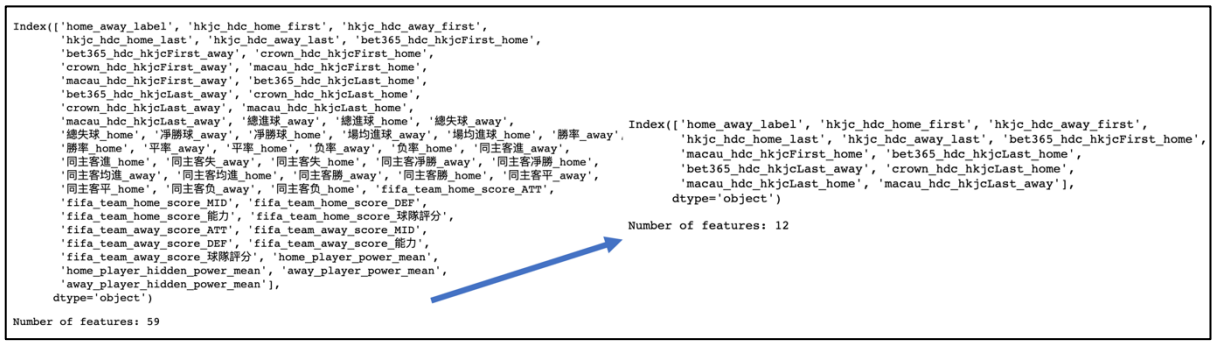

Figure 70. The feature before and after RFE.

We will evaluate these selected features through our all-league pipeline. Stacked autoencoder and deep autoencoder will be used because we want to see if the second approach is always better or not.

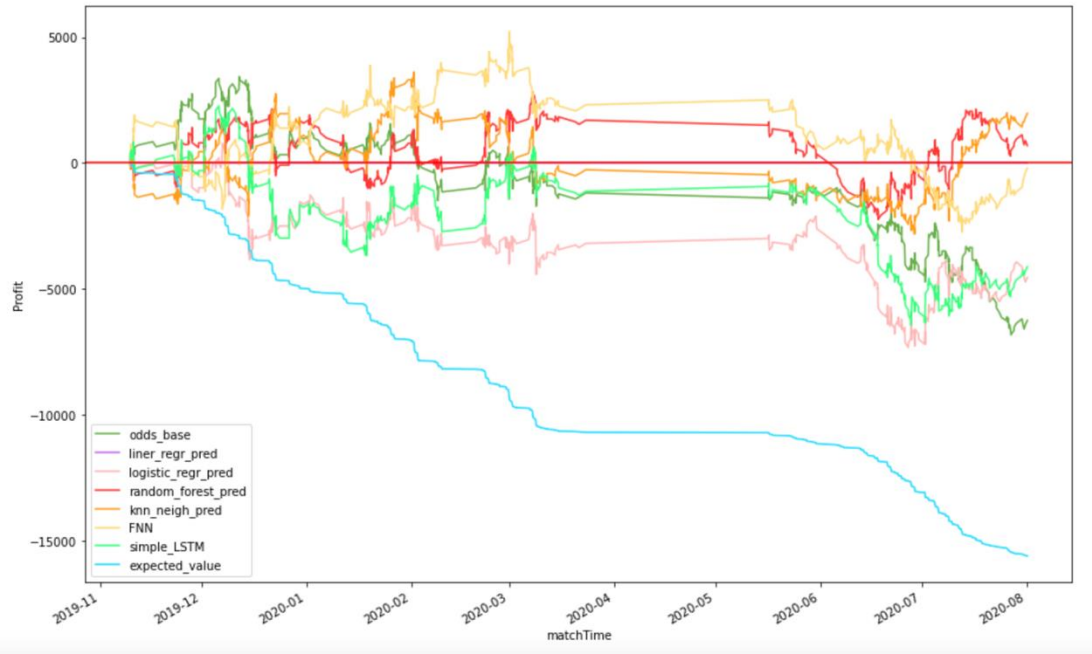

Figure 71. Stack Autoencoder for RFE

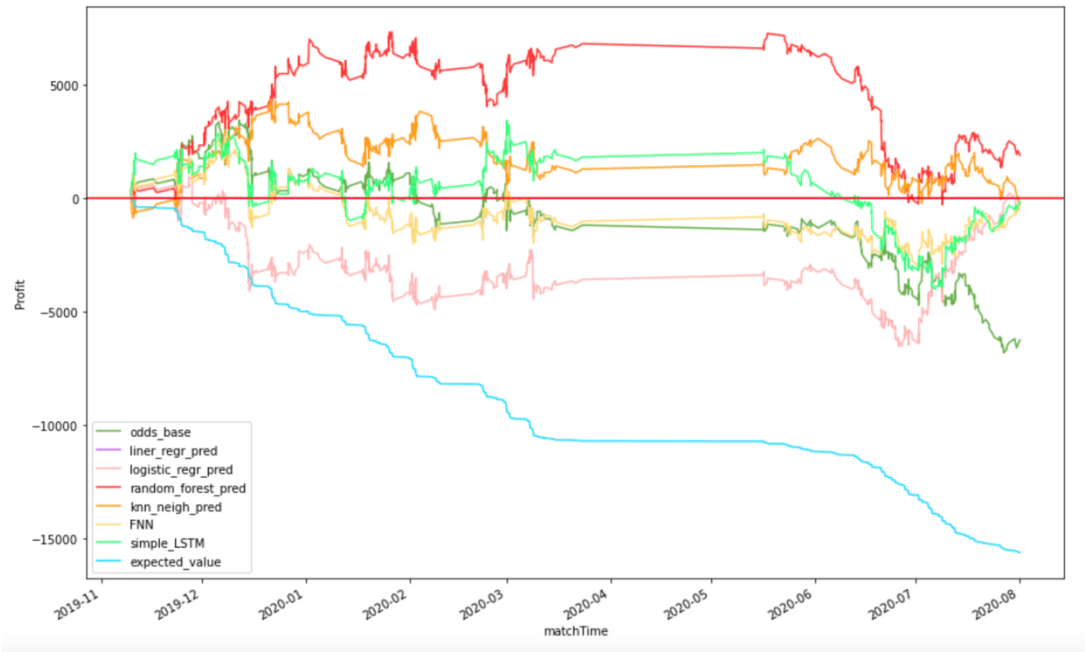

Figure 72. Deep Autoencoder for RFE

Figure 71 and Figure 72 show the performance of each approach. Once again, deep autoencoder was able to give a better result with all models' profit are close to or greater than $0. Overall, feature selection using RFE did improve the model results drastically. It is because we ruled out the non-significate variables so a better quantity dataset can be modelled. Since RFE selected only odds related variable. We are interested in this behaviour and a further experiment on modelling with just odds will be discuss later in Section 6.7.

6.6.3 Feature Selection with Feedforward Neural Network

Besides using the linear approach to do the feature selection, the Feedforward Neural Network (FNN 1.0) which introduce in Section 5.5.1 was also used for performing feature selection. We tried a backward selection approach (RFE); hence, a forward feature selection approach will be adopted in this section. We did not grid search on the best number of features because it took too long for the program to terminate as in too many FNNs needed to be trained. We set 40 as the number of features to be selected. Afterward, the best 40 features will be used for evaluation. Figure 73 illustrates the selected 40 variables. The order shows in the figure represents the order of feature selected. Unlike RFE, the first 12 features were not non-odds features. In fact, out of the 40 selected features, only 10 of them were odds related.

```
array(['home_away_label', 'hkjc_hdc_home_first', '平率_home',
       'bet365_hdc_hkjcLast_away', '場均進球_away',
       'away_player_hidden_power_mean', '同主客平_away', '负率_away',
       'home_player_hidden_power_mean', '同主客進_away', '同主客失_home',
       '總進球_home', 'fifa_team_home_score_MID', 'crown_hdc_hkjcLast_away',
       'hkjc_hdc_away_last', '平率_away', 'bet365_hdc_hkjcFirst_home',
       'fifa_team_away_score_球隊評分', 'fifa_team_home_score_能力',
       '場均進球_home', 'hkjc_hdc_home_last', '勝率_home', '總失球_away',
       'bet365_hdc_hkjcLast_home', '同主客均進_away',
       'crown_hdc_hkjcFirst_away', '同主客均進_home', '總失球_home', '勝率_away',
       '同主客失_away', '同主客勝_home', 'bet365_hdc_hkjcFirst_away',
       'away_player_power_mean', '同主客负_away', 'hkjc_hdc_away_first',
       'macau_hdc_hkjcLast_away', '同主客淨勝_home',
       'fifa_team_away_score_ATT', '同主客平_home', '淨勝球_home'], dtype='<U29')
```

Figure 73. Forward Selection by FNN

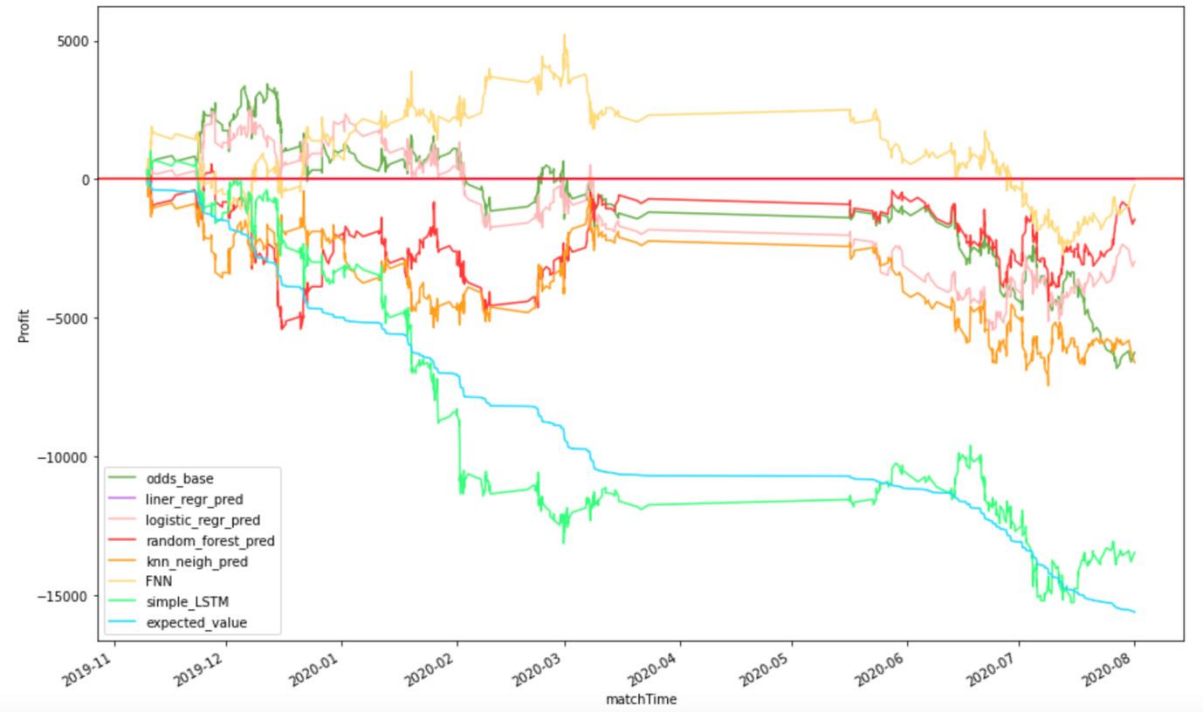

Figure 74. Stacked Autoencoder for Forward Selection

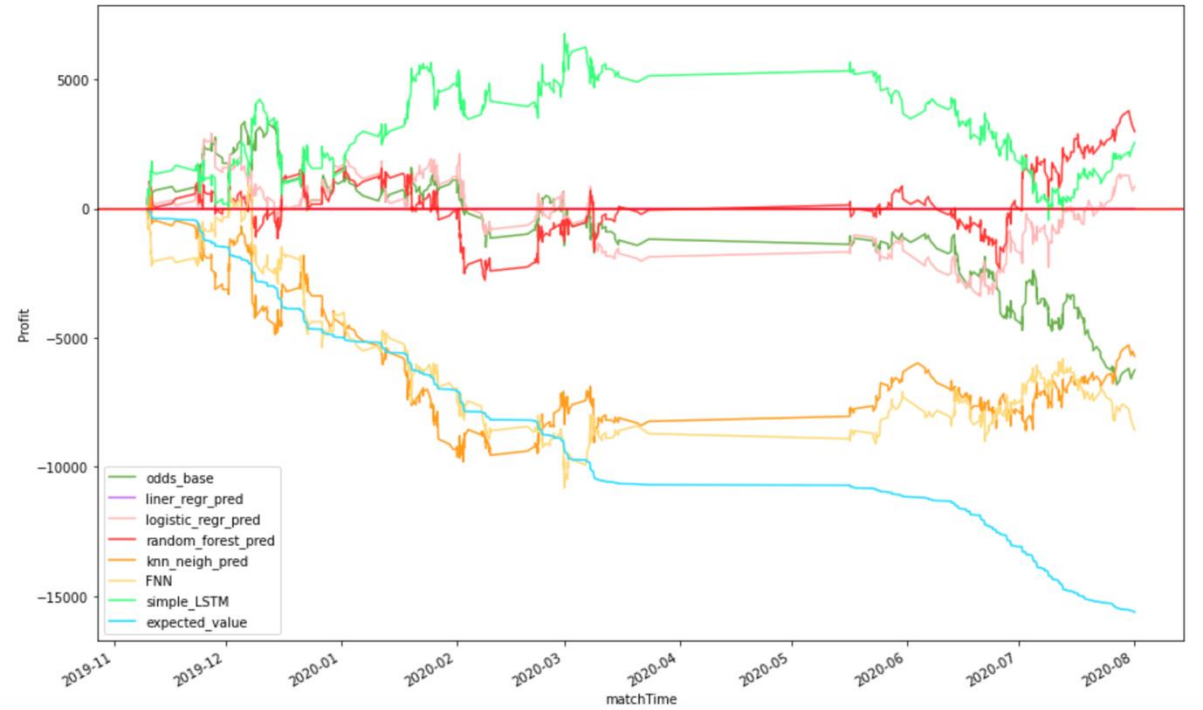

Figure 75. Deep Autoencoder for Forward Selection

Figure 74 and Figure 75 depict the profit gain by using two different dimension reduction methods. Deep autoencoder again gave a better result with 3 models have a return larger than 0. Similarly, the performances of the two neural network models are different in the two approaches, when one performs well the other perform worst. Interestingly, their performances

are consistent with what we mentioned in Section 6.5 where LSTM performs better in deep autoencoder where FNN performs better in stacked autoencoder. Perhaps, the code generated by deep autoencoder is more suitable for LSTM.

Table 13 illustrates an overall statistic of this section. The RFE method generally performs better than FNN forward selection. Perhaps it is because no grid search on the ***number of features*** was adopted on the FNN forward selection due to the limit on computation power. In particular, we are interested in the odds-only features selected by RFE and will be discussed in the next section.

(all-league)	Expected Value	Odds based	Linear Regr	Logistic Regr	Random forest	KNN	FNN	LSTM	AVG
Base	-16000	-6250	-50	1250	100	-4500	-2000	-12500	-2950
Stacked RFE	-16000	-6250	0	-4500	500	2000	-500	-4000	-1083
Deep RFE	-16000	-6250	0	-100	2000	-100	-200	-10	265
Stacked FNNFS	-16000	-6250	0	-3000	-1500	-6300	-10	-13000	-3968
Deep FNNFS	-16000	-6250	0	750	3000	-800	-8000	2500	-425

Table 13. Comparison of dimension reduction

6.7 Odds only Features

According to Section 4.8.1, it is proved that the odds have a high relationship with the handicap results, which is corresponding to the result of feature selection, in which all selected features are odd-based features. While selecting the best model in Section 6.4, there is a finding that the performances of LSTM model were not good in all criteria if containing all the features. Mainly because the features except odds are not really in a time series manner. Therefore, there is an evaluation to check how do the models perform with odd-based features only. Moreover, since we believe that the odds from different bookmakers are time series data, we assume that LSTM will get a better performance. Similar to the evaluation above, the first problem of the evaluation is also the league so the evaluation is distributed into 2 parts, all-league and by-league, in which only contain the leagues witch more than 200 matches.

6.7.1 All-league Model

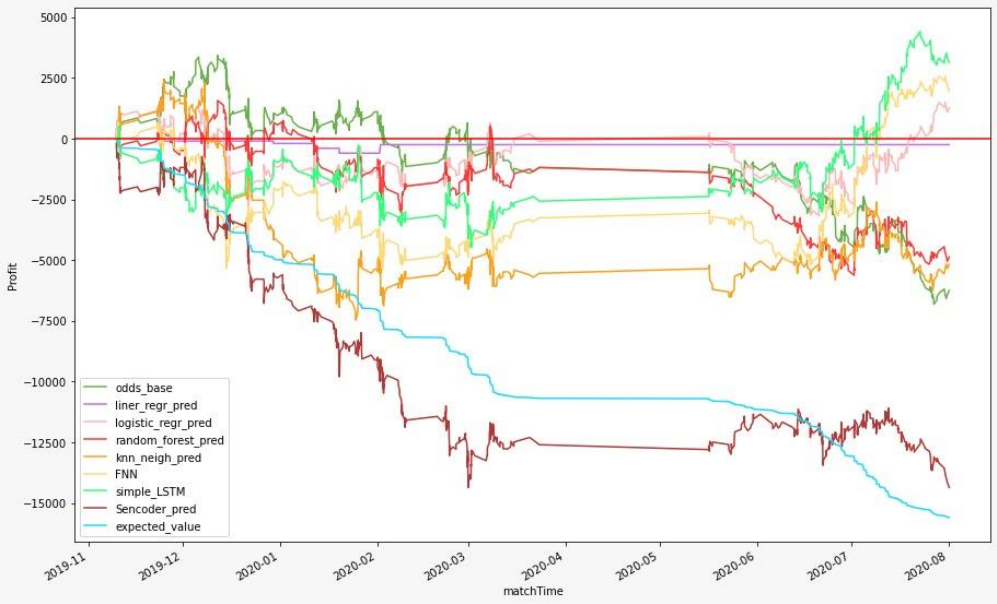

Figure 76. All-league model with odd-based features

According to Figure 76, the performances of most of the models are similar to the selected best model in Section 6.4. However, there is a significant difference that is the LSTM model. Comparing with the LSTM from the best model, LSTM with odd-based features improved a lot, and even got the best accumulated profit on the last day. However, the performances of all models are also unstable so they are still not a profitable model.

