



Department of Computer Science and Engineering,  
The Chinese University of Hong Kong

Final Year Project, Final Report Term 1

*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. The goal of this project is to use different machine learning techniques and models to predict the handicap result for soccer games. In this report, odds from different bookmakers, past 10 encounter record of 2 teams, FIFA estimated player scores and FIFA estimated team scores are used to compare the performances of different models. In addition, various ways of feature engineering and dimension reduction techniques were used to conduct a better dataset for model training. Overall, the most stable and profitable model is the by-league model using the odds-related features only.

## **Acknowledgements**

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

# Table of Contents

<b>1. Introduction .....</b>	<b>7</b>
1.1 Overview .....	7
1.2 Motivation .....	7
1.3 Objective .....	8
1.4 Project Workflow .....	8
1.5 Glossary .....	9
1.6 Definition of Handicap .....	10
1.7 Past Research .....	12
<b>2. Methodology .....</b>	<b>13</b>
<b>3. Dataset .....</b>	<b>14</b>
3.1 Weather and temperature.....	14
3.2 Odds.....	14
3.3 The lineup of team .....	15
3.4 League.....	16
3.5 Recent 10 matches .....	17
3.5.1 Recent 10 matches of home/away team.....	17
3.5.2 Recent 10 encounters.....	17
3.6 Statistics of the season of the match .....	18
3.7 Player Information .....	19
3.7.1 Win007 Player Information .....	19
3.7.2 FIFA Player Information .....	20
3.8 FIFA Soccer Team Score .....	21
3.9 Overview .....	22
3.10 Evaluation Graph.....	22
<b>4. Feature Engineering .....</b>	<b>23</b>
4.1 HKJC time relative odds .....	24
4.2 Players and team scores .....	25
4.3 Encounters of each match .....	25
4.4 One hot encoding on League Name .....	26
4.4.1 Categorical League.....	26
4.4.2 One Hot League .....	27
4.5 Recent 10 Matches .....	28
4.6 Team names merging .....	29
4.7 Player names merging .....	30
4.8 Data Analysis.....	31



4.8.1	Association between odds and handicap result .....	31
4.8.2	The number of groups in handicap result.....	33
<b>4.9</b>	<b>Pre-processing.....</b>	<b>34</b>
4.9.1	Imputation .....	35
4.9.2	Principal Components Analysis.....	37
4.9.3	Autoencoder .....	38
4.9.4	Stacked, Deep Autoencoder .....	39
<b>5.</b>	<b>Modelling.....</b>	<b>41</b>
<b>5.1</b>	<b>Metrics .....</b>	<b>41</b>
<b>5.2</b>	<b>Hyperparameter tuning .....</b>	<b>41</b>
<b>5.1</b>	<b>Benchmark Models.....</b>	<b>42</b>
5.1.1	Odds-based Prediction Model, strategical betting .....	42
5.1.2	Naïve Betting Model, strategical betting.....	42
5.1.3	Expected Value Model, statistical metric .....	43
<b>5.2</b>	<b>Statistical models .....</b>	<b>44</b>
5.2.1	Linear Regression.....	44
5.2.2	Logistic regression .....	45
5.2.3	Random Forest .....	46
5.2.4	K-nearest neighbour .....	47
<b>5.3</b>	<b>Neural Network models.....</b>	<b>47</b>
5.3.1	Feedforward Neural Network.....	48
5.3.2	Long short-term memory (LSTM) .....	51
5.3.3	Stacked Autoencoder with supervised fine tuning .....	52
<b>5.4</b>	<b>Cluster-then-Predict Model .....</b>	<b>54</b>
<b>6.</b>	<b>Evaluation.....</b>	<b>55</b>
<b>6.1</b>	<b>Cluster-then-Predict Model .....</b>	<b>56</b>
<b>6.2</b>	<b>By-league Model .....</b>	<b>62</b>
<b>6.3</b>	<b>All-league Model .....</b>	<b>63</b>
<b>6.4</b>	<b>Best model .....</b>	<b>64</b>
<b>6.5</b>	<b>Dimension reduction .....</b>	<b>65</b>
<b>6.6</b>	<b>Feature Selection.....</b>	<b>72</b>
6.6.1	Akaike Information Criterion (AIC) .....	72
6.6.2	Recursive Feature Elimination .....	77
6.6.3	Feature Selection with Feedforward Neural Network.....	79
<b>6.7</b>	<b>Odds only features .....</b>	<b>81</b>
6.7.1	All-league Model.....	82
6.7.2	By-league Model.....	83
6.7.3	Leagues with best performance .....	84
6.7.4	Leagues with worst performance .....	85
<b>7.</b>	<b>Limitation .....</b>	<b>86</b>
<b>7.1</b>	<b>The size of dataset.....</b>	<b>86</b>
<b>7.2</b>	<b>Merging player information.....</b>	<b>86</b>
<b>7.3</b>	<b>Not all features are tested .....</b>	<b>86</b>
<b>8.</b>	<b>Conclusion and Future Work.....</b>	<b>87</b>

<b>9. References .....</b>	<b>88</b>
----------------------------	-----------

# 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 first semester, including the data collection, data selection, feature engineering, modelling and evaluation.

## 1.2 Motivation

Soccer Games have become one of the most popular sports around the world. Nowadays, hundreds of countries have their national team and there are 712 soccer clubs in the FIFA20 database. Although we do not play soccer game, we are fascinated by it. Due to the popularity of it, many commercial businesses related to soccer grow continuously. Among them, the growth of gambling business is the largest so it is not difficult to find the sponsor logos of some bookmakers in the 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.

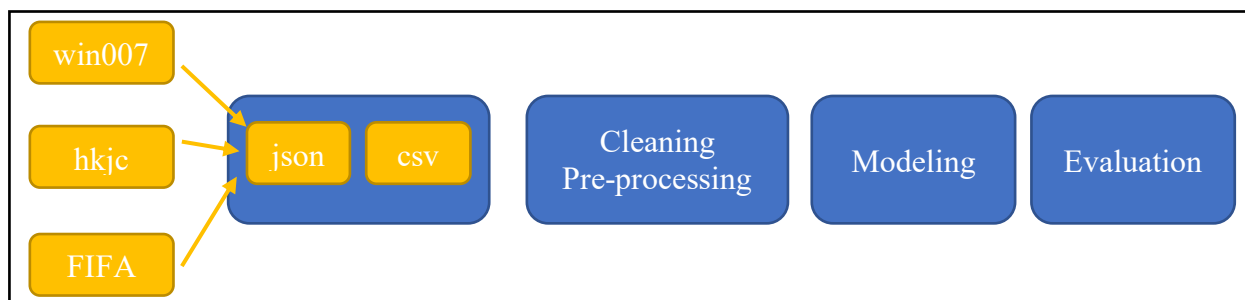
In semester one, the main focus will be on data cleaning and modelling.

- 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, optimizing the models, formulation of a betting policy and a real time demo will be focused.

### 1.4 Project Workflow

First, data were collected from three different websites<sup>1</sup> 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.



*Figure 1. Project Workflow*

<sup>1</sup> <http://live.win007.com/>, <https://g10oal.com/> and <https://www.fifaindex.com/>

## 1.5 Glossary

Terms	Explanations
Betting category	Various types of gambling games.
Commission fee	$\left( \sum_{s=odds} \frac{1}{s} \right) - 1$ <p>For example, if both of the handicap odds for the home team and away team are 1.85.</p> $\text{Commission} = \frac{1}{1.85} + \frac{1}{1.85} - 1 \approx 8\%$
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

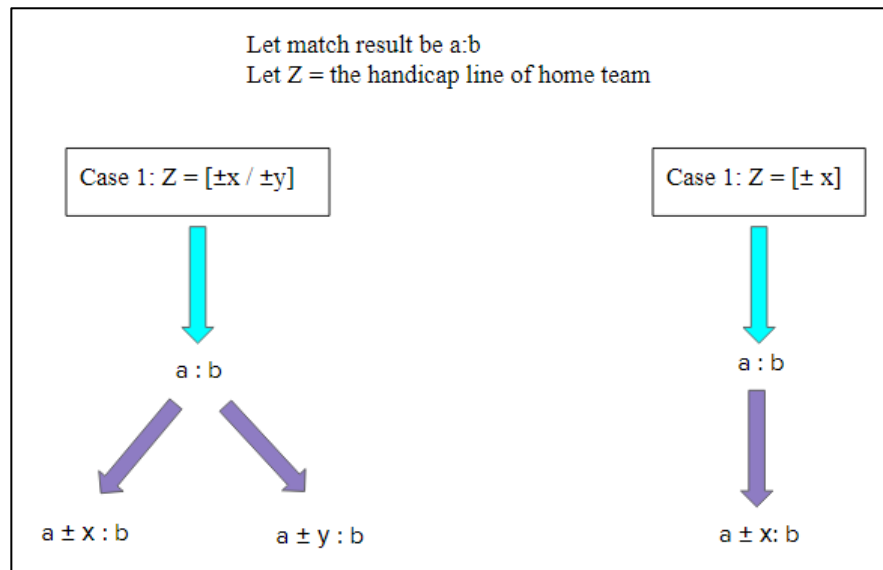


Figure 2. Illustration of handicap

Handicap is a betting category that makes the match being fairer and evenner 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  $[\pm x / \pm y]$  and  $[\pm x]$ . Then, the result ( $a : b$ ) after calculating the handicap will be  $(a \pm x : b \pm y)$  or  $(a \pm x : b)$ .

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	$[-1]$	$3:1 \rightarrow 2:1$
1	winning half of the money	$[+0/0.5]$	$1:1 \rightarrow 1:1/1.5:1$
0	getting back original money	$[-1]$	$2:1 \rightarrow 1:1$
-1	losing half of the money	$[-2/-2.5]$	$3:1 \rightarrow 1:1/0.5:1$
-2	losing all the money	$[-1/-1.5]$	$0:0 \rightarrow -1:0/-1.5:0$

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 football 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 the 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 tried to model the game 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. Hence, odds will be the main focus in this project.

Each bookie has their betting odds for each match. Usually, this betting odds will change over time so to maximise the bookie's profit. Thus, all changes of the handicap odds *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 may not be same. Therefore, we proposed a by-league model and all-league model, this former one is essentially train the dataset on a league basis and the latter one is train with one-hot label on the league.

The type of handicap results will be used as the label because it has the least average commission fee (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 maybe 0.1.

There are 5 possible classes in handicap result, introduced in [section1.6 \(Definition of Handicap\)](#). 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<sup>2</sup>. 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

Odds 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, odds from different bookmakers are collected, in which companies like HKJC, 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	Away
Tuesday Matches					
TUE 3		Lokomotiv Moscow[+2.5/+3] vs Bayern Munich[-2.5/-3]	28/10 01:55	<input type="checkbox"/> 1.76	<input type="checkbox"/> 2.11
TUE 4		Shakhtar Donetsk[+1/+1.5] vs Inter Milan[-1/-1.5]	28/10 01:55	<input type="checkbox"/> 1.98	<input type="checkbox"/> 1.86
TUE 8		Barnsley[0/-0.5] vs QPR[0/+0.5]	28/10 03:00	<input type="checkbox"/> 2.02	<input type="checkbox"/> 1.82

Figure 4. Handicap odds of HKJC

<sup>2</sup> 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

### 3.3 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 6, 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 6. 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.




















 English	 Italian	 German
 Spanish	 French	 Dutch
 Scottish	 Portuguese	 Swedish
 Norwegian	 Japanese	 Brazilian
 American	 Argentina	 Australian
 Mexican	 Chilean	 Russian
 Belgian	 Korean	

Figure 7. List of the countries whose leagues are supported by HKJC

 Euro Cups	 American Cups	 Asian Cups
 African Cups	 International Matches	

Figure 8. 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 <input type="checkbox"/> 主場 <input checked="" type="checkbox"/> 歐霸盃 <input checked="" type="checkbox"/> 德甲 <input checked="" type="checkbox"/> 德國盃 <input checked="" type="checkbox"/> 球會友誼															
類型	日期	主場	比分(半場)	角球	客場	皇冠	終盤	平均歐賠	終盤	主	和	客	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	諾德施泰(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 9. 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 <input type="checkbox"/> 主客相同 <input checked="" type="checkbox"/> 德甲 <input checked="" type="checkbox"/> 球會友誼 <input checked="" type="checkbox"/> 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	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 10. 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 11, 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												[德甲-9]奧格斯堡											
全場	賽	勝	平	負	得	失	淨	得分	排名	勝率	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%	總	4	2	1	1	5	3	2	7	9	50.0%		
主	1	0	1	0	1	1	0	1	14	0.0%	主	2	1	0	1	2	2	0	3	9	50.0%		
客	3	1	2	0	2	1	1	5	3	33.3%	客	2	1	1	0	3	1	2	4	5	50.0%		
近6	4	1	3	0	3	2	1	6		25.0%	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	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%	總	4	2	1	1	2	1	1	7	8	50.0%		
主	1	0	1	0	1	1	0	1	12	0.0%	主	2	1	0	1	1	1	0	3	9	50.0%		
客	3	2	1	0	2	0	2	7	2	66.7%	客	2	1	1	0	1	0	1	4	9	50.0%		
近6	4	2	2	0	3	1	2	8		50.0%	近6	4	2	1	1	2	1	1	7		50.0%		

Figure 11. Example of a statistic table for a match

Figure 11 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 12. 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.

	簡體名/簡稱:	拉菲尔·莱奥/莱奥	英文名:	Rafael Leao <a href="#">報錯</a>
	繁體名:	拉菲利亞奧	預計身價:	2160萬英鎊
	生日:	1999-06-10	體重:	kg
	身高:	188 cm	慣用腳:	右腳
	國籍:	葡萄牙、安哥拉	合同截止期:	2024-06-30

Figure 12. 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 13. 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 14. 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)

### 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 20 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 (Figure 15). 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
	水戸蜀葵	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

*Figure 15. 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 16. Example of the HKJC handicap odds

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

Figure 17. 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 16), the handicap line of HKJC is “[0]” (Figure 16) which is same as the Chinese word “平手” (Figure 17). For non-HKJC bookies, the odds in the blue rectangle (Figure 17) 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 to prevent too many features 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 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

Figure 18. 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.

