카레맨이 설명하는

NBA 데이터 분석

Contents

Data
Crawling

NBA 농구 데이터 수집

수집한 데이터 가공

Data
Processing

Applying
ML model

가공된 Data에
'Keras' 모델 적용

배운 점 및 느낀 점

The end

주제 선정

NBA

주제 선정

BASKETBALL REFERENCE

Enter Person, Team, Section, etc

Players | Teams | Seasons | Leaders | Scores 4 | Playoffs | Draft | Play Index | Full Site Menu Below ▾

October | November | December | January | February | March | April | May | June

October Schedule

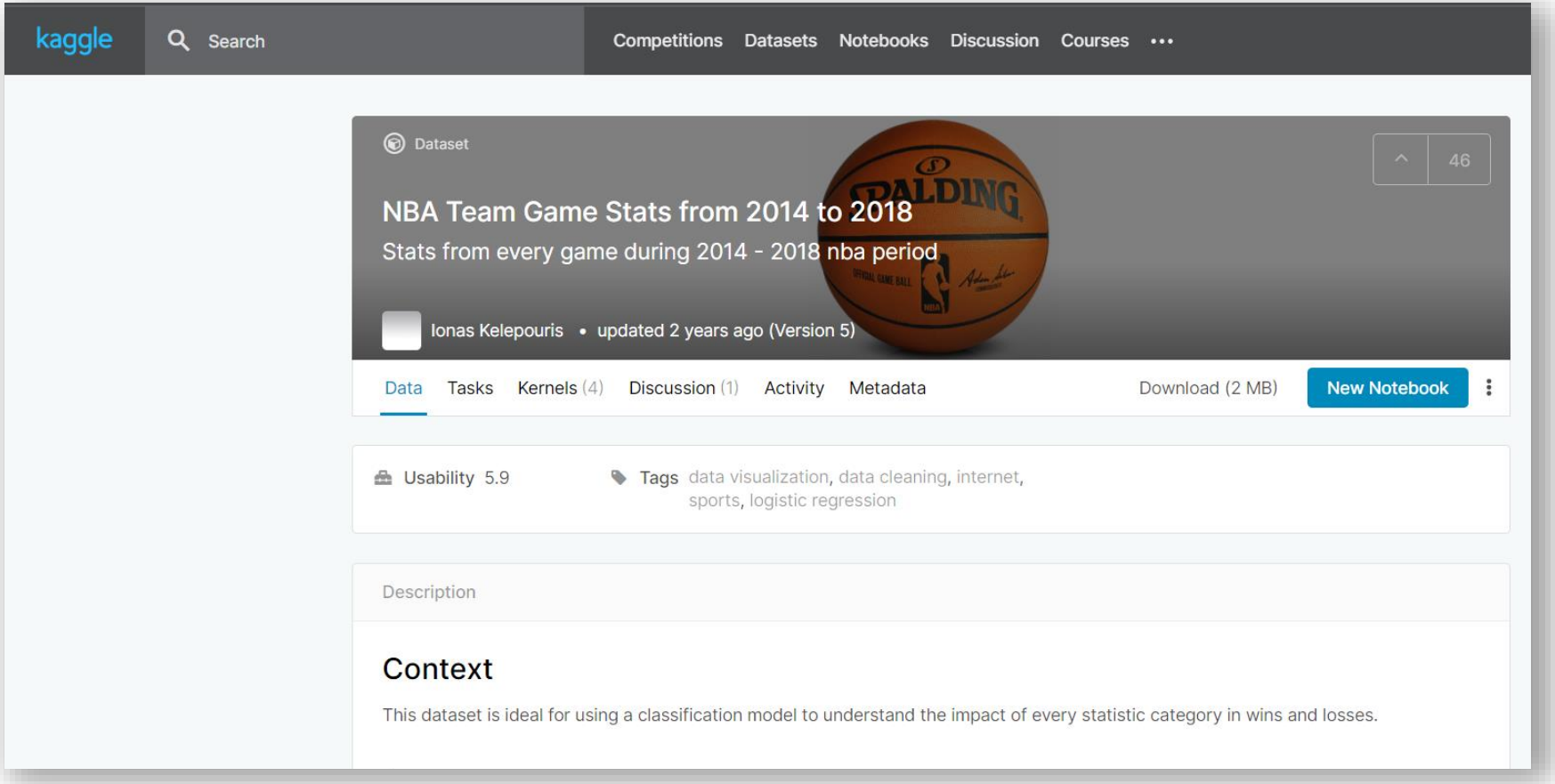
Share & more ▾

Date	Start (ET)	Visitor/Neutral	PTS	Home/Neutral	PTS		Attend.	Notes
Tue, Oct 16, 2018	8:00p	Philadelphia 76ers	87	Boston Celtics	105	Box Score	18,624	
Tue, Oct 16, 2018	10:30p	Oklahoma City Thunder	100	Golden State Warriors	108	Box Score	19,596	
Wed, Oct 17, 2018	7:00p	Milwaukee Bucks	113	Charlotte Hornets	112	Box Score	17,889	
Wed, Oct 17, 2018	7:00p	Brooklyn Nets	100	Detroit Pistons	103	Box Score	20,332	
Wed, Oct 17, 2018	7:00p	Memphis Grizzlies	83	Indiana Pacers	111	Box Score	17,923	
Wed, Oct 17, 2018	7:00p	Miami Heat	101	Orlando Magic	104	Box Score	19,191	
Wed, Oct 17, 2018	7:30p	Atlanta Hawks	107	New York Knicks	126	Box Score	18,249	
Wed, Oct 17, 2018	7:30p	Cleveland Cavaliers	104	Toronto Raptors	116	Box Score	19,915	
Wed, Oct 17, 2018	8:00p	New Orleans Pelicans	131	Houston Rockets	112	Box Score	18,055	
