DNN을 활용한 축구경기 결과 예측

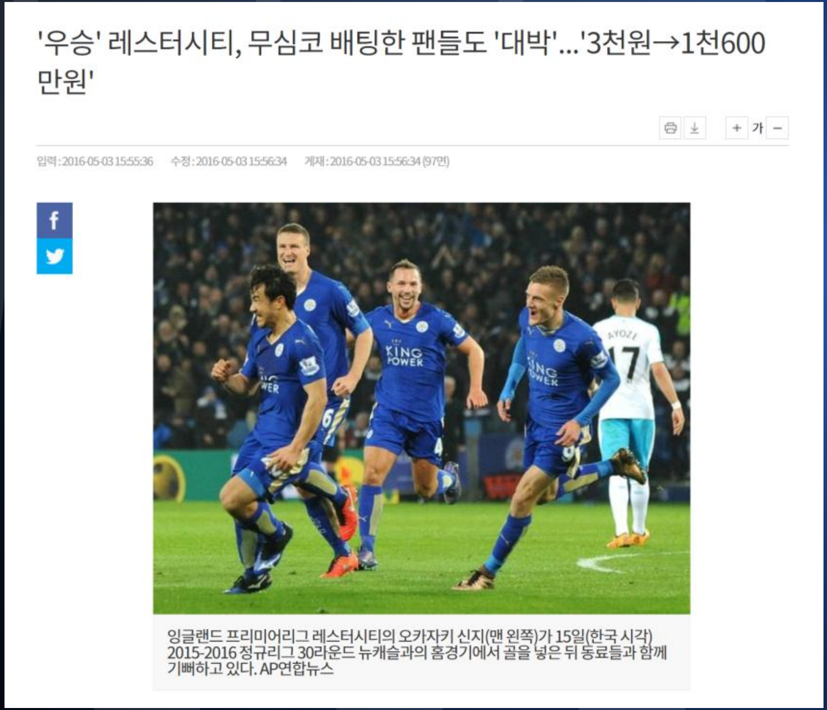
201910206 김태호

승부예측의 중요성

- 스포츠 팬들에게 더욱 흥미로운 경기 제공
- 경기결과에 대한 토론과 예측을 통해 사회적인 연결고리 형성
- 경기에 대한 지식과 분석력 향상
- **스포츠 배팅 배당률에 속지 않기 위함**
  - 배팅회사들의 배당률 함정 피할 수 있음

    : 배팅회사들이 회사에 유리하게 배당률 설정

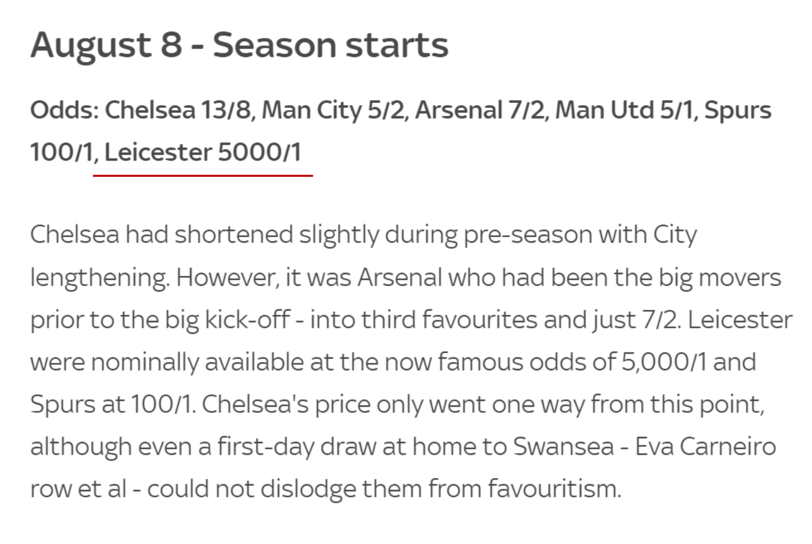
승부예측의 중요성



'우승' 레스터시티, 무심코 배팅한 팬들도 '대박'…'3천원→1천600만원'

입력 : 2016-05-03 15:55:36 수정 : 2016-05-03 15:56:34 게재 : 2016-05-03 15:56:34 (97면)

잉글랜드 프리미어리그 레스터시티의 오카자키 신지(맨 왼쪽)가 15일(한국 시각) 2015-2016 정규리그 30라운드 뉴캐슬과의 홈경기에서 골을 넣은 뒤 동료들과 함께 기뻐하고 있다. AP연합뉴스



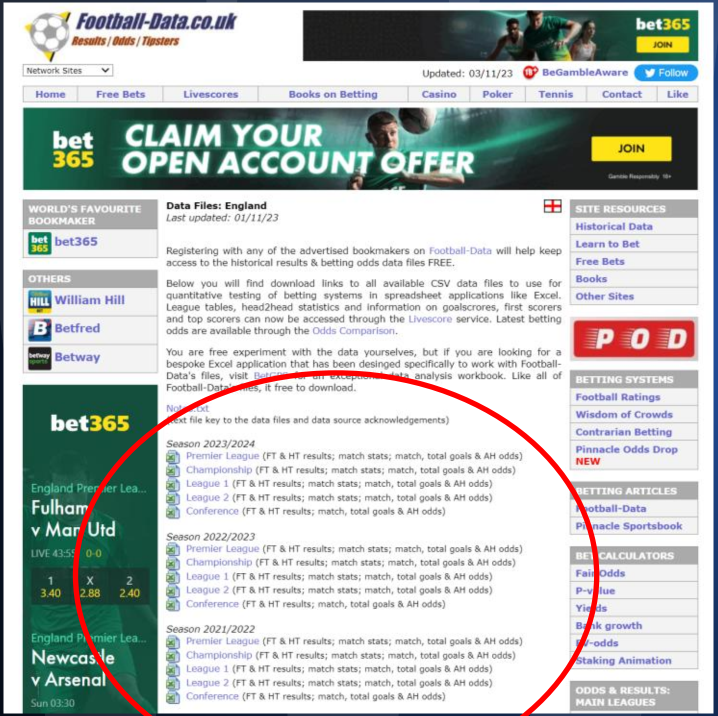
August 8 - Season starts

Odds: Chelsea 13/8, Man City 5/2, Arsenal 7/2, Man Utd 5/1, Spurs 100/1, Leicester 5000/1

Chelsea had shortened slightly during pre-season with City lengthening. However, it was Arsenal who had been the big movers prior to the big kick-off - into third favourites and just 7/2. Leicester were nominally available at the now famous odds of 5,000/1 and Spurs at 100/1. Chelsea's price only went one way from this point, although even a first-day draw at home to Swansea - Eva Carneiro row et al - could not dislodge them from favouritism.

딥러닝을 활용한 잉글랜드 프리미어리그 순위 예측

I. 데이터 수집



Football-Data.co.uk
Results / Odds / Tipsters

Network Sites

Updated: 03/11/23 BeGambleAware Follow

Home | Free Bets | Livescores | Books on Betting | Casino | Poker | Tennis | Contact | Like

WORLD'S FAVOURITE BOOKMAKER
bet365

OTHERS
William Hill
Betfred
Betway

Data Files: England
Last updated: 01/11/23

Registering with any of the advertised bookmakers on Football-Data will help keep access to the historical results & betting odds data files FREE.

Below you will find download links to all available CSV data files to use for quantitative testing of betting systems in spreadsheet applications like Excel. League tables, head2head statistics and information on goalscrores, first scorers and top scorers can now be accessed through the Livescore service. Latest betting odds are available through the Odds Comparison.

You are free experiment with the data yourselves, but if you are looking for a bespoke Excel application that has been desinged specifically to work with Football-Data's files, visit BetGPS for an exceptional data analysis workbook. Like all of Football-Data's files, it free to download.

Notes.txt
(text file key to the data files and data source acknowledgements)

Season 2023/2024
Premier League (FT & HT results; match stats; match, total goals & AH odds)
Championship (FT & HT results; match stats; match, total goals & AH odds)
League 1 (FT & HT results; match stats; match, total goals & AH odds)
League 2 (FT & HT results; match stats; match, total goals & AH odds)
Conference (FT & HT results; match, total goals & AH odds)

Season 2022/2023
Premier League (FT & HT results; match stats; match, total goals & AH odds)
Championship (FT & HT results; match stats; match, total goals & AH odds)
League 1 (FT & HT results; match stats; match, total goals & AH odds)
League 2 (FT & HT results; match stats; match, total goals & AH odds)
Conference (FT & HT results; match, total goals & AH odds)

Season 2021/2022
Premier League (FT & HT results; match stats; match, total goals & AH odds)
Championship (FT & HT results; match stats; match, total goals & AH odds)
League 1 (FT & HT results; match stats; match, total goals & AH odds)
League 2 (FT & HT results; match stats; match, total goals & AH odds)
Conference (FT & HT results; match, total goals & AH odds)

SITE RESOURCES
Historical Data
Learn to Bet
Free Bets
Books
Other Sites

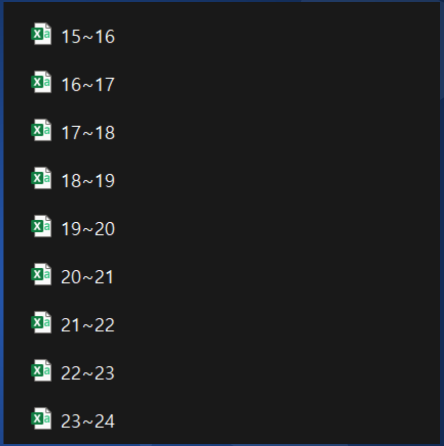
P O D

BETTING SYSTEMS
Football Ratings
Wisdom of Crowds
Contrarian Betting
Pinnacle Odds Drop NEW

BETTING ARTICLES
Football-Data
Pinnacle Sportsbook

BET CALCULATORS
Fair Odds
P-value
Yields
Bank growth
EV-odds
Staking Animation

ODDS & RESULTS: MAIN LEAGUES



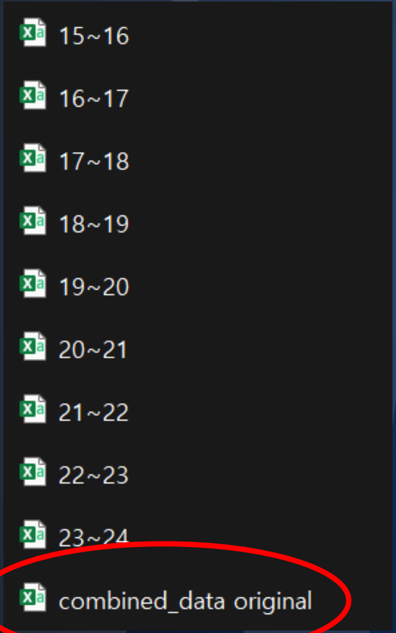
15~16
16~17
17~18
18~19
19~20
20~21
21~22
22~23
23~24

15-16 시즌부터 23-24 시즌까지 총 9개 시즌 정보

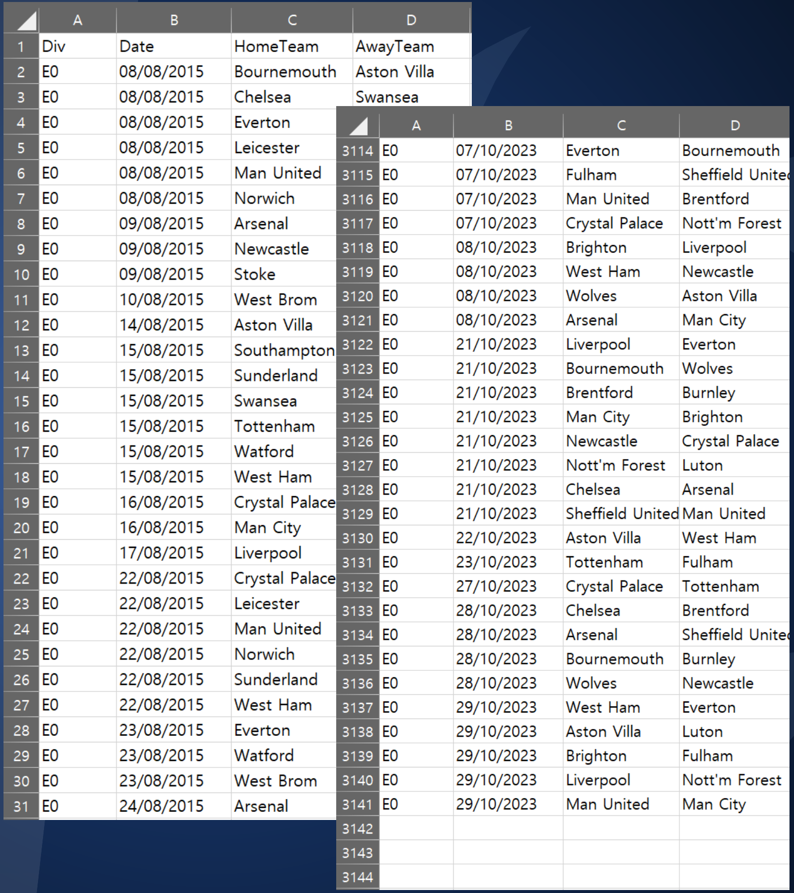
20팀 - 1R 당 10경기 - 한 시즌은 38R

380 X 8 + 100 = 3140 경기

I. 데이터 수집



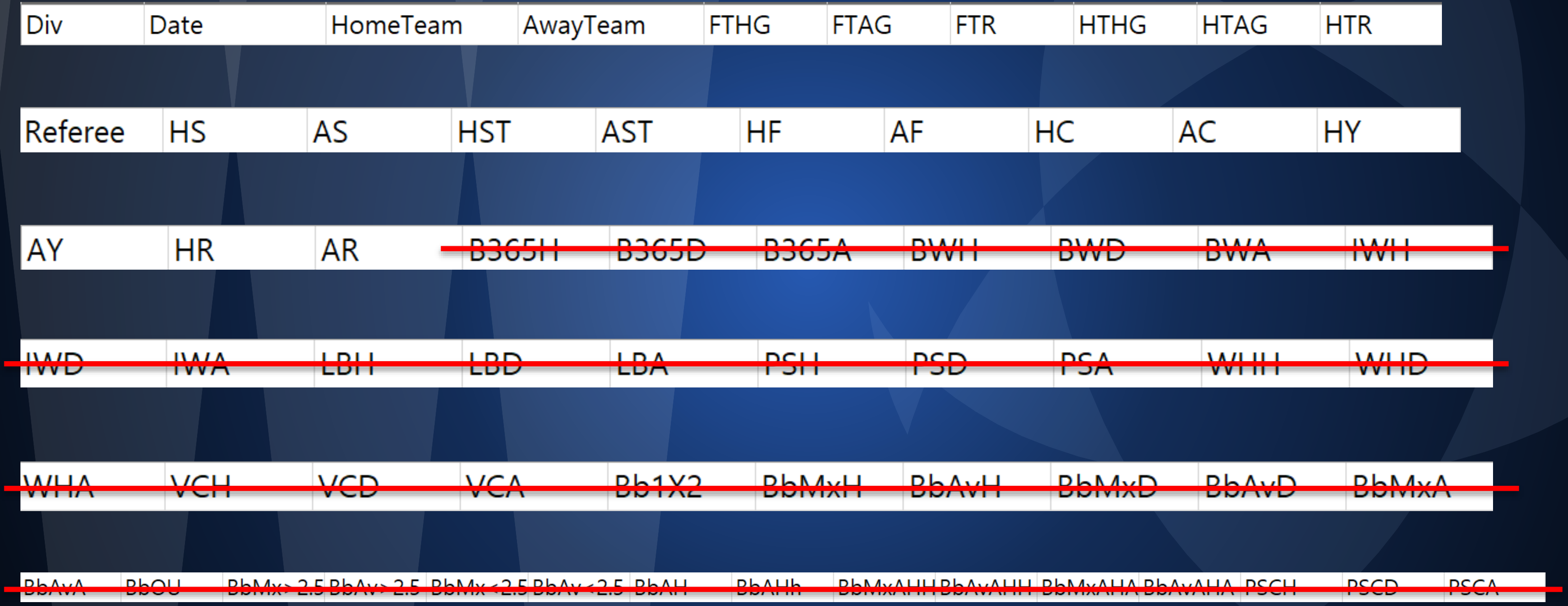
- 15~16
- 16~17
- 17~18
- 18~19
- 19~20
- 20~21
- 21~22
- 22~23
- 23~24
- combined_data original