6.7.2 By-league Model

Similar to the by-league model in Section 6.2, the evaluation of by-league model will combine the profit gained from each league and show in one plot.

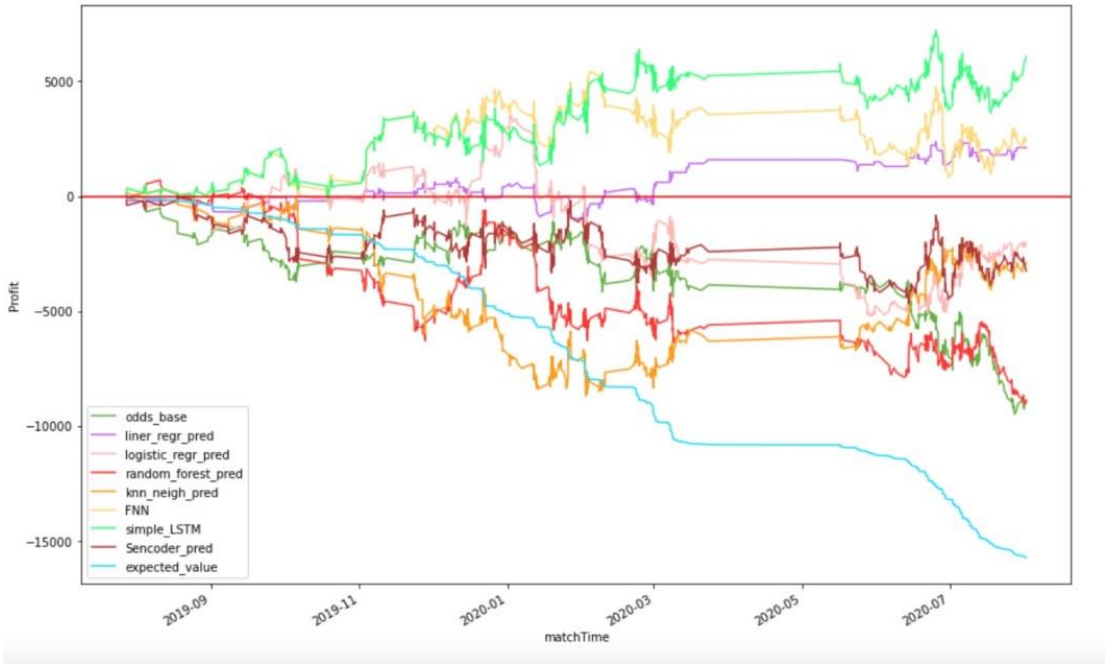

Figure 77. By-league model with odd-based features

According to the Figure 77, it is different from the by-league model in Section 6.2. There are several differences between them, such as the performance of the logistic model and random forest model worsened. However, the most significant difference is that the performances of all neural network models (except Sencoder) improved a lot, especially the LSTM model. Although the final profit of LSTM from the all-league model is also positive, the performance of by-league one is more stable and more likely to be a profitable model. Thus, there is an in-depth exploration below.

An overall comparison of all-league model and by-league that used only odds related features is illustrates in Table 7. On average the by-league model improved dramatically. This shows our original hypothesis: “bookmakers will have a different standard on calculating the odds for different leagues” might be correct. However, it is more difficult for model to capture the signals after adding extra features. As a result, the original by-league model perform worst.

	Expected Value	Odds based	Linear Regr	Logistic Regr	Random forest	KNN	FNN	LSTM	Sencoder	AVG
by-league	-16000	-9000	-4250	-4250	2250	-12500	-750	-16000	-3000	-5500
by-league (odds only)	-16000	-9000	2000	-2200	-9000	-2700	2500	6300	-3000	-871
all-league	-16000	-6250	-50	1250	100	-4500	-2000	-12500	-14000	-4529
all-league (odds only)	-16000	-6250	-100	1250	-4900	-5100	2000	3000	-14500	-2621

Table 14. Comparison of odds only Model

6.7.3 Leagues with Best Performance

After comparing to average performance from all leagues, the overall performance from the Premier League is the best.

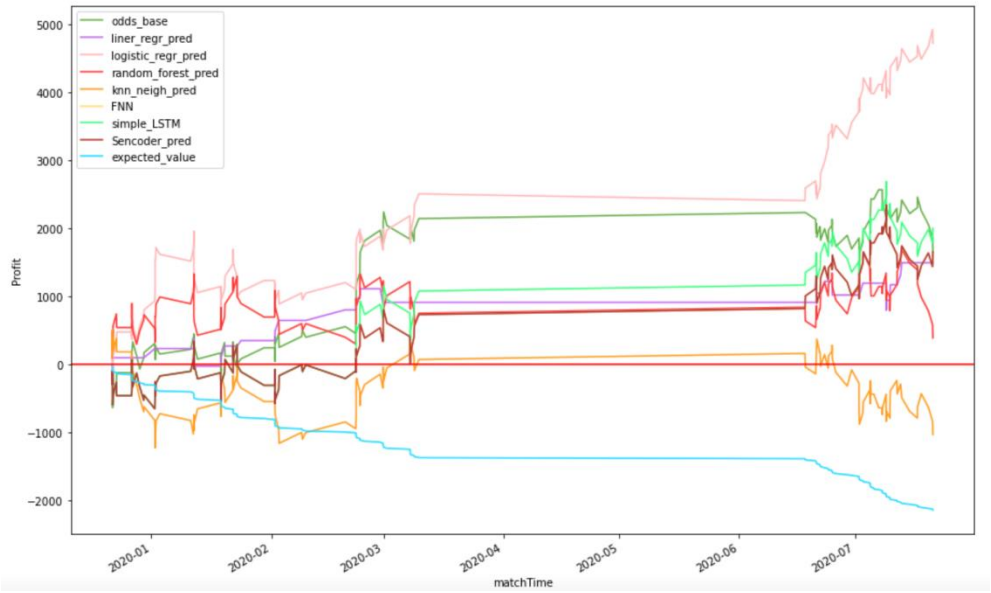

Figure 78. Evaluation result with odd-based features from the Premier League

In Premier League, almost all models had positive profit, especially the logistic regression models. Also, it is interesting that the odds-based model which is a model that always buy the teams with lowest odds got a fairly good profit. However, most of the models got similar or worser performance than it. Thus, although they are profitable models, there is still a room for improvements.

6.7.4 Leagues with Worst Performance

After comparing to average performance from all leagues, the overall performance from the La Liga is the worst.

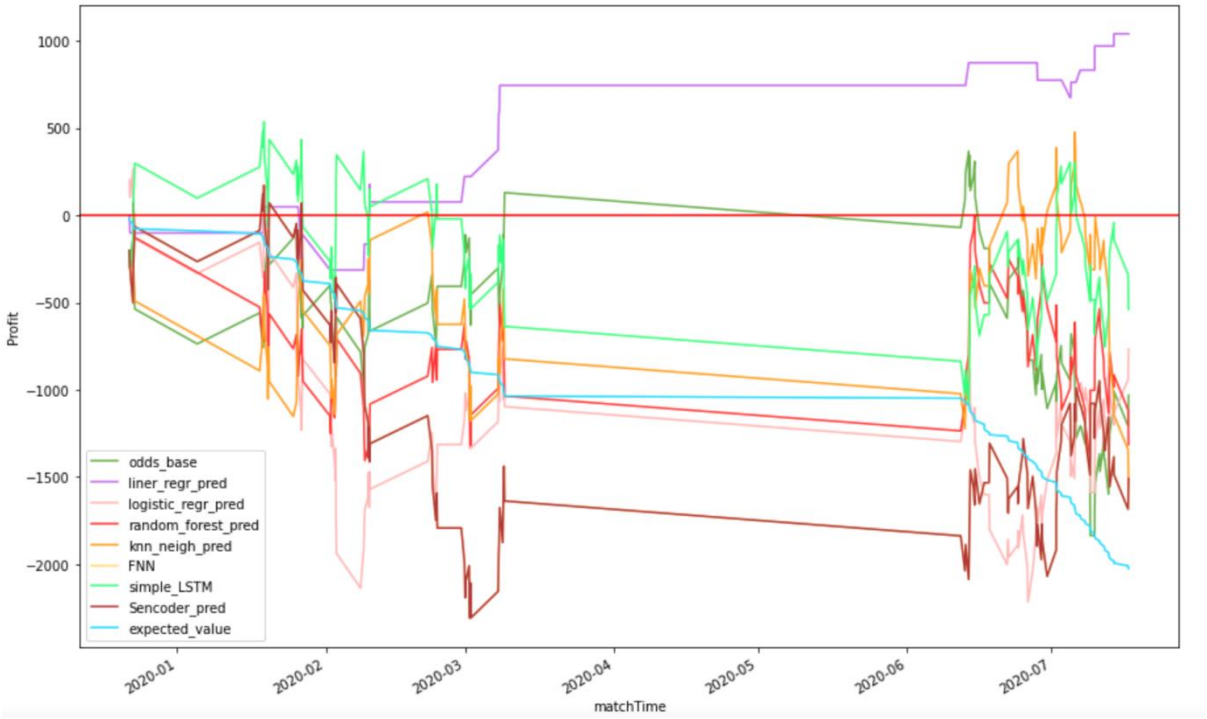

Figure 79. Evaluation result with odd-based features from the La Liga

According to Figure 79, the performances of all model keep fluctuating whereas only linear regression model can keep a stable performance. It is because linear regression decided not to bet on any team most of the time. Relatively, the performances of the neural network models are not really bad. However, it seems that the models "learnt" nothing and likes a random guess.

7. Evaluation, Phrase 2

In this section, we will mainly focus on the experiments we did during the second semester, three neural network (FNN 2.0, Sencoder 2.0, CNN) models will be focused. In the first semester, we show that all-league model is the most suitable one for the experiment that we will perform in the second term, hence, we will use it as the default baseline. Odds only features were proved to have the best performance in term of profit (Section 6.7). We did extra experiments on this and found that by just using initial odds features, the performance s of the models can be further improved. The detail and rationale of this will be discussed in Section 7.3. All metrics (Section 5.1) we used to evaluate the performances of the models are same as term 1. Since we have already beaten the bookmakers expected values and odds-base benchmark model in the first term. Thus, we will focus on having a profitable return in project phrase 2. In fact, after the refinement of neural network models and the help of other new modelling setting, we are able to achieve a much better performance, this comparison will be shown in Section 7.6.

7.1 Cluster-then-Predict Model, Phrase 2

Since there are more data available in the second semester, therefore, it is a good time for us to re-evaluate the Cluster-then-Predict model on the new data set. The hypothesis we made in term 1 is that the trend we discovered will exist in term 2, we will verify if the hypothesis is correct at the beginning of term 2. According to Figure 80 to Figure 88, we can see that the trend we observed previously is no longer exist in the new dataset. One possible explanation is that the distribution of the dataset might vary over time. Thus, the idea of Cluster-then-Predict was discarded in project phrase two. Nevertheless, this inspire us the idea of adding a test-like discriminator when we are working on GAN (Section 4.10.2).

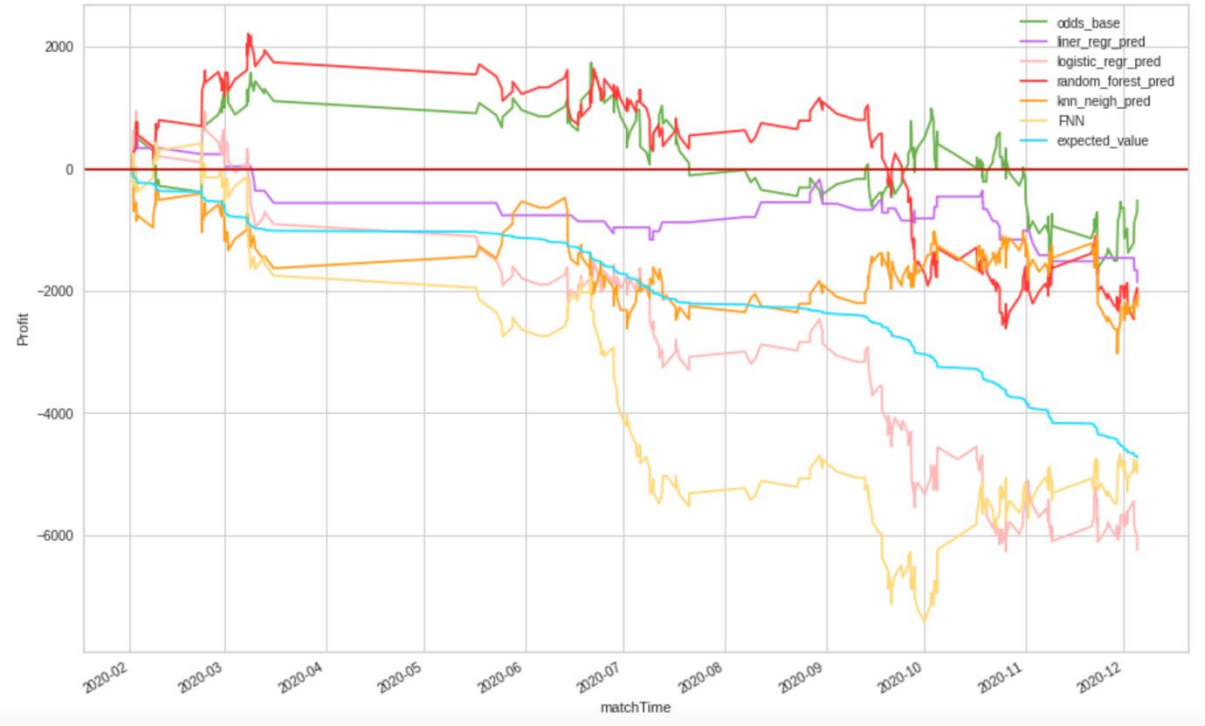

Figure 80. K-means cluster 1 (phrase 2)

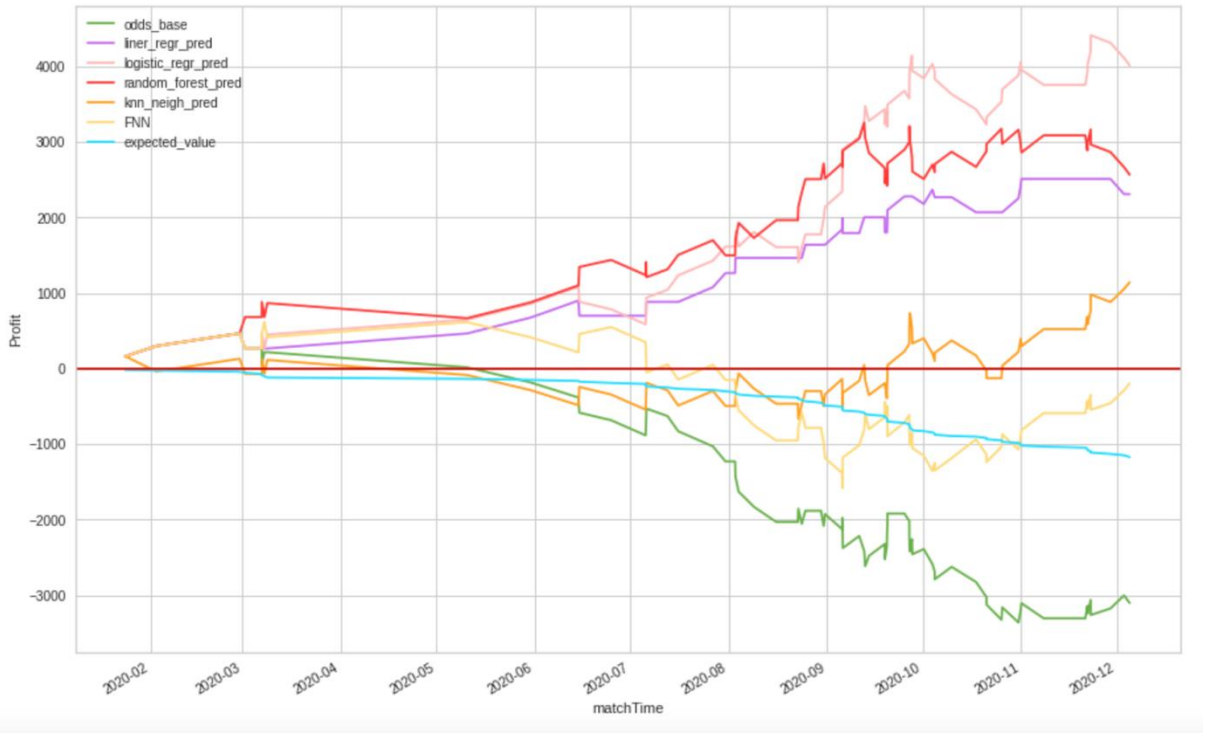

Figure 81. K-means cluster 2 (phrase 2)