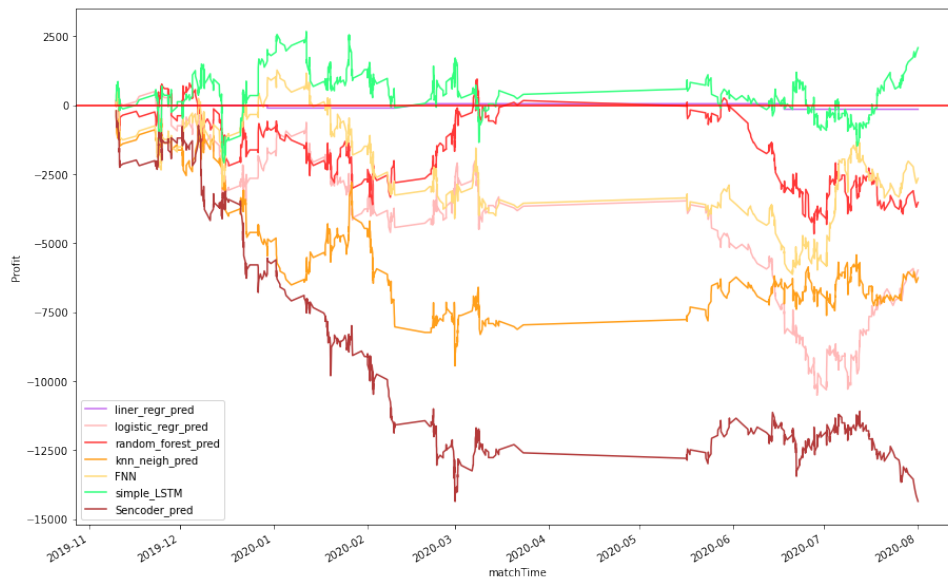


Figure 19. Example of the categorical league

#### 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

Figure 20. Example of the categorical league

Similarly, we pass the dataset which contains one hot feature and odds to the modelling pipeline. According to Figure 21, 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.

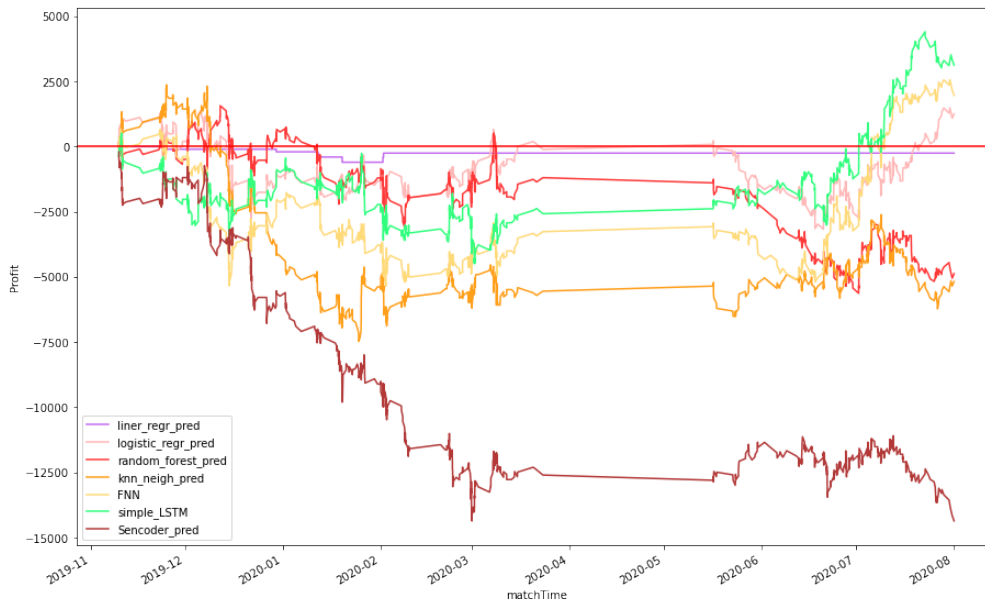


Figure 21. Example of the categorical league

## 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 <input type="checkbox"/> 主客相同 <input checked="" type="checkbox"/> 德甲 <input checked="" type="checkbox"/> 球會友誼 <input checked="" type="checkbox"/> 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	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 22. 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 22

, 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 even. 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.



## 4.6 Team names merging

Since most of the team data are collected from 2 websites which are win007 and FIFAIndex, a merging process should be implemented on them. In the beginning, the merging process compared the traditional names from both websites because the English team names are not provided by win007. However, the merging performance is not what we expected as the many team names are different from different websites. For example, “卡山魯賓” in FIFAIndex called “魯賓卡山” in win007.

After a deep 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 2.

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 2. A part from the team mapping table*

After applying the team mapping table, 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, the 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<sup>3</sup> 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.