	A	B	C	D
1	Div	Date	HomeTeam	AwayTeam
2	E0	08/08/2015	Bournemouth	Aston Villa
3	E0	08/08/2015	Chelsea	Swansea
4	E0	08/08/2015	Everton	
5	E0	08/08/2015	Leicester	
6	E0	08/08/2015	Man United	
7	E0	08/08/2015	Norwich	
8	E0	09/08/2015	Arsenal	
9	E0	09/08/2015	Newcastle	
10	E0	09/08/2015	Stoke	
11	E0	10/08/2015	West Brom	
12	E0	14/08/2015	Aston Villa	
13	E0	15/08/2015	Southampton	
14	E0	15/08/2015	Sunderland	
15	E0	15/08/2015	Swansea	
16	E0	15/08/2015	Tottenham	
17	E0	15/08/2015	Watford	
18	E0	15/08/2015	West Ham	
19	E0	16/08/2015	Crystal Palace	
20	E0	16/08/2015	Man City	
21	E0	17/08/2015	Liverpool	
22	E0	22/08/2015	Crystal Palace	
23	E0	22/08/2015	Leicester	
24	E0	22/08/2015	Man United	
25	E0	22/08/2015	Norwich	
26	E0	22/08/2015	Sunderland	
27	E0	22/08/2015	West Ham	
28	E0	23/08/2015	Everton	
29	E0	23/08/2015	Watford	
30	E0	23/08/2015	West Brom	
31	E0	24/08/2015	Arsenal	

	A	B	C	D
3114	E0	07/10/2023	Everton	Bournemouth
3115	E0	07/10/2023	Fulham	Sheffield Unite
3116	E0	07/10/2023	Man United	Brentford
3117	E0	07/10/2023	Crystal Palace	Nott'm Forest
3118	E0	08/10/2023	Brighton	Liverpool
3119	E0	08/10/2023	West Ham	Newcastle
3120	E0	08/10/2023	Wolves	Aston Villa
3121	E0	08/10/2023	Arsenal	Man City
3122	E0	21/10/2023	Liverpool	Everton
3123	E0	21/10/2023	Bournemouth	Wolves
3124	E0	21/10/2023	Brentford	Burnley
3125	E0	21/10/2023	Man City	Brighton
3126	E0	21/10/2023	Newcastle	Crystal Palace
3127	E0	21/10/2023	Nott'm Forest	Luton
3128	E0	21/10/2023	Chelsea	Arsenal
3129	E0	21/10/2023	Sheffield United	Man United
3130	E0	22/10/2023	Aston Villa	West Ham
3131	E0	23/10/2023	Tottenham	Fulham
3132	E0	27/10/2023	Crystal Palace	Tottenham
3133	E0	28/10/2023	Chelsea	Brentford
3134	E0	28/10/2023	Arsenal	Sheffield Unite
3135	E0	28/10/2023	Bournemouth	Burnley
3136	E0	28/10/2023	Wolves	Newcastle
3137	E0	29/10/2023	West Ham	Everton
3138	E0	29/10/2023	Aston Villa	Luton
3139	E0	29/10/2023	Brighton	Fulham
3140	E0	29/10/2023	Liverpool	Nott'm Forest
3141	E0	29/10/2023	Man United	Man City
3142				
3143				
3144				

I I. 데이터 전처리



Div	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR

Referee	HS	AS	HST	AST	HF	AF	HC	AC	HY

AY	HR	AR	~~B365H~~	~~B365D~~	~~B365A~~	~~BWH~~	~~BWD~~	~~BWA~~	~~IWH~~

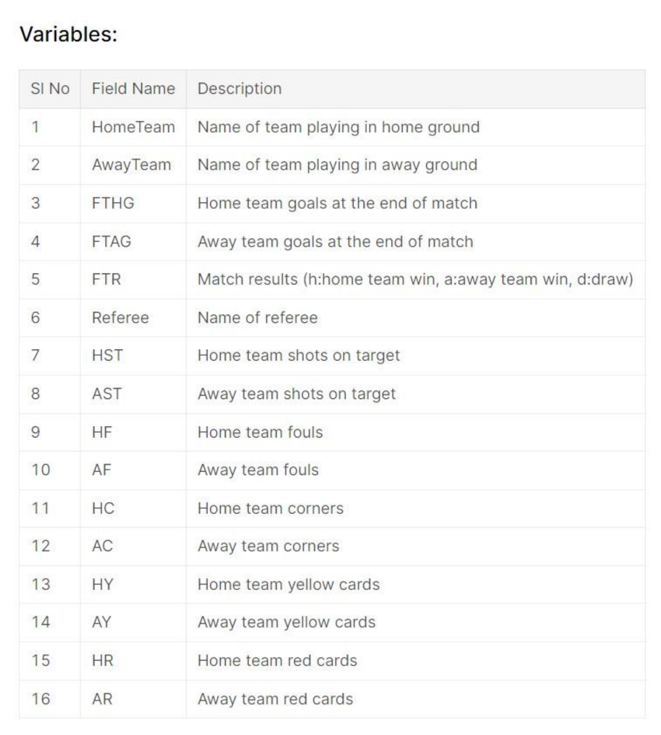
~~IWD~~	~~IWA~~	~~LBH~~	~~LBD~~	~~LBA~~	~~PSH~~	~~PSD~~	~~PSA~~	~~WHH~~	~~WHD~~

~~WHA~~	~~VCH~~	~~VCD~~	~~VCA~~	~~Bb1X2~~	~~BbMxH~~	~~BbAvH~~	~~BbMxD~~	~~BbAvD~~	~~BbMxA~~

~~BbAvA~~	~~BbOU~~	~~BbMx>2.5~~	~~BbAv>2.5~~	~~BbMx<2.5~~	~~BbAv<2.5~~	~~BbAH~~	~~BbAHh~~	~~BbMxAHH~~	~~BbAvAHH~~	~~BbMxAHA~~	~~BbAvAHA~~	~~PSCH~~	~~PSCD~~	~~PSCA~~

Column 총 56 개... -----------------→ Column 총 23 개!

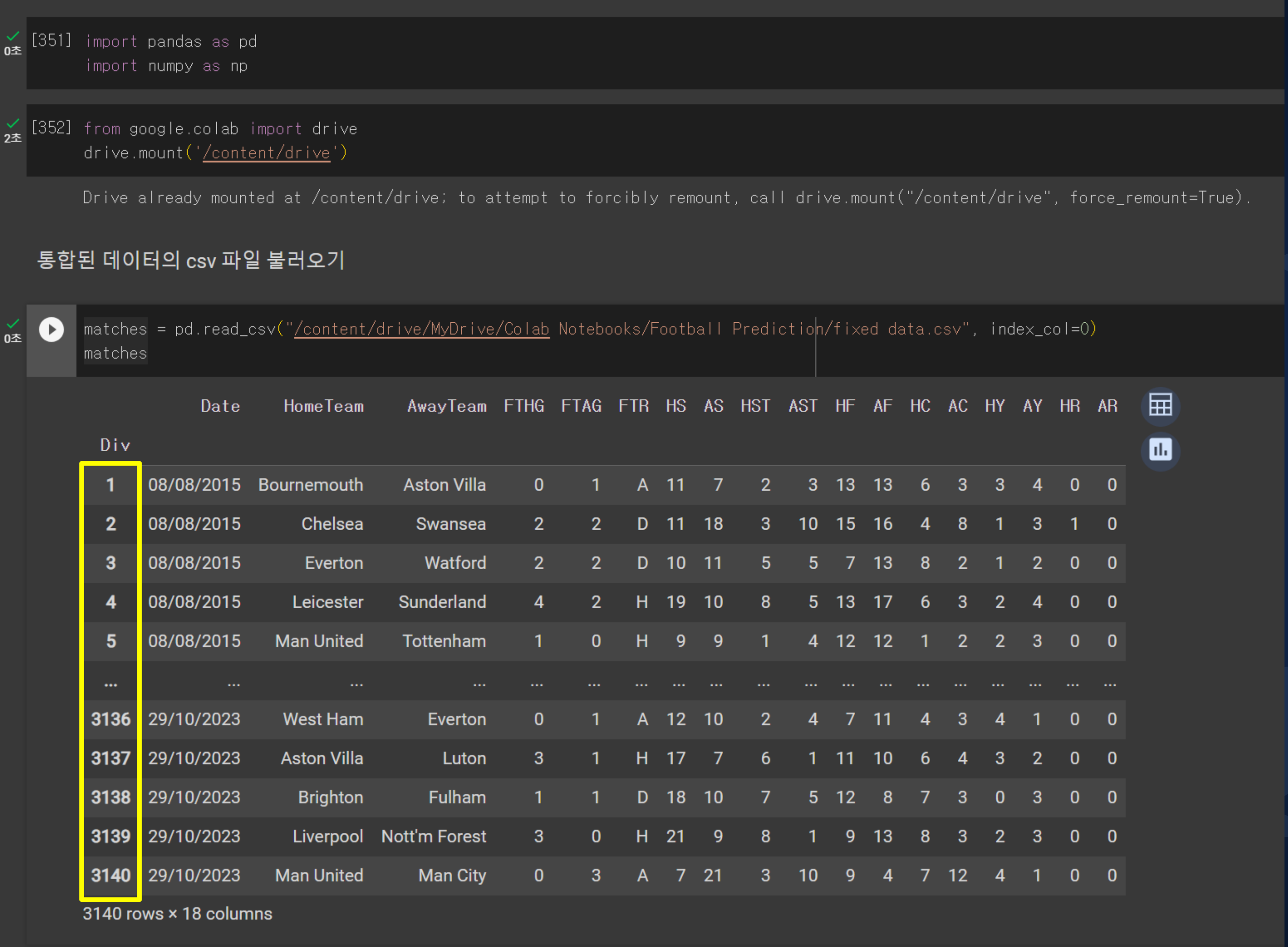
I I. 데이터 전처리



Variables:

Sl No	Field Name	Description
1	HomeTeam	Name of team playing in home ground
2	AwayTeam	Name of team playing in away ground
3	FTHG	Home team goals at the end of match
4	FTAG	Away team goals at the end of match
5	FTR	Match results (h:home team win, a:away team win, d:draw)
6	Referee	Name of referee
7	HST	Home team shots on target
8	AST	Away team shots on target
9	HF	Home team fouls
10	AF	Away team fouls
11	HC	Home team corners
12	AC	Away team corners
13	HY	Home team yellow cards
14	AY	Away team yellow cards
15	HR	Home team red cards
16	AR	Away team red cards

Date : 경기 날짜
HomeTeam : 홈에서 경기한 팀
AwayTeam : 원정에서 경기한 팀
FTHG : 홈 팀이 넣은 골
FTAG : 원정 팀이 넣은 골
FTR : 경기결과 (H - 홈 팀 승, A - 원정 팀 승, D - 무승부
Referee : 주심의 이름
HS : 홈 팀의 슛
AS : 원정 팀의 슛
HST : 홈 팀의 유효슈팅
AST : 원정 팀의 유효슈팅
HF : 홈 팀의 파울
AF : 원정 팀의 파울
HC : 홈 팀의 코너킥
AC : 원정 팀의 코너킥
HY : 홈 팀이 받은 옐로카드
AY : 원정 팀이 받은 옐로카드
HR : 홈 팀이 받은 레드카드
AR : 원정 팀이 받은 레드카드



```
[351] import pandas as pd
      import numpy as np
```

0초

```
[352] from google.colab import drive
      drive.mount('/content/drive')
```