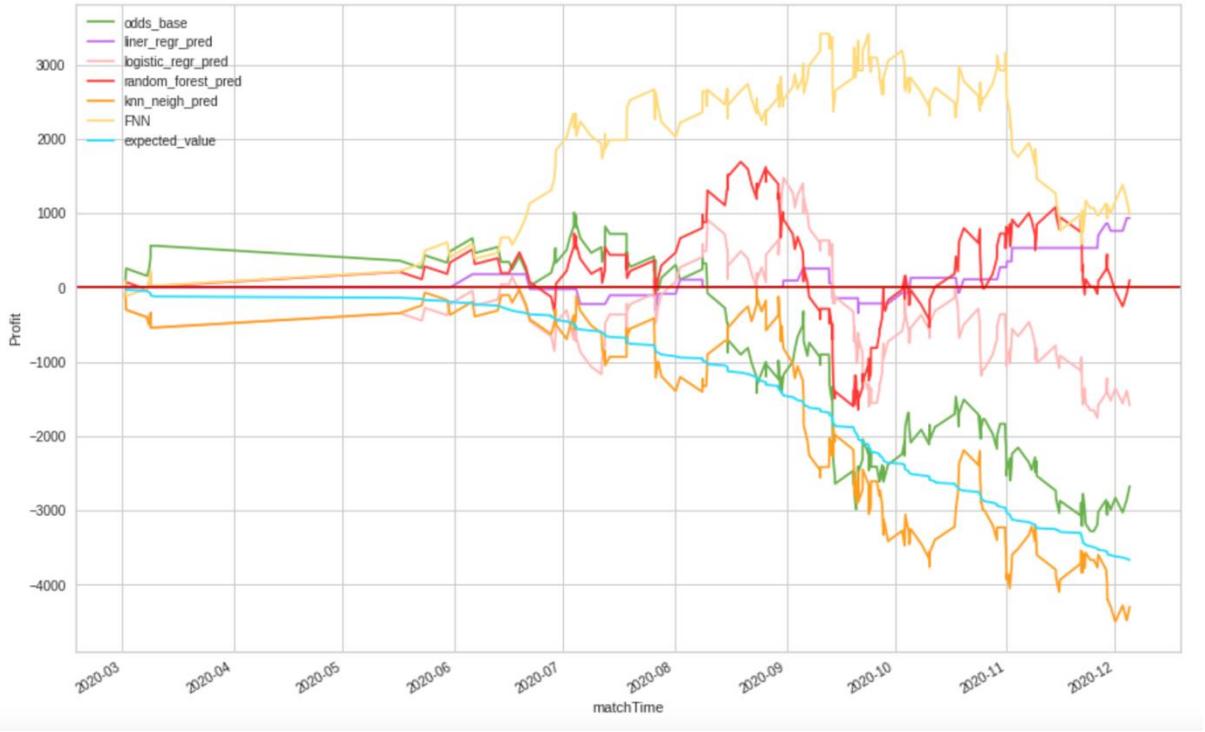

Figure 82. K-means cluster 3 (phrase 2)

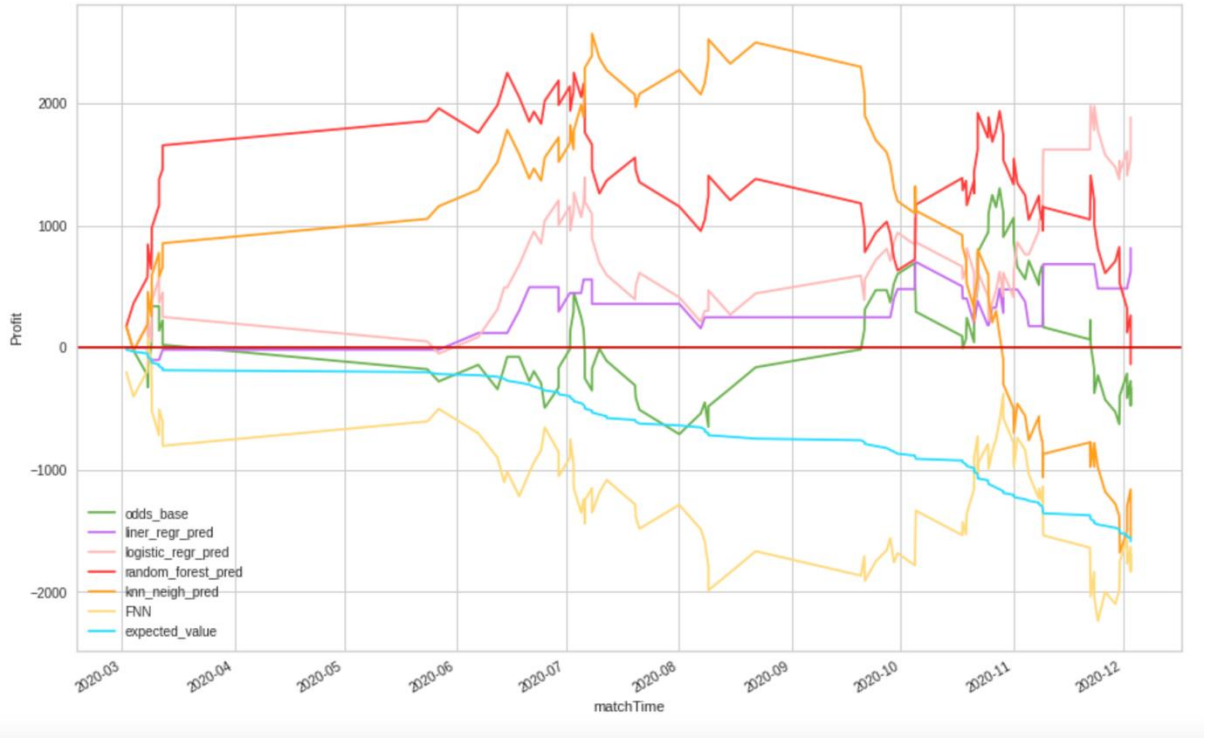

Figure 83. K-means cluster 4 (phrase 2)

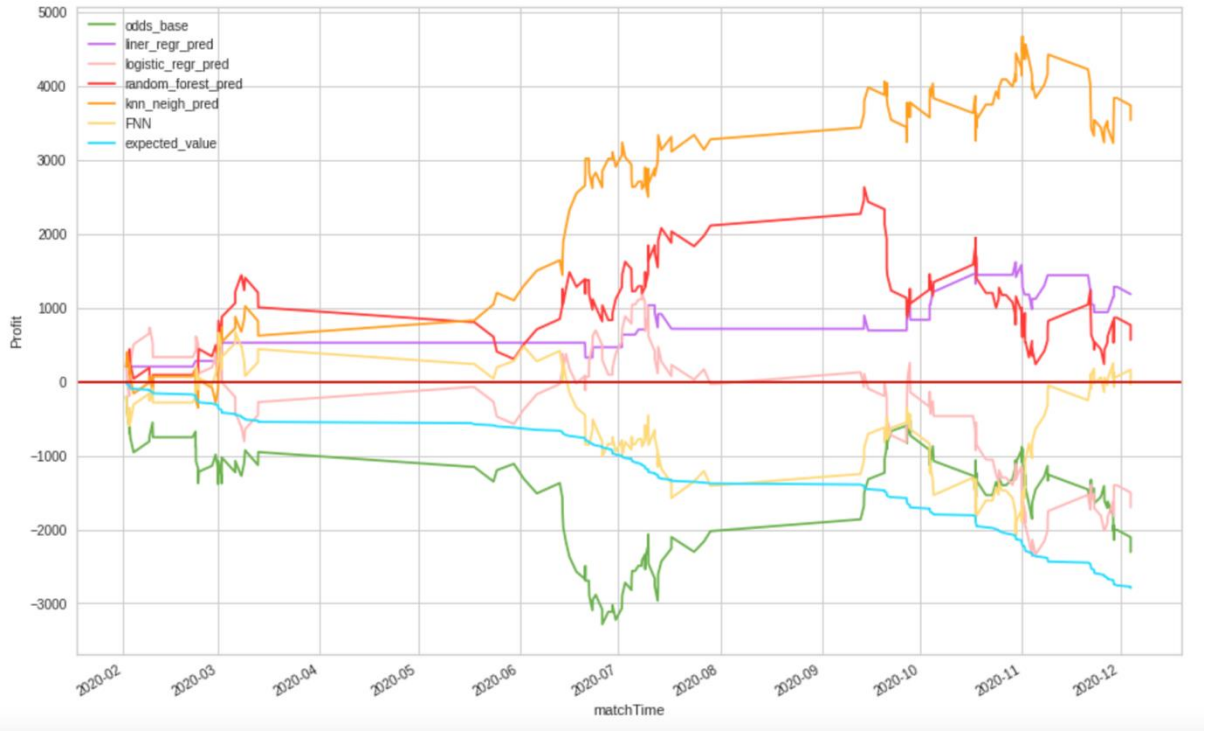

Figure 84. K-means cluster 5 (phrase 2)

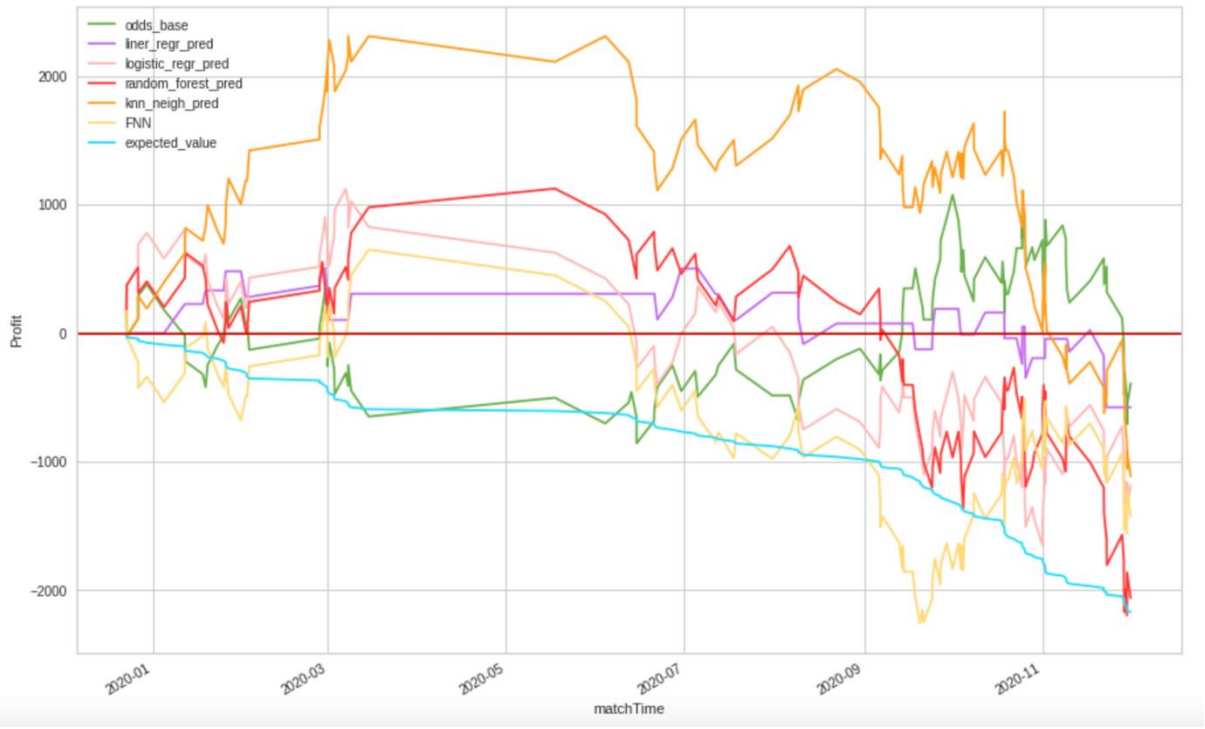

Figure 85. K-means cluster 6 (phrase 2)

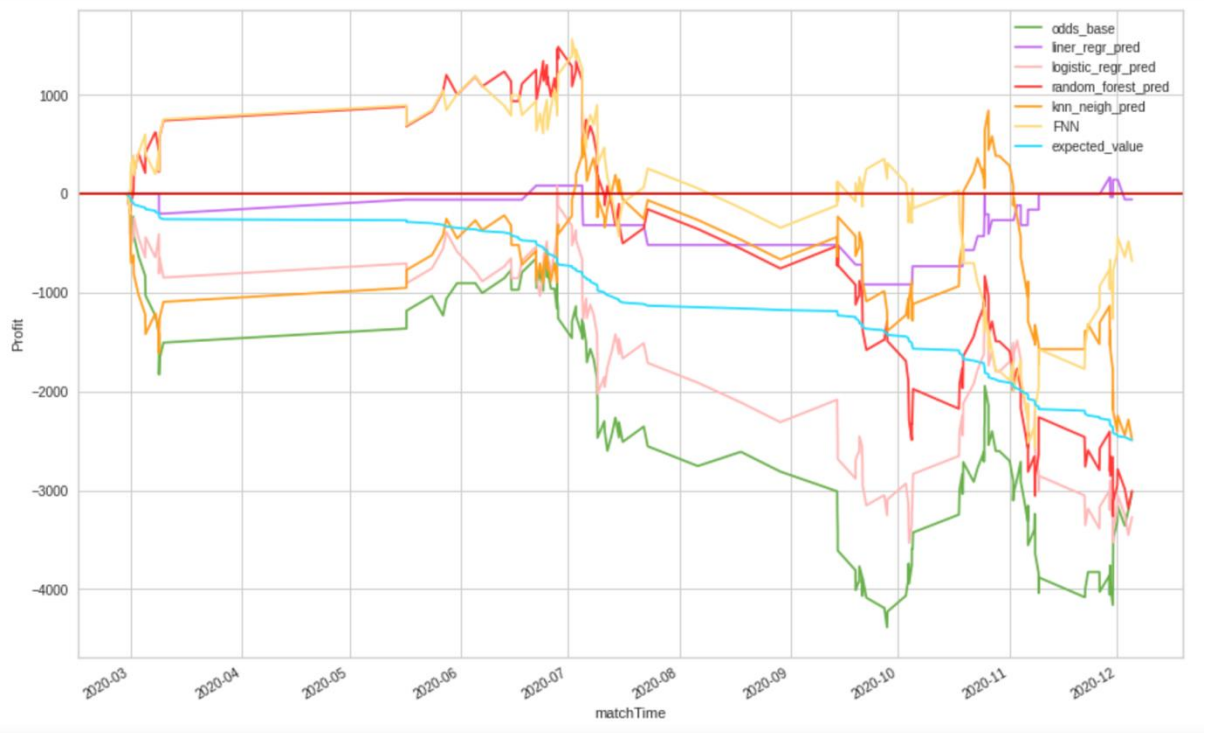

Figure 86. K-means cluster 7 (phrase 2)

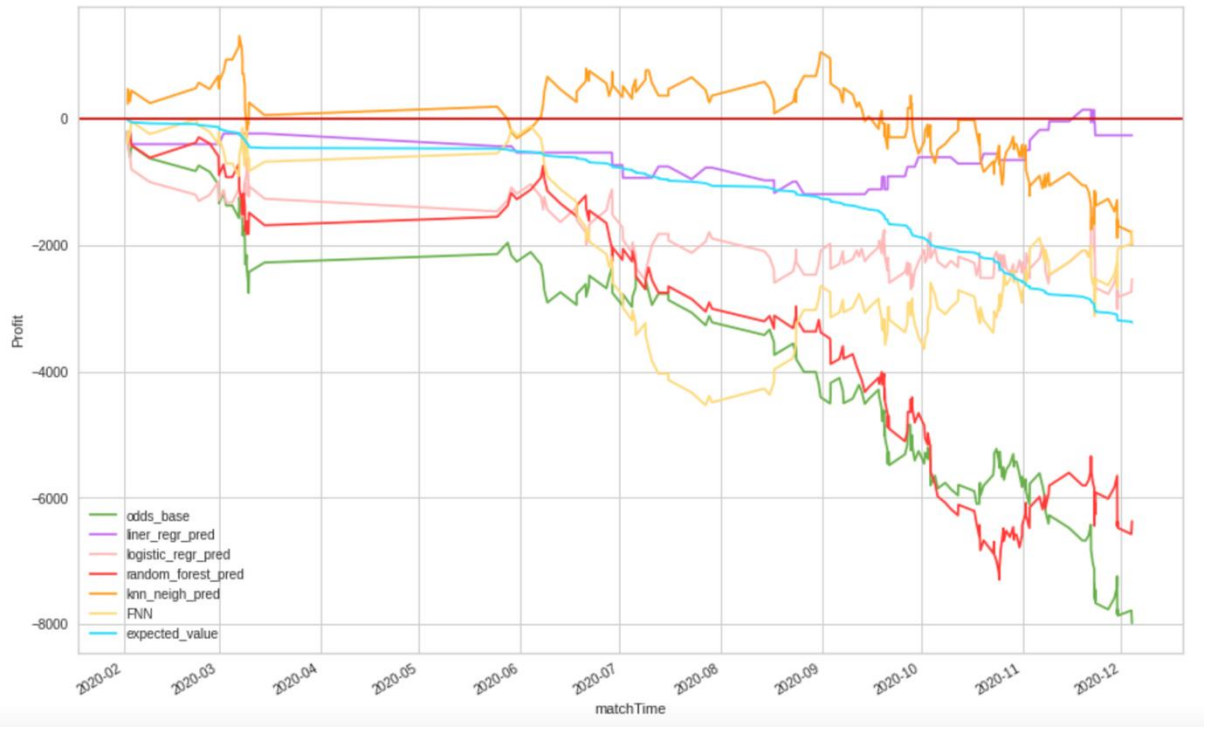

Figure 87. K-means cluster 8 (phrase 2)