---

<sup>3</sup> <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 generate it 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. Also, 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. Then, we used R to calculate the test and the result was as below Figure.

```
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 23. Kruskal-Wallis H Test between initiated odds and handicap result*

According to Table 3, 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 3. 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. 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.

For finding the differences between the groups, we used 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]. Since taking all bookmakers and odds to perform the test is a little bit messy, we used the HKJC initialled handicap odd 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 24. Pairwise Wilcoxon Rank Sum Tests between the groups in handicap result

According to the Figure 24, there is a sufficient difference between the winning/losing half money and all money. Thus, we will not combine them in this project. Also, there is an interesting finding is that the winning half money has no significant difference in losing all money and winning half money has no significant difference with losing all money. Therefore, it will be considered when setting the betting strategy in semester 2.

## 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 25, 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.

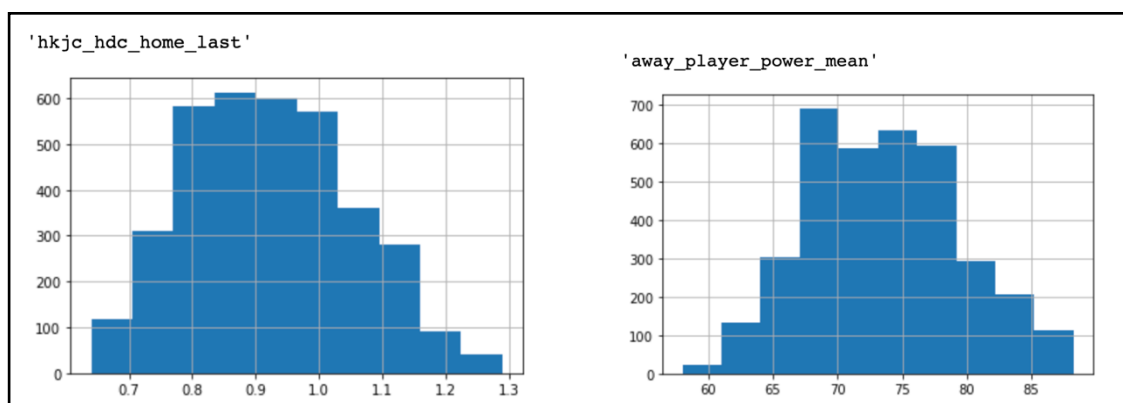


Figure 25. Distribution of two HKJC odds and player score

The histograms below (Figure 26) show the difference between before and after normalization. Each color 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.

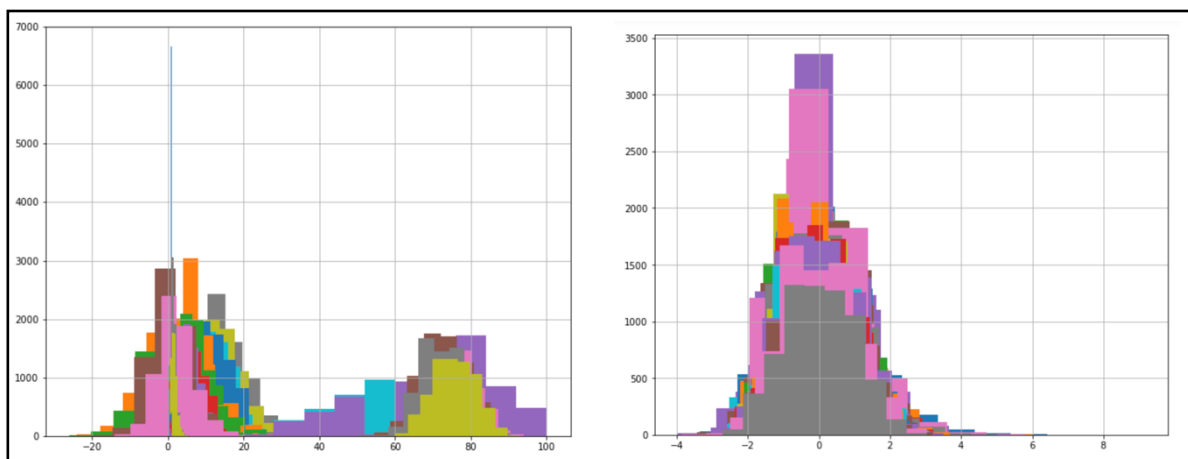


Figure 26. Normalization Results

### 4.9.1 Imputation

Due to the fact that our datasets were crawled from different websites. 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 in 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.

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

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:

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

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

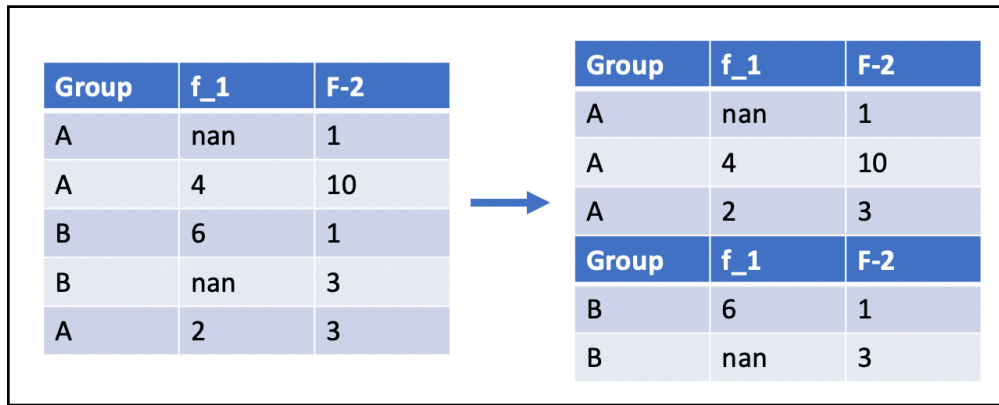


Figure 27. Illustration of separate by [team, league]

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 27) 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 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, a 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 of the input layer with fewer dimension. This inner 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. The autoencoder will compress the input features into a new set of features which dimension is 70% 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. Thus, a linear activation function was used. A illustration of our autoencoder is depicted on Figure 28.

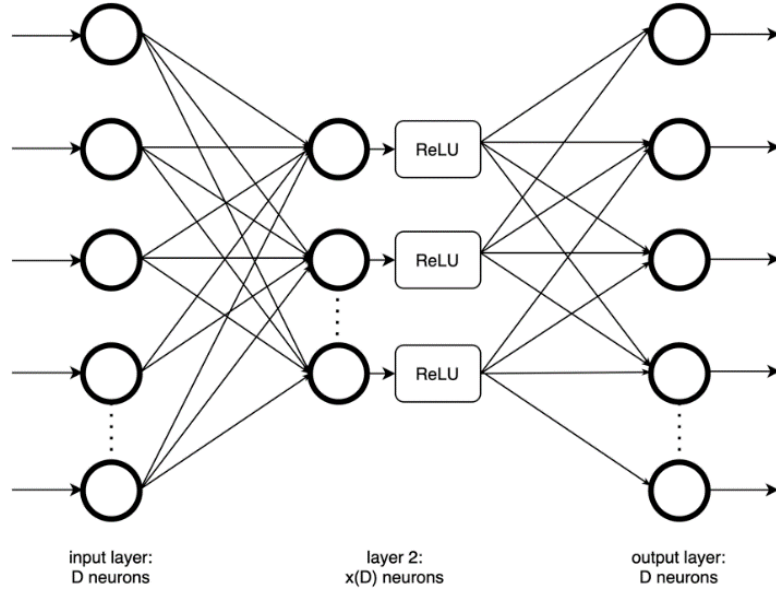


Figure 28. Architecture of Autoencoder

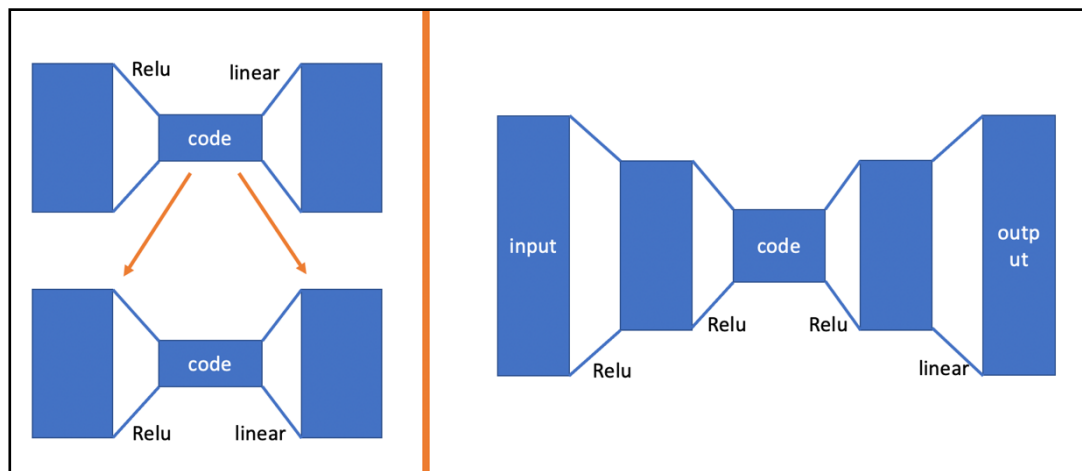
#### 4.9.4 Stacked, Deep Autoencoder

A multi-layer approach on autoencoder which is called Stacked Autoencoder is also tried. The main difference between them 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](#).

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. The other way is to train all the layers at once (Deep Autoencoder), setting up one universal loss function 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 are 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 29 shows their difference.