2초

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

통합된 데이터의 csv 파일 불러오기

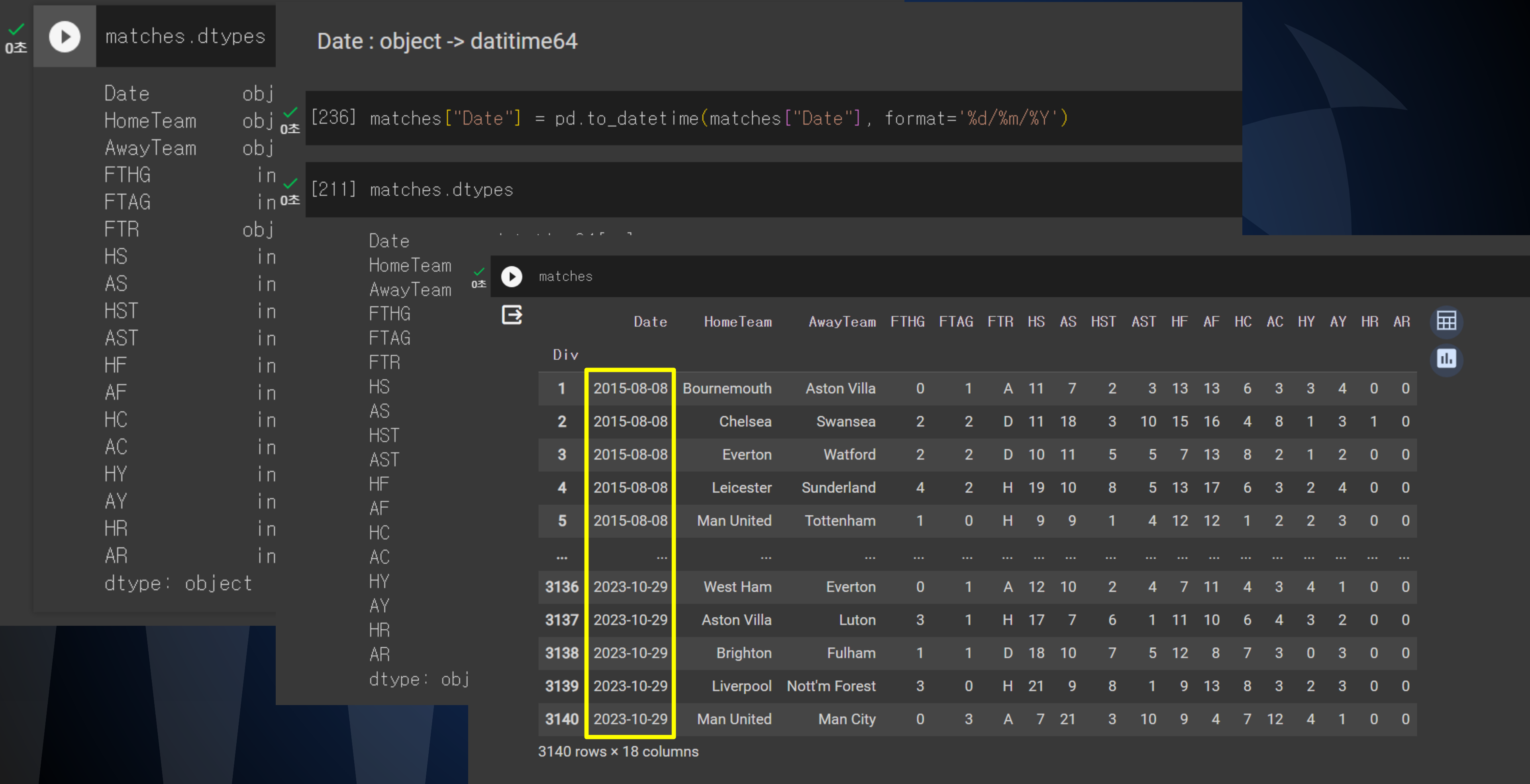
```
matches = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Football Prediction/fixed data.csv", index_col=0)
matches
```

0초

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HS	AS	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
Div																		
1	08/08/2015	Bournemouth	Aston Villa	0	1	A	11	7	2	3	13	13	6	3	3	4	0	0
2	08/08/2015	Chelsea	Swansea	2	2	D	11	18	3	10	15	16	4	8	1	3	1	0
3	08/08/2015	Everton	Watford	2	2	D	10	11	5	5	7	13	8	2	1	2	0	0
4	08/08/2015	Leicester	Sunderland	4	2	H	19	10	8	5	13	17	6	3	2	4	0	0
5	08/08/2015	Man United	Tottenham	1	0	H	9	9	1	4	12	12	1	2	2	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3136	29/10/2023	West Ham	Everton	0	1	A	12	10	2	4	7	11	4	3	4	1	0	0
3137	29/10/2023	Aston Villa	Luton	3	1	H	17	7	6	1	11	10	6	4	3	2	0	0
3138	29/10/2023	Brighton	Fulham	1	1	D	18	10	7	5	12	8	7	3	0	3	0	0
3139	29/10/2023	Liverpool	Nott'm Forest	3	0	H	21	9	8	1	9	13	8	3	2	3	0	0
3140	29/10/2023	Man United	Man City	0	3	A	7	21	3	10	9	4	7	12	4	1	0	0

3140 rows × 18 columns

각 Column의 dtypes



```
matches.dtypes
```

```
Date         obj
HomeTeam     obj
AwayTeam     obj
FTHG          in
FTAG          in
FTR          obj
HS            in
AS            in
HST           in
AST           in
HF            in
AF            in
HC            in
AC            in
HY            in
AY            in
HR            in
AR            in
dtype: object
```

Date : object -> datitime64

```
[236] matches["Date"] = pd.to_datetime(matches["Date"], format='%d/%m/%Y')
```

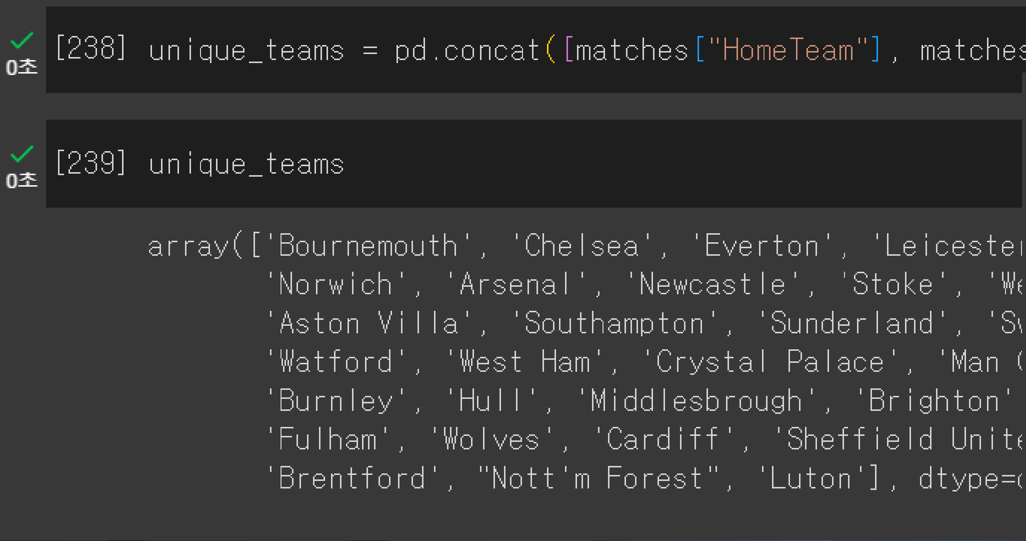
```
[211] matches.dtypes
```

```
Date
HomeTeam
AwayTeam
FTHG
FTAG
FTR
HS
AS
HST
AST
HF
AF
HC
AC
HY
AY
HR
AR
dtype: obj
```

```
matches
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HS	AS	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
Div																		
1	2015-08-08	Bournemouth	Aston Villa	0	1	A	11	7	2	3	13	13	6	3	3	4	0	0
2	2015-08-08	Chelsea	Swansea	2	2	D	11	18	3	10	15	16	4	8	1	3	1	0
3	2015-08-08	Everton	Watford	2	2	D	10	11	5	5	7	13	8	2	1	2	0	0
4	2015-08-08	Leicester	Sunderland	4	2	H	19	10	8	5	13	17	6	3	2	4	0	0
5	2015-08-08	Man United	Tottenham	1	0	H	9	9	1	4	12	12	1	2	2	3	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3136	2023-10-29	West Ham	Everton	0	1	A	12	10	2	4	7	11	4	3	4	1	0	0
3137	2023-10-29	Aston Villa	Luton	3	1	H	17	7	6	1	11	10	6	4	3	2	0	0
3138	2023-10-29	Brighton	Fulham	1	1	D	18	10	7	5	12	8	7	3	0	3	0	0
3139	2023-10-29	Liverpool	Nott'm Forest	3	0	H	21	9	8	1	9	13	8	3	2	3	0	0
3140	2023-10-29	Man United	Man City	0	3	A	7	21	3	10	9	4	7	12	4	1	0	0

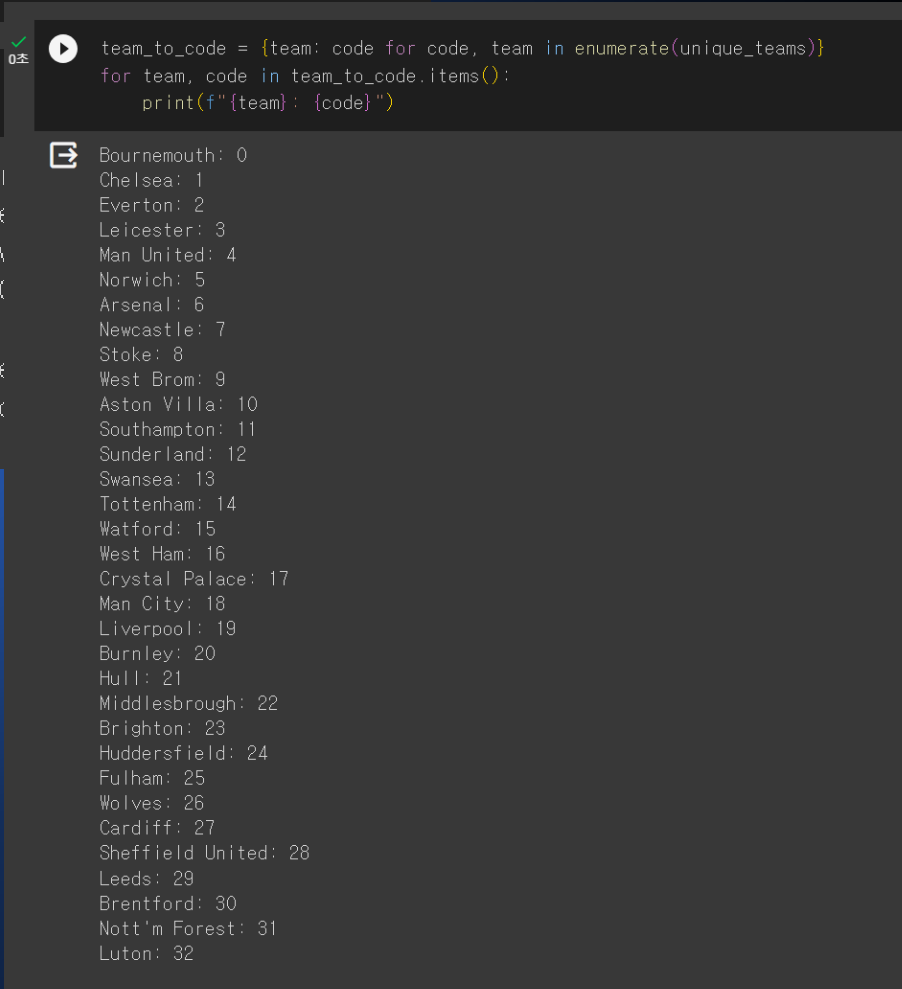
3140 rows × 18 columns



```
[238] unique_teams = pd.concat([matches["HomeTeam"], matches["AwayTeam"]]).unique()
```

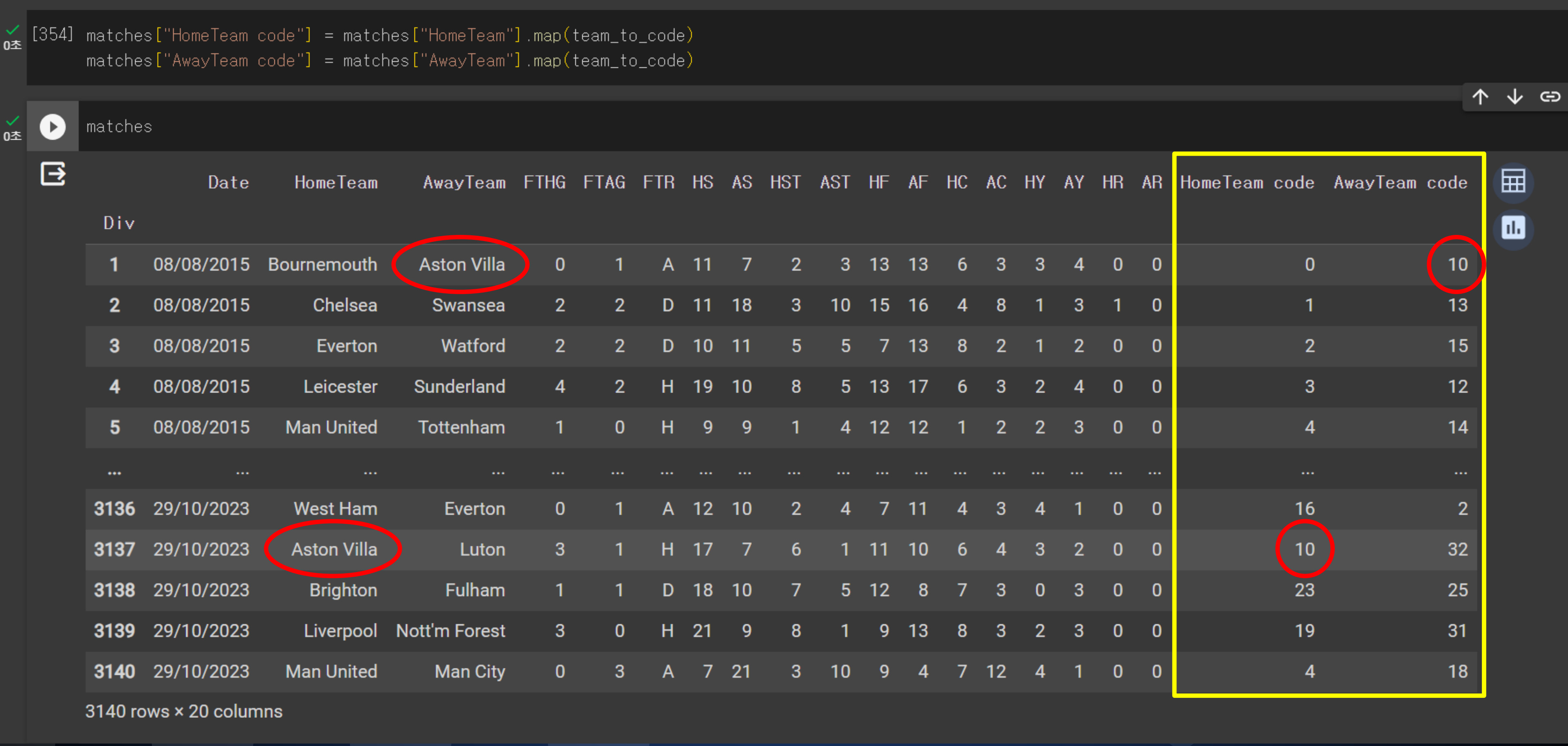
```
[239] unique_teams
```

```
array(['Bournemouth', 'Chelsea', 'Everton', 'Leiceste
       'Norwich', 'Arsenal', 'Newcastle', 'Stoke', 'We
       'Aston Villa', 'Southampton', 'Sunderland', 'S
       'Watford', 'West Ham', 'Crystal Palace', 'Man (
       'Burnley', 'Hull', 'Middlesbrough', 'Brighton'
       'Fulham', 'Wolves', 'Cardiff', 'Sheffield Unite
       'Brentford', "Nott'm Forest", 'Luton'], dtype=(
```



```
team_to_code = {team: code for code, team in enumerate(unique_teams)}
for team, code in team_to_code.items():
    print(f"{team}: {code}")
```