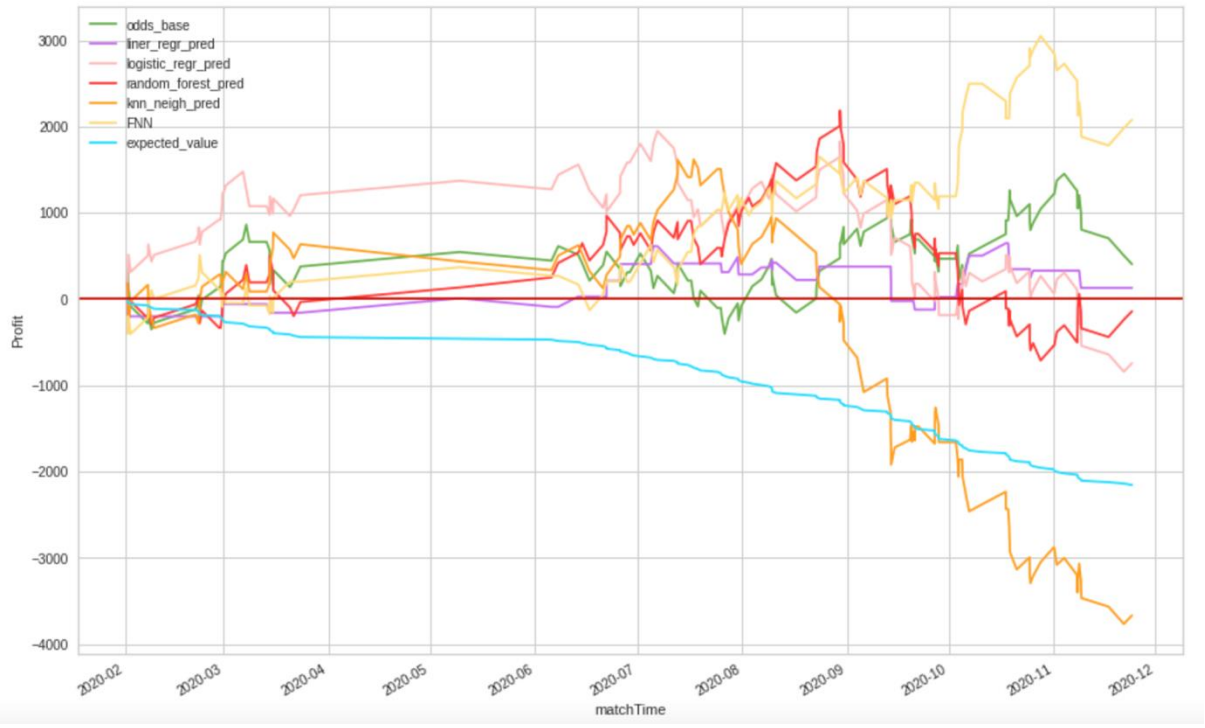

Figure 88. K-means cluster 9 (phrase 2)

7.1 XGBoost

In this section, we will discuss about the performance on XGBoost. The features used for modelling are the ***odds only features*** (Section 6.7). Several statistical models and the two benchmark models are used for comparison. According to Figure 89, the blue curve is the XGBoost accumulative profits over time. The performance is considered to be stable over time and have a positive return at the end. Although XGBoost is a boosting version of decision tree, it did not overperform the random forest.

Figure 89. XGBoost performance

7.2 Best Model for League

In this section, background of best model for league strategy will be discussed and the results will be shown afterwards.

In the entire project, we have one assumption that is bookmaker calculate the odds by league differently. Hence, we assume that some models might be able learn some useful representation for some leagues but not the others, even under the same training environment. In other words, each model has their own "best" league. We will now discuss our approach to select the best model for each league. The original dataset is divided into three datasets which are training, selecting and testing. The training set is used for modelling whereas the selecting set is used to select the best model for each league, as a result, a combined best model is created. This combined best model will then be used for final evaluation in the testing set.

We propose three ways to select the "best" model on a league basis, ***final***, ***slope*** and ***auto***. ***final*** is to compare the final profit earn by each model at each league, higher ***final*** is better. While ***slope*** is to fit a linear regression on the profit earned across time and get the corresponding slope, higher ***slope*** is preferred. ***final*** can ensure to have the most profit at the end whereas ***slope*** can ensure the model is earning money constantly across time. The two metrics mentioned above yield the same results most of the time. Hence, it is a good idea to compare both methods so to have a more stable and general model. ***auto*** combine both ideas from ***final*** and ***slope***, it returns the average of the two above normalized metrics so ***auto*** is used for default. Mathematically, $auto = sum\big(normalized(final, slope)\big)$. Although the all-league approach is the default in project phrase 2, currently, this strategy is trained in a by-league model approach (Section 6.2) because we believe train the model in a by league basis and select the best model using by league approach might make more sense. In fact, when we perform further analysis and experiment on this approach, we found that the all-league approach works better and therefore, is adopted in the real-time prediction (Section 8).

One should not confuse ***best model for league*** with ***ensemble***, the former refer to choosing a model for each league whereas the latter one is making decision by majority vote regardless on the league.

In term of best model for league, the modelling setting is by-league training and using odds related features as explained in the beginning of this section. Figure 90 illustrates the performance of the best model for league strategy. The purple curve refers to best model for league (***auto***) and the black curve refer to the ensemble aggregating method (***weighted majority vote***) (It will be further discussed in Section 7.4.1). The performance of best model for league

clearly outweighs ensemble (aggregating). Moreover, it also proves that some model did perform better in some league. For example, let us consider model A and B with league X, Y and Z. Model A is selected by the ***weighted majority vote*** (ensemble) in league X and Y and at the same time, the ***auto*** metrics (best model for league) for A in X and Y are the highest. This means A learn some useful latent features for league X and Y so that it can give a good profit in both leagues. In other words, best model for league is a strategy to choose the model regarding to league smartly whereas ensemble (aggregating) is to consider the decision for each model carefully.

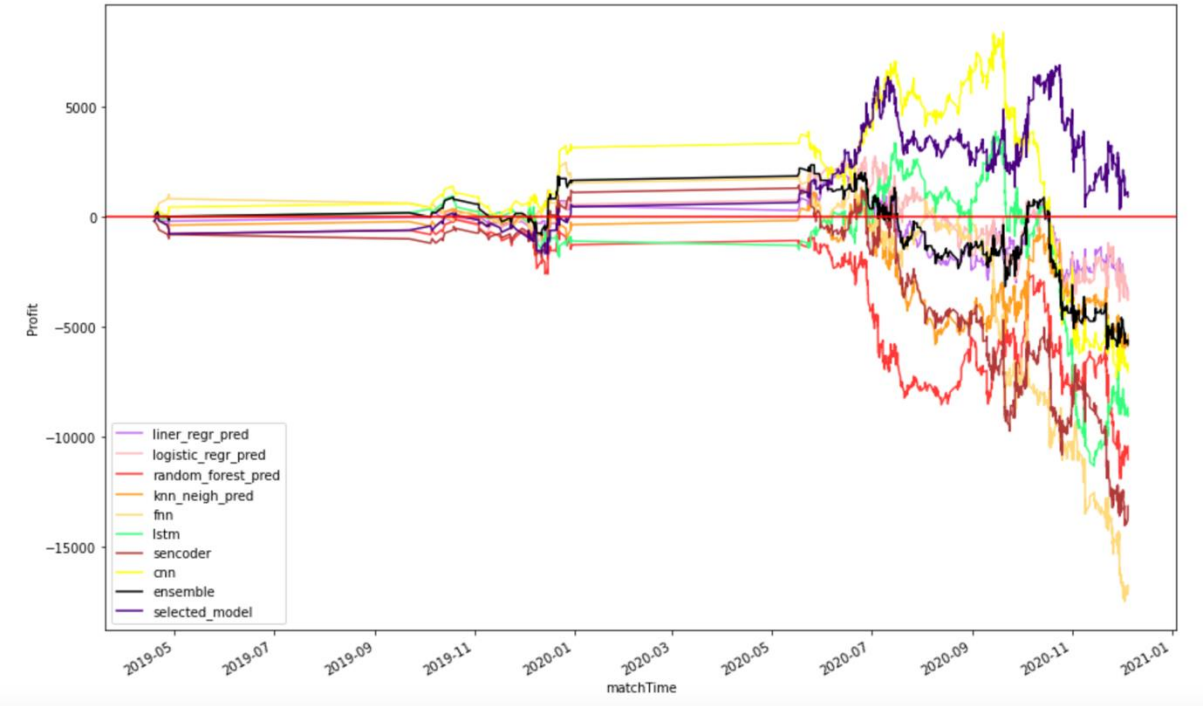

Figure 90. Best Model on each league performance

7.3 Calculated Handicap Odds and First Odds Only Features

7.3.1 What Are Odds?

Odds are the ratio of payoff if one winning in a gambling and are issued by the bookmaker. According to Pete Nordsted who is a professional sports trader and the contributor of Goal.com[4], there are many methods to calculate the odds and one of them is based on the recent performance of the teams [41]. In overall, the odds contain the probability that the betting item occurs and the betting margin (defined in glossary). Since the betting margin is designed by the bookmaker and will not affect our betting decision, the main focus is how to calculate the handicap odds based on probability without betting margin.

7.3.2 Probabilities Under HAD Odds

Before finding the handicap odds based on probability without betting margin, we need to find the probability that the team wins the match first because the handicap odds are calculated based on the probabilities [42]. When ignoring the betting margin, the odds should be

$$\text{Home Odds} = \frac{(1 - \mathrm{p}_D)}{\mathrm{p}_H}$$

$$\text{Away Odds} = \frac{(1 - \mathrm{p}_D)}{\mathrm{p}_A}$$

where $\mathrm{p}_H, \mathrm{p}_A$ and p_D represent the probability of home team winning the match, away team winning the match and draw respectively. Then, moving to the realistic case, the rate of return is needed when tackling the betting margin. Rate of return is actual ratio of turnover after betting so it helps the HAD odds remove the betting margin.

$$\text{Rate of return} = \frac{\mathrm{O}_H * \mathrm{O}_A * \mathrm{O}_D}{\mathrm{O}_H * \mathrm{O}_A + \mathrm{O}_H * \mathrm{O}_D + \mathrm{O}_A * \mathrm{O}_D}$$

$$\frac{\text{Home}\,\frac{\text{Win}}{Away}\,\text{Win}}{\text{Draw}} = \text{Target Odds} * \text{Rate of return}$$

where $\mathrm{O}_H, \mathrm{O}_A$ and O_D represent the odds of home team winning the game, away team winning the game and draw. Then, our target probabilities will be

[4] https://www.goal.com/

$$P(\text{Home team wins}) = \frac{\text{Rate of return}}{O_H}$$

$$P(\text{Away team wins}) = \frac{\text{Rate of return}}{O_A}$$

$$P(\text{Draw}) = \frac{\text{Rate of return}}{O_D}$$

7.3.3 Calculated Handicap Odds based on Probability

Comparing to the probabilities under HAD odds, the probabilities under handicap odds are much more complex because we should calculate the probability that the goal of a team is larger than a specific number. There are some examples for different handicap lines.

	Handicap odds based on probability without betting margin	
Handicap Line	Home Team	Away Team
[0]	$\frac{1 - P(\text{Draw})}{P(Home \text{ team wins})}$	$\frac{1 - P(\text{Draw})}{P(Away \text{ team wins})}$
[0/+0.5]	$\frac{1 - \frac{P(\text{Draw})}{2}}{P(Home \text{ team wins}) + \frac{P(\text{Draw})}{2}}$	$\frac{1 - \frac{P(\text{Draw})}{2}}{P(Away \text{ team wins})}$
[0/-0.5]	$\frac{1 - \frac{P(\text{Draw})}{2}}{P(Home \text{ team wins})}$	$\frac{1 - \frac{P(\text{Draw})}{2}}{P(Away \text{ team wins}) + \frac{P(\text{Draw})}{2}}$

Table 15. Example of Handicap odds based on probability without betting margin

After averaging the calculated handicap odds, the results are as below.

Handicap Line	Type	Odds							
[0]	H	2.500	2.550	2.600	2.650	2.700	2.750	2.800	2.850
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	2.900	2.950	3.000	3.050	3.100	3.150	3.200	
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[0/-0.5]	H	2.000	2.030	2.070	2.100	2.130	2.170	2.200	2.230
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	2.270	2.300	2.330	2.370	2.400	2.430	2.470	
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[-0.5/-1]	H	1.500	1.520	1.530	1.550	1.570	1.580	1.600	1.620
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	1.630	1.650	1.670	1.680	1.700	1.720	1.730	
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[-1]	H	1.380	1.390	1.400	1.410	1.430	1.440	1.450	1.460
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	1.480	1.490	1.500	1.510	1.530	1.540	1.550	
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[-1/-1.5]	H	1.300	1.310	1.320	1.330	1.340	1.350	1.360	1.370
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	1.380	1.390	1.400	1.410	1.420	1.430	1.440	
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[-1.5/-2]	H	1.210	1.220		1.240		1.250	1.260	
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	1.270	1.280	1.290		1.300	1.310		
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[-2]	H	1.190		1.200	1.210		1.220	1.230	
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	1.240		1.250	1.260		1.270	1.280	
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.100	
[-2/-2.5]	H	1.170		1.180		1.190		1.200	1.210
	HDC_H	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925
	H	1.210	1.220		1.230		1.240		
	HDC_H	0.950	0.975	1.000	1.025	1.050	1.075	1.1	

Table 16. Average of the transformation from HAD odds to Handicap odds

where H means the HAD odds and HDC_H means the calculated handicap odds using HAD odds. Thus, the handicap odds based on probability without betting margin can be calculated by the HAD odds so they are one of our features. Then, the calculated handicap odds are added as our features and the performances of most of the models were improved a lot.

7.3.1 First Odds

First odds are the initialized odds issued by the bookmakers so it will only be influenced by the probability of the team winning the match and the betting margin. On the other hand, the odds after the first odds will be influenced by the customers. For example, if there are too many customers buying the home team handicap odds, the odds will decrease in order to balance the risk and return. Hence, the influence of customers can be considered as noise to the bookie released odds, which might increase the difficultly of the modelling process. The bookmaker will keep the handicap odds against the turnover as below:

$$\frac{T_H}{T_A} = \frac{O_H}{O_A}$$

T_H: The turnover of home team handicap

T_A: The turnover of away team handicap

O_H: The odds of home team handicap

O_A: The odds of away team handicap

Thus, the bookmakers can make sure that they can earn from the customers due to the betting margin. Therefore, only the first odds can represent the probabilities that the team wins (predicted by bookie) the match which is the reason that we chose to use first odds only.

In fact, our sencoder (Section 5.5.3) has a drastic improvement by using the first odds features. One explanation is that after removing the noisy features (last odds), the autoencoder is able to capture the important (latent) signal from the first odds from different companies. Hence, the encoding functionality of the sencoder can perform a lot better and is able to give a good prediction at the end.

7.4 Ensemble Model

In this section, three approach of ensemble methods, aggregating, stacking and bagging, will be discussed.

7.4.1 Aggregating

Recall from Section 5.7 that there are two metrics used for ensemble (aggregating), ***mean*** and ***weighted majority vote*** where the former is a more conservative approach and the latter is more aggressive prediction. Their results are shown in **Error! Reference source not found.** and **Error! Reference source not found.** respectively. The features used in modelling phrase are the odds related features (Section 6.7) under all-league modelling (Section 6.4). The results

match our hypothesis that mean is more conservative. According to both figures, it is obvious that the logistic regression (pink curve) has a high cross-validation score such that it has a high contribution in the ensemble model.

Figure 91. Ensemble (aggregating) using "mean"

Figure 92. Ensemble (aggregating) using "weighted majority vote"

7.4.2 Stacking Ensemble

Stacking is one of the ensemble methods in which multiple classifiers are combined liked a stack so this idea is like combining the effort of multiple classifiers [43]. First, the dataset should be divided into different groups in order to distribute them to different heterogeneous classifiers. Then, the output of all the heterogeneous classifiers will be combined as features for a meta-classifier. Finally, the meta-classifier will generate the final prediction. Generally, the error rate of an ensemble model will decrease greatly as the wrong prediction of the ensemble model will only be generated when more than 50% predictions from heterogeneous classifiers are wrong. For example, if there are 5 independent heterogeneous classifiers with 0.3 error rate, the error rate of the ensemble classifier while ignoring the meta-classifier is as below.

$$\sum_{i=\lceil \frac{5}{2} \rceil}^{5} \binom{5}{i} * 0.3^{i} * (1-0.3)^{5-i} \approx 0.163$$

In our case, we used bookmaker to be the criteria to distribute the groups. Thus, the odds from different bookmakers will be trained by different heterogeneous classifiers such as Bet365 NN classifier, Crown NN classifier and HKJC NN classifier. The output of those NN classifiers will be the input of the final meta-classifier and then the final predicted match result is generated.