*Figure 29. Illustration of Stacked Autoencoder and Deep Autoencoder*

## 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. 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 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

*Figure 30. 5-fold cross validation*

## 5.1 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.

### 5.1.1 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. 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.

### 5.1.2 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 change 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 31. Pseudocode for navie model*

### 5.1.3 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 trend 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 [16], 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 football 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 commission fee hided in the odds. Therefore, the bookmakers can always earn money from the gambling.

## 5.2 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. [17] Notwithstanding this, statistical models still have a pretty good performance.

### 5.2.1 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. As there is more than one feature in our model, multiple linear 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

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

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.



### 5.2.2 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  $i^{th}$  match belonging to class  $k$  is

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

If  $k$  is the last class,

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

If  $k$  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.

### 5.2.3 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  $entropy = -\sum_i^k p_i \log_2(p_i)$ . [18] 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.

### 5.2.4 K-nearest neighbour

K-nearest neighbour (KNN) 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 dealing with the NAN.

Consider  $x$  as an unlabelled input data and  $k$  as the parameter.  $d_k(x)$  is the distance from  $x$  to the  $k$ -nearest neighbour. The label of  $x$  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  $k$  will be pass to hyperparameter tuning.

## 5.3 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.

### 5.3.1 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  $x$  and output is  $y$  [19]. 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 neuron in the output later represent the probability of each class.

In the architecture for our FNN, 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  $x = [x_1, x_2, x_3]$  determine the number of neurons in each hidden layer. For example, a  $D$  dimension of vector and the  $x=[0.8, 0.5, 0.3]$  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 ( $x$ ) 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 32 visualizes the overall structure of the FNN in our project.

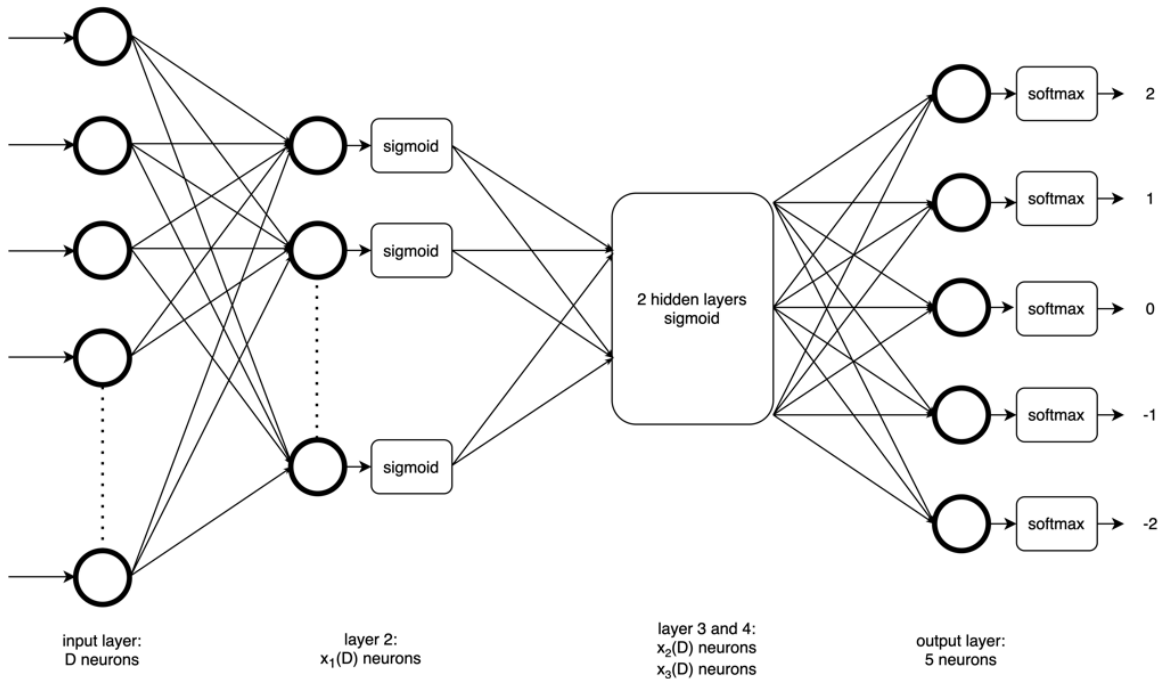


Figure 32. Network structure of the Feed Forward Neural Network

Although the capability of the neural network models is very strong, overfitting happens easily in such model. In this project, two ways to prevent overfitting has adopted. The first one is early stop [20], 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 performance of validation set will be worsen.