```
Bournemouth: 0
Chelsea: 1
Everton: 2
Leicester: 3
Man United: 4
Norwich: 5
Arsenal: 6
Newcastle: 7
Stoke: 8
West Brom: 9
Aston Villa: 10
Southampton: 11
Sunderland: 12
Swansea: 13
Tottenham: 14
Watford: 15
West Ham: 16
Crystal Palace: 17
Man City: 18
Liverpool: 19
Burnley: 20
Hull: 21
Middlesbrough: 22
Brighton: 23
Huddersfield: 24
Fulham: 25
Wolves: 26
Cardiff: 27
Sheffield United: 28
Leeds: 29
Brentford: 30
Nott'm Forest: 31
Luton: 32
```

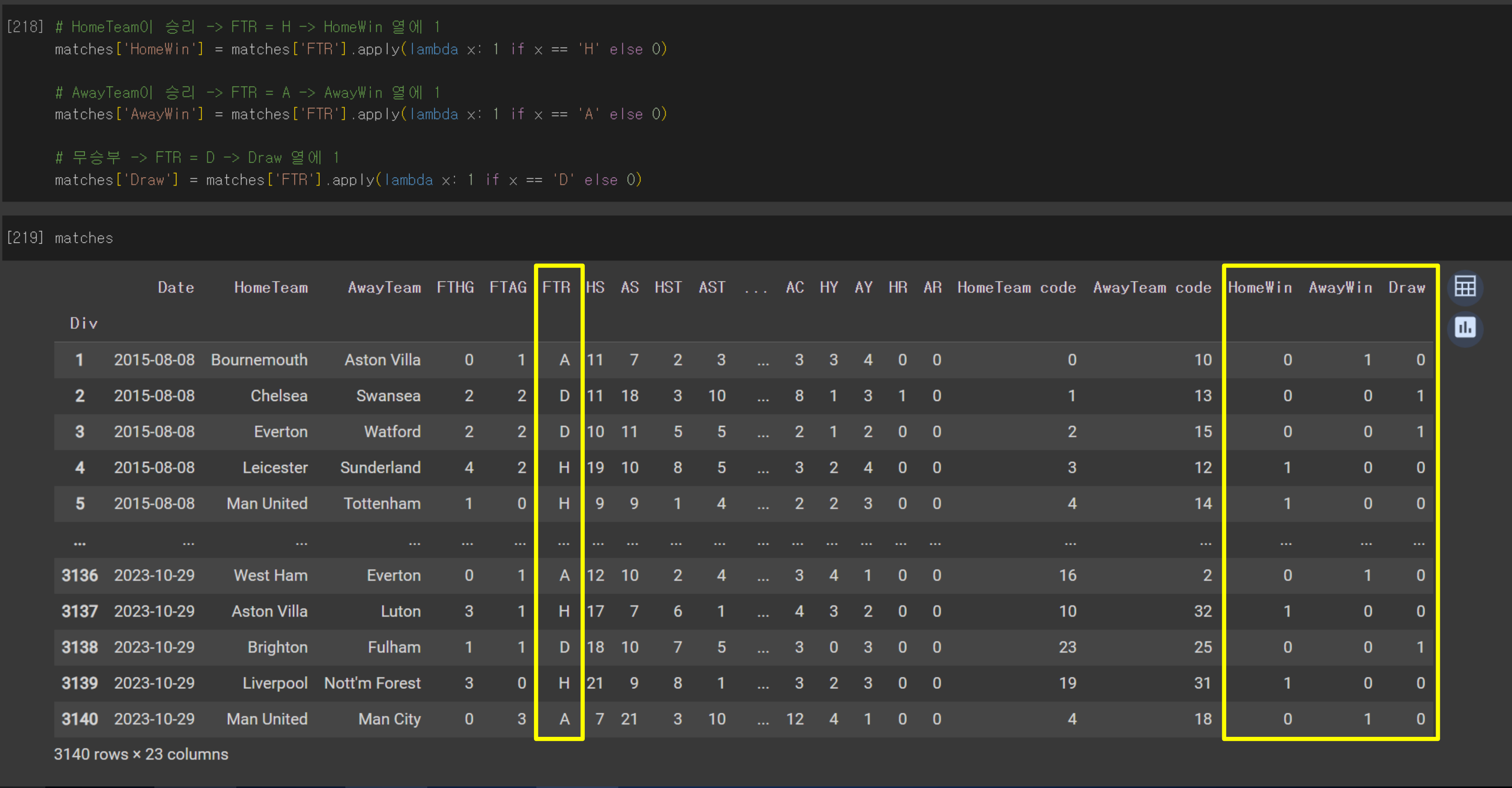


```
[354] matches["HomeTeam code"] = matches["HomeTeam"].map(team_to_code)
      matches["AwayTeam code"] = matches["AwayTeam"].map(team_to_code)
```

```
matches
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HS	AS	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR	HomeTeam code	AwayTeam code
Div																				
1	08/08/2015	Bournemouth	Aston Villa	0	1	A	11	7	2	3	13	13	6	3	3	4	0	0	0	10
2	08/08/2015	Chelsea	Swansea	2	2	D	11	18	3	10	15	16	4	8	1	3	1	0	1	13
3	08/08/2015	Everton	Watford	2	2	D	10	11	5	5	7	13	8	2	1	2	0	0	2	15
4	08/08/2015	Leicester	Sunderland	4	2	H	19	10	8	5	13	17	6	3	2	4	0	0	3	12
5	08/08/2015	Man United	Tottenham	1	0	H	9	9	1	4	12	12	1	2	2	3	0	0	4	14
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3136	29/10/2023	West Ham	Everton	0	1	A	12	10	2	4	7	11	4	3	4	1	0	0	16	2
3137	29/10/2023	Aston Villa	Luton	3	1	H	17	7	6	1	11	10	6	4	3	2	0	0	10	32
3138	29/10/2023	Brighton	Fulham	1	1	D	18	10	7	5	12	8	7	3	0	3	0	0	23	25
3139	29/10/2023	Liverpool	Nott'm Forest	3	0	H	21	9	8	1	9	13	8	3	2	3	0	0	19	31
3140	29/10/2023	Man United	Man City	0	3	A	7	21	3	10	9	4	7	12	4	1	0	0	4	18

3140 rows × 20 columns

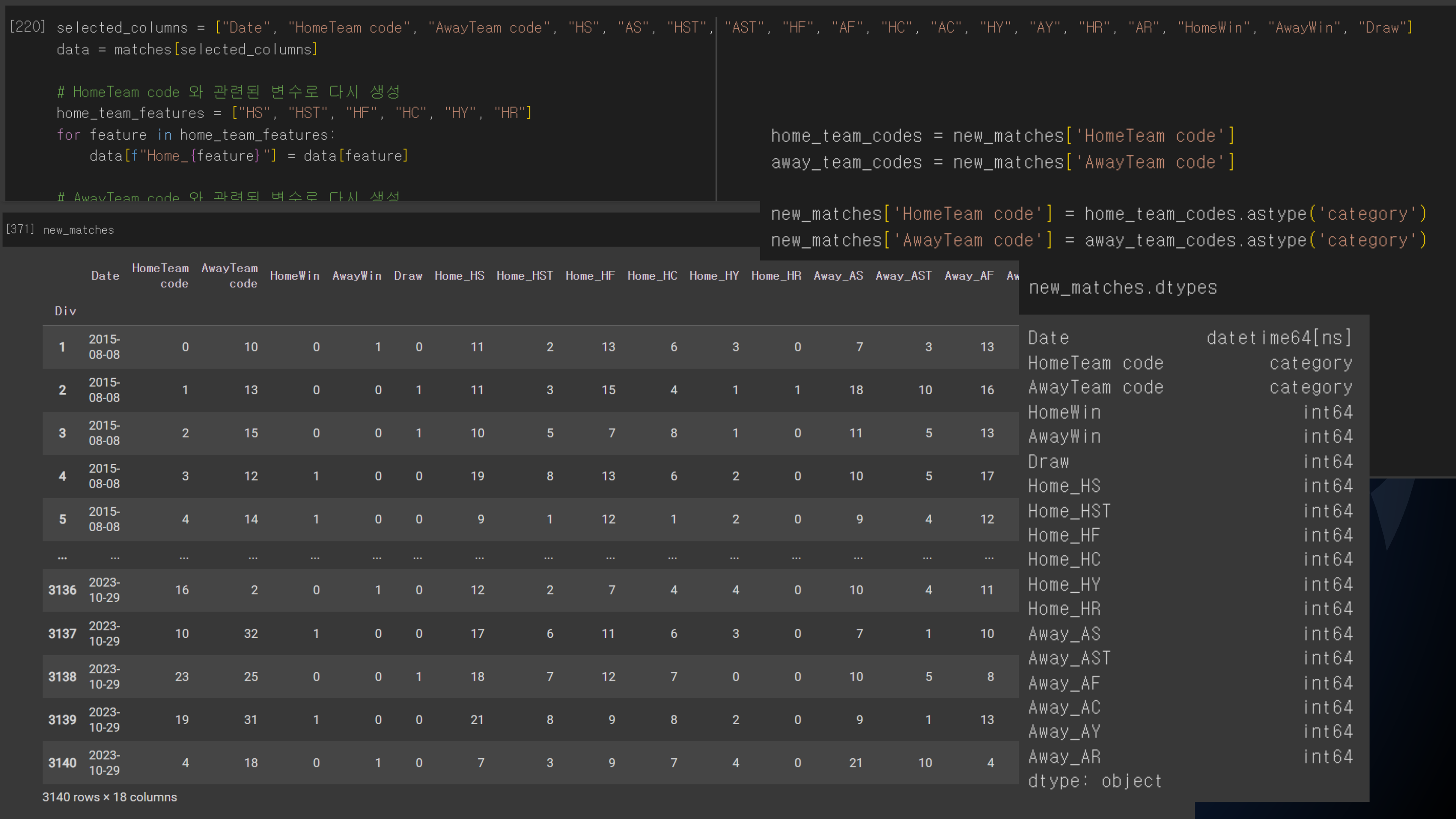


```
[218] # HomeTeam이 승리 -> FTR = H -> HomeWin 열에 1
      matches['HomeWin'] = matches['FTR'].apply(lambda x: 1 if x == 'H' else 0)

      # AwayTeam이 승리 -> FTR = A -> AwayWin 열에 1
      matches['AwayWin'] = matches['FTR'].apply(lambda x: 1 if x == 'A' else 0)

      # 무승부 -> FTR = D -> Draw 열에 1
      matches['Draw'] = matches['FTR'].apply(lambda x: 1 if x == 'D' else 0)
```

```
[219] matches
```

	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HS	AS	HST	AST	...	AC	HY	AY	HR	AR	HomeTeam code	AwayTeam code	HomeWin	AwayWin	Draw
Div																					
1	2015-08-08	Bournemouth	Aston Villa	0	1	A	11	7	2	3	...	3	3	4	0	0	0	10	0	1	0
2	2015-08-08	Chelsea	Swansea	2	2	D	11	18	3	10	...	8	1	3	1	0	1	13	0	0	1
3	2015-08-08	Everton	Watford	2	2	D	10	11	5	5	...	2	1	2	0	0	2	15	0	0	1
4	2015-08-08	Leicester	Sunderland	4	2	H	19	10	8	5	...	3	2	4	0	0	3	12	1	0	0
5	2015-08-08	Man United	Tottenham	1	0	H	9	9	1	4	...	2	2	3	0	0	4	14	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3136	2023-10-29	West Ham	Everton	0	1	A	12	10	2	4	...	3	4	1	0	0	16	2	0	1	0
3137	2023-10-29	Aston Villa	Luton	3	1	H	17	7	6	1	...	4	3	2	0	0	10	32	1	0	0
3138	2023-10-29	Brighton	Fulham	1	1	D	18	10	7	5	...	3	0	3	0	0	23	25	0	0	1
3139	2023-10-29	Liverpool	Nott'm Forest	3	0	H	21	9	8	1	...	3	2	3	0	0	19	31	1	0	0
3140	2023-10-29	Man United	Man City	0	3	A	7	21	3	10	...	12	4	1	0	0	4	18	0	1	0

3140 rows × 23 columns



```
[220] selected_columns = ["Date", "HomeTeam code", "AwayTeam code", "HS", "AS", "HST", "AST", "HF", "AF", "HC", "AC", "HY", "AY", "HR", "AR", "HomeWin", "AwayWin", "Draw"]
      data = matches[selected_columns]

      # HomeTeam code 와 관련된 변수로 다시 생성
      home_team_features = ["HS", "HST", "HF", "HC", "HY", "HR"]
      for feature in home_team_features:
          data[f"Home_{feature}"] = data[feature]

      # AwayTeam code 와 관련된 변수로 다시 생성
```

```
home_team_codes = new_matches['HomeTeam code']
away_team_codes = new_matches['AwayTeam code']

new_matches['HomeTeam code'] = home_team_codes.astype('category')
new_matches['AwayTeam code'] = away_team_codes.astype('category')
```

```
[371] new_matches
```

	Date	HomeTeam code	AwayTeam code	HomeWin	AwayWin	Draw	Home_HS	Home_HST	Home_HF	Home_HC	Home_HY	Home_HR	Away_AS	Away_AST	Away_AF	Aw
Div																
1	2015-08-08	0	10	0	1	0	11	2	13	6	3	0	7	3	13	
2	2015-08-08	1	13	0	0	1	11	3	15	4	1	1	18	10	16	
3	2015-08-08	2	15	0	0	1	10	5	7	8	1	0	11	5	13	
4	2015-08-08	3	12	1	0	0	19	8	13	6	2	0	10	5	17	
5	2015-08-08	4	14	1	0	0	9	1	12	1	2	0	9	4	12	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
3136	2023-10-29	16	2	0	1	0	12	2	7	4	4	0	10	4	11	
3137	2023-10-29	10	32	1	0	0	17	6	11	6	3	0	7	1	10	
3138	2023-10-29	23	25	0	0	1	18	7	12	7	0	0	10	5	8	
3139	2023-10-29	19	31	1	0	0	21	8	9	8	2	0	9	1	13	
3140	2023-10-29	4	18	0	1	0	7	3	9	7	4	0	21	10	4	

3140 rows × 18 columns

```
new_matches.dtypes
```

```
Date             datetime64[ns]
HomeTeam code          category
AwayTeam code          category
HomeWin                   int64
AwayWin                   int64
Draw                      int64
Home_HS                   int64
Home_HST                  int64
Home_HF                   int64
Home_HC                   int64
Home_HY                   int64
Home_HR                   int64
Away_AS                   int64
Away_AST                  int64
Away_AF                   int64
Away_AC                   int64
Away_AY                   int64
Away_AR                   int64
dtype: object
```

I I I. 알고리즘 선택

기본적인 모델 구축 계획

모델 1 - MLP(Multi-layer Perceptron) 구조의 DNN(Deep Neural Network)