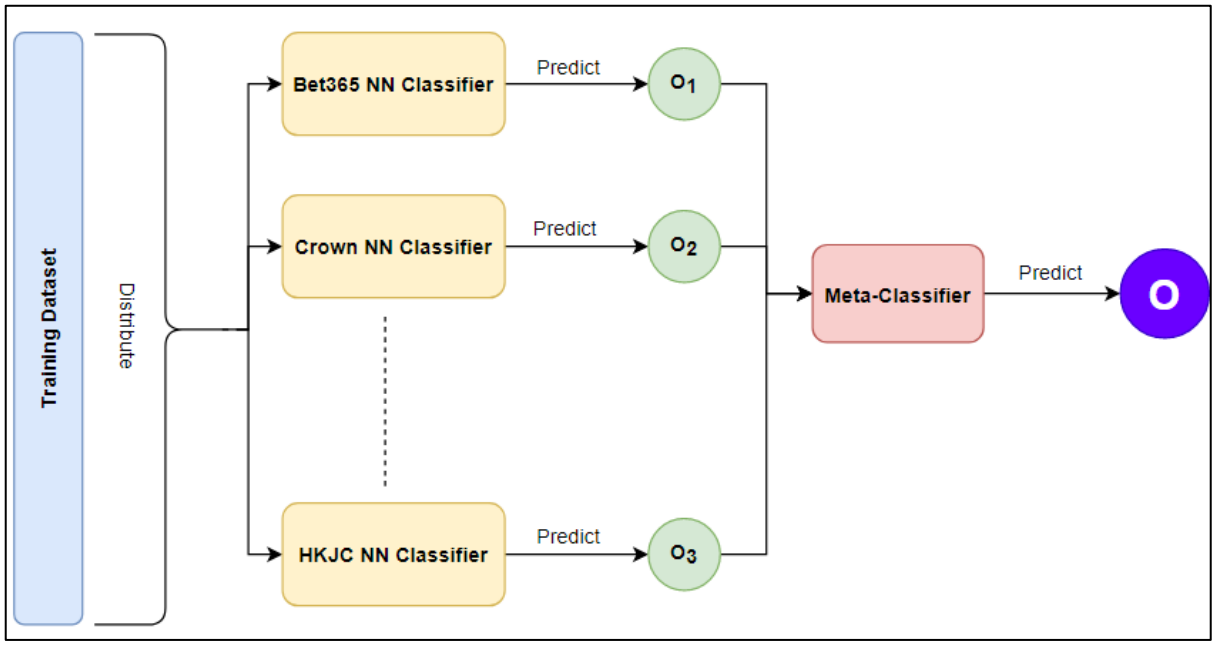

Figure 93. Structure of the Stacking Ensemble Model

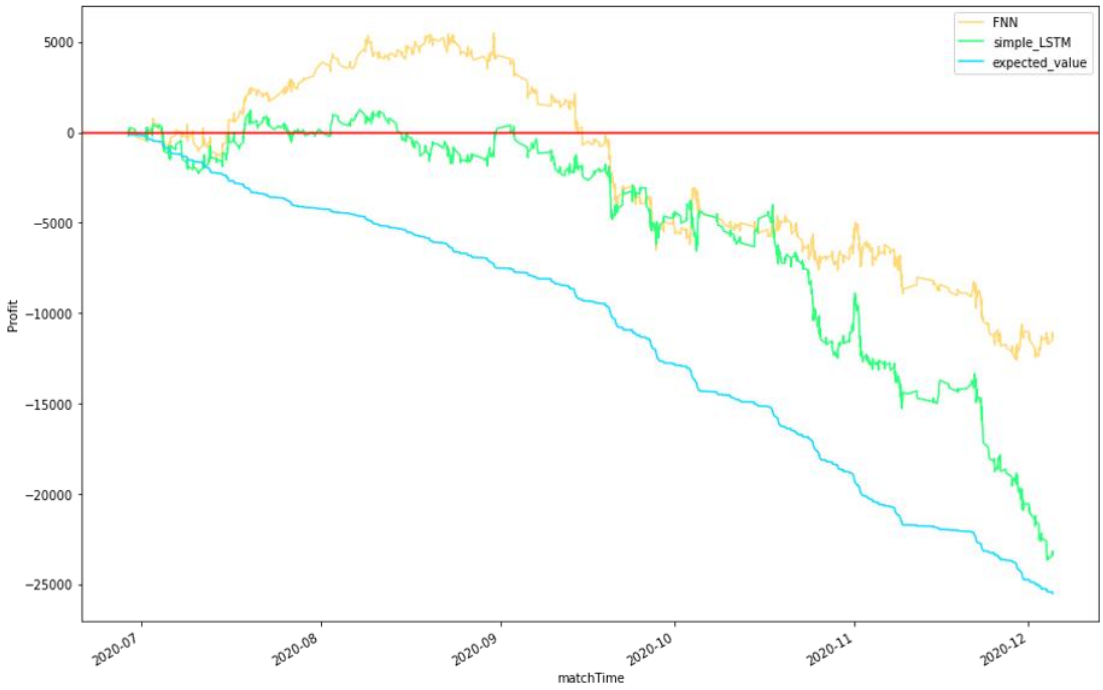

Figure 94. Performance of the Stacking Ensemble Model

In Figure 94, it is clear that both LSTM and FNN stacking ensemble models performs bad. Recalling the formation of the error of ensemble formula stated as above, if the error rate of each NN classifiers is ε, the error rate our ensemble model will be $\sum_{i=\lceil \frac{5}{2} \rceil}^{5} \binom{5}{i} * \varepsilon^{i} * (1-\varepsilon)^{5-i}$. Thus, if ε is larger than 0.5, the error rate of the ensemble model will increase actually. For example, if ε is 0.6, the error rate of the ensemble model is similar to 0.683. Moreover, since our ensemble method is stacking but not bagging, the error rate of our ensemble model will be increased by the meta-classifier too. Therefore, the main reason of the poor performance of our stacking ensemble model is that our heterogeneous classifiers are too "weak".

According to Table 6, we used Kruskal-Wallis H Test to prove that there are association between the odds from different bookmakers. The most probable reason of weak heterogeneous classifiers is that the association of the odds from different bookmakers is the key of training a successful model for soccer prediction. Since the generated outputs of heterogeneous classifiers will be independent to each other as each of them is trained separately with odds from a specific bookmaker, the original association will be lost. In order to solve this problem and reduce the influence of meta-classifier, we create a bootstrap ensemble model in which each heterogeneous classifier is trained with odds from all bookmakers in section 7.4.3.

7.4.3 Bagging Model

In this section, we will focus on the results obtained using bootstrap aggregating (bagging). Recall that there are 5 bootstrap samples created in advanced before fitting them into modelling phrase. The features used odds related features with HAD (using last odds also) and train in all-league model. Figure 95 to Figure 99 illustrate the performance for each bootstrap sample individually. The yellow (FNN) and brown (Sencoder) curve are the main focus here. Sencoder perform relatively stable among the 5 bootstrap samples, the position for the curves are the almost the same. On the other hands, FNN performance is inconsistent, this might suggest FNN cannot capture the important signal from the dataset. Overall, all models plunged after September 2020, this can confirm with the observation we have on Cluster-then-Predict Model (Section 7.1), "the distribution of the dataset might vary over time".

The next step of bootstrap is aggregating, Figure 100 shows the results of aggregating. The performance look noisy, each curve fluctuate a lot among itself. One possible explanation is that the last odds (emotion of customer) is taken into account. This is also a main reason we used first odds and HAD eventually (Section 7.3).

One interesting phenomenon we observed is that the ***ensemble aggregating*** (black) perform better than ***best model for league*** (purple) on the 5 bootstrap samples, however when it comes to bootstrap aggregating, ***best model for league*** overperform ***ensemble aggregating***.

For the bootstrap model, that uses first odds only and train in all-league strategy, achieved a significant improvement. Moreover, the FNN bootstrap is selected to be one of the candidate of the final best model.