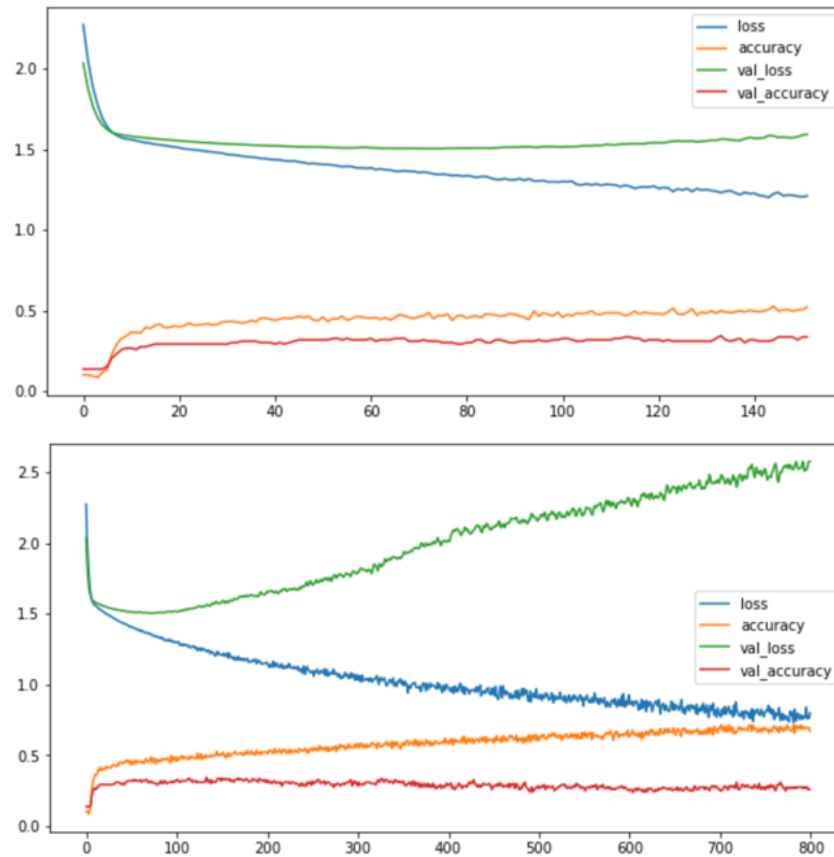


Figure 33. With early stop VS without early stop

The second approach is using dropout layer, 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). 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) [21]. 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.

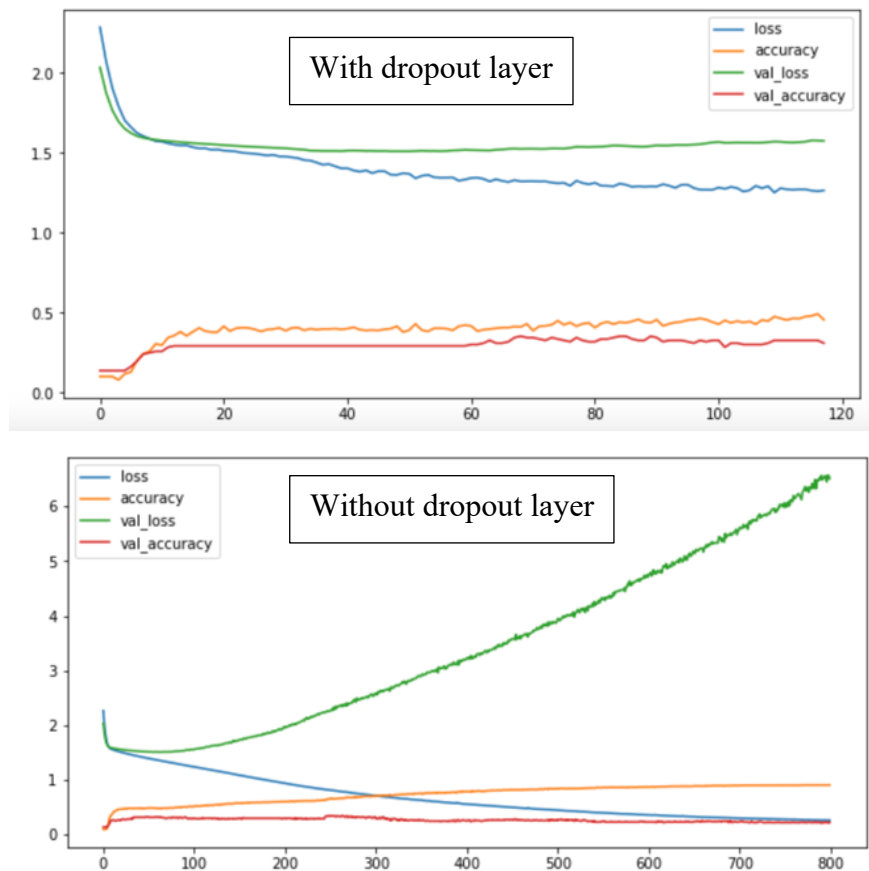


Figure 34. Using dropout vs without dropout

### 5.3.2 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  $h_i$  of a node  $n_i$  are the output from last node  $n_{i-1}$  and the new data  $x_i$ . Relatively, the output data are not only the output  $y_i$ , but also the output hidden state  $h_{i+1}$  which is one of the input data of node  $n_{i+1}$ . 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 [22]. The implementation of LSTM model is adding one more input and output cell state  $c_i$ . In overview, the cell state  $c_i$  change slowly whereas the hidden state  $h_i$  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  $N_f$ , the number of neurons is **round-up( $N_f * (4 - i + 1) * 0.2$ )** in the  $i^{th}$  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.

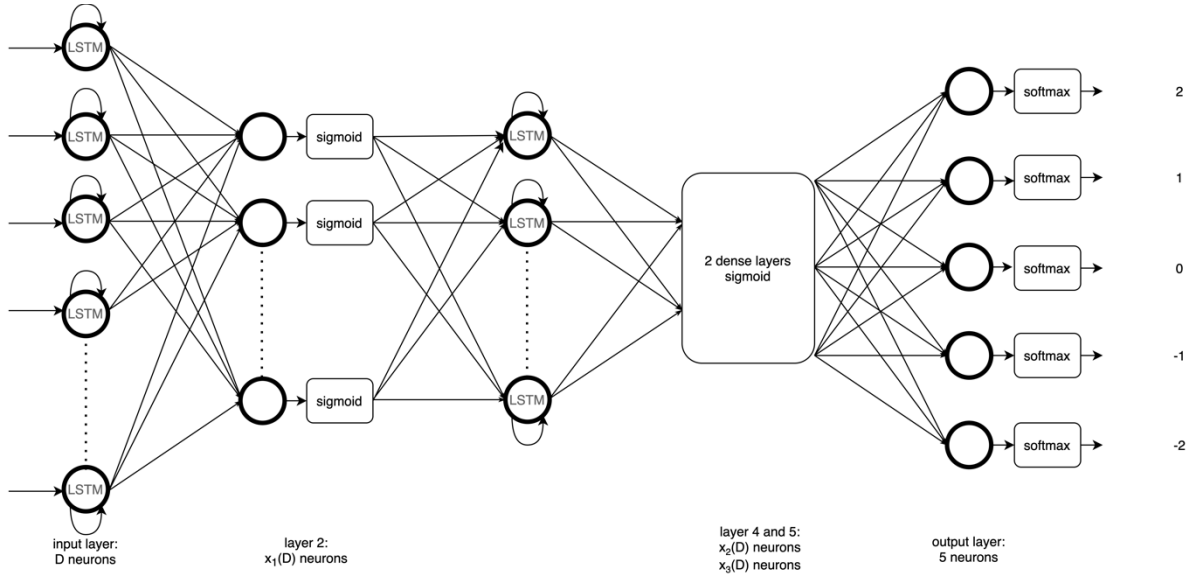


Figure 35. Structure of the LSTM Model

### 5.3.3 Stacked Autoencoder with supervised fine tuning

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 [23]. 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 36.

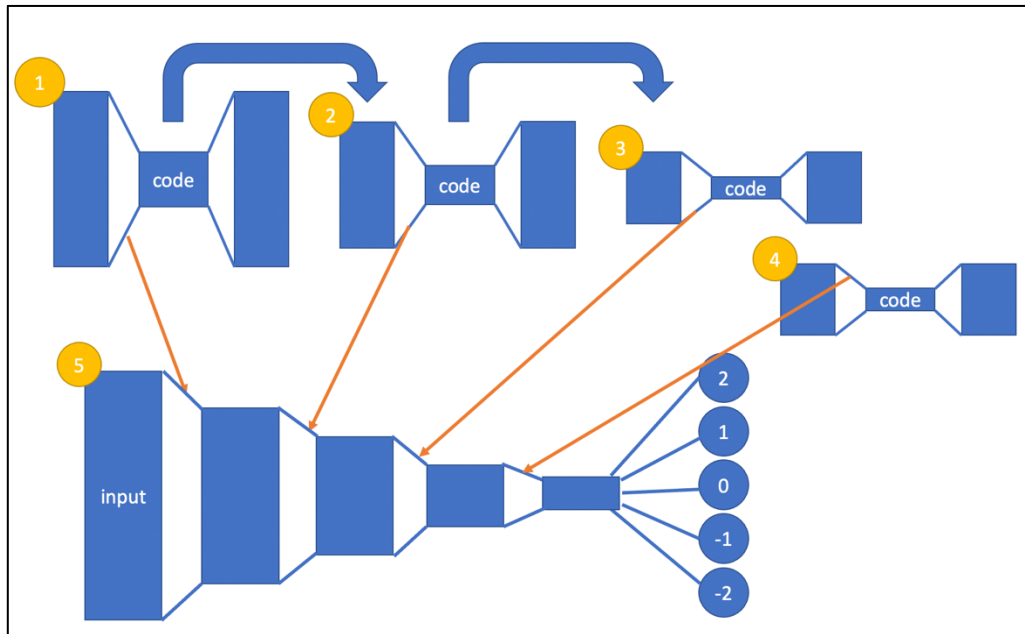


Figure 36. Visualization of Stacked Autoencoder with supervised method

Although the last part 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.

## 5.4 Cluster-then-Predict Model

Although the main focus of the project is choosing between a by-league model or an all-league 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 [24]. In that paper, they cluster the twitter with similar words and model on it. The rationale of this is to form 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 [25] 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  $centre_X$  is the cluster’s centre of  $X$ . Obvious when  $k$  increase, distortion decrease. According to Figure 37, the “elbow” occurred when  $k$  equal to 9 in our dataset. Hence, nine cluster will be used for K-means algorithm.

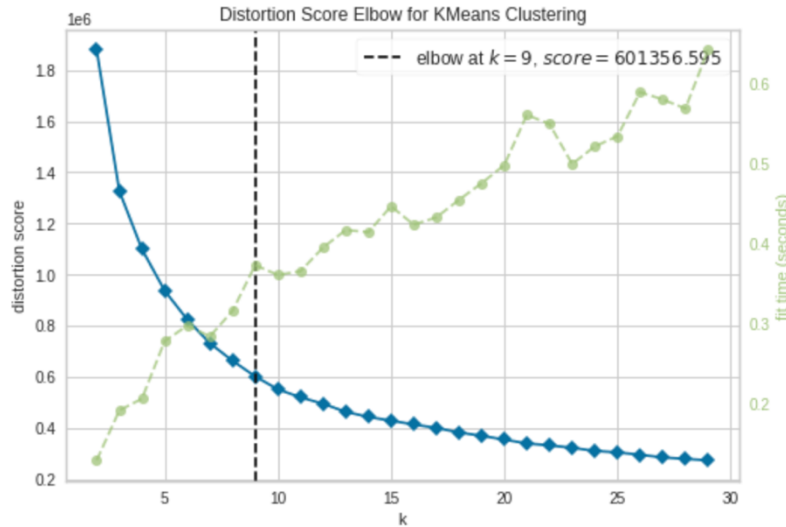


Figure 37. Elbow Method plot

## 6. Evaluation

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 is higher than 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.

In this section, we will first choose the best way to do the prediction among three methods, which are Cluster-Than-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.

When we are doing the evaluation. All of the statistical models, 3 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

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

*Figure 38. 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 future, we will keep monitoring this behaviour by adding the data from Aug-2020 to Dec-2020. We want to see if the best models in each cluster now are still the best in the new dataset or not. If that's 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 39), we are suspecting 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.

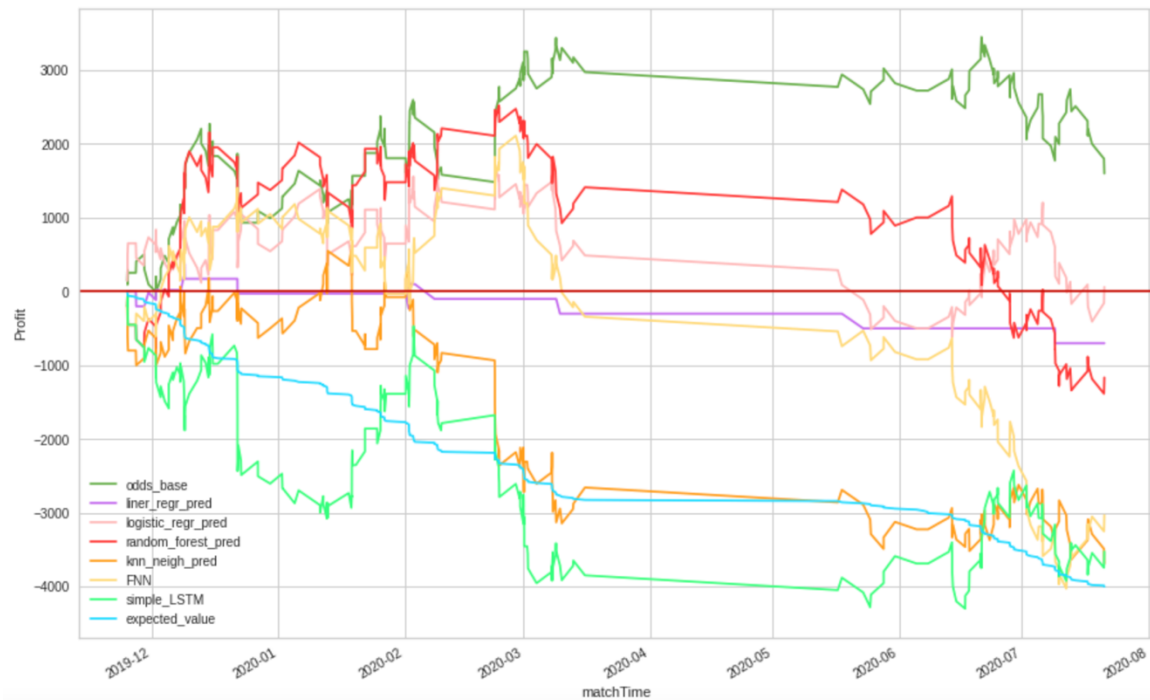


Figure 39. K-means cluster 1

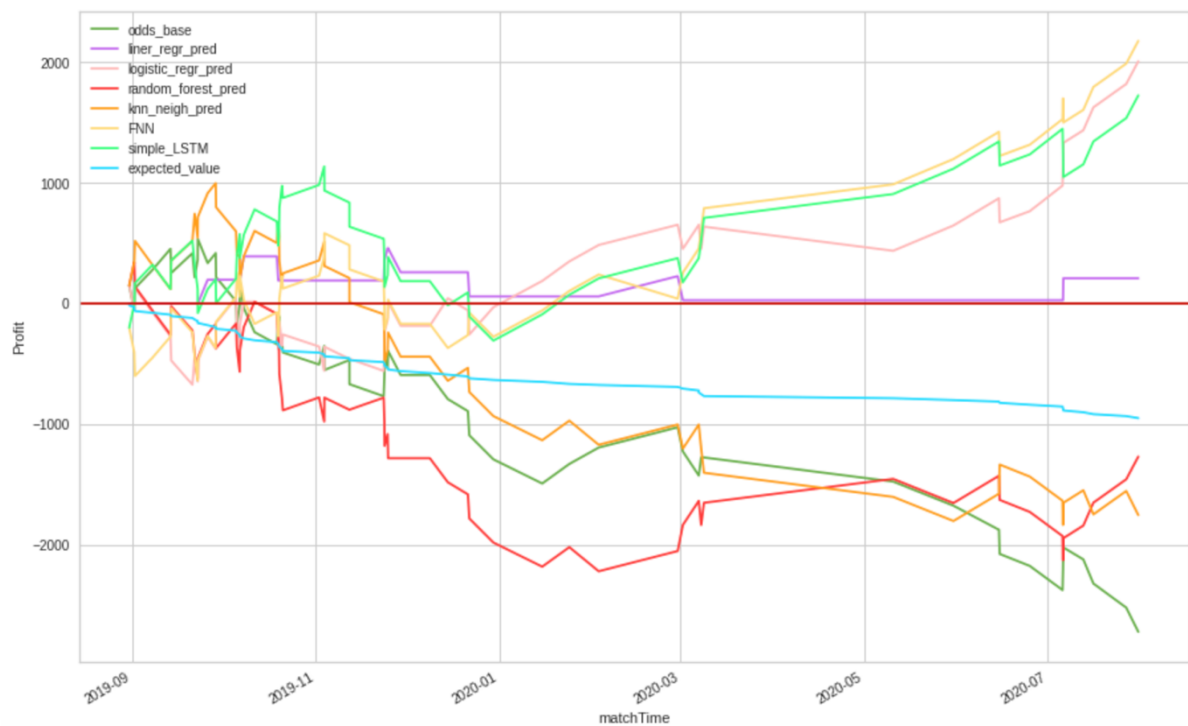


Figure 40. K-means cluster 2

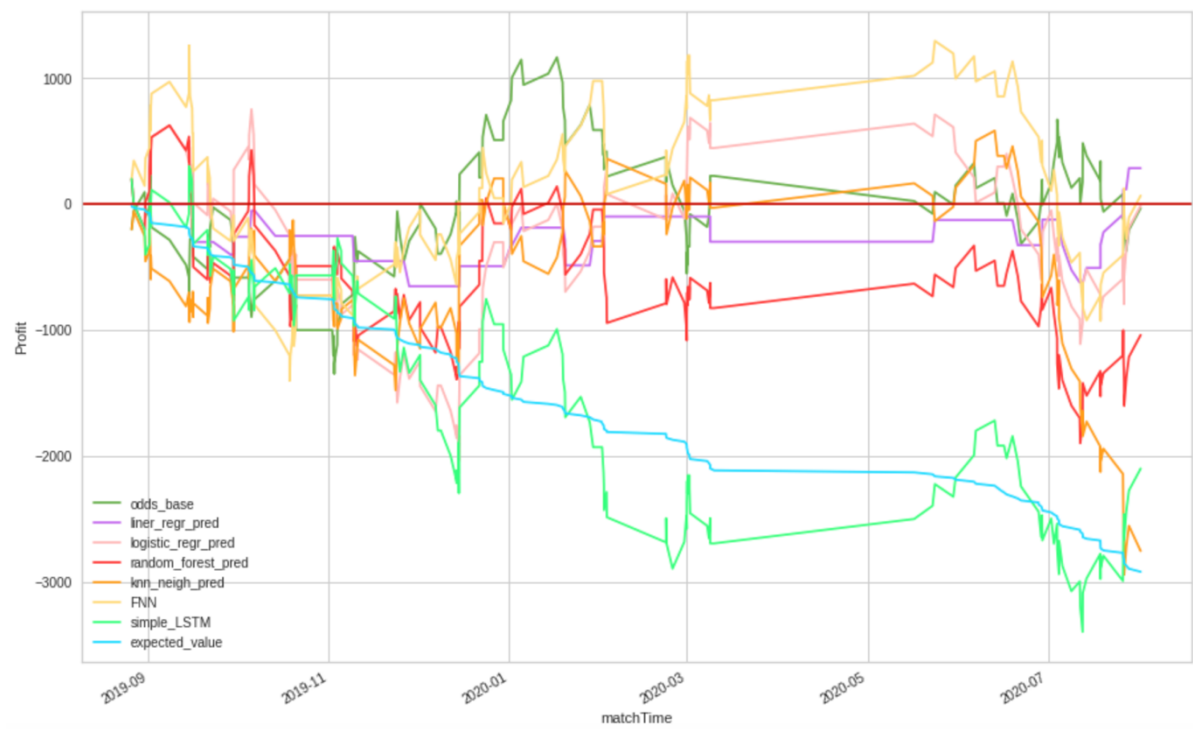


Figure 41. K-means cluster 3

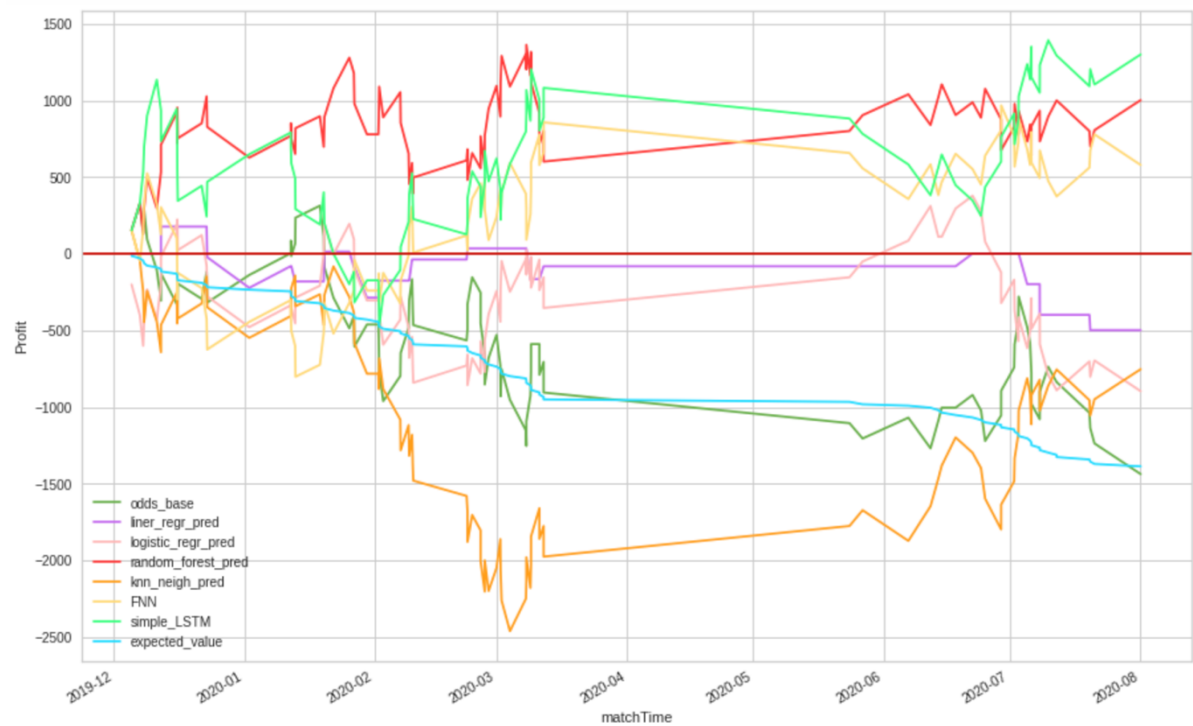


Figure 42. K-means cluster 4