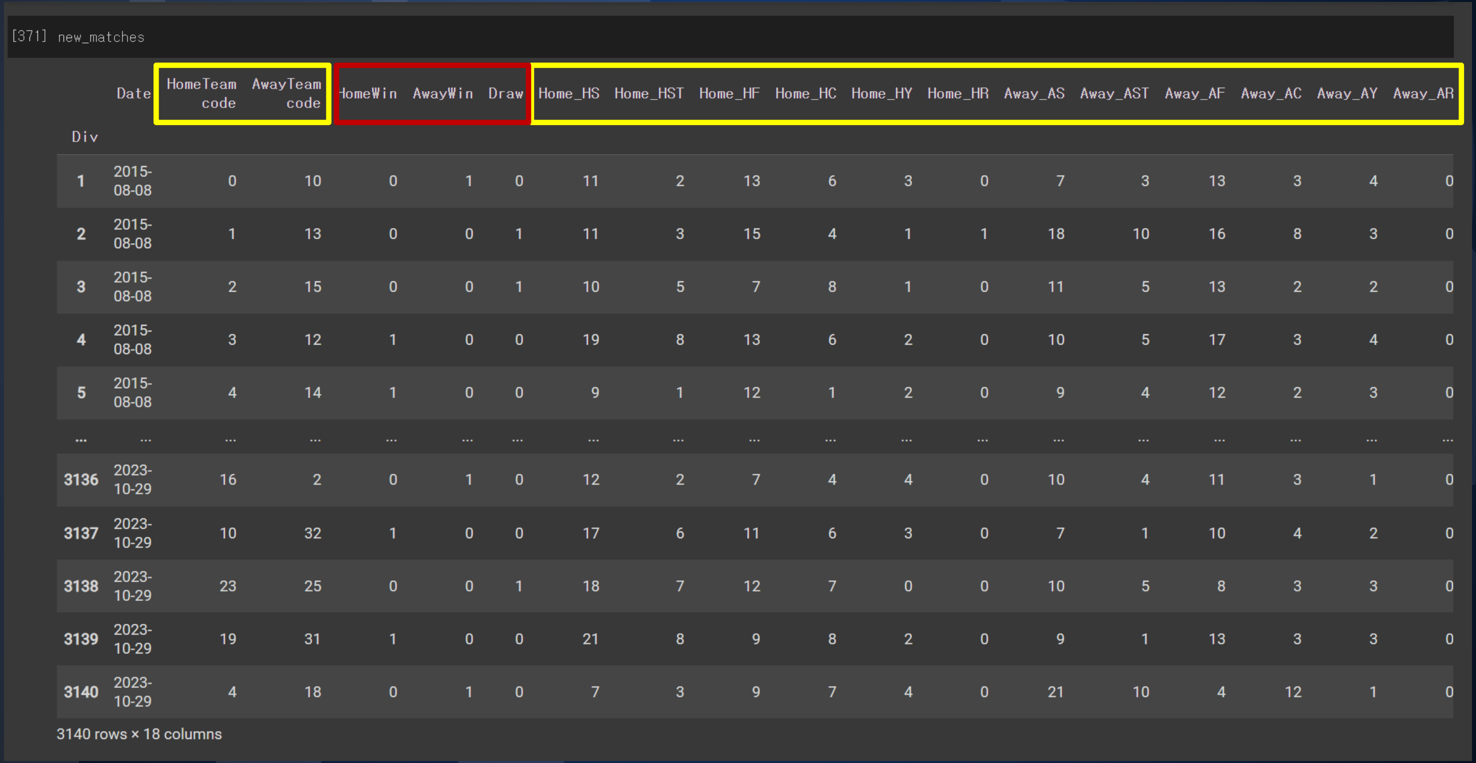
- X : 다양한 경기 정보 / y : 경기결과 승, 무, 패 분류

모델 2 - 모델 1 을 전이학습 시킨 모델

- X : HomeTeam code, AwayTeam code / Y : 경기의 승, 무, 패 확률 예측

IV. 모델 1

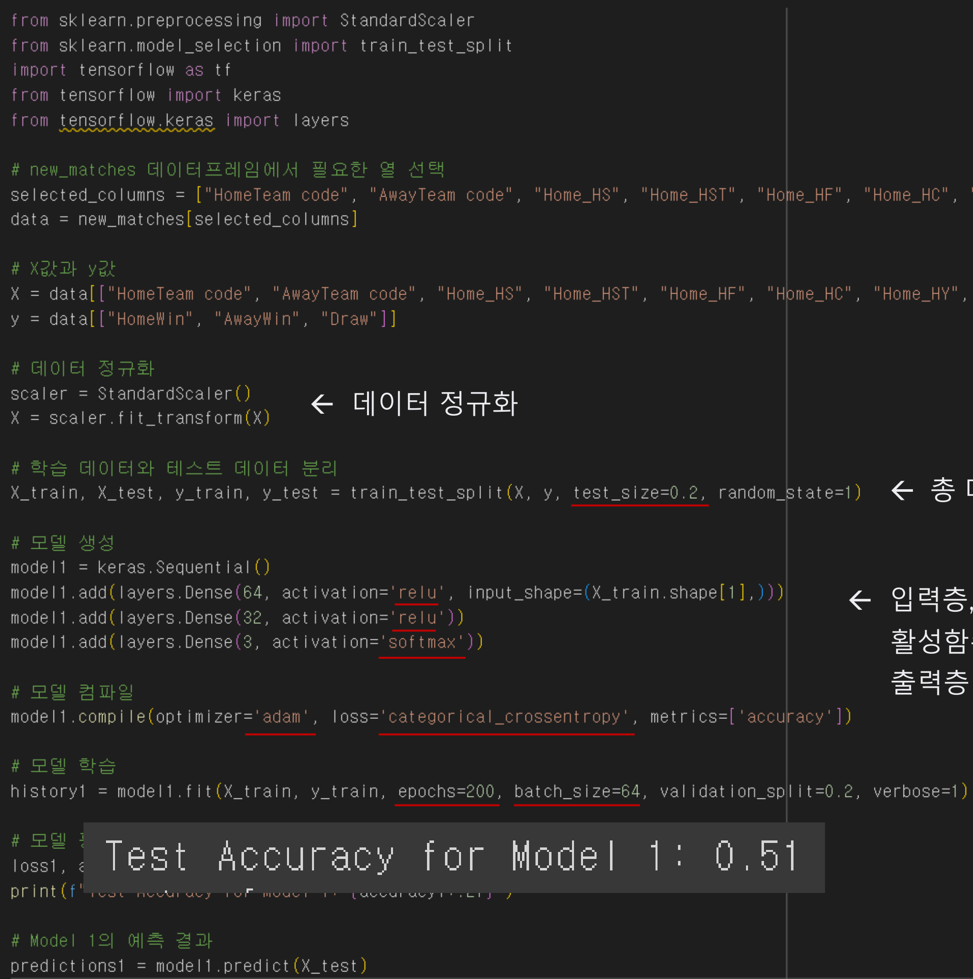
X값 / y값



```
[371] new_matches
```

Div	Date	HomeTeam code	AwayTeam code	HomeWin	AwayWin	Draw	Home_HS	Home_HST	Home_HF	Home_HC	Home_HY	Home_HR	Away_AS	Away_AST	Away_AF	Away_AC	Away_AY	Away_AR
1	2015-08-08	0	10	0	1	0	11	2	13	6	3	0	7	3	13	3	4	0
2	2015-08-08	1	13	0	0	1	11	3	15	4	1	1	18	10	16	8	3	0
3	2015-08-08	2	15	0	0	1	10	5	7	8	1	0	11	5	13	2	2	0
4	2015-08-08	3	12	1	0	0	19	8	13	6	2	0	10	5	17	3	4	0
5	2015-08-08	4	14	1	0	0	9	1	12	1	2	0	9	4	12	2	3	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3136	2023-10-29	16	2	0	1	0	12	2	7	4	4	0	10	4	11	3	1	0
3137	2023-10-29	10	32	1	0	0	17	6	11	6	3	0	7	1	10	4	2	0
3138	2023-10-29	23	25	0	0	1	18	7	12	7	0	0	10	5	8	3	3	0
3139	2023-10-29	19	31	1	0	0	21	8	9	8	2	0	9	1	13	3	3	0
3140	2023-10-29	4	18	0	1	0	7	3	9	7	4	0	21	10	4	12	1	0

3140 rows × 18 columns



```
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# new_matches 데이터프레임에서 필요한 열 선택
selected_columns = ["HomeTeam code", "AwayTeam code", "Home_HS", "Home_HST", "Home_HF", "Home_HC", "Home_HY", "Home_HR", "Away_AS", "Away_AST", "Away_AF", "Away_AC", "Away_AY", "Away_AR", "HomeWin"
data = new_matches[selected_columns]

# X값과 y값
X = data[["HomeTeam code", "AwayTeam code", "Home_HS", "Home_HST", "Home_HF", "Home_HC", "Home_HY", "Home_HR", "Away_AS", "Away_AST", "Away_AF", "Away_AC", "Away_AY", "Away_AR"]]
y = data[["HomeWin", "AwayWin", "Draw"]]

# 데이터 정규화
scaler = StandardScaler()
X = scaler.fit_transform(X)

# 학습 데이터와 테스트 데이터 분리
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)

# 모델 생성
model1 = keras.Sequential()
model1.add(layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)))
model1.add(layers.Dense(32, activation='relu'))
model1.add(layers.Dense(3, activation='softmax'))

# 모델 컴파일
model1.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# 모델 학습
history1 = model1.fit(X_train, y_train, epochs=200, batch_size=64, validation_split=0.2, verbose=1)

# 모델 
loss1, a
print(f'

# Model 1의 예측 결과
predictions1 = model1.predict(X_test)
```

```
Test Accuracy for Model 1: 0.51
```

← 데이터 정규화

← 총 데이터의 0.8

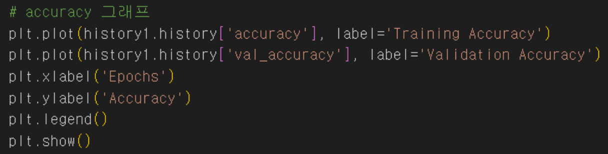
사용

← 입력층, 은닉층, 출

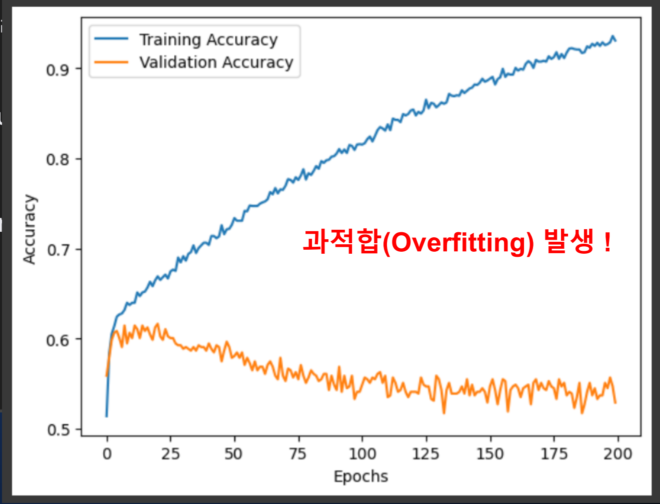
활성함수 (activatio

출력층 – softmax fu

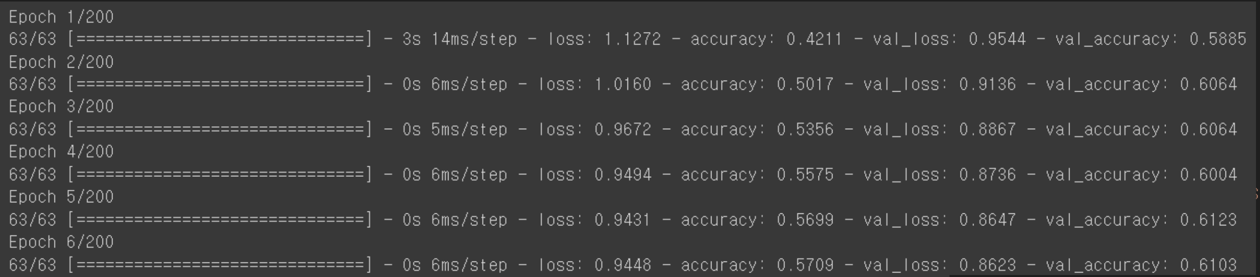
← Hyper



```
# accuracy 그래프
plt.plot(history1.history['accuracy'], label='Training Accuracy')
plt.plot(history1.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

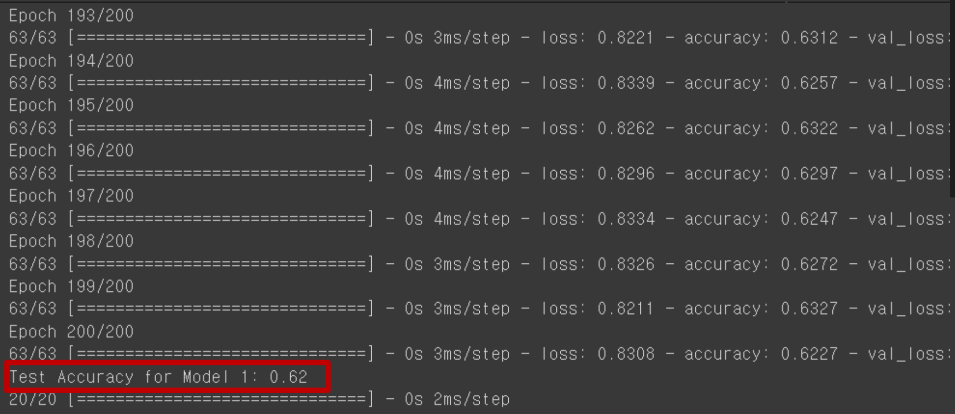


과적합(Overfitting) 발생 !

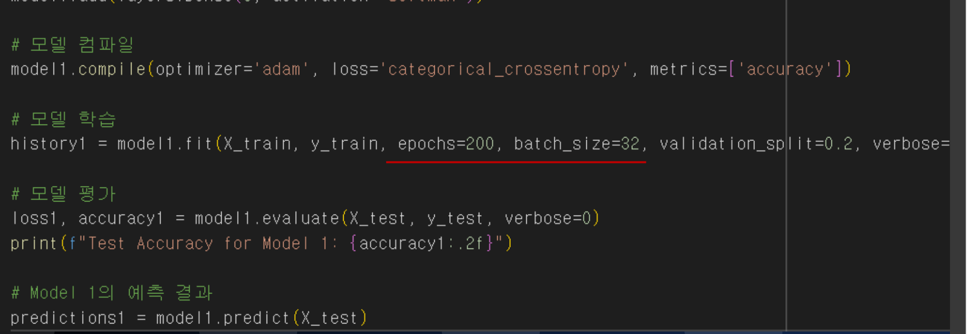