Wed, Oct 17, 2018	8:30p	Minnesota Timberwolves	108	San Antonio Spurs	112	Box Score	18,354	
Wed, Oct 17, 2018	10:00p	Utah Jazz	123	Sacramento Kings	117	Box Score	17,583	
Wed, Oct 17, 2018	10:30p	Denver Nuggets	107	Los Angeles Clippers	98	Box Score	19,068	
Wed, Oct 17, 2018	10:30p	Dallas Mavericks	100	Phoenix Suns	121	Box Score	18,055	
Thu, Oct 18, 2018	8:00p	Chicago Bulls	108	Philadelphia 76ers	127	Box Score	20,302	
Thu, Oct 18, 2018	8:00p	Miami Heat	113	Washington Wizards	112	Box Score	20,409	
Thu, Oct 18, 2018	10:30p	Los Angeles Lakers	119	Portland Trail Blazers	128	Box Score	19,996	
Fri, Oct 19, 2018	7:00p	Charlotte Hornets	120	Orlando Magic	88	Box Score	17,668	
Fri, Oct 19, 2018	7:30p	New York Knicks	105	Brooklyn Nets	107	Box Score	17,732	
Fri, Oct 19, 2018	8:00p	Atlanta Hawks	117	Memphis Grizzlies	131	Box Score	17,019	
Fri, Oct 19, 2018	8:00p	Cleveland Cavaliers	123	Minnesota Timberwolves	131	Box Score	18,978	
Fri, Oct 19, 2018	8:00p	Sacramento Kings	129	New Orleans Pelicans	149	Box Score	18,337	
Fri, Oct 19, 2018	8:00p	Boston Celtics	101	Toronto Raptors	113	Box Score	19,800	
Fri, Oct 19, 2018	8:30p	Indiana Pacers	101	Milwaukee Bucks	118	Box Score	17,341	
Fri, Oct 19, 2018	10:30p	Oklahoma City Thunder	92	Los Angeles Clippers	108	Box Score	14,816	
Fri, Oct 19, 2018	10:30p	Golden State Warriors	124	Utah Jazz	123	Box Score	18,306	