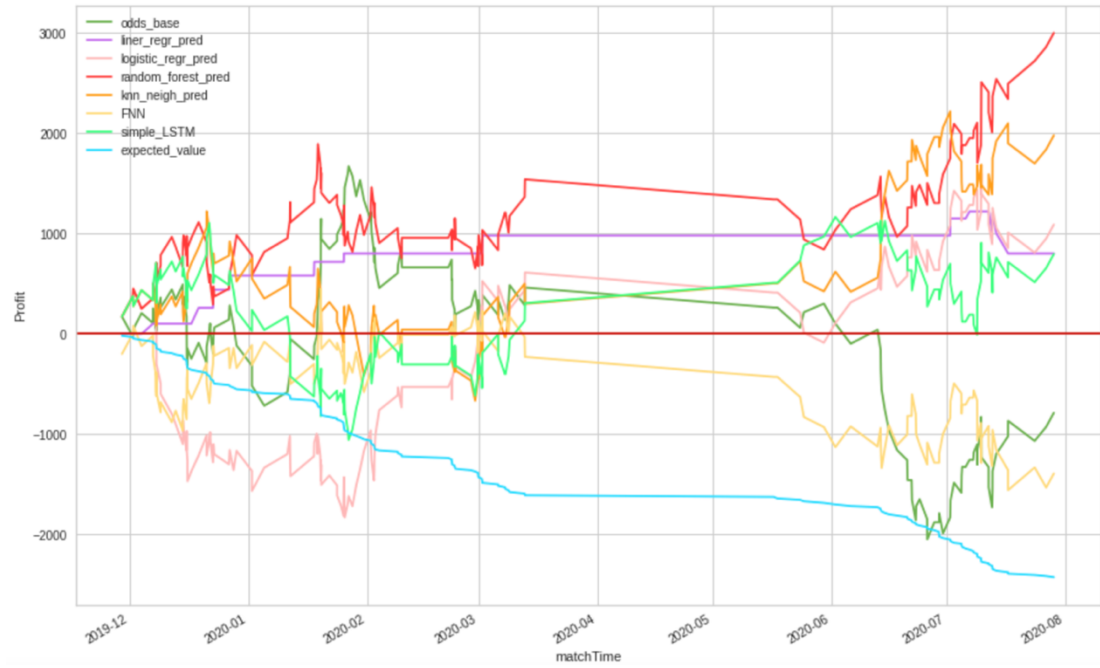


Figure 43. K-means cluster 5

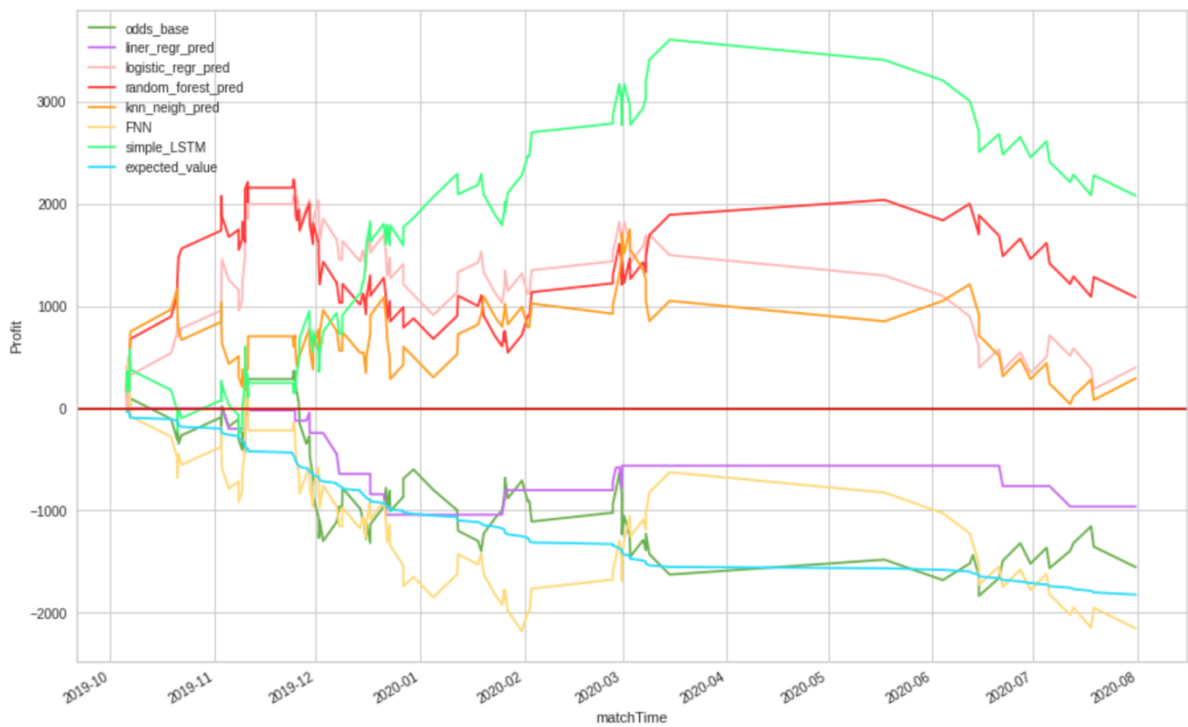


Figure 44. K-means cluster 6



Figure 45. K-means cluster 7

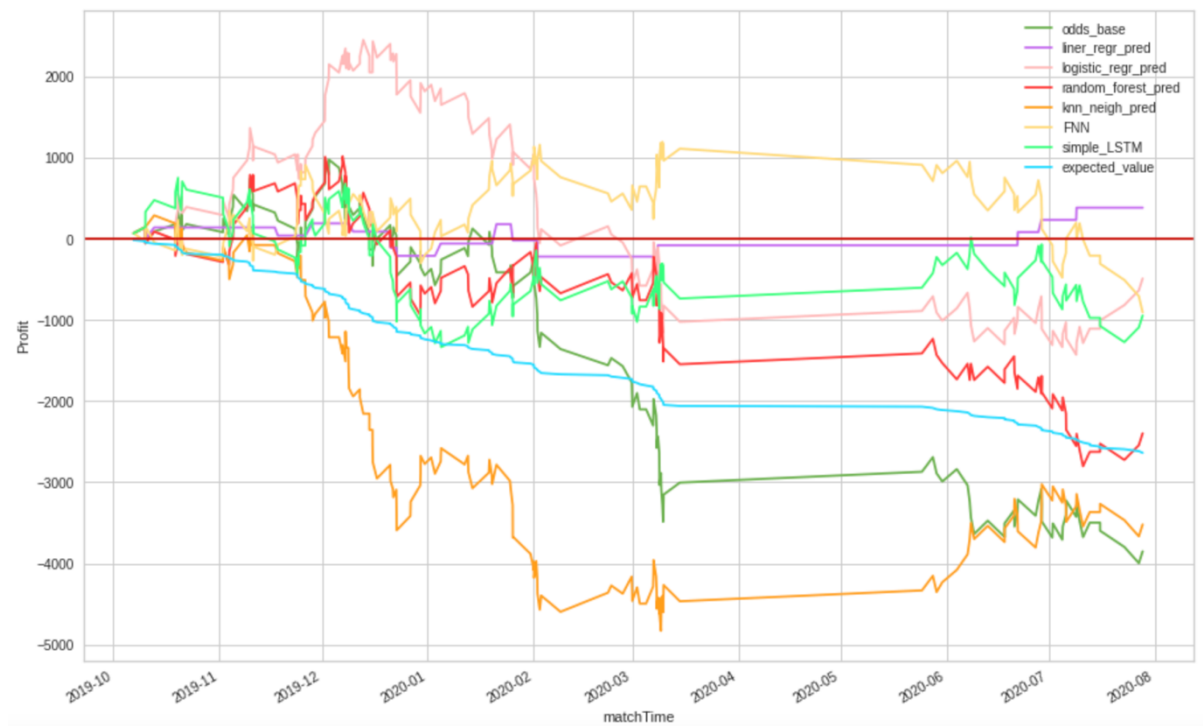


Figure 46. K-means cluster 8



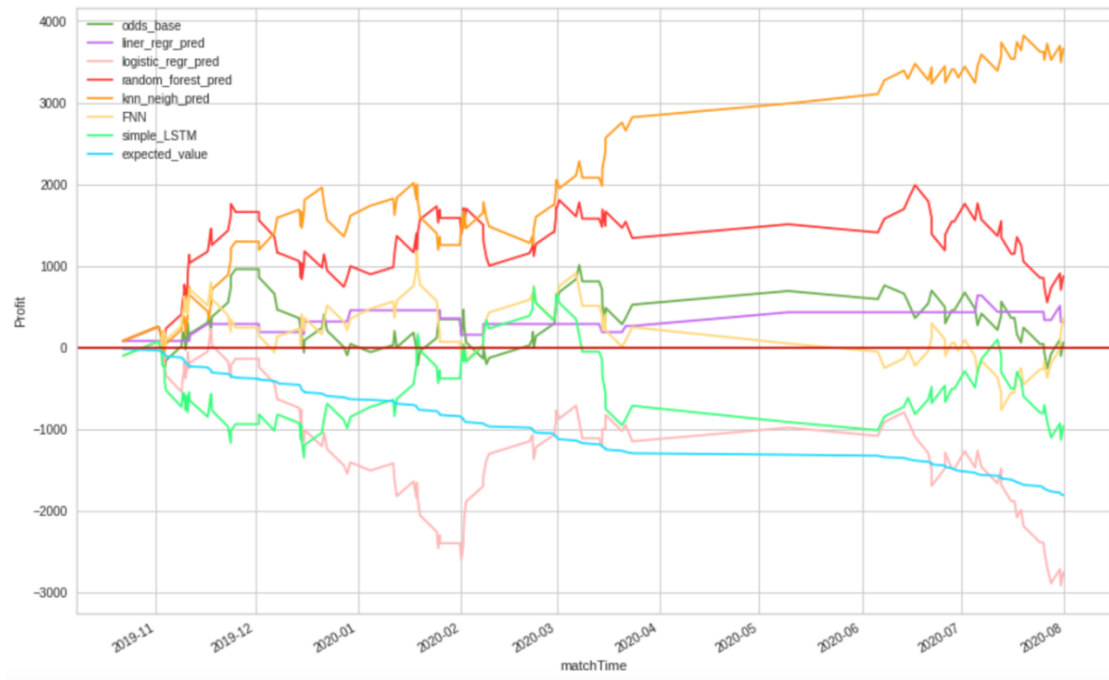


Figure 47. K-means cluster 9

Besides looking from cluster-level, we would also want to know how's the overall performance be so that it is easier for us to compare with the other two approaches. Figure 48 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.

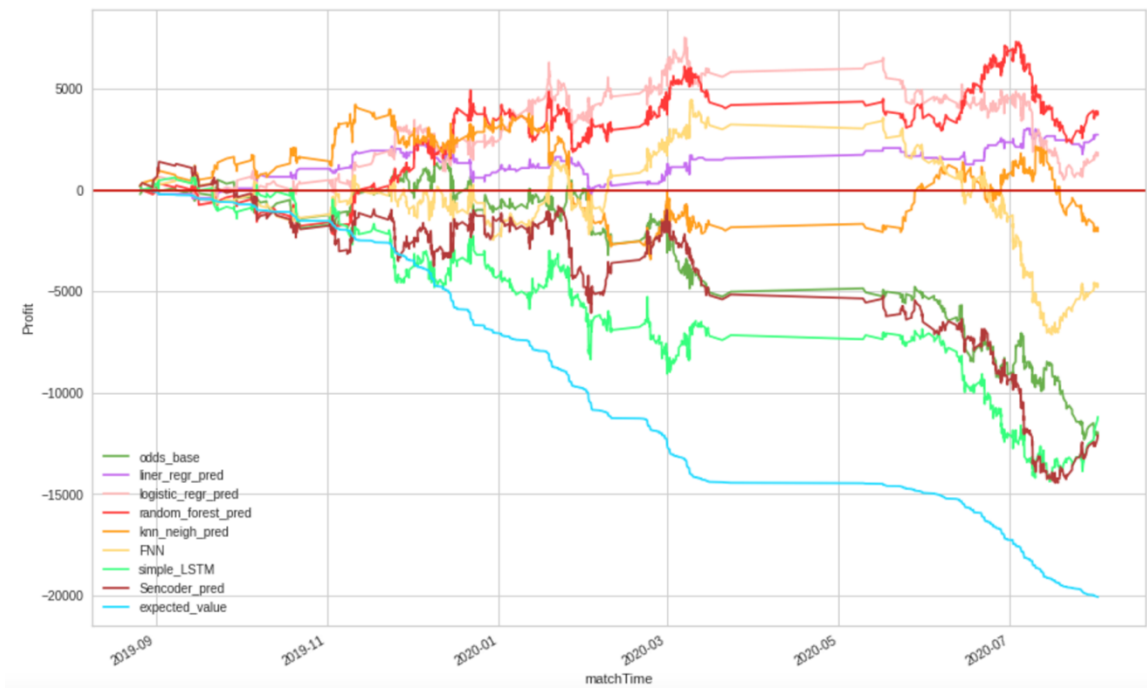


Figure 48. Overall k-means profit

## 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 49 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 50 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 49. League to train for by-league and all-league Model

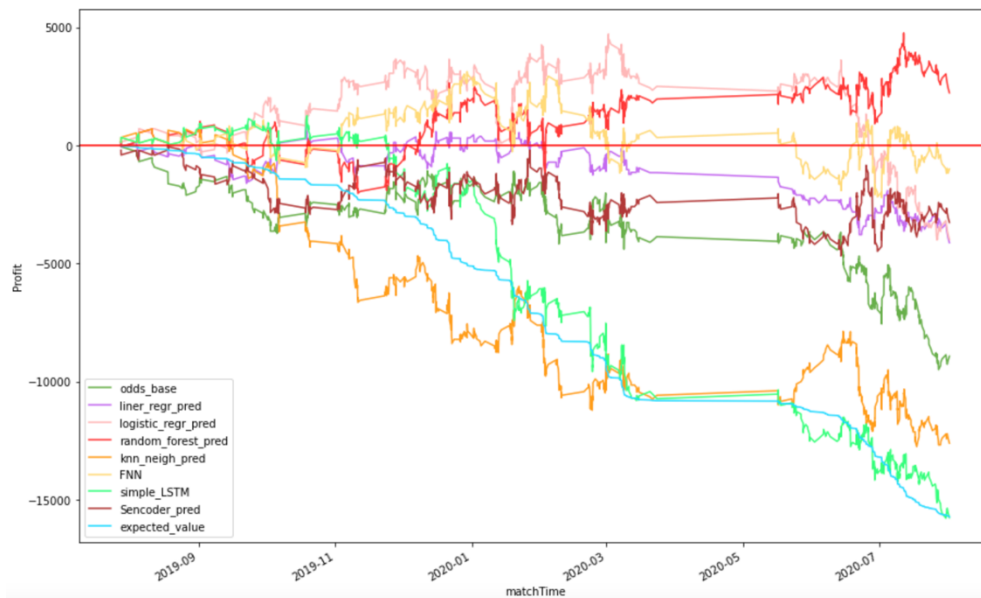


Figure 50. By-league Model Results

### 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 49). In this model, 80% of combined and selected were trained and the remaining were tested. Figure 51 shows the all-league performance.

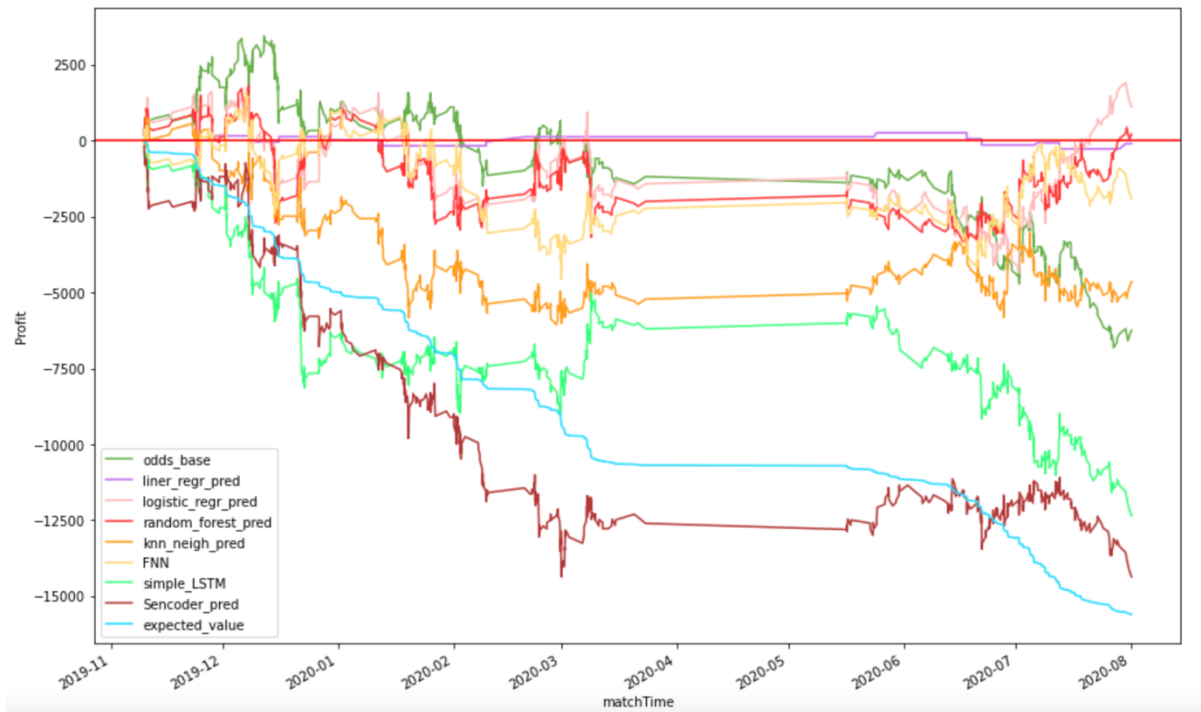


Figure 51. All-league Model Results

## 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 bad 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 the next semester, 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 4 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 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. Moreover, since we will more focus on neural network models in the next semester, the all-league model is the most suitable for our future task as its neural network models (except the Sencoder) got the best performance. Therefore, we will use all-league model to do the further evaluation in the following parts.

	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 4. 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 52. 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 53 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.

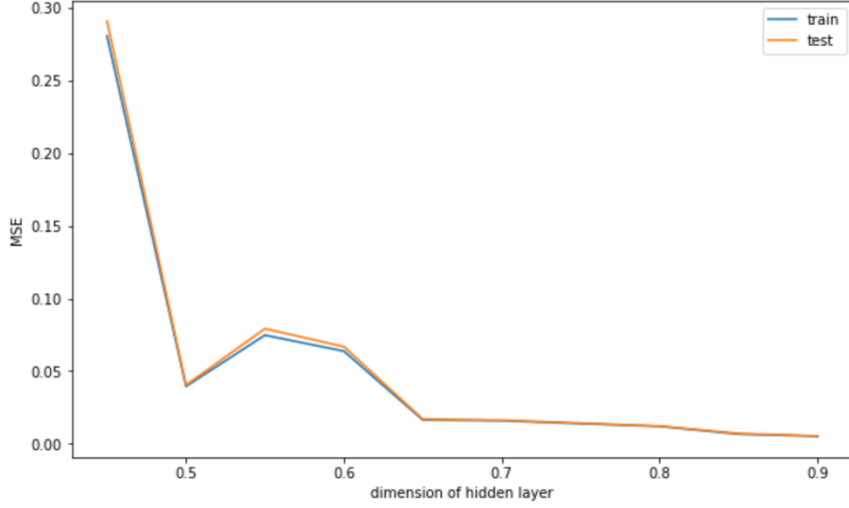


Figure 53. Autoencoder MSE

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.

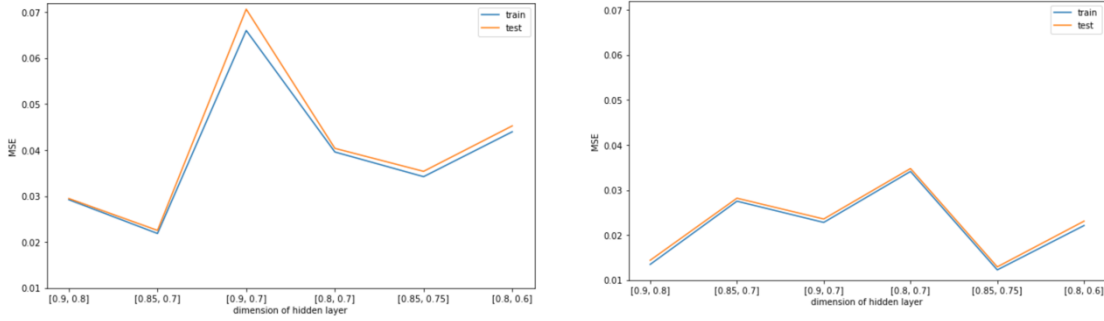
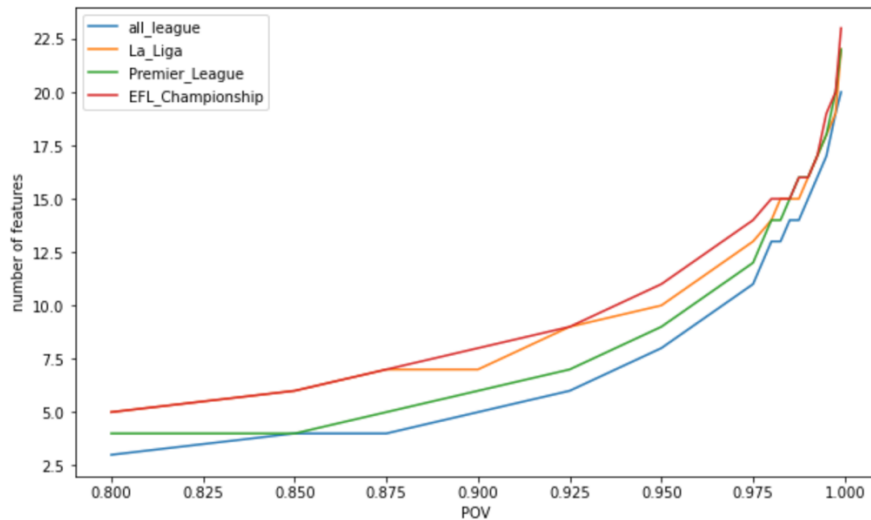


Figure 54. MSE on Deep Autoencoder and Stacked Autoencoder

According to Figure 54, 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  $100 \rightarrow 90 \rightarrow 80 \rightarrow 90 \rightarrow 100$ . 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 55. Number of features selected, PCA\_POV*

Figure 55 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.

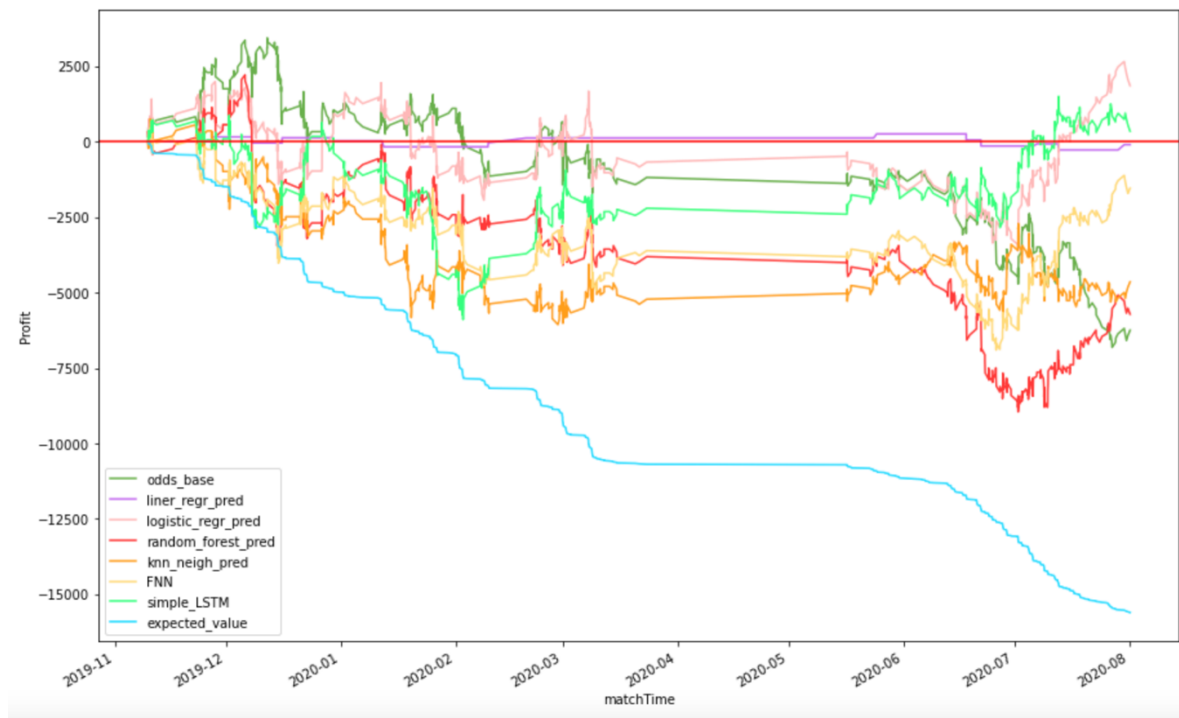


Figure 56. PCA with  $POV=0.999$

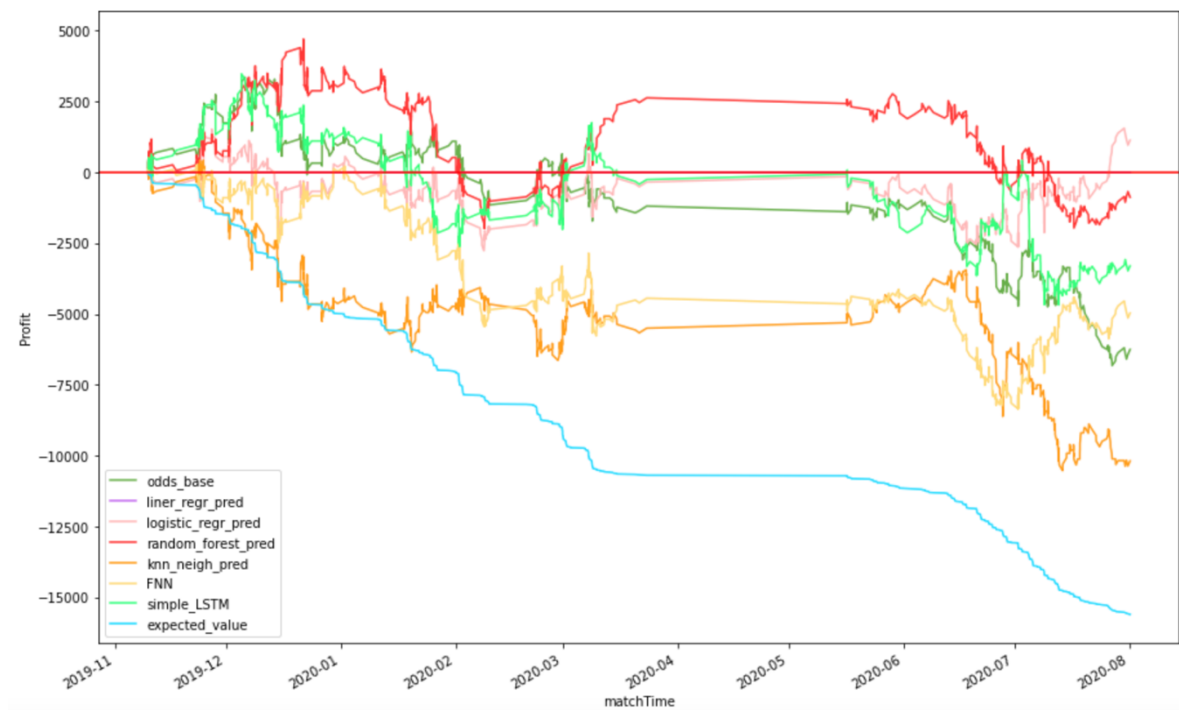


Figure 57. PCA with  $POV=0.95$

Figure 56 and Figure 57 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 51; 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.

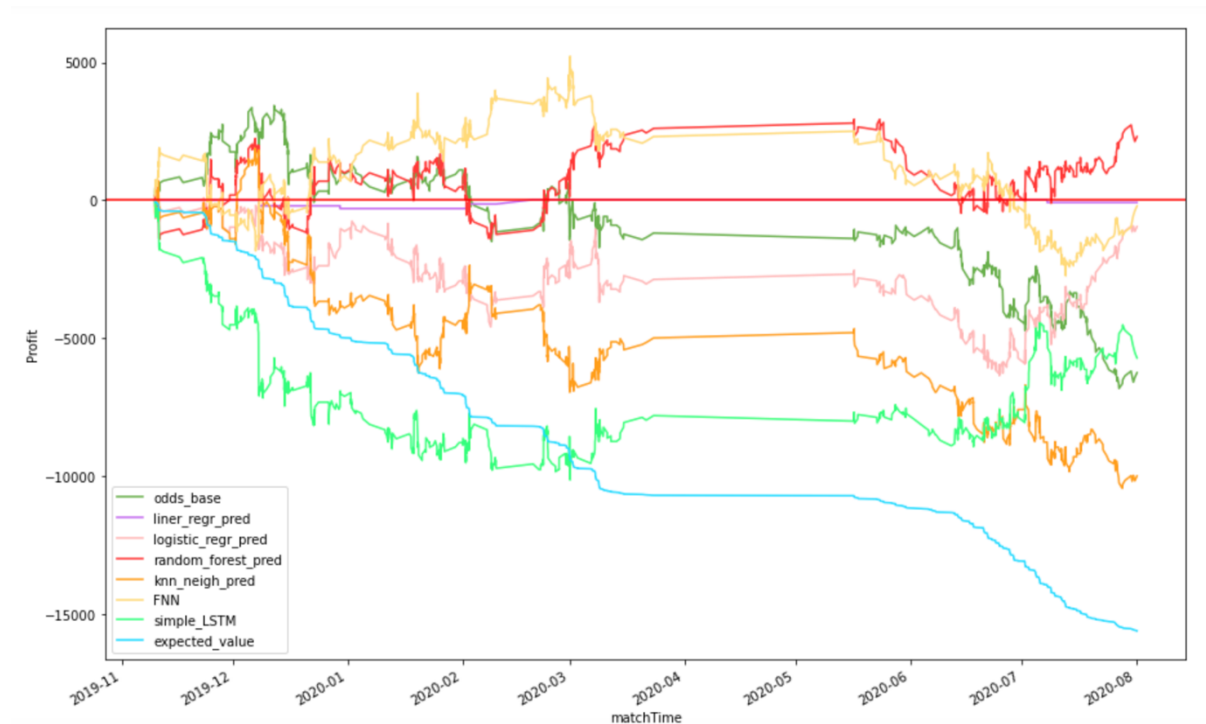


Figure 58. Stacked Autoencoder Result



Figure 59. Deep Autoencoder Results

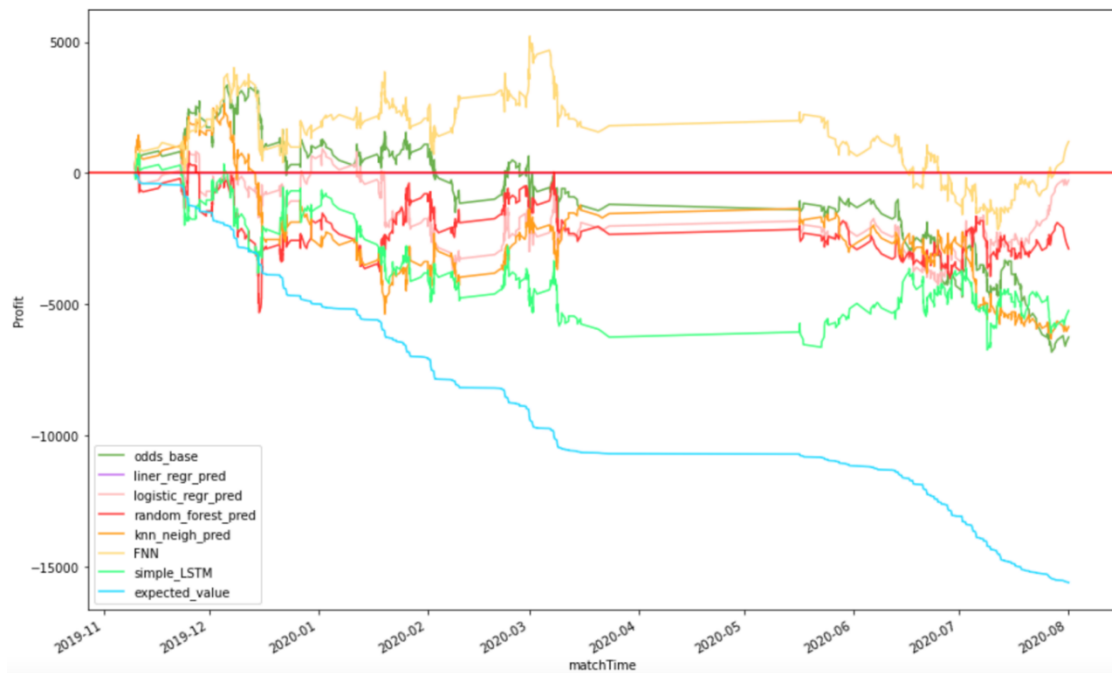


Figure 60. Autoencoder Results

Last but not least, Figure 60 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 5 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 5. 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 [26].

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$同主客淨勝_home + fd$bet365_hdc_initialFirst_away +
  fd$bet365_hdc_initialFirst_home

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