```
Epoch 1/200
63/63 [==============================] - 3s 14ms/step - loss: 1.1272 - accuracy: 0.4211 - val_loss: 0.9544 - val_accuracy: 0.5885
Epoch 2/200
63/63 [==============================] - 0s 6ms/step - loss: 1.0160 - accuracy: 0.5017 - val_loss: 0.9136 - val_accuracy: 0.6064
Epoch 3/200
63/63 [==============================] - 0s 5ms/step - loss: 0.9672 - accuracy: 0.5356 - val_loss: 0.8867 - val_accuracy: 0.6064
Epoch 4/200
63/63 [==============================] - 0s 6ms/step - loss: 0.9494 - accuracy: 0.5575 - val_loss: 0.8736 - val_accuracy: 0.6004
Epoch 5/200
63/63 [==============================] - 0s 6ms/step - loss: 0.9431 - accuracy: 0.5699 - val_loss: 0.8647 - val_accuracy: 0.6123
Epoch 6/200
63/63 [==============================] - 0s 6ms/step - loss: 0.9448 - accuracy: 0.5709 - val_loss: 0.8623 - val_accuracy: 0.6103
```

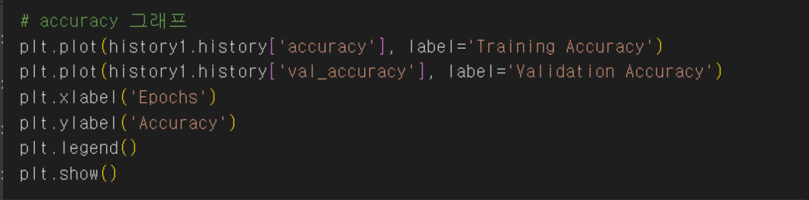
```
", "Away_AST", "Away_AF", "Away_AC", "Away_AY", "Away_AR", "HomeWin"
```



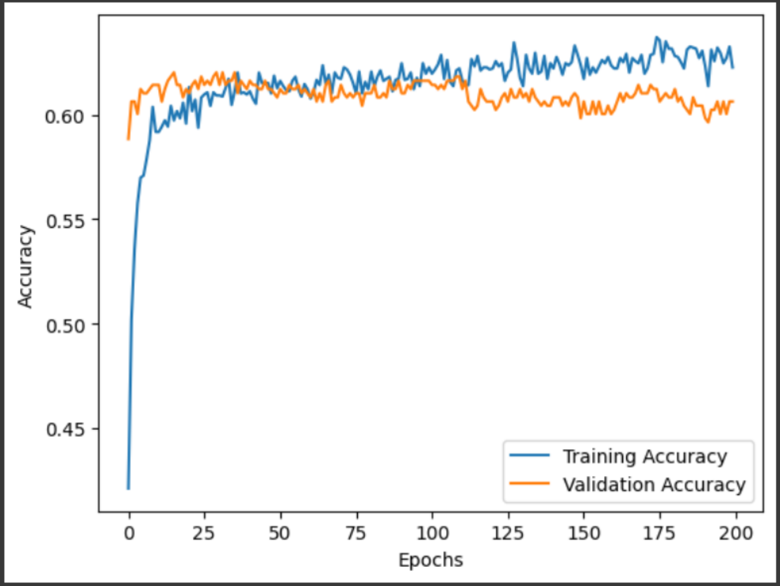
```
Epoch 193/200
63/63 [==============================] - 0s 3ms/step - loss: 0.8221 - accuracy: 0.6312 - val_loss:
Epoch 194/200
63/63 [==============================] - 0s 4ms/step - loss: 0.8339 - accuracy: 0.6257 - val_loss:
Epoch 195/200
63/63 [==============================] - 0s 4ms/step - loss: 0.8262 - accuracy: 0.6322 - val_loss:
Epoch 196/200
63/63 [==============================] - 0s 4ms/step - loss: 0.8296 - accuracy: 0.6297 - val_loss:
Epoch 197/200
63/63 [==============================] - 0s 4ms/step - loss: 0.8334 - accuracy: 0.6247 - val_loss:
Epoch 198/200
63/63 [==============================] - 0s 3ms/step - loss: 0.8326 - accuracy: 0.6272 - val_loss:
Epoch 199/200
63/63 [==============================] - 0s 3ms/step - loss: 0.8211 - accuracy: 0.6327 - val_loss:
Epoch 200/200
63/63 [==============================] - 0s 3ms/step - loss: 0.8308 - accuracy: 0.6227 - val_loss:
Test Accuracy for Model 1: 0.62
20/20 [==============================] - 0s 2ms/step
```

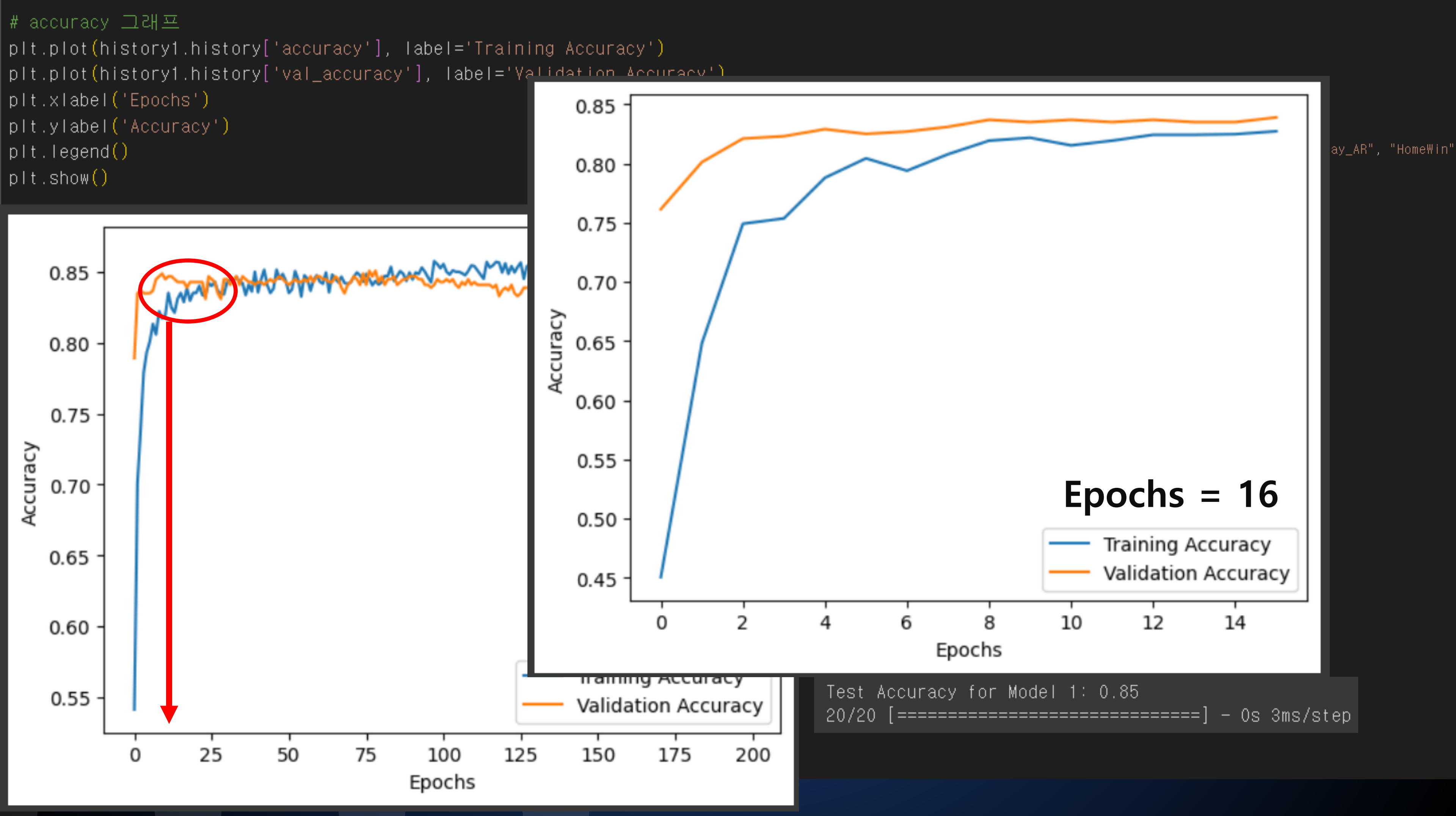


```python
# 모델 컴파일
model1.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# 모델 학습
history1 = model1.fit(X_train, y_train, epochs=200, batch_size=32, validation_split=0.2, verbose=

# 모델 평가
loss1, accuracy1 = model1.evaluate(X_test, y_test, verbose=0)
print(f"Test Accuracy for Model 1: {accuracy1:.2f}")

# Model 1의 예측 결과
predictions1 = model1.predict(X_test)
```



```python
# accuracy 그래프
plt.plot(history1.history['accuracy'], label='Training Accuracy')
plt.plot(history1.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```





```
# accuracy 그래프
plt.plot(history1.history['accuracy'], label='Training Accuracy')
plt.plot(history1.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

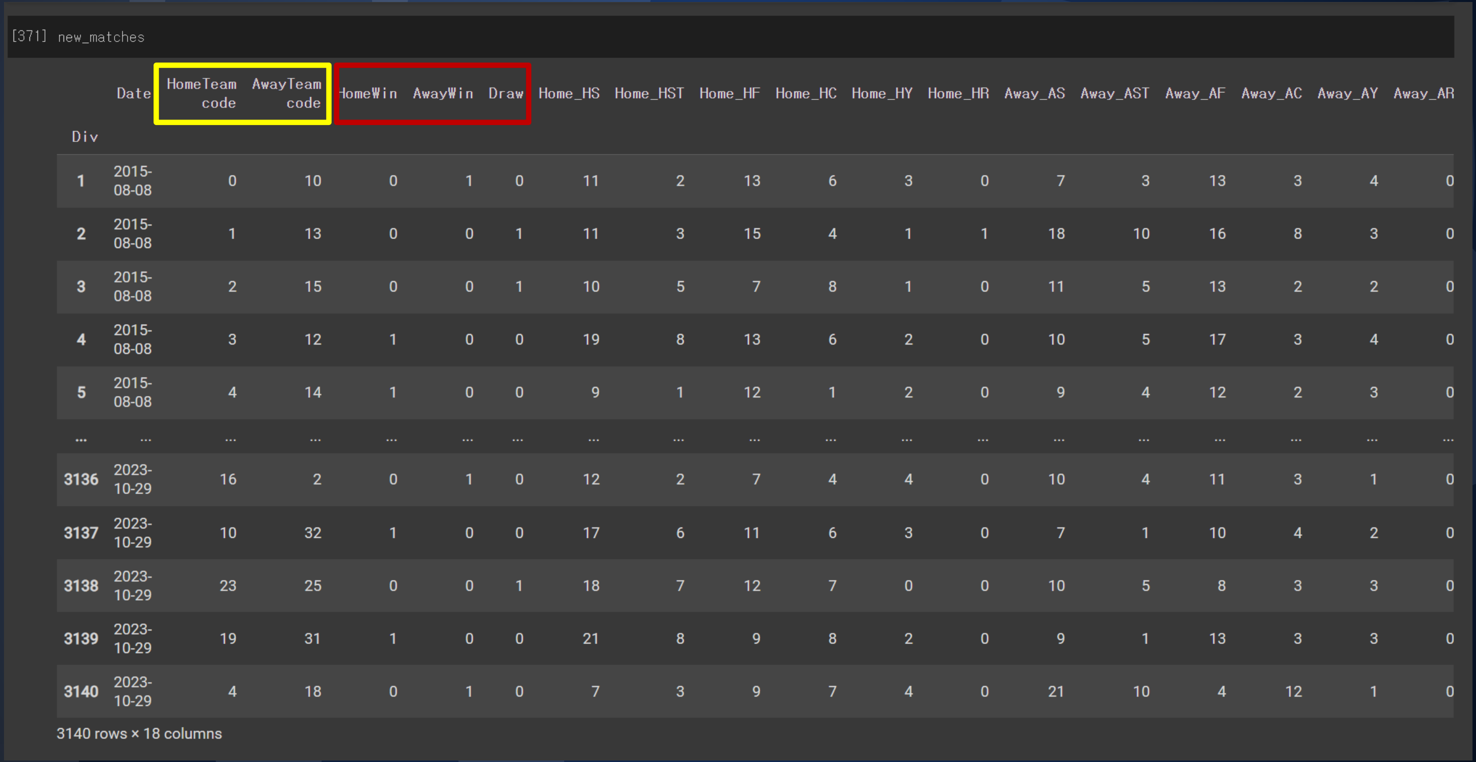
```
ay_AR", "HomeWin"
```

Epochs = 16

```
Test Accuracy for Model 1: 0.85
20/20 [==============================] - 0s 3ms/step
```

v. 모델 2

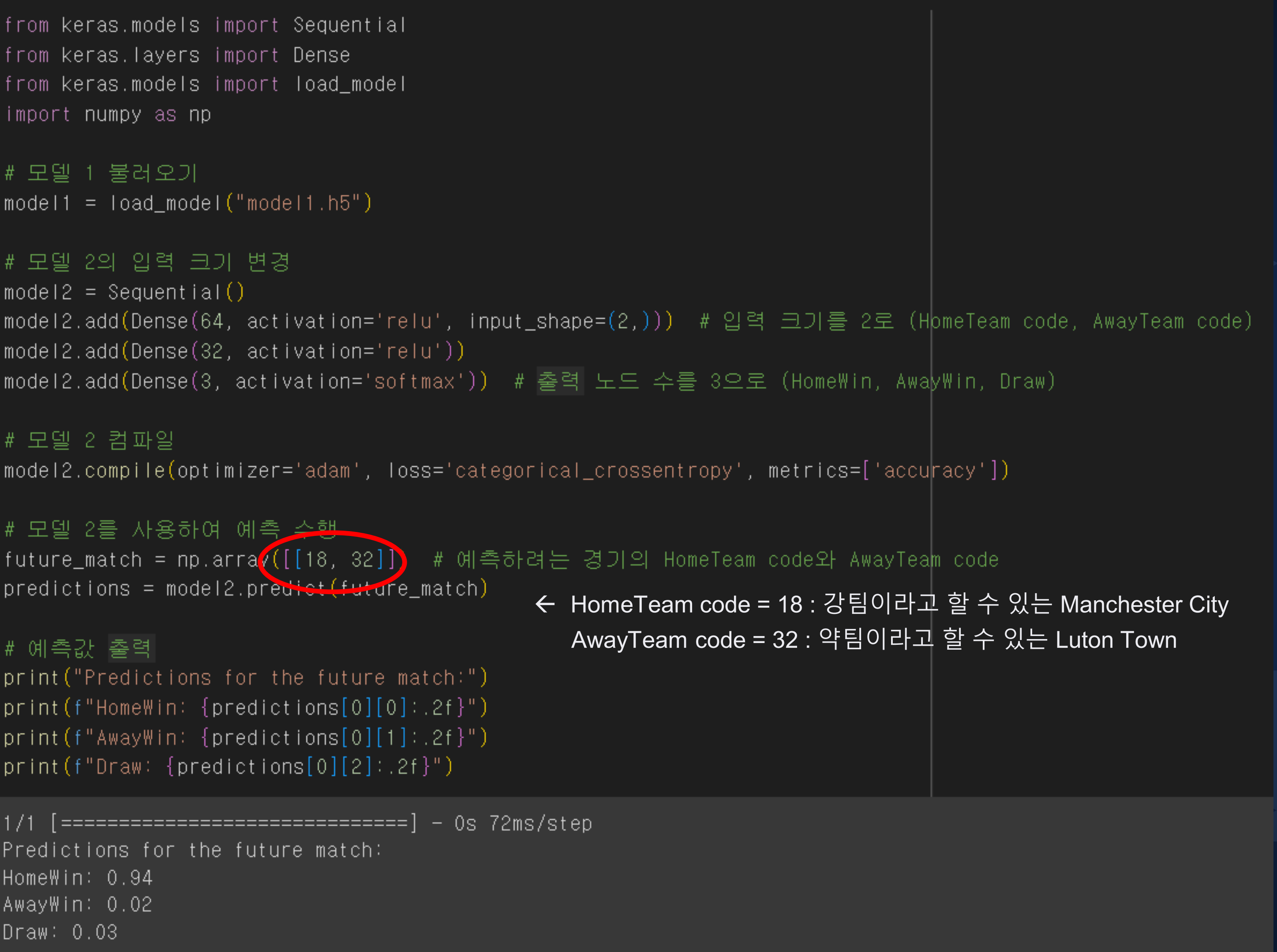
X값 / y값