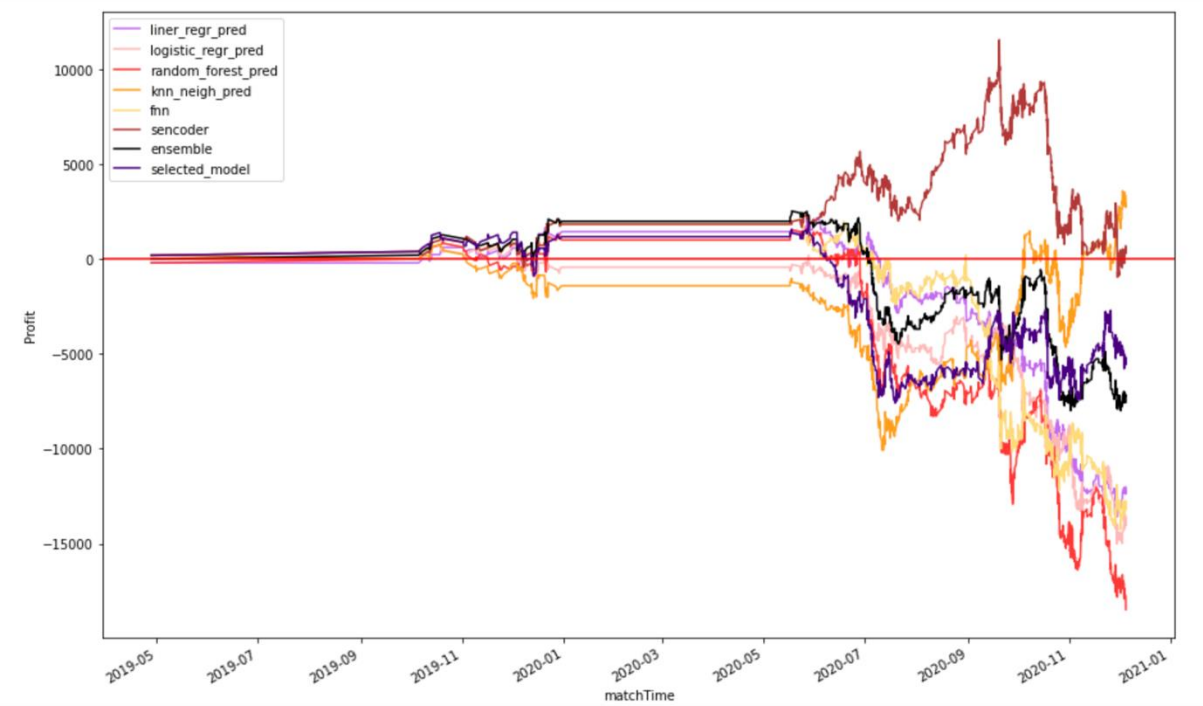

Figure 95. Bootstrap sample 1 performance

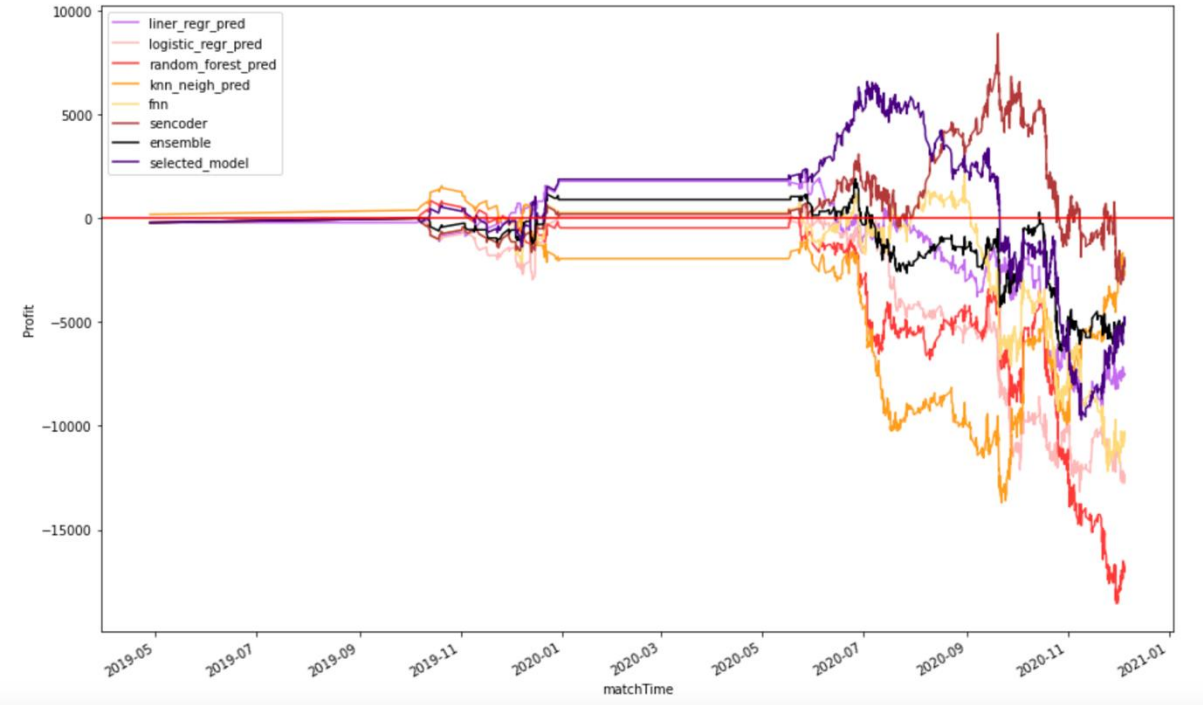

Figure 96. Bootstrap sample 2 performance

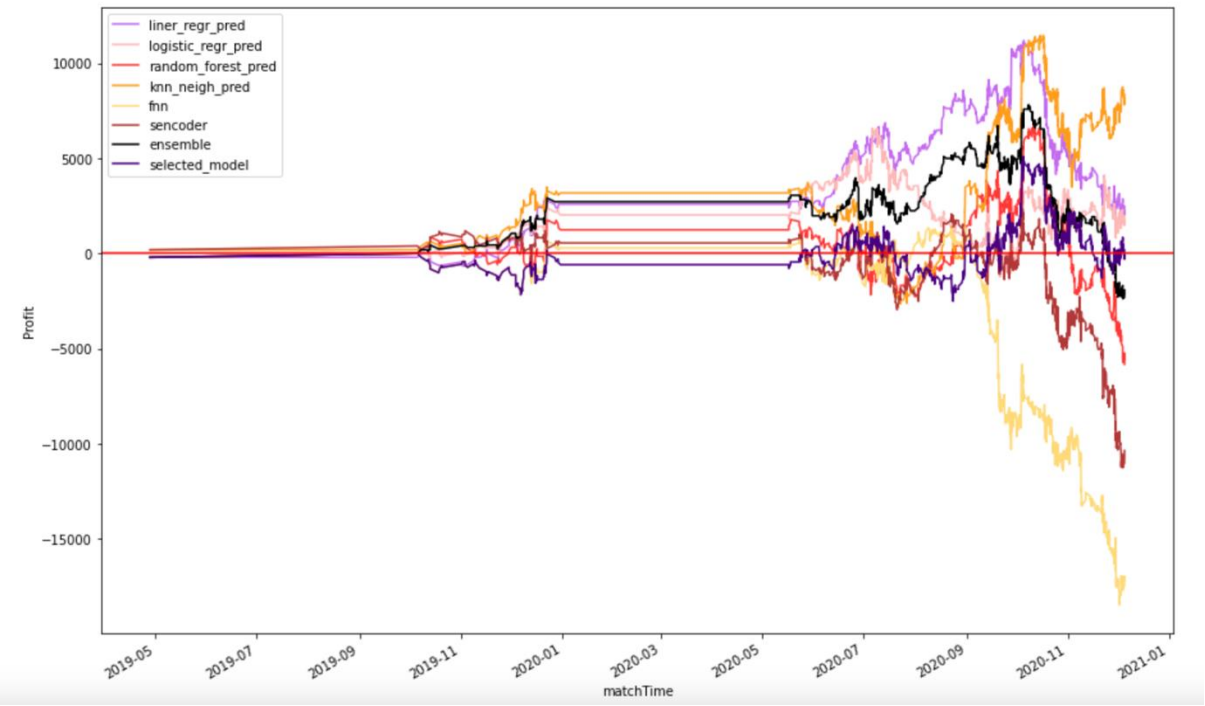

Figure 97. Bootstrap sample 3 performance

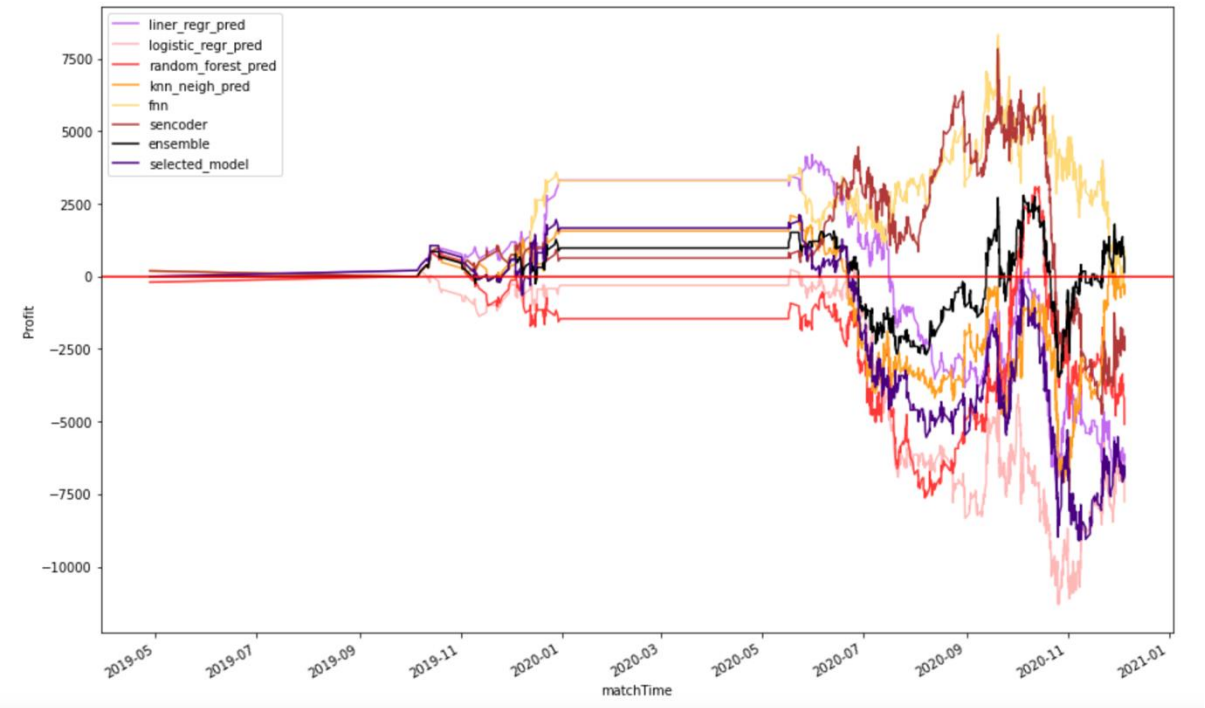

Figure 98. Bootstrap sample 4 performance

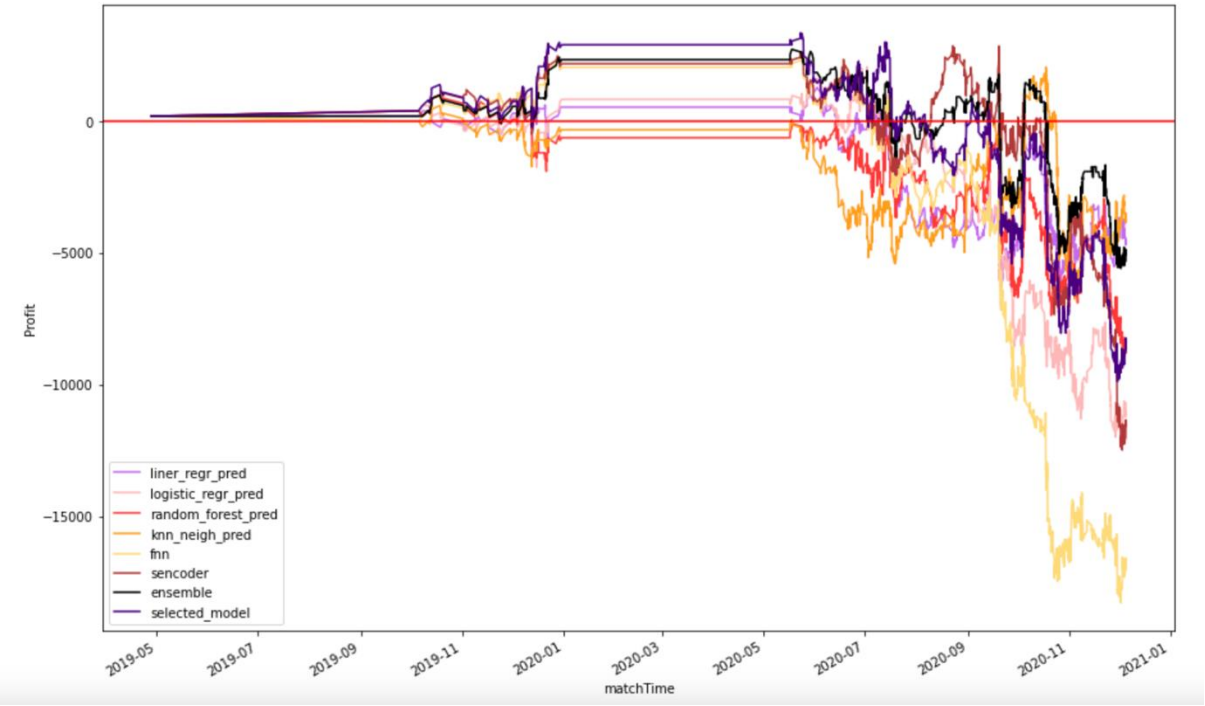

Figure 99. Bootstrap sample 5 performance

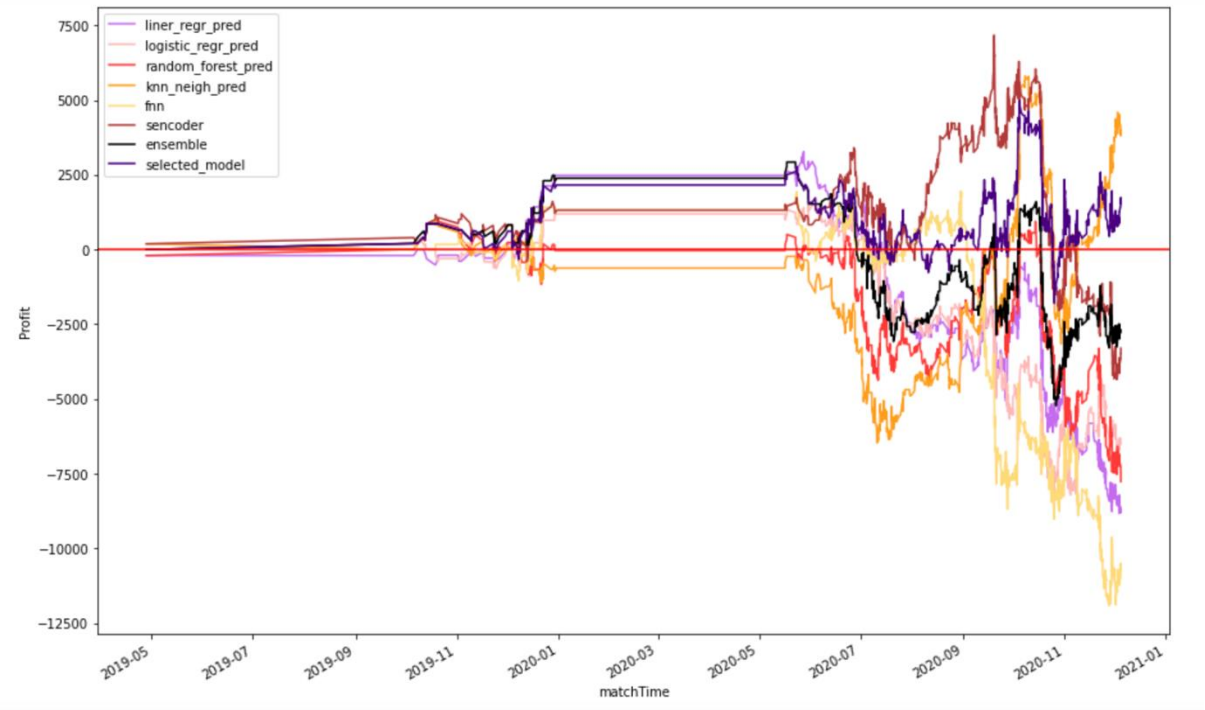

Figure 100. Bootstrap Aggregating performance

7.5GAN

In this section, we will discuss and analysis the results we obtain from GAN, namely CTGAN, CTGAN with ***bookie_expected*** and CTGAN with ***test_like***. Same as bootstrap, 40% of new data is generated. The features used for modelling are first odds with HAD and all-league model training strategy. Figure 101 shows the performance of CTGAN. The FNN used in this setting is the FNN 2.0 (the re-work one), it obtains a very well profit at the end. However, the prediction made from October to December is relatively unchanged. On the other hands, other models performance are not very promising, they earn a negative profit at the end. After removing the last odds features, the fluctuate within a curve is reduced, in other words, the dataset is less noisy.

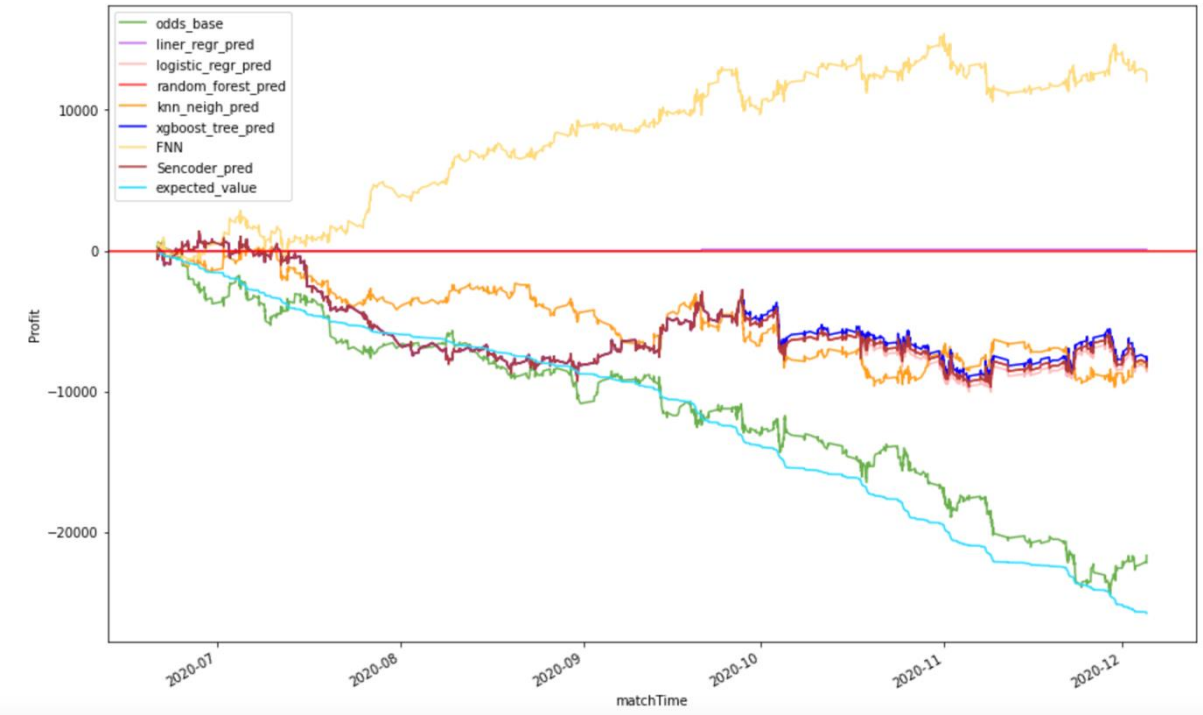

Figure 101. CTGAN result

Now we will talk about the approach of modelling the bookie prediction (Bookie Prediction). Table illustrates the statistic of the three newly added columns. The mean of ***bookie_expected*** and ***customer_expected*** are around 37%, at the first glance, this imply bookie and customer predict incorrectly most of the time. However, the standard deviation is around 50%, this imply both newly added columns are actually very noisy and might not contain specific usually information. Indeed, according to Figure 102, the performance for most models remain the same but FNN performance is a lot worser than before. One of the explanation is that FNN is not as robust to noise as other models are.

	bookie_expected	customer_expected	bookie_customer_expected
count	8404.00	8404.00	8404.00
mean	0.38	0.37	0.70
std	0.48	0.48	0.46
min	0.00	0.00	0.00
25%	0.00	0.00	0.00
50%	0.00	0.00	1.00
75%	1.00	1.00	1.00
max	1.00	1.00	1.00

Table 17. Statistics for bookie and customer prediction

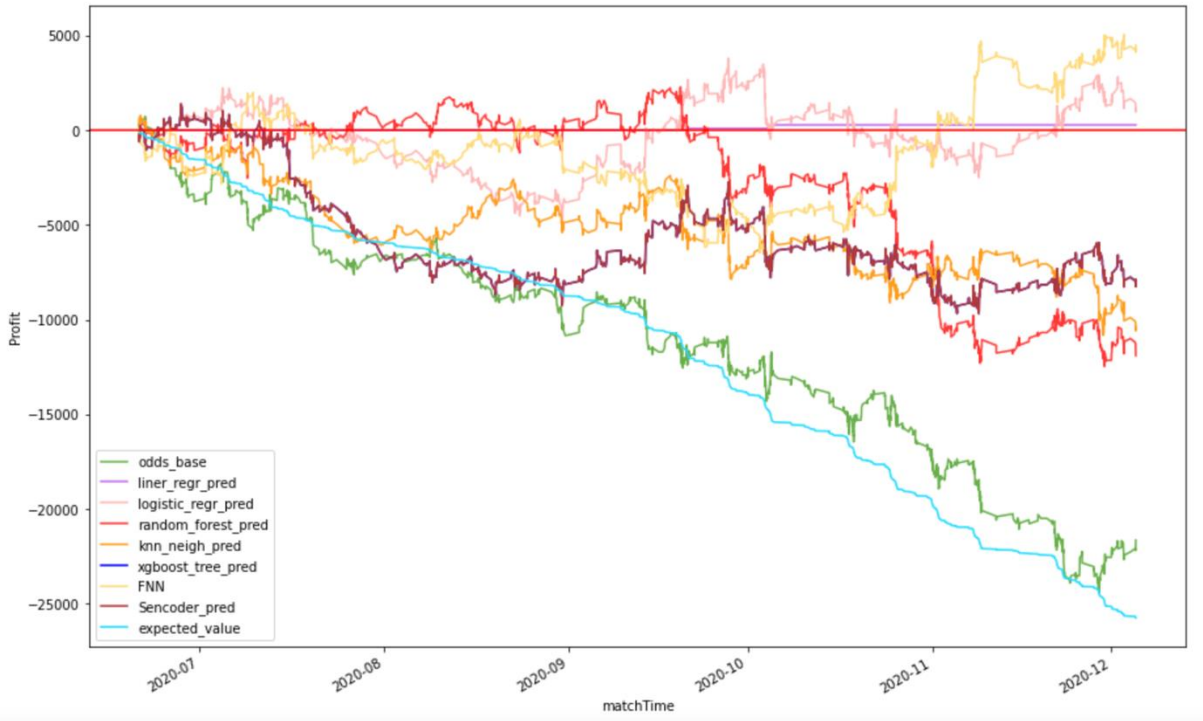

Figure 102. CTGAN with bookie_expected result

Finally, we will talk about the test-like methods (Test-like Data). The pipeline setting is same as above, using first odds with HAD and all-league training strategy. Figure 103 shows the performance under this approach, the result is better than the previous two approaches. Two neural network models (FNN 2.0 and sencoder 2.0) achieved a positive profit at the end. The logistic models has a flying colour results. This infer our hypothesis is correct that the distribution (information) of the data vary over time.

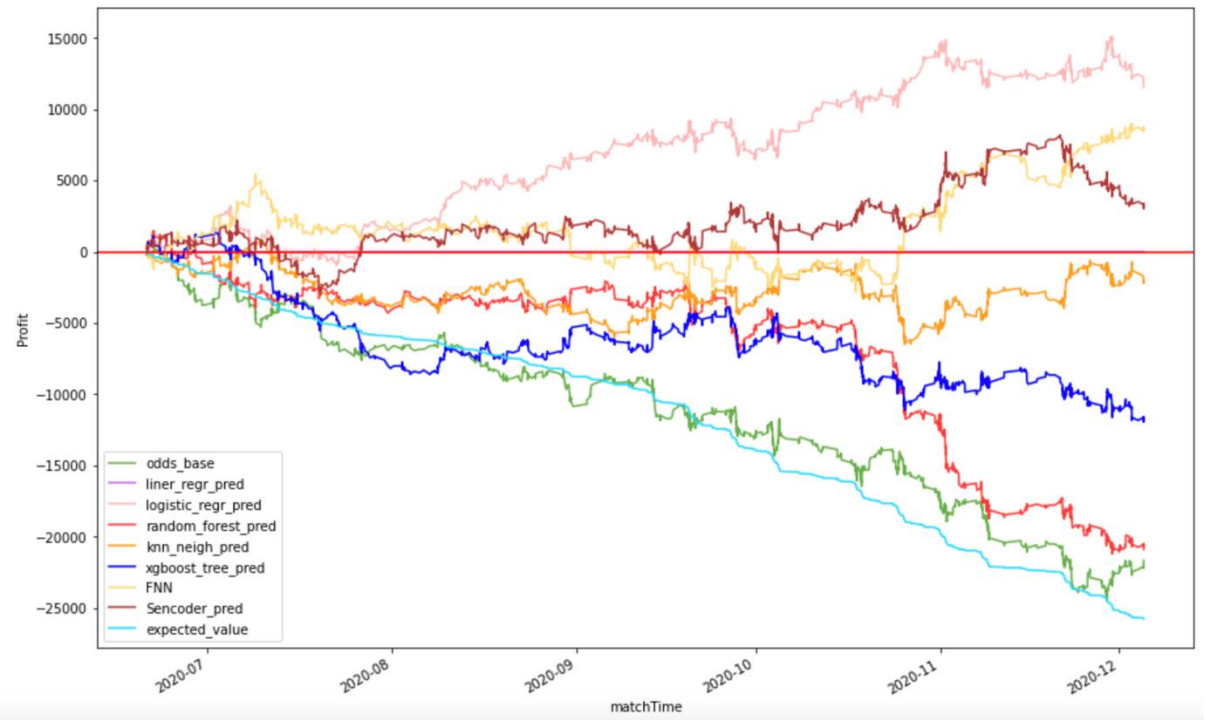

Figure 103. CTGAN with test-like data result

7.6 Model Comparison

In this section, we present all our best model data. During the myriad of experiments we did in the entire project, we have been selecting some model as a candidate of the final best model. The final best model is to combine all the strength of these candidates, in particular, to select the best model for each league. This will be introduced in Section 8.

Table shows the profit obtained from the models that use first odds features and train by all-league strategy, eventually, evaluated in an all-league basis.