Figure 61. 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 62. 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.

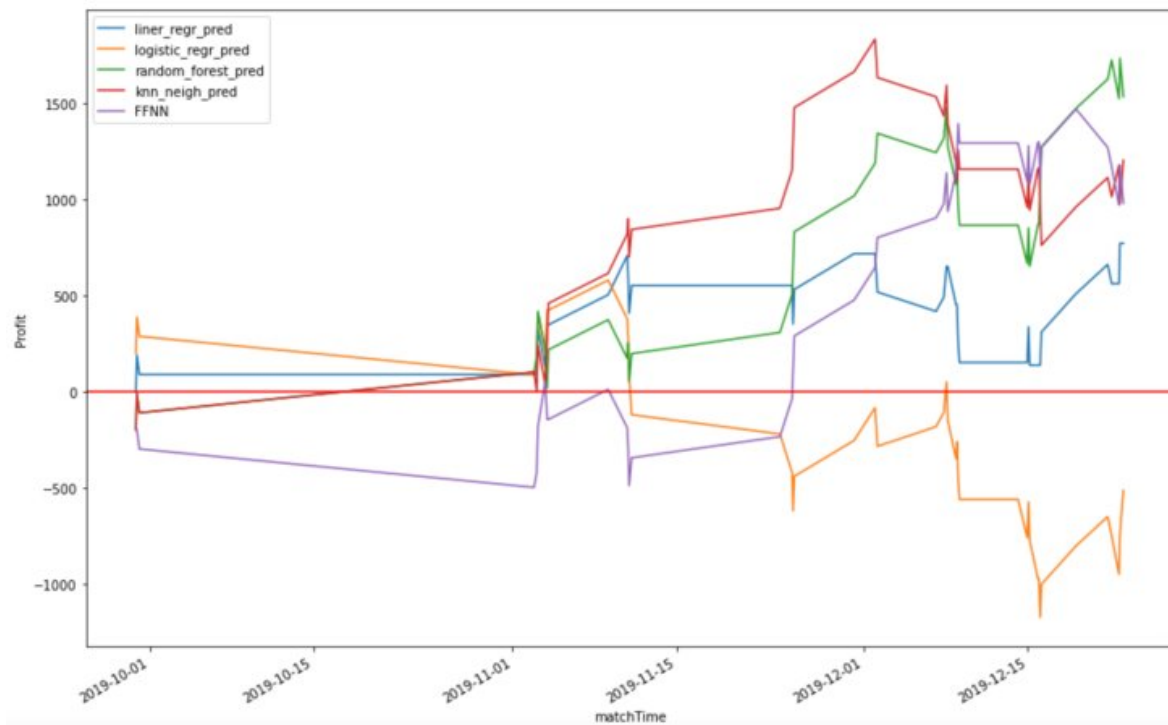


Figure 63. Predicted profit using selected features from best linear regression model

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 64. 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$scrown_hdc_home_first +
  fd$scrown_hdc_away_first + fd$scrown_hdc_home_last + fd$scrown_hdc_away_last +
  fd$scrown_hdc_initialFirst_home + fd$scrown_hdc_initialFirst_away +
  fd$scrown_hdc_initialLast_home + fd$scrown_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$scrown_hdc_hkjcFirst_home +
  fd$scrown_hdc_hkjcFirst_away + fd$macau_hdc_hkjcFirst_home +
  fd$macau_hdc_hkjcFirst_away + fd$bet365_hdc_hkjcLast_home +
  fd$bet365_hdc_hkjcLast_away + fd$scrown_hdc_hkjcLast_home +
  fd$scrown_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 65. 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.

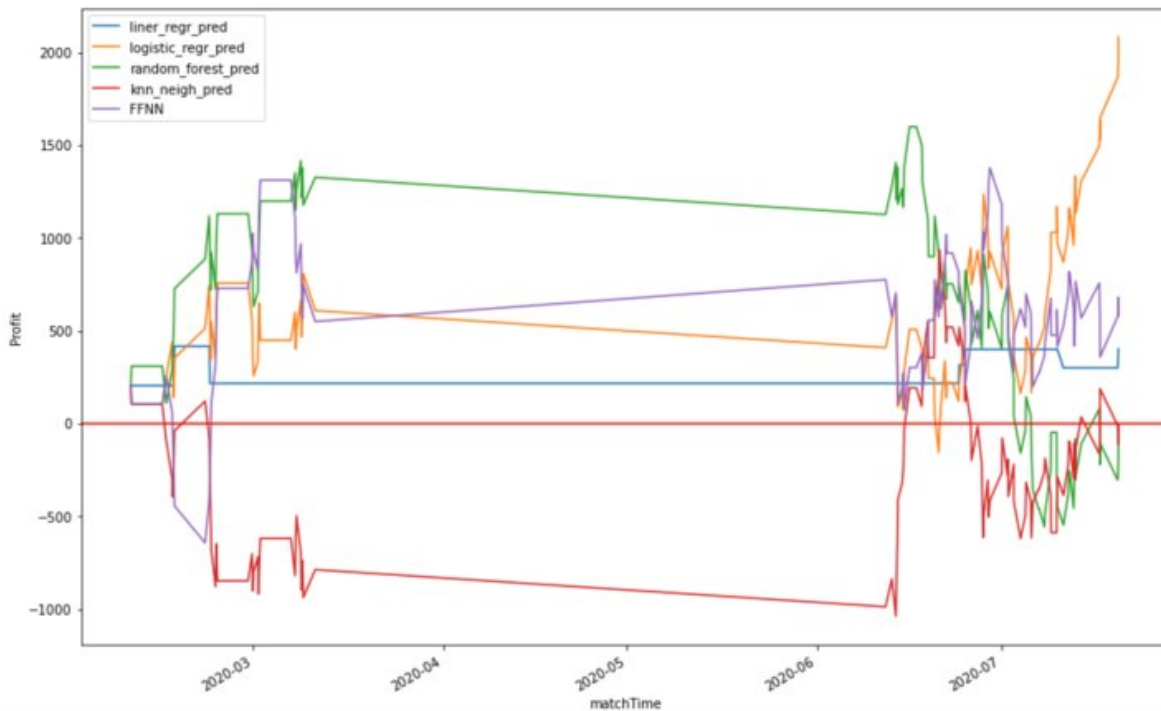


Figure 66. Predicted profit using selected features from best logistic regression model

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 [27]. In our case, the Support Vector Machine model with linear kernel will be used for recursive feature elimination (RFE) as mentioned in this paper [28].

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 [28] 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.

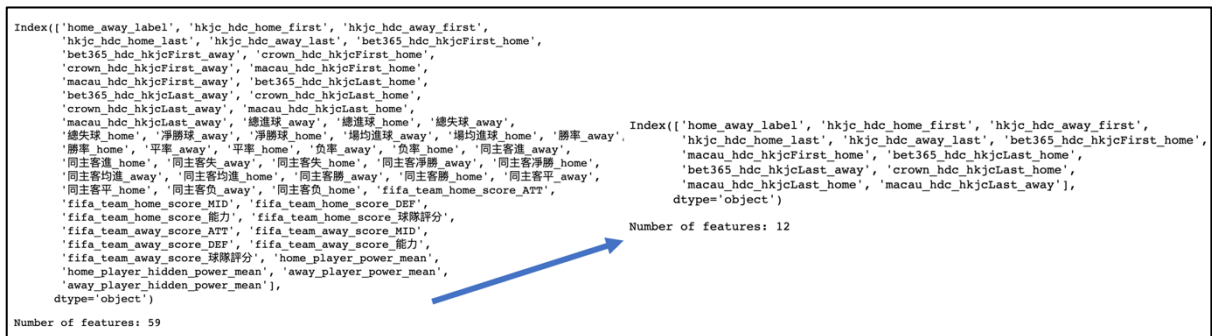


Figure 67. The feature before and after RFE.

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.

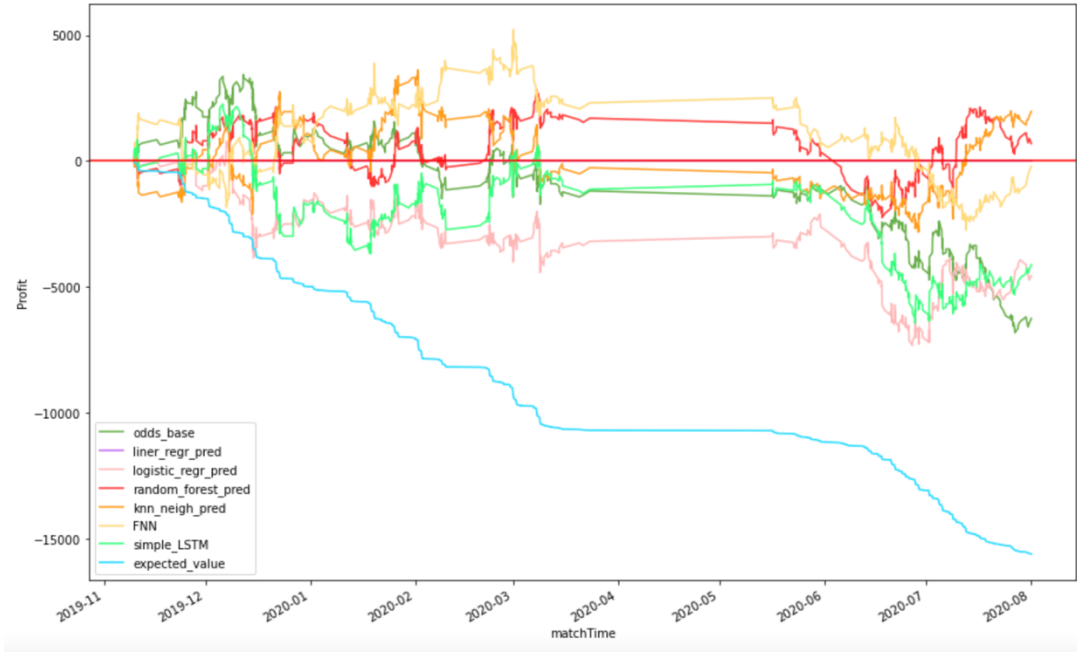


Figure 68. Stack Autoencoder for RFE

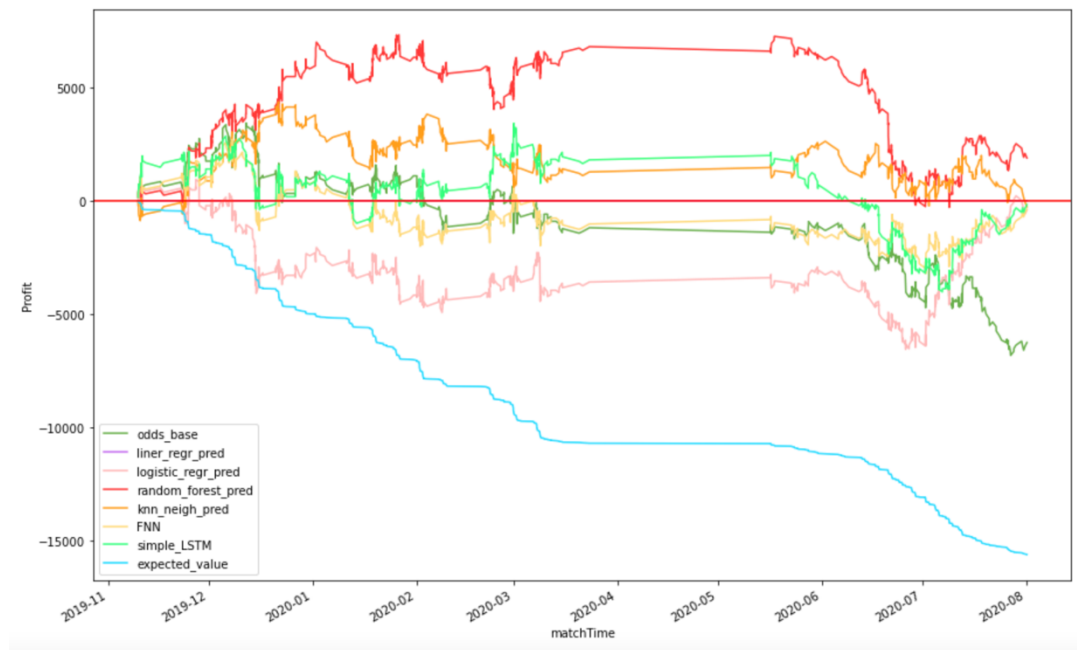


Figure 69. Deep Autoencoder for RFE

Figure 68 and Figure 69 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) which introduce in [Section 5.3.1](#) 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 70 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', '負率_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', '同主客負_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 70. Forward Selection by FNN*

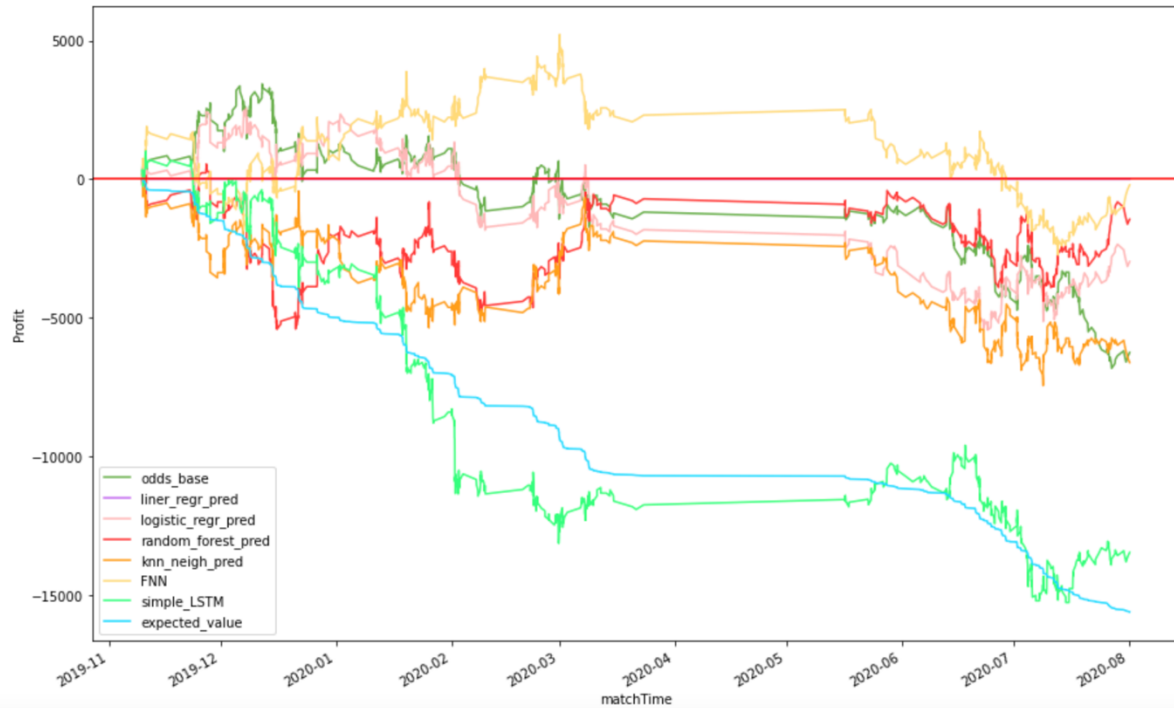


Figure 71. Stacked Autoencoder for Forward Selection

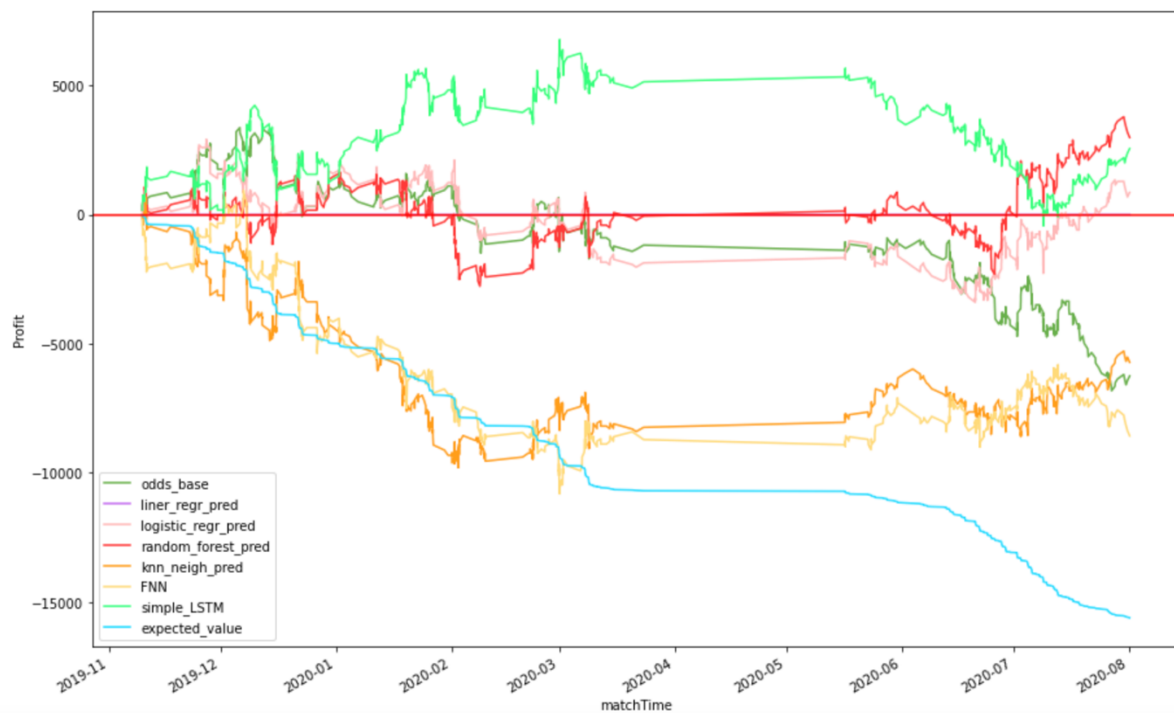


Figure 72. Deep Autoencoder for Forward Selection

Figure 71 and Figure 72 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 6 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.

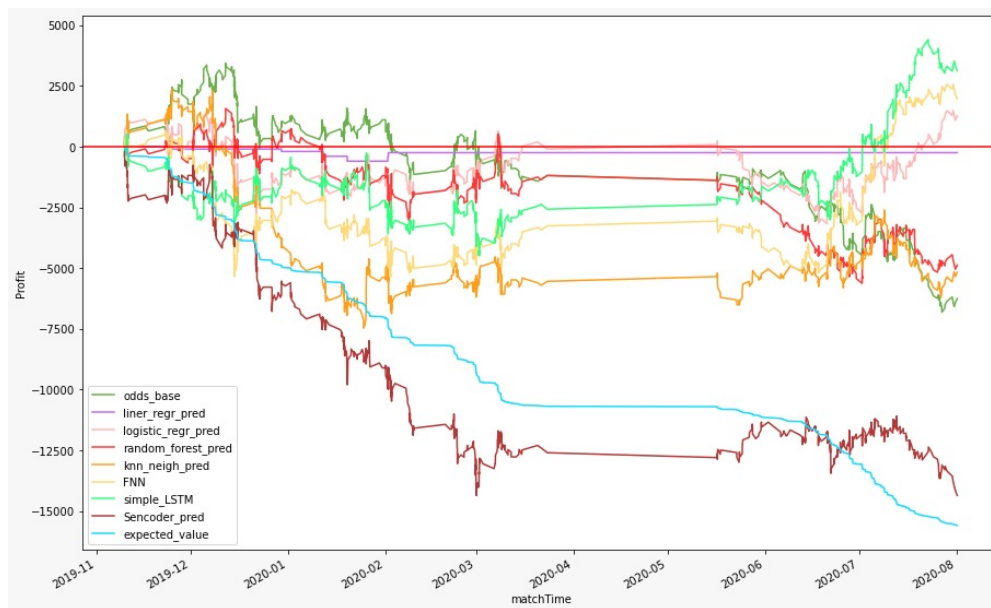
(all-league)	Expected Value	Odds based	Linear Regr	Logistic Regr	Random forest	KNN	FNN	LSTM	AVG
Base	-16000	-6250	-50	1250	100	-4500	-2000	-12500	-2950
Stacked RFE	-16000	-6250	0	-4500	500	2000	-500	-4000	-1083
Deep RFE	-16000	-6250	0	-100	2000	-100	-200	-10	265
Stacked FNNFS	-16000	-6250	0	-3000	-1500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

*Table 6. 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 with more than 200 matches.

### 6.7.1 All-league Model



*Figure 73. All-league model with odd-based features*

According to Figure 73, 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.

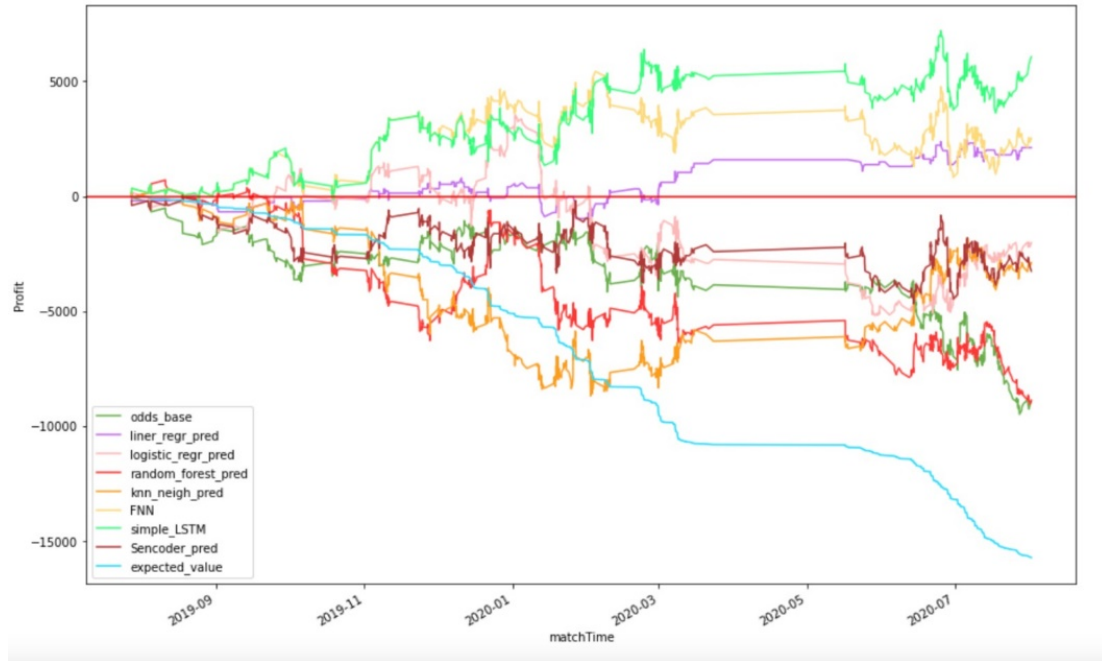


Figure 74. By-league model with odd-based features

According to the Figure 74, it is totally 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 7. 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.

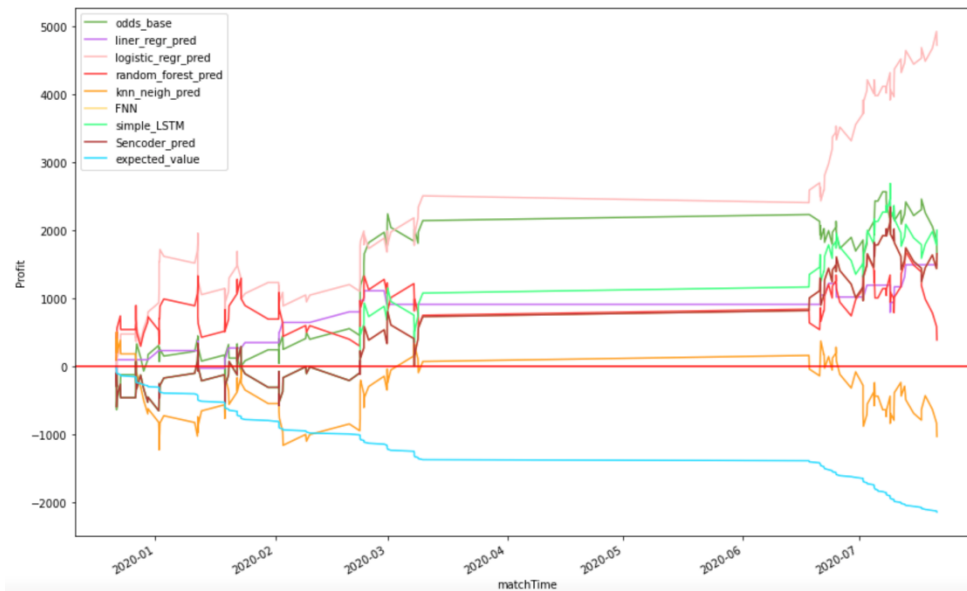


Figure 75. Evaluation result with odd-based features from the Premier League