```
[371] new_matches
```

	Date	HomeTeam code	AwayTeam code	HomeWin	AwayWin	Draw	Home_HS	Home_HST	Home_HF	Home_HC	Home_HY	Home_HR	Away_AS	Away_AST	Away_AF	Away_AC	Away_AY	Away_AR
Div																		
1	2015-08-08	0	10	0	1	0	11	2	13	6	3	0	7	3	13	3	4	0
2	2015-08-08	1	13	0	0	1	11	3	15	4	1	1	18	10	16	8	3	0
3	2015-08-08	2	15	0	0	1	10	5	7	8	1	0	11	5	13	2	2	0
4	2015-08-08	3	12	1	0	0	19	8	13	6	2	0	10	5	17	3	4	0
5	2015-08-08	4	14	1	0	0	9	1	12	1	2	0	9	4	12	2	3	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3136	2023-10-29	16	2	0	1	0	12	2	7	4	4	0	10	4	11	3	1	0
3137	2023-10-29	10	32	1	0	0	17	6	11	6	3	0	7	1	10	4	2	0
3138	2023-10-29	23	25	0	0	1	18	7	12	7	0	0	10	5	8	3	3	0
3139	2023-10-29	19	31	1	0	0	21	8	9	8	2	0	9	1	13	3	3	0
3140	2023-10-29	4	18	0	1	0	7	3	9	7	4	0	21	10	4	12	1	0

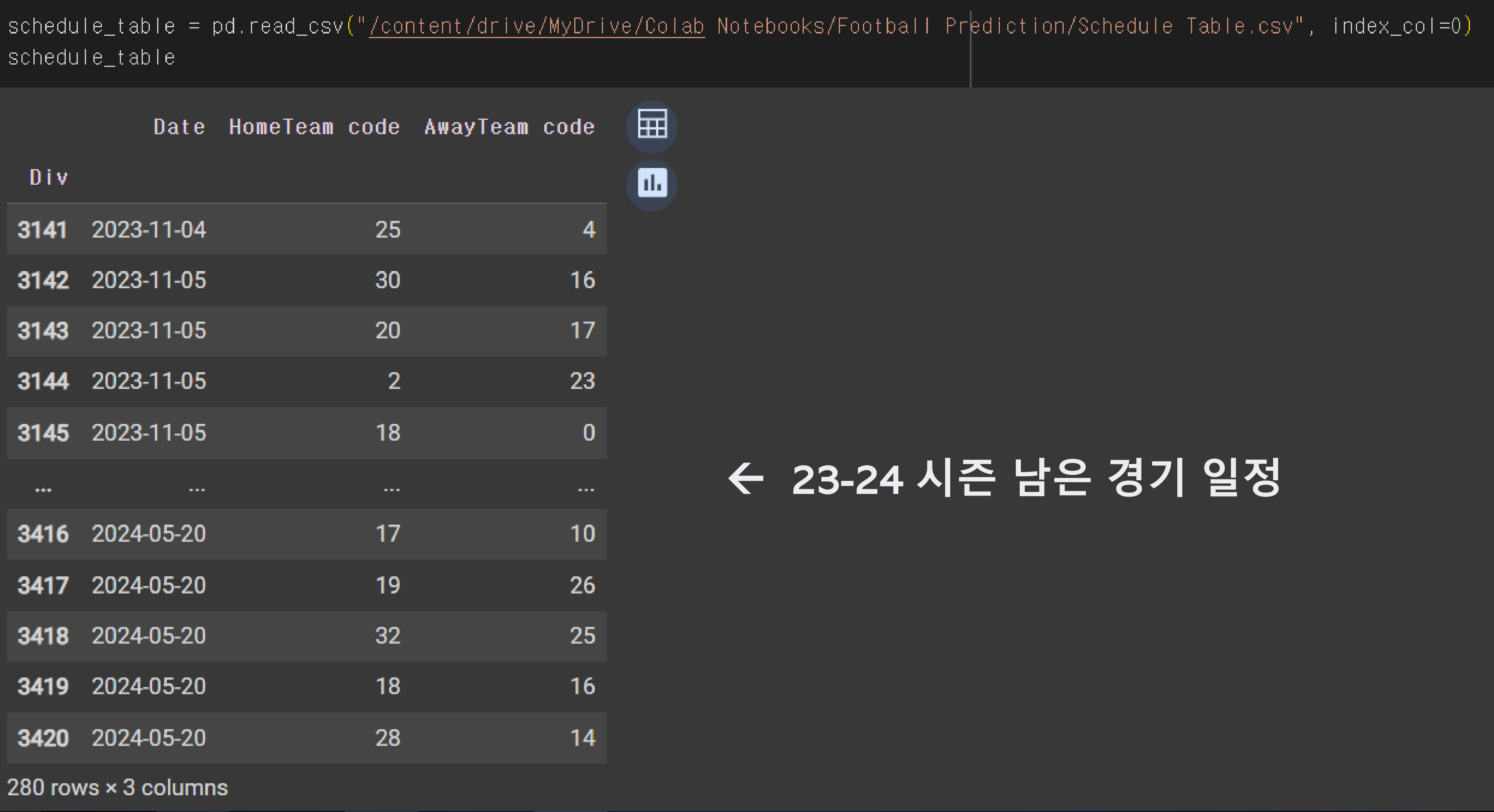
3140 rows × 18 columns



```
from keras.models import Sequential
from keras.layers import Dense
from keras.models import load_model
import numpy as np

# 모델 1 불러오기
model1 = load_model("model1.h5")

# 모델 2의 입력 크기 변경
model2 = Sequential()
model2.add(Dense(64, activation='relu', input_shape=(2,)))  # 입력 크기를 2로 (HomeTeam code, AwayTeam code)
model2.add(Dense(32, activation='relu'))
model2.add(Dense(3, activation='softmax'))  # 출력 노드 수를 3으로 (HomeWin, AwayWin, Draw)

# 모델 2 컴파일
model2.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# 모델 2를 사용하여 예측 수행
future_match = np.array([[18, 32]])  # 예측하려는 경기의 HomeTeam code와 AwayTeam code
predictions = model2.predict(future_match)

# 예측값 출력
print("Predictions for the future match:")
print(f"HomeWin: {predictions[0][0]:.2f}")
print(f"AwayWin: {predictions[0][1]:.2f}")
print(f"Draw: {predictions[0][2]:.2f}")
```

```
1/1 [==============================] - 0s 72ms/step
Predictions for the future match:
HomeWin: 0.94
AwayWin: 0.02
Draw: 0.03
```

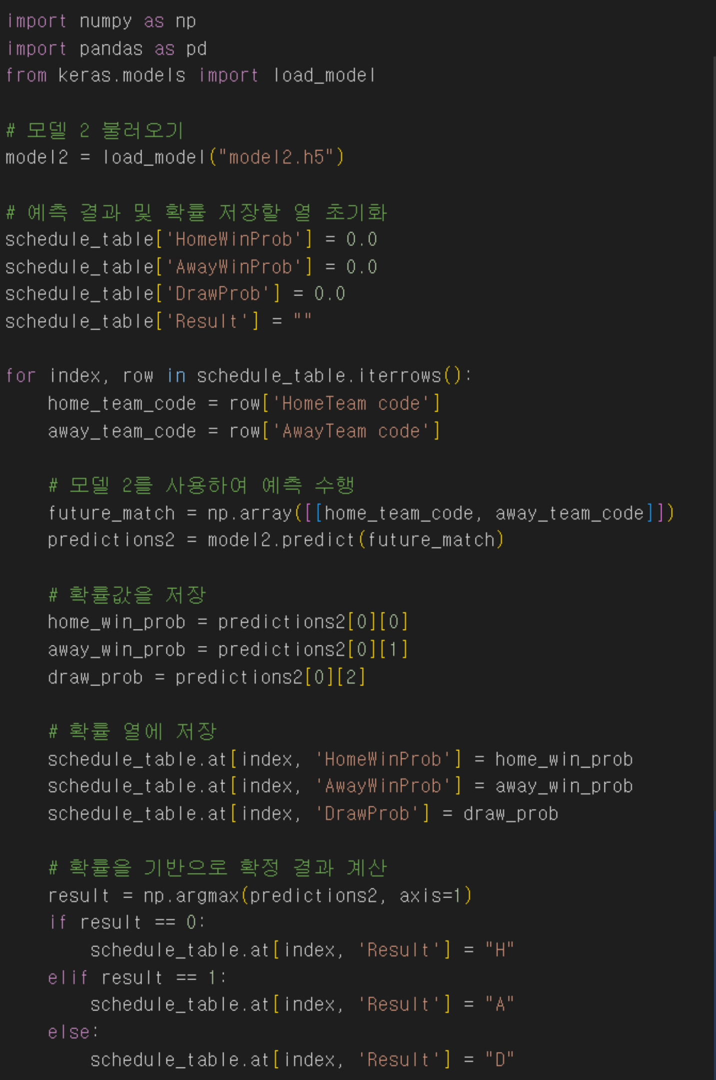
← HomeTeam code = 18 : 강팀이라고 할 수 있는 Manchester City
AwayTeam code = 32 : 약팀이라고 할 수 있는 Luton Town



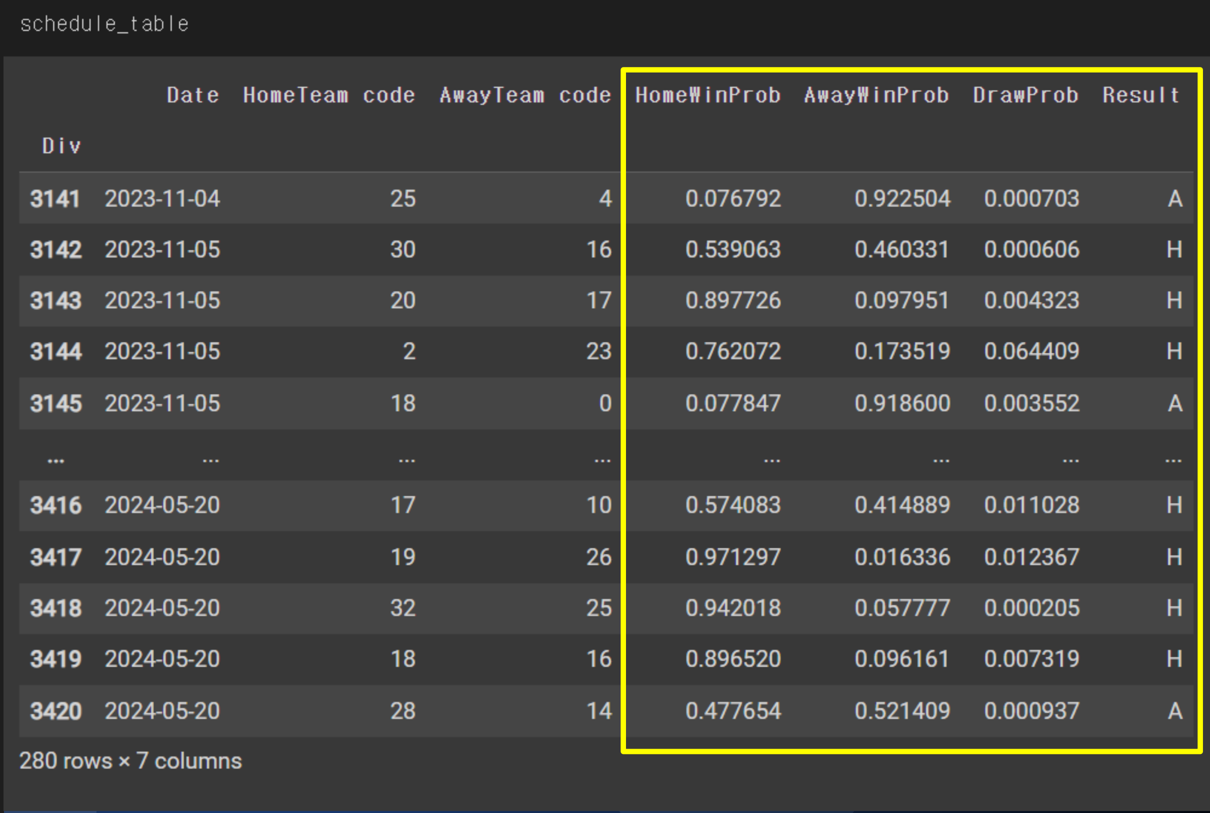
```
schedule_table = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Football Prediction/Schedule Table.csv", index_col=0)
schedule_table
```