	FNN_v2.0	Sencoder_v2.0	LSTM_v2.0	Logistic_v2.0	Logistic_v2.1	FNN_Bootstrap	XGBoost	CNN
Phrase 2	21818	12506	9877	12203	18430	8291	271	7934
Phrase 1	2000	-14500	3000	1250	Nan	Nan	Nan	Nan
	FNN_v1.0	Sencoder_v1.0	LSTM_v1.0	Logistic_v1.0				

Table 18. Model comparison

8. Real-time Prediction

8.1 Metric

For the model selection, lots of metric can be chosen. Among those metrics, the most suitable should be the accuracy and the *auto* (defined in section 7.2) or. In the preliminary evaluation, *auto* seems to be most suitable because it can ensure the model can earn the most in the test data. Then, we take a look into the winning odds.

matchTime	homeTeamName	awayTeamName	leagueName	Profit	y_test	y_pred	hkjc_hdc_home_last	hkjc_hdc_away_last
2020-06-21 20:00:00	切爾達	艾拉維斯	西甲	252	2.0	2	1.26	0.66
2020-06-21 21:00:00	紐卡素	錫菲聯	英超	156	2.0	1	0.78	1.08
2020-06-21 21:30:00	海登海默	漢堡	德乙	168	2.0	2	0.84	1.00
	聖保利	雷根斯堡	德乙	102	-1.0	-2	0.82	1.02
2020-06-21 22:30:00	莫斯科火車頭	奧倫堡	俄超	86	1.0	1	0.86	0.96

Table 19. Odds that having correct prediction

According to profit table above, it is clear that the odds that having correct prediction are different. In other words, there is not specific odds can have a favour in prediction. Since odds are random in the soccer matches, using *auto* as the metric can ensure that the models return maximum profit in test data but not in the future data.

On the other hand, accuracy can ensure the selected models performing well in the generally case. Although their performance may not be the best one, they should be the most stable one. Therefore, the real-time prediction program will choose the model with the highest accuracy under the specific league.

8.2 Best Model

Based on the reason stated in section 8.1, the metric of best model selection will be accuracy. Thus, all the prediction results towards the testing dataset are listed.

8.2.1 Prediction Result

In the follow figures, lists of predicted result of all models are shown. In the first column of each figure, ***pos***, ***neg*** and ***neu*** mean the number of matches that winning money, losing money and returning original money respectively.

1) FNN_v2.0 (Feedforward Neural Network)

	德甲	英冠	西甲	意甲	挪超	日職乙	瑞典超	德乙	俄超	英超	葡超	日職聯	澳洲甲	美職業	歐霸盃	歐冠盃	巴西甲	法乙	法甲	荷甲
pos	43	52	81	61	32	81	36	34	48	74	24	107	6	68	28	39	51	16	52	23
neg	29	39	53	71	29	58	30	36	33	63	13	77	7	46	24	39	32	11	32	23
neu	5	18	9	9	2	10	5	3	6	11	0	10	1	5	2	4	4	3	3	4

Table 20. Predicted result of FNN_v2.0 for test data

2) Sencoder

	德甲	英冠	俄超	挪超	英超	西甲	意甲	瑞典超	德乙	日職乙	葡超	日職聯	澳洲甲	美職業	歐冠盃	巴西甲	法乙	法甲	荷甲
pos	42	47	42	35	77	93	64	32	31	79	19	90	7	57	23	44	12	46	24
neg	32	43	31	30	69	45	72	33	34	51	16	74	6	44	22	35	14	35	20
neu	5	17	7	2	12	10	9	5	3	11	0	10	1	4	1	4	3	3	4

Table 21. Predicted result of Sencoder for test data

3) FNN Bootstrap

	澳洲甲	挪超	瑞典超	英超	法乙	德乙	英冠	西甲	意甲	葡超	德甲	法甲	美職業	日職乙	巴西甲	日職聯	俄超	荷甲	歐冠盃
pos	12.0	16.0	14.0	33.0	9.0	32.0	27.0	44.0	39.0	16.0	27.0	20.0	14.0	27.0	15.0	21.0	12.0	19.0	12.0
neg	8.0	9.0	7.0	29.0	7.0	14.0	11.0	22.0	17.0	7.0	19.0	27.0	16.0	17.0	10.0	20.0	7.0	7.0	8.0
neu	7.0	4.0	8.0	19.0	8.0	12.0	21.0	11.0	10.0	0.0	13.0	5.0	11.0	12.0	9.0	13.0	8.0	4.0	5.0

Table 22. Predicted result of FNN Bootstrap for test data

4) XGboost

	德甲	英冠	俄超	挪超	英超	西甲	意甲	瑞典超	德乙	日職乙	葡超	日職聯	澳洲甲	美職業	歐冠盃	巴西甲	法乙	法甲	荷甲
pos	39.0	42.0	40.0	43.0	72.0	64.0	54.0	30.0	38.0	55.0	15.0	83.0	3.0	51.0	27.0	47.0	10.0	43.0	17.0
neg	34.0	42.0	30.0	19.0	69.0	67.0	75.0	32.0	25.0	67.0	19.0	72.0	7.0	41.0	16.0	28.0	12.0	35.0	27.0
neu	6.0	23.0	10.0	5.0	17.0	17.0	16.0	8.0	5.0	19.0	1.0	19.0	4.0	13.0	3.0	8.0	7.0	6.0	4.0

Table 23. Predicted result of XGboost for test data

5) LogisticsRegression_v2.0

	德甲	英冠	俄超	挪超	英超	西甲	意甲	瑞典超	德乙	日職乙	葡超	日職聯	澳洲甲	美職業	歐冠盃	巴西甲	法乙	法甲	荷甲
pos	40.0	44.0	45.0	41.0	78.0	89.0	62.0	33.0	34.0	71.0	23.0	94.0	8.0	55.0	26.0	39.0	11.0	42.0	20.0
neg	34.0	46.0	28.0	24.0	68.0	49.0	74.0	32.0	31.0	59.0	12.0	70.0	5.0	46.0	19.0	40.0	15.0	39.0	24.0
neu	5.0	17.0	7.0	2.0	12.0	10.0	9.0	5.0	3.0	11.0	0.0	10.0	1.0	4.0	1.0	4.0	3.0	3.0	4.0

Table 24. Predicted result of LogisticsRegression_v2.0 for test data

6) LogisticsRegression_V2.1

	挪超	英超	意甲	西甲	俄超	英冠	日職乙	瑞典超	德甲	德乙	...	日職聯	澳洲甲	美職業	歐霸盃	歐冠盃	巴西甲	法乙	法甲	荷乙	荷甲
pos	37	71	59	89	46	53	75	37	46	41	...	111	8	61	28	42	46	13	43	16	25
neg	27	75	75	52	36	40	65	30	28	29	...	73	5	53	24	36	37	14	41	21	21
neu	2	11	9	9	7	19	11	5	5	3	...	10	1	5	2	4	4	3	3	1	4

Table 25. Predicted result of LogisticsRegression_v2.1 for test data

7) CNN

	德甲	英冠	俄超	挪超	英超	西甲	意甲	瑞典超	德乙	日職乙	葡超	日職聯	澳洲甲	美職業	歐冠盃	巴西甲	法乙	法甲	荷甲
pos	40	38	39	39	80	77	66	29	29	54	21	85	4	51	23	44	16	42	21
neg	33	42	30	24	56	56	63	33	32	66	14	73	6	43	22	32	7	36	23
neu	6	27	11	4	22	15	16	8	7	21	0	16	4	11	1	7	6	6	4

Table 26. Predicted result of CNN for test data

8) LSTM_V2.0 (Long Short-Term Memory)

	德甲	英冠	西甲	意甲	挪超	日職乙	瑞典超	德乙	俄超	英超	葡超	日職聯	澳洲甲	美職業	歐霸盃	歐冠盃	巴西甲	法乙	法甲	荷甲
pos	47	62	84	85	32	84	38	41	39	62	27	103	8	74	26	44	54	12	40	28
neg	27	29	50	47	29	55	28	29	42	75	10	81	5	40	26	34	29	15	44	18
neu	5	18	9	9	2	10	5	3	6	11	0	10	1	5	2	4	4	3	3	4

Table 27. Predicted result of LSTM for test data

8.2.1 Overall Accuracy

According to the predicted result of all the models shown as above, the accuracy for a model will be ***pos*** / (***pos*** + ***neg***).

	FNN_v2.0	Sencoder	FNN Bootstrap	XGboost	Logistics_v2.0	Logistics_v2.1	CNN	LSTM_v2.0
德甲	0.60	0.57	0.59	0.53	0.54	0.62	0.55	0.64
英冠	0.57	0.52	0.71	0.50	0.49	0.57	0.48	0.68
西甲	0.60	0.67	0.67	0.49	0.64	0.63	0.58	0.63
意甲	0.46	0.47	0.70	0.42	0.46	0.44	0.51	0.64
挪超	0.52	0.54	0.64	0.69	0.63	0.58	0.62	0.52
日職乙	0.58	0.61	0.61	0.45	0.55	0.54	0.45	0.60
瑞典超	0.55	0.49	0.67	0.48	0.51	0.55	0.47	0.58
德乙	0.49	0.48	0.70	0.60	0.52	0.59	0.48	0.59
俄超	0.59	0.58	0.63	0.57	0.62	0.56	0.57	0.48
英超	0.54	0.53	0.53	0.51	0.53	0.49	0.59	0.45
葡超	0.65	0.54	0.70	0.44	0.66	0.70	0.60	0.73
日職聯	0.58	0.55	0.51	0.54	0.57	0.60	0.54	0.56
澳洲甲	0.46	0.54	0.60	0.30	0.62	0.62	0.40	0.62
美職業	0.60	0.56	0.47	0.55	0.54	0.54	0.54	0.65
歐冠盃	0.50	0.51	0.60	0.63	0.58	0.54	0.51	0.56
巴西甲	0.61	0.56	0.60	0.63	0.49	0.55	0.58	0.65
法乙	0.59	0.46	0.56	0.45	0.42	0.48	0.70	0.44
法甲	0.62	0.57	0.43	0.55	0.52	0.51	0.54	0.48
荷甲	0.50	0.55	0.73	0.39	0.45	0.54	0.48	0.61

Table 28. Accuracy of all models grouped by league

Recall from Section 7.6 that we want to create a final best model such that this model combines all the strength of each candidates we selected during the project. The strategy to create such model is to pick the model with the highest accuracy by league and the selected model will be responsible for predicting the corresponding league. Eventually, we achieve a model that is capable for predicting each league. Table 29 depicts each league and the model that will be participating for prediction.

	Model	Accuracy
德甲	LSTM_v2.0	0.635135
英冠	FNN Bootstrap	0.710526
西甲	Sencoder	0.673913
意甲	FNN Bootstrap	0.696429
挪超	XGboost	0.693548
日職乙	FNN Bootstrap	0.613636
瑞典超	FNN Bootstrap	0.666667
德乙	FNN Bootstrap	0.695652
俄超	FNN Bootstrap	0.631579
英超	CNN	0.588235
葡超	LSTM_v2.0	0.72973
日職聯	LogisticsRegression_v2.1	0.61413
澳洲甲	LSTM_v2.0	0.615385
美職業	LSTM_v2.0	0.649123
歐霸盃	LogisticsRegression_v2.1	0.538462
歐冠盃	XGboost	0.627907
巴西甲	LSTM_v2.0	0.650602
法乙	CNN	0.695652
法甲	FNN_v2.0	0.619048
荷甲	FNN Bootstrap	0.730769

Table 29. Best model for each league with the accuracy

8.3 Process

Step 1:

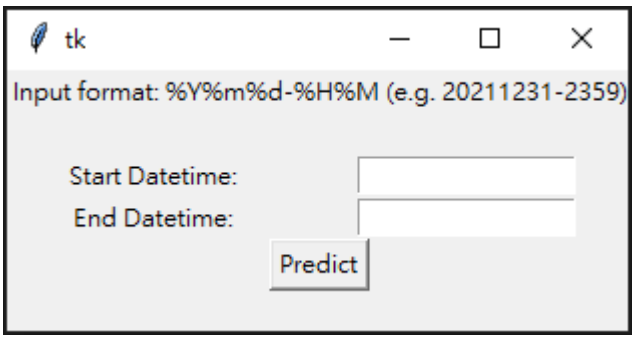

Figure 104. Main page of the real-time prediction program

At the beginning, the user should enter the time range for the soccer matches which are needed to be predicted. If it is null, the program will select all the matches today in which Hong Kong Jockey Club defines today in GMT-4.

Tuesday Matches												
TUE 1	Newcastle Jets vs Perth Glory		13/04 17:05			2.30	3.55	2.47				
TUE 2	Huddersfield vs Bournemouth		14/04 00:30			4.15	3.40	1.70				
TUE 3	Sheff Wednesday vs Swansea		14/04 01:00			2.78	2.95	2.35				
TUE 4	Crewe vs Portsmouth		14/04 01:00			2.80	3.15	2.23				
TUE 6	AFC Wimbledon vs Ipswich		14/04 01:30			2.90	2.92	2.30				
TUE 7	Rotherham vs QPR		14/04 02:00			2.33	3.20	2.62				
TUE 8	Blackpool vs Accrington Stanley		14/04 02:00			1.70	3.40	4.15				
TUE 13	Paris Saint Germain vs Bayern Munich		14/04 03:00		643	2.62	3.90	2.10				2
TUE 14	Chelsea vs Porto		14/04 03:00		644	1.74	3.35	4.15				2
Wednesday Matches												

Figure 105.All soccer matches of Hong Kong Jockey Club in in 13/04/2021

Thus, there are matches that will start on 14/04/2021 shown on 13/04/2021 match list.

Step 2:

After entering the time range for the desired matches and pressing the predict button shown in Figure 104. The program will start scrapy to crawl and get information from LeiSu[5], win007[6], HKJC[7] and TransferMarkt[8].

	AUS BNE Reserves League	16:30	未	Centenary Stormers Reserves	-	The Gap FC Reserve						
	A FFA Cup	17:00	未	Richmond	-	Dandenong City SC		0.850 0.800	-1.5 3.75	0.950 1.000		
	AUS VWC	17:00	未	Preston Lions Women	-	Boroondara Eagles Women		0.900 0.850	0.25 3.5	0.900 0.950		
	A FFA Cup	17:00	未	PCYC Parramatta Eagles	-	Dulwich Hill SC						
	AUS A-League	17:05	未	**Newcastle Jets**	-	**Perth Glory**		0.860 0.930	0 2.75	1.040 0.930		
	A FFA Cup	17:15	未	Hills Brumbies	-	Blacktown Spartans		0.800 0.850	0.0 3.0	1.000 0.950		
	Aus BCC	17:30	未	Kangaroo Point Rovers FC	-	Oxley United		0.880 0.900	0.5 3.5	0.930 0.900		
	A FFA Cup	17:30	未	Tarragindi Tigers	-	Capalaba Bulldogs						
	A FFA Cup	17:30	未	Pine Hills	-	Caloundra		0.970 0.900	0.0 3.5	0.820 0.900		

Figure 106.Future soccer match list in LeiSu

選	04月13日	時間	狀態	比賽球隊	比分	比賽球隊	角/半	動畫	Crown指數			走	數據
	哈薩超	16:00		波格特約	-	阿克圖比	-		1.02 0.95	平手 2/2.5	0.80 0.85		析 亞 大 歐
	V推荐资讯				球吧资讯						V猜球资讯		
	女奧亞預	16:00		中國女足	-	南韓女足	-		0.93 0.87	半/一 2	0.89 0.93		析 亞 大 歐
	澳洲甲	17:05		紐卡素噴射機(中)	-	珀斯光輝	- -		0.85 0.95	平手 2.5/3	1.05 0.93		析 亞 大 歐
	高清直播												
	國際友誼	17:30		斯洛文尼亞女足	-	斯洛伐克女足	-		0.81 0.88	平/半 2.5	1.01 0.92		析 亞 大 歐
	哈薩超	18:00		卡拉簡迪	-	安察利克	- -						析 亞 大 歐
	土甲	18:30		土茲拉士邦	-	艾斯基沙希	- -		0.82 0.96	兩球 3/3.5	1.00 0.84		析 亞 大 歐
	澳布超	18:30		史托姆斯世紀	-	甲普	- -						析 亞 大 歐
	阿美超	19:00		羅瑞瓦納佐爾	-	阿拉特阿美尼亞	-						析 亞 大 歐
	斯伐盃	19:30		FK柯西斯	-	特倫辛	- -						析 亞 大 歐
	國際友誼	20:00		匈牙利女足	-	波斯尼亞女足	-						析 亞 大 歐
	國際友誼	20:00		北馬其頓U19	-	黑山U19	-						析 亞 大 歐
	捷甲	20:30		帕爾杜比斯	-	史洛特	- -		0.85 0.98	*半球 2/2.5	1.03 0.88		析 亞 大 歐

Figure 107.Future soccer match list in win007

[5] https://live.leisu.com/
[6] http://www.win007.com/
[7] https://bet.hkjc.com/football/index.aspx?lang=en
[8] https://www.transfermarkt.co.uk/

Tuesday Matches					
TUE 1	Newcastle Jets[0] vs Perth Glory[0]	13/04 17:05	1.84	2.00
TUE 2	Huddersfield[+0.5/+1] vs Bournemouth[-0.5/-1]	14/04 00:30	1.76	2.02
TUE 3	Sheff Wednesday[0] vs Swansea[0]	14/04 01:00	2.07	1.72
TUE 7	Rotherham[0] vs QPR[0]	14/04 02:00	1.77	2.01
TUE 13	Paris Saint Germain[0/+0.5] vs Bayern Munich[0/-0.5]	14/04 03:00	1.86	1.98
TUE 14	Chelsea[-0.5/-1] vs Porto[+0.5/+1]	14/04 03:00	2.13	1.74

Figure 108.Future soccer match list for handicap in HKJC

Date	Home team		Away team
Quarter-Finals 2nd leg			
Tue Apr 13, 2021	Chelsea	8:00 PM	FC Porto
	Paris SG	8:00 PM	FC Bayern
Wed Apr 14, 2021	Bor. Dortmund	8:00 PM	Man City
	Liverpool	8:00 PM	Real Madrid

Figure 109.Future soccer match list in TransferMarkt

Since the win007 will block the IP address if the same IP address requires information too much time within 10 minutes, the original scrapy program will take a long time. Thus, we added a random Proxy Middleware into the scrapy. When the scrapy is banned by the website, it will random draw an available IP address and port from https://www.sslproxies.org/ as the proxy. Therefore, the speed of the scrapy in real-time prediction program is improved a lot.