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 pretty 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.

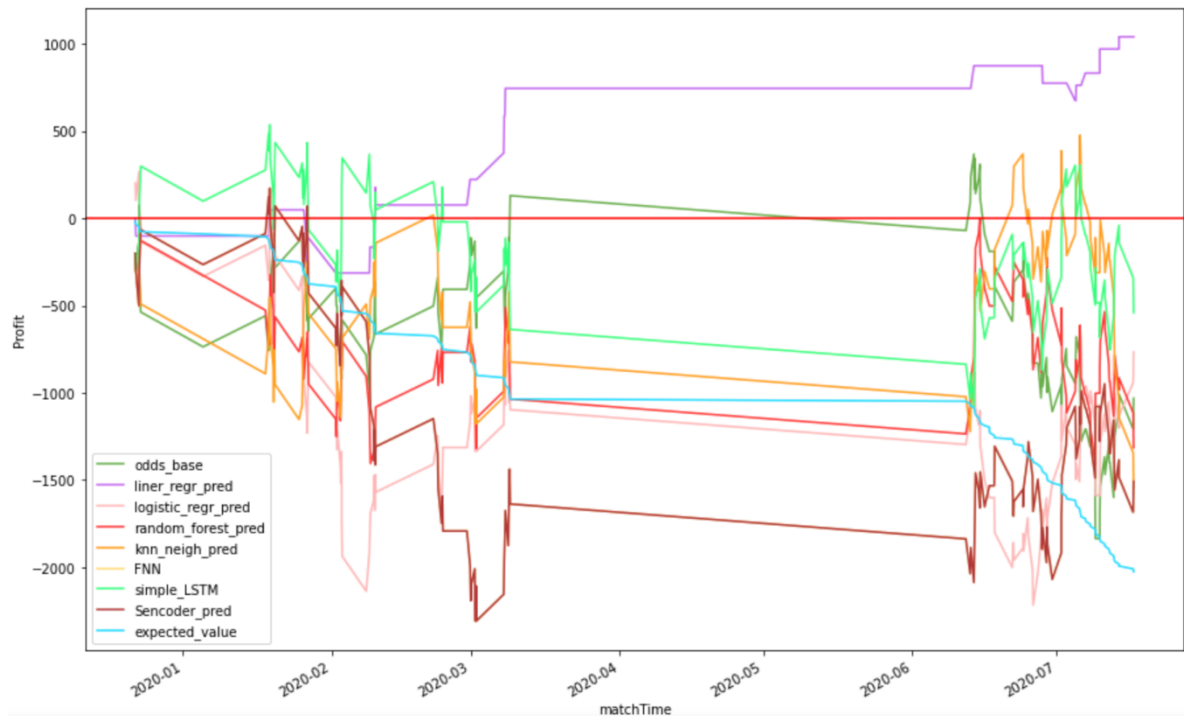


Figure 76. Evaluation result with odd-based features from the La Liga

According to Figure 76, 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. Limitation

### 7.1 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.

### 7.2 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 77. 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.

### 7.3 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.

## 8. Conclusion and Future Work

According to different analysis, it is proved that the handicap odds and handicap result are not dependent. Based on this finding, varying models are designed to predict the handicap result. In this report, combination of features and dimension reduction techniques are evaluated. Although most of the models can get a better performance than the expected value, the most stable and profitable model is still using the odd-based features only mentioned in Section 6.7.2. Thus, the main focus in the next semester will be improving the neural network models and the feature engineering, and designing a betting strategy.

Moreover, the Cluster-then-Predict Model will be followed up in the next semester. Further experiments on whether each cluster has their own specific meaning will be conducted. Moreover, betting strategy will be a one of the important directions in semester 2. For example, only the best machine learning model in each league and cluster in the by-league and Cluster-then-Predict model respectively will be considered.

Another goal is to build a deployable real-time betting program. The program should be able to use the pre-trained model and finalized betting strategy to perform prediction on the future matches in real time.

## 9. 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](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] 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>.

- [17] X. Song, A. Mitnitskib, J. Cox and K. Rockwood, "Comparison of Machine Learning Techniques with Classical Statistical Models in Predicting Health Outcomes," IOS Press, Amsterdam, 2004.
- [18] "Gini Index vs Information Entropy," [Online]. Available: <https://towardsdatascience.com/gini-index-vs-information-entropy-7a7e4fed3fcb>. [Accessed 20 11 2020].
- [19] Ian Goodfellow and Yoshua Bengio and Aaron Courville, "Deep Feedforward Networks," in *Deep Learning*, MIT Press, 2016, p. 164.
- [20] "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].
- [21] 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.
- [22] Z. Lin, *Recurrent Neural Network Models of Human Mobility*, University of California, Berkeley, 2018.
- [23] Liu Guifang, Bao Huaqian, Han Baokun, "A Stacked Autoencoder-Based Deep Neural Network for Achieving Gearbox Fault Diagnosis," 2018.
- [24] Rishabh Soni and K. James Mathai , "Improved Twitter Sentiment Prediction through 'Cluster-then-Predict Model'," 2015.
- [25] "K-Means Clustering in Python," [Online]. Available: <https://towardsdatascience.com/k-means-clustering-in-python-4061510145cc>. [Accessed 19 11 2020].
- [26] 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.
- [27] Kjell Johnson, Max Kuhn, "Forward, Backward, and Stepwise Selection," in *Applied Predictive Modeling*, Springer, 2013, p. 494.
- [28] Isabelle Guyon, Jason Weston, Stephen Barnhill, M.D., Vladimir Vapnik, Barnhill, "Gene Selection for Cancer Classification using Support Vector Machines".