```
            Date  HomeTeam code  AwayTeam code
 Div
3141  2023-11-04             25              4
3142  2023-11-05             30             16
3143  2023-11-05             20             17
3144  2023-11-05              2             23
3145  2023-11-05             18              0
 ...         ...            ...            ...
3416  2024-05-20             17             10
3417  2024-05-20             19             26
3418  2024-05-20             32             25
3419  2024-05-20             18             16
3420  2024-05-20             28             14
280 rows × 3 columns
```

← 23-24 시즌 남은 경기 일정



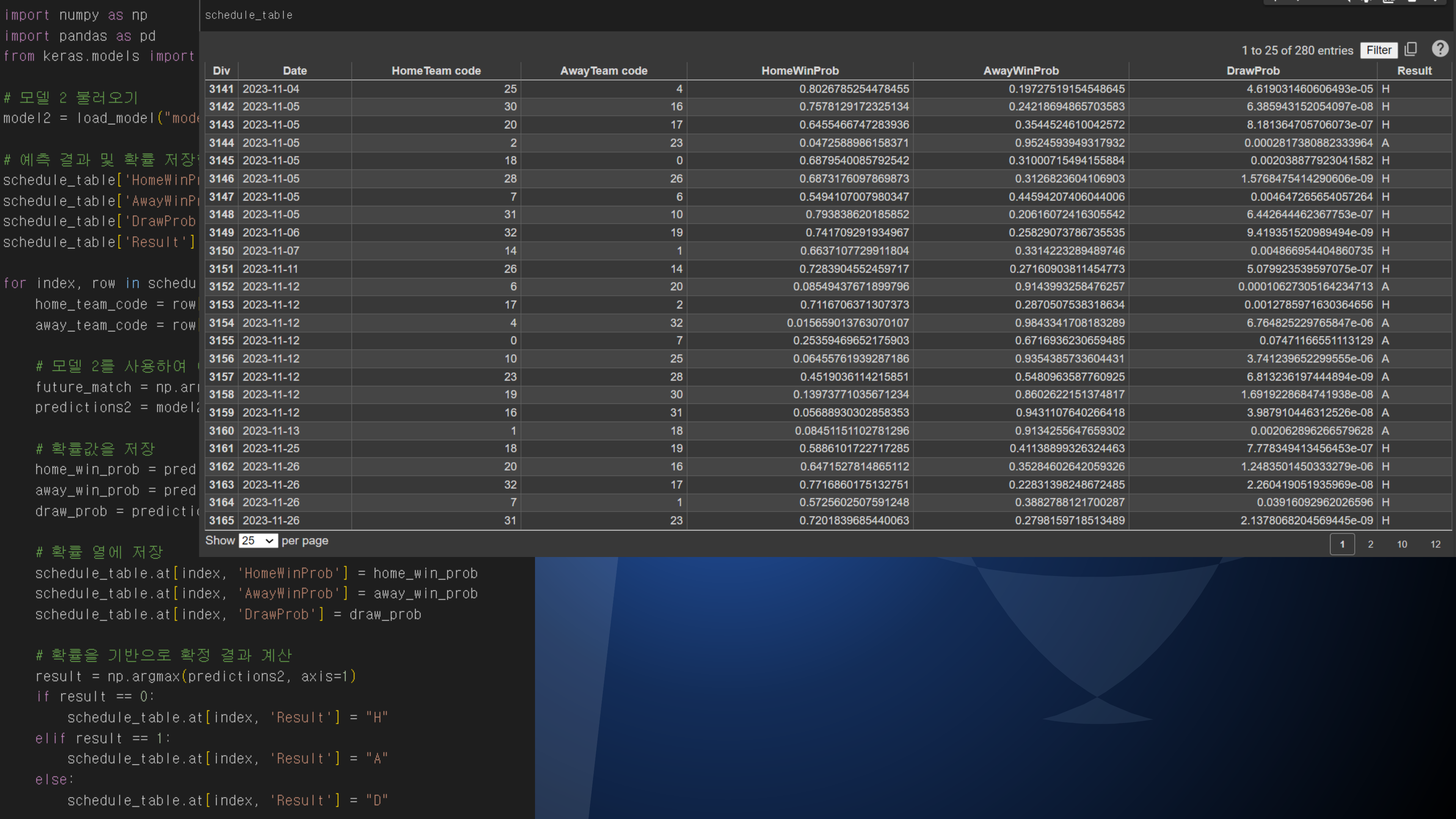
```
import numpy as np
import pandas as pd
from keras.models import load_model

# 모델 2 불러오기
model2 = load_model("model2.h5")

# 예측 결과 및 확률 저장할 열 초기화
schedule_table['HomeWinProb'] = 0.0
schedule_table['AwayWinProb'] = 0.0
schedule_table['DrawProb'] = 0.0
schedule_table['Result'] = ""

for index, row in schedule_table.iterrows():
    home_team_code = row['HomeTeam code']
    away_team_code = row['AwayTeam code']

    # 모델 2를 사용하여 예측 수행
    future_match = np.array([[home_team_code, away_team_code]])
    predictions2 = model2.predict(future_match)

    # 확률값을 저장
    home_win_prob = predictions2[0][0]
    away_win_prob = predictions2[0][1]
    draw_prob = predictions2[0][2]

    # 확률 열에 저장
    schedule_table.at[index, 'HomeWinProb'] = home_win_prob
    schedule_table.at[index, 'AwayWinProb'] = away_win_prob
    schedule_table.at[index, 'DrawProb'] = draw_prob

    # 확률을 기반으로 확정 결과 계산
    result = np.argmax(predictions2, axis=1)
    if result == 0:
        schedule_table.at[index, 'Result'] = "H"
    elif result == 1:
        schedule_table.at[index, 'Result'] = "A"
    else:
        schedule_table.at[index, 'Result'] = "D"
```



```
schedule_table
```

	Date	HomeTeam code	AwayTeam code	HomeWinProb	AwayWinProb	DrawProb	Result
Div							
3141	2023-11-04	25	4	0.076792	0.922504	0.000703	A
3142	2023-11-05	30	16	0.539063	0.460331	0.000606	H
3143	2023-11-05	20	17	0.897726	0.097951	0.004323	H
3144	2023-11-05	2	23	0.762072	0.173519	0.064409	H
3145	2023-11-05	18	0	0.077847	0.918600	0.003552	A
...	...	...	...	...	...	...	...
3416	2024-05-20	17	10	0.574083	0.414889	0.011028	H
3417	2024-05-20	19	26	0.971297	0.016336	0.012367	H
3418	2024-05-20	32	25	0.942018	0.057777	0.000205	H
3419	2024-05-20	18	16	0.896520	0.096161	0.007319	H
3420	2024-05-20	28	14	0.477654	0.521409	0.000937	A

280 rows × 7 columns



```
import numpy as np
import pandas as pd
from keras.models import

# 모델 2 불러오기
model2 = load_model("mode

# 예측 결과 및 확률 저장
schedule_table['HomeWinPr
schedule_table['AwayWinPr
schedule_table['DrawProb
schedule_table['Result']

for index, row in schedu
    home_team_code = row
    away_team_code = row

    # 모델 2를 사용하여
    future_match = np.ar
    predictions2 = model

    # 확률값을 저장
    home_win_prob = pred
    away_win_prob = pred
    draw_prob = predicti

    # 확률 열에 저장
    schedule_table.at[index, 'HomeWinProb'] = home_win_prob
    schedule_table.at[index, 'AwayWinProb'] = away_win_prob
    schedule_table.at[index, 'DrawProb'] = draw_prob

    # 확률을 기반으로 확정 결과 계산
    result = np.argmax(predictions2, axis=1)
    if result == 0:
        schedule_table.at[index, 'Result'] = "H"
    elif result == 1:
        schedule_table.at[index, 'Result'] = "A"
    else:
        schedule_table.at[index, 'Result'] = "D"
```

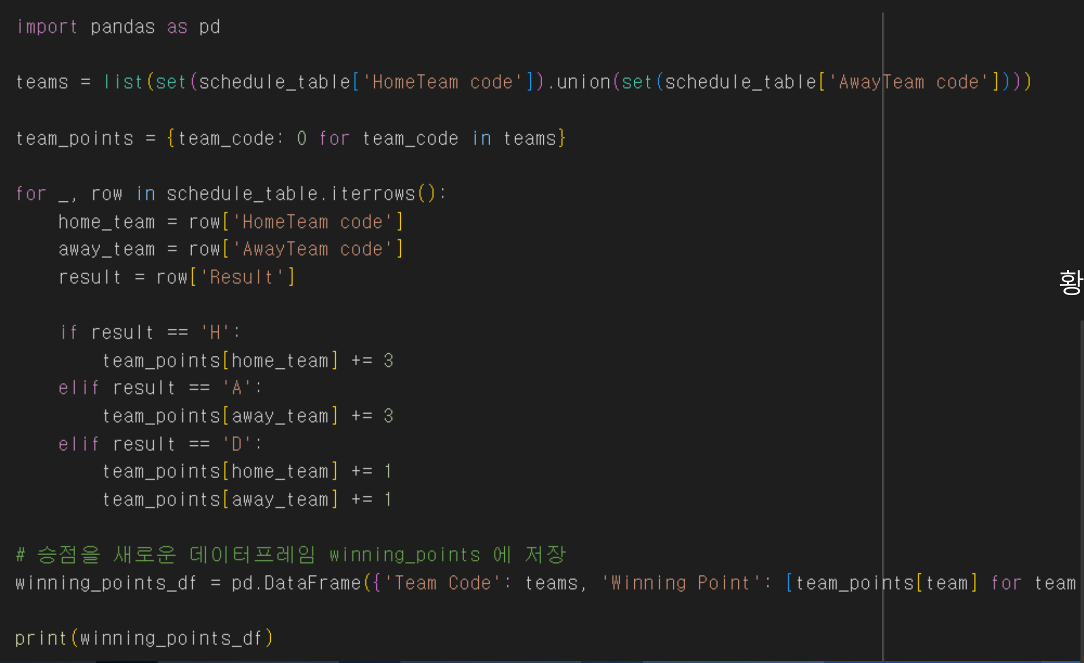
schedule_table

1 to 25 of 280 entries Filter

Div	Date	HomeTeam code	AwayTeam code	HomeWinProb	AwayWinProb	DrawProb	Result
3141	2023-11-04	25	4	0.8026785254478455	0.19727519154548645	4.619031460606493e-05	H
3142	2023-11-05	30	16	0.7578129172325134	0.24218694865703583	6.385943152054097e-08	H
3143	2023-11-05	20	17	0.6455466747283936	0.3544524610042572	8.181364705706073e-07	H
3144	2023-11-05	2	23	0.0472588986158371	0.9524593949317932	0.0002817380882333964	A
3145	2023-11-05	18	0	0.6879540085792542	0.31000715494155884	0.002038877923041582	H
3146	2023-11-05	28	26	0.6873176097869873	0.3126823604106903	1.5768475414290606e-09	H
3147	2023-11-05	7	6	0.5494107007980347	0.44594207406044006	0.004647265654057264	H
3148	2023-11-05	31	10	0.793838620185852	0.20616072416305542	6.442644462367753e-07	H
3149	2023-11-06	32	19	0.741709291934967	0.25829073786735535	9.419351520989494e-09	H
3150	2023-11-07	14	1	0.6637107729911804	0.3314223289489746	0.004866954404860735	H
3151	2023-11-11	26	14	0.7283904552459717	0.27160903811454773	5.079923539597075e-07	H
3152	2023-11-12	6	20	0.08549437671899796	0.9143993258476257	0.00010627305164234713	A
3153	2023-11-12	17	2	0.7116706371307373	0.2870507538318634	0.0012785971630364656	H
3154	2023-11-12	4	32	0.015659013763070107	0.9843341708183289	6.764825229765847e-06	A
3155	2023-11-12	0	7	0.25359469652175903	0.6716936230659485	0.07471166551113129	A
3156	2023-11-12	10	25	0.06455761939287186	0.9354385733604431	3.741239652299555e-06	A
3157	2023-11-12	23	28	0.4519036114215851	0.5480963587760925	6.813236197444894e-09	A
3158	2023-11-12	19	30	0.13973771035671234	0.8602622151374817	1.6919228684741938e-08	A
3159	2023-11-12	16	31	0.05688930302858353	0.9431107640266418	3.987910446312526e-08	A
3160	2023-11-13	1	18	0.08451151102781296	0.9134255647659302	0.002062896266579628	A
3161	2023-11-25	18	19	0.5886101722717285	0.41138899326324463	7.778349413456453e-07	H
3162	2023-11-26	20	16	0.6471527814865112	0.35284602642059326	1.2483501450333279e-06	H
3163	2023-11-26	32	17	0.7716860175132751	0.22831398248672485	2.260419051935969e-08	H
3164	2023-11-26	7	1	0.5725602507591248	0.3882788121700287	0.03916092962026596	H
3165	2023-11-26	31	23	0.7201839685440063	0.2798159718513489	2.1378068204569445e-09	H

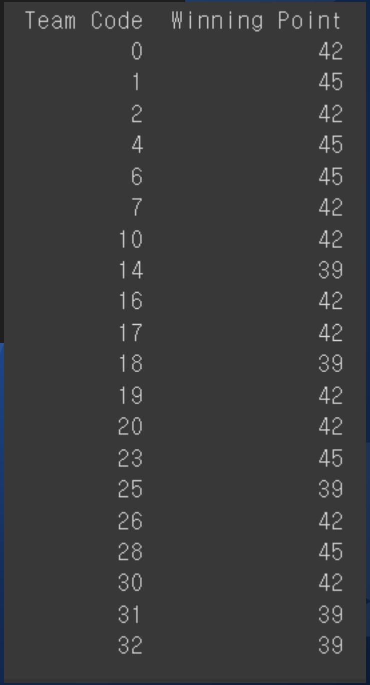
Show 25 per page

1 2 10 12



```python
import pandas as pd

teams = list(set(schedule_table['HomeTeam code']).union(set(schedule_table['AwayTeam code'])))

team_points = {team_code: 0 for team_code in teams}

for _, row in schedule_table.iterrows():
    home_team = row['HomeTeam code']
    away_team = row['AwayTeam code']
    result = row['Result']

    if result == 'H':
        team_points[home_team] += 3
    elif result == 'A':
        team_points[away_team] += 3
    elif result == 'D':
        team_points[home_team] += 1
        team_points[away_team] += 1

# 승점을 새로운 데이터프레임 winning_points 에 저장
winning_points_df = pd.DataFrame({'Team Code': teams, 'Winning Point': [team_points[team] for team

print(winning_points_df)
```

황당한 승점 결과 발생 – 예측실패,,



```
Team Code  Winning Point
        0             42
        1             45
        2             42
        4             45
        6             45
        7             42
       10             42
       14             39
       16             42
       17             42
       18             39
       19             42
       20             42
       23             45
       25             39
       26             42
       28             45
       30             42
       31             39
       32             39
```

예측 실패의 원인

- 불완전한 코딩/ 코드의 미흡

: 모델 1 에서의 코딩실수 / 모델2로 전이학습 실패

- 모델1에 비해 급격히 감소한 모델2의 입력값

: 모델 1의 X값 - ["HomeTeam code", "AwayTeam code", "Home_HST", "Home_HC", "Home_HR", "Away_AST", "Away_AC", "Away_AR", "Draw“]

모델 2의 X값 - [["HomeTeam code", "AwayTeam code“]

- 일반적으로 축구경기 결과를 예측할 때 사용할 수 있는 많은 변수의 부재

ex) 경기당일의 날씨, 주요 선수들의 평균평점, 선발라인업 etc,,,