IP Address	Port	Code	Country	Anonymity	Google	Https	Last Checked
37.120.192.154	8080	NL	Netherlands	anonymous	no	yes	3 seconds ago
201.45.163.114	80	BR	Brazil	elite proxy	no	yes	3 seconds ago
103.240.77.98	30093	IN	India	elite proxy	no	yes	3 seconds ago
208.80.28.208	8080	US	United States	elite proxy	no	yes	3 seconds ago
169.57.1.85	80	MX	Mexico	elite proxy	no	yes	3 seconds ago
147.135.195.42	8020	FR	France	elite proxy	no	yes	3 seconds ago
157.90.4.20	8080	DE	Germany	elite proxy	no	yes	3 seconds ago
186.4.186.36	3128	EC	Ecuador	elite proxy	no	yes	3 seconds ago

Figure 110. Example of the available IP address and corresponding port in sslproxies.com

After all information of desired soccer matches is got, the program will merge the information from different website so that the information can divided by each match. Then, the program will select the soccer matches that HKJC offered handicap odds and pass all related information to the prediction program.

Step 3:

When the prediction program received all the information from each match, it will select the best model according to the league of the match. Thus, it will predict the result for the match in range [2, 1, 0, -1, -2].

Predicted Result	Meaning
2	Buy home team and will win all the money
1	Buy home team and will win half of the money
0	Return the original money (Prefer not to bet)
-1	Buy away team and will win half of the money
-2	Buy away team and will win all the money

Table 30. Meaning of the predicted result of real-time prediction program

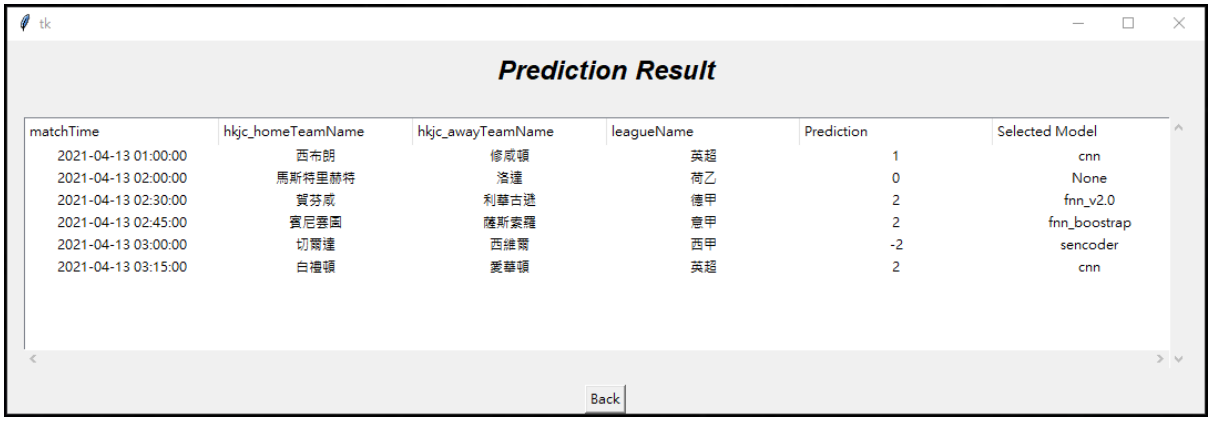

Figure 111.Predicted result of the real-time prediction program for the matches on 12/4/2021 in HKJC

If the selected model is “None”, it means that there is not a suitable model for that league since all the models are trained with the leagues that containing matches larger than 150 or 200 only. In other word, if the selected model is “None”, that league contains fewer than 150 matches in training dataset.

9. Limitation

9.1 The Size of Dataset

There are some caveats to the model results: the number of records in each league might be small due to the fact that not many matches are being played in each season. Thus, although data from several years are collected, the number of matches in each league is also small. Therefore, each record in validation dataset will affect the profit greatly. We tried to tackle it via data augmentation technique, for instance, bootstrap sampling and GAN. Eventually, one of the model from bootstrap is selected to be a candidate for the final best model. However, solving this problem via data augmentation might not be promising and it is very limited. In collusion, if there are more data it will be beneficial for the entire project.

9.2 Merging Player Information

Since different websites use different English names for the players, it is tough to merge the statistics of players from different websites.

Figure 112. Example of a player name in FIFA and player name in Win007

From the example above, although these two names belong to the same player, they are inconsistent. Unlike merging team names, since the number of soccer teams is much fewer, it is possible to build a mapping table manually. Currently, when the player names from different websites are not same, we have proposed different solutions to tackle this problem like search online and set methods. Although these methods helped in the process of merging players, there are still lots of records merged unsuccessfully.

9.3 Not All Features Are Tested

Due to not enough research support, some features are not used. For example, Moreover, the feature “lineup” is not used because we cannot find a plausible way to turn it into a useful feature for modelling. It is because lineup is not a numeric number and not easy to compare. Second, the weather-related information is not considered because of only less than 20% of data containing such information due to the limitation of the source.

9.1 Clustering Method

In the analysis result (Table 29) of our real-time prediction program, it shows that models will have different performance in different league. Thus, it proves that the type of league will influence the training process of the models. Recalling section 5.6, we constructed a cluster-then-predict model to reduce the influence of league. However, its performance is not good. The reason may be that we used Euclidean distance as our clustering method which can help grouping the leagues with linearly relationship only. Thus, a more suitable clustering method should be found in further research.

10. Contributions to The Project

At the beginning process, I was responsible for the scraping of HKJC data, the team score of FIFA and the player score of FIFA. Also, I tried to do some feature engineering on the data such as HKJC time relative odds, the recent performance score of the team (Past ten record) and one-hot encoding. The most difficult part was to merging all the match from different websites as websites would use different team name and player name, for example, “費雷堡” in HKJC called “弗賴堡” in Win007. For merging the teams, I designed a merging table in which I collected all the team names in HKJC and found the corresponding team name in Win007 one by one. For merging the player, I designed a function to find the serval names of the player from the TransferMarkt if the player could not be merged.

After all the data was collected, I used R to analyze the data to ensure that our topic was possible. First, the Kruskal-Wallis H Test was used to prove that there was an association between the initialized odds from different bookmakers and the handicap results so the project could focus on the odds data. Then, the Pairwise Wilcoxon Rank Sum Test was used to prove that there were significant differences between the 5 types of handicap results. Thus, this project was possible.

After the data analysis, the next process was feature engineering. Since the odds were changed from time to time, I designed the HKJC time relative odds to replace the list of odds which were very messy and complex to hand. Also, the past ten records of the home team, away team and encounter were converted to a number according to the corresponding net handicap results. In other words, the recent performance of a team was quantified. Also, the one-hot league was used as it proved that the prediction performance will improve.

When the feature engineering was done, feature selection was needed to find the suitable features for the best model. Thus, I used forward and backward AIC (Akaike Information Criterion) to select the features that had a linear relationship with the handicap result. Also, the forward and backward AIC for logistic regression was done too. Overall, the result showed that the most suitable features for the linear and logistic models were odds-related features only.

Hence, the LSTM model, linear model, logistic regression model, XGBoost model and stacking ensemble model were designed to predict the handicap result via selected features. Although the hyperparameter tuning was done by grid search, both models could not get a profit larger than $10000. Since poor features would always train a poor model, I started to reflect on the features that we chose.

In order to improve the data, I started to read lots of papers relating to football prediction. Then, I discovered lots of possible features from different past research, such as attacking strength of a team [44], the number of recent corners kicked [45], player score [46], home field identifier [47] and attributes of players [48]. All these features were similar to our original features but they were removed in feature selection because odds included that information.

After that, a deeper analysis of handicap odds was needed. According to a research that used the odds feature [47], they used the HAD odds and handicap odds to get a quite good performance. Thus, it was thus doubted that there was an association between the HAD odds and handicap odds so analysis between them was needed. After studying several papers, I found that the HAD odds can be used to calculate the handicap odds based on probability without the betting margin. Thus, a function that transformed the HAD odds to the calculated handicap odds was added and the performances of most of the models were improved a lot.

Afterward, I created the real-time prediction program which was not simply applying the trained models. First, it added the random proxy function because real-time prediction was needed to crawl the features quickly. However, our original scrapy is quite slow as the IP addressed would not be blocked by the websites. Thus, the added random proxy function will change the IP address and corresponding port frequently so the spend will improved a lot. Moreover, the real-time prediction program would select the best model with different normalization methods corresponding to different leagues.

11. Conclusion and Future Work

After a year of analysis, it is proved it is possible to predict the handicap result using the odds related features, especially the first odds. Although neural network is powerful, its prediction does not always be better than the statistical models such as logistic regression and XGBoost in our project. One of the probable reasons is that the architectures of our neural network models are not optimal. Although we used random search to tunning the parameters of the model structure, it does not guarantee the best global optimal parameter set can be found. Moreover, we found that some models did a great job in some specific league. In other words, model has their preferred league, so we added a function in the real-time program to find the suitable model for the target match according to its league.

Although the models can get a positive return at the end, the payoff is only 1/4 of the maximum profit. Thus, there are still a lot of improvement needed. The further work will be trying some more complex hidden layers, exploring more possible features from the past research and finding a more accurate method to merge similar league instead of our cluster-then-predict model. The future target program should get a return which more than 2/4 of the maximum profit.

12. References

[1] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, , "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825--2830, 2011.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, L, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/.

[3] W. K. Wong and C. Y. Wong, "Exploiting Betting Odds using Machine Learning," The Chinese University of Hong Kong, 2020.

[4] E. O. Thorp, L. C. MacLean and W. T. Ziemba, World Scientific Handbook in Financial Economics Series: Kelly Capital Growth Investment Criterion, World Scientific Publishing Company, 2011, pp. 509-523.

[5] E. Strumbelj and R. M. Sikonja, "Online bookmakers' odds as forecasts: The case of Europeansoccer leagues," *International Journal of Forecasting*, vol. 26, no. 3, pp. 482-488, 7-9 2010.

[6] J. Goddard, Forecasting football results and the efficiency of fixed-odds betting, John Wiley & Sons, Ltd., 2004.

[7] M. Kumar and A. Deshpande, Artificial Intelligence for Big Data, Packt Publishing Ltd, 2018, pp. 171-172.

[8] O. Institute, Pearson's Correlation Coefficient and the Consolidated State Performance Report (2012-2013): High School Science Proficiency Across the U.S. States, Sage Publications Limited, 2015.

[9] E. P. McKight and J. Najab, "Kruskal-Wallis Test".

[10] L. M. Surhone, M. T. Timpledon and S. F. Marseken, Wilcoxon Signed-Rank Test, VDM Publishing, 2010.

[11] J. K. DIXON, "Pattern Recognition with Partly Missing Data," 1979.

[12] E. Alpaydin, "Dimensionality Reduction," in *Introduction To Machine Learning 3Rd Edition*, MIT Press, 2015, p. 120.

[13] A. Ng, "Sparse autoencoder," [Online]. Available: https://web.stanford.edu/class/cs294a/sparseAutoencoder_2011new.pdf.

[14] Heung-Il Suk, Seong-Whan Lee, Dinggang Shen, "Latent feature representation with stacked auto-encoder for AD/MCI diagnosis," 2013.

[15] "Stacked shallow autoencoders vs. deep autoencoders," [Online]. Available: https://stats.stackexchange.com/questions/393572/stacked-shallow-autoencoders-vs-deep-autoencoders. [Accessed 20 11 2020].

[16] J. Brownlee, "A Gentle Introduction to the Bootstrap Method," [Online]. Available: https://machinelearningmastery.com/a-gentle-introduction-to-the-bootstrap-method/.

[17] Kjell Johnson, Max Kuhn, "The Bootstrap," in *Applied Predictive Modeling*, p. 72.

[18] I. J. Goodfellow, "Generative Adversarial Nets," 2014.

[19] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, Kalyan Veeramachaneni, “Modeling Tabular Data using Conditional GAN,” 2019.

[20] “GANs for tabular data,” [Online]. Available: https://towardsdatascience.com/review-of-gans-for-tabular-data-a30a2199342. [Accessed 13 4 2021].

[21] Smarkets, “How to calculate expected value in betting,” 15 7 2016. [Online]. Available: https://help.smarkets.com/hc/en-gb/articles/214554985-How-to-calculate-expected-value-in-betting.

[22] X. Song, A. Mitnitskib, J. Coxb and K. Rockwood, “Comparison of Machine Learning Techniques with Classical Statistical Models in Predicting Health Outcomes,” IOS Press, Amsterdam, 2004.

[23] “Gini Index vs Information Entropy,” [Online]. Available: https://towardsdatascience.com/gini-index-vs-information-entropy-7a7e4fed3fcb. [Accessed 20 11 2020].

[24] J. H. Friedman, “Stochastic gradient boosting,” Elsevier Science B.V., 2002.

[25] E. Alpaydin, “k-Nearest Neighbor Estimator,” in *Introduction to Machine Learning 3Rd Edition*, 2015, p. 190.

[26] Ian Goodfellow and Yoshua Bengio and Aaron Courville, “Deep Feedforward Networks,” in *Deep Learning*, MIT Press, 2016, p. 164.

[27] “Why are neural networks becoming deeper, but not wider?,” [Online]. Available: https://stats.stackexchange.com/questions/222883/why-are-neural-networks-becoming-deeper-but-not-wider. [Accessed 15 4 2021].

[28] “Introduction to Early Stopping: an effective tool to regularize neural nets,” [Online]. Available: https://towardsdatascience.com/early-stopping-a-cool-strategy-to-regularize-neural-networks-bfdeca6d722e. [Accessed 19 11 2020].

[29] Nitish Srivastava and Geoffrey Hinton and Alex Krizhevsky and Ilya Sutskever and Ruslan Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” 2014.

[30] Sergey Ioffe, Christian Szegedy, “Batch Normalization: Accelerating Deep Network Training b y Reducing Internal Covariate Shift,” 2015.

[31] Z. Lin, Recurrent Neural Network Models of Human Mobility, University of California, Berkeley, 2018.

[32] Liu Guifang, Bao Huaiqian, Han Baokun, “A Stacked Autoencoder-Based Deep Neural Network for Achieving Gearbox Fault Diagnosis,” 2018.

[33] “Transfer Learning,” [Online]. Available: https://keras.io/guides/transfer_learning/. [Accessed 15 4 2020].

[34] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”.

[35] “CS231n Convolutional Neural Networks for Visual Recognition Course Website, Data Preprocessing,” [Online]. Available: https://cs231n.github.io/neural-networks-2/#datapre.

[36] Rishabh Soni and K. James Mathai , “Improved Twitter Sentiment Prediction through ‘Cluster-then-Predict Model’,” 2015.

[37] “K-Means Clustering in Python,” [Online]. Available: https://towardsdatascience.com/k-means-clustering-in-python-4061510145cc. [Accessed 19 11 2020].

[38] H. Arabnia and Q. N. Tran, Emerging Trends in Computational Biology, Bioinformatics, and Systems Biology, 1st Edition ed., Elsevier, Morgan Kaufmann, 2015, pp. 577-602.

[39] Kjell Johnson, Max Kuhn, “Forward, Backward, and Stepwise Selection,” in *Applied Predictive Modeling*, Springer, 2013, p. 494.

[40] Isabelle Guyon, Jason Weston, Stephen Barnhill, M.D., Vladimir Vapnik, Barnhill, “Gene Selection for Cancer Classification using Support Vector Machines”.

[41] P. Nordsted, “Pete Nordsted's betting guide: Calculate your own odds to find value,” Goal.com, 2011.

[42] T. Salonen, How to Calculate Asian Handicap Odds, http://sportsdiscover.com/, 2002.

[43] A. Kumar and M. Jain, Ensemble Learning for AI Developers: Learn Bagging, Stacking, and Boosting Methods with Use Cases, Apress, 2020.

[44] D. Berrar, P. Lopes and W. Dubitzky, “Incorporating domain knowledge in machine learning,” *Machine Learning,* pp. 97-126, 2019.

[45] K. Y. Huang and K. J. Chen, “Multilayer Perceptron for Prediction of 2006 World Cup Football Game,” *2010 International Joint Conference on Neural Network*, pp. 1-8, 2010.

[46] D. Prasetio and D. Harlili, “Predicting Football Match Results with Logistic Regression,” *2016 International Conference On Advanced Informatics: Concepts, Theory And Application,* pp. 1-5, 2016.

[47] N. Tax and Y. Joustra, “Predicting The Dutch Football Competition Using Public Data: A Machine Learning Approach,” *Transactions on Knowledge and Data Engineering,* pp. 1-13, 2015.

[48] N. Danisik, P. Lacko and M. Farkas, “Football Match Prediction using Players Attributes,” *2018 World Symposium on Digital Intelligence for Systems and Machines,* pp. 201-206, 2018.