Crawling

```python
for current in current_page: # 조건문이 참인 "동안" 실행

    print('₩n{}월 경기부터 크롤링을 시작합니다.'.format(current))
    url="https://www.basketball-reference.com/leagues/NBA_2019_games-" + current + ".html"
    web=requests.get(url).content
    source = BeautifulSoup(web, 'html.parser')
    score = source.find_all('td', {'class' : 'right'})
    nbateamname=source.find_all('td', {'class' : 'left'})

    for urls in score:
        if urls["data-stat"].startswith("visitor_pts"):
            visitor_team_points.append(urls.get_text())

    for urls in score:
        if urls["data-stat"].startswith("home_pts"):
            home_team_points.append(urls.get_text())

    for urls in nbateamname:
        if urls["data-stat"].startswith("visitor_team_name"):
            visitor_teamname.append(urls.get_text())


    for urls in nbateamname:
        if urls["data-stat"].startswith("home_team_name"):
            home_teamname.append(urls.get_text())
```

Kaggle – NBA Team Game Stat from 2014 to 2018



데이터 전처리

Opp.FieldGoals
Assists
Opp.3PointShotsAttempted
Turnovers

⋮

총 41개 Columns

⋮

Opp.FreeThrows
Opp.OffRebounds
TotalFouls
Opp.Turnovers

6개 Columns

데이터 전처리 code

```
# extract certain date that i need

data['new_dates'] = pd.to_datetime(data['Date'])
start_date = '10/01/2017'
end_date = '04/30/2018'

mask = (data['new_dates'] > start_date) & (data['new_dates'] <= end_date)
data = data.loc[mask]

#change the name of team (str type) to int type
da = [data]
team_titles = {"ATL": 1, "BOS": 2, "BRK": 3, "CHO": 4, "CHI": 5,
               "CLE": 6, "DAL": 7, "DEN": 8, "DET": 9, "GSW": 10,
              "HOU": 11, "IND": 12, "LAC": 13, "LAL": 14, "MEM": 15,
              "MIA": 16, "MIL": 17, "MIN": 18, "NOP": 19, "NYK" :20,
              "OKC": 21, "ORL": 22, "PHI": 23, "PHO": 24, "POR": 25,
              "SAC": 26, "SAS": 27, "TOR":  28, "UTA": 29, "WAS": 30}

for data in da:
    data['Team'] = data['Team'].map(team_titles)
    data['Opponent'] = data['Opponent'].map(team_titles)
```

데이터 전처리 code

```
#if the game was on the home ground -> '1', away -> '0'
#win -> '1' | loss -> '0'
data['int_home'] = data['Home'].apply(lambda x : '1' if x == 'Home' else '0')
data['int_win_loss'] = data['WINorLOSS'].apply(lambda x : '1' if x == 'W' else '0')

# delete some rows that seems useless
data.drop(['Date', 'Game', 'new_dates', 'Home', 'WINorLOSS', 'Unnamed: 0'], axis = 1, inplace = True)

# separate the data, featrues and target
x_data = new_raw.copy()
y_data = new_raw[['int_win_loss']].copy()


del x_data['int_win_loss']
x_data['int_home'] = x_data['int_home'].apply(pd.to_numeric, errors='coerce')
y_data['int_win_loss'] = y_data['int_win_loss'].apply(pd.to_numeric, errors='coerce')

data.head()
```

전처리 한 data

	Team	Opponent	FieldGoals.	FreeThrowsAttempted	FreeThrows.	TotalRebounds	Turnovers	Opp.FieldGoals.	int_home
0	1	26	0.507	31	0.774	37	16	0.475	0
1	1	28	0.520	29	0.655	45	21	0.402	1
2	1	15	0.500	26	0.769	46	18	0.378	0
3	1	4	0.519	18	0.778	50	16	0.330	1
4	1	12	0.417	19	0.789	37	18	0.476	0
...	...	...	...	...	...	...	...	...	...
1915	30	27	0.541	18	0.778	50	16	0.448	1
1916	30	9	0.416	15	0.800	43	15	0.488	0
1917	30	4	0.455	12	0.750	44	12	0.455	1
1918	30	5	0.457	12	1.000	37	16	0.524	0
1919	30	11	0.528	23	0.696	36	17	0.512	0

1920 rows × 9 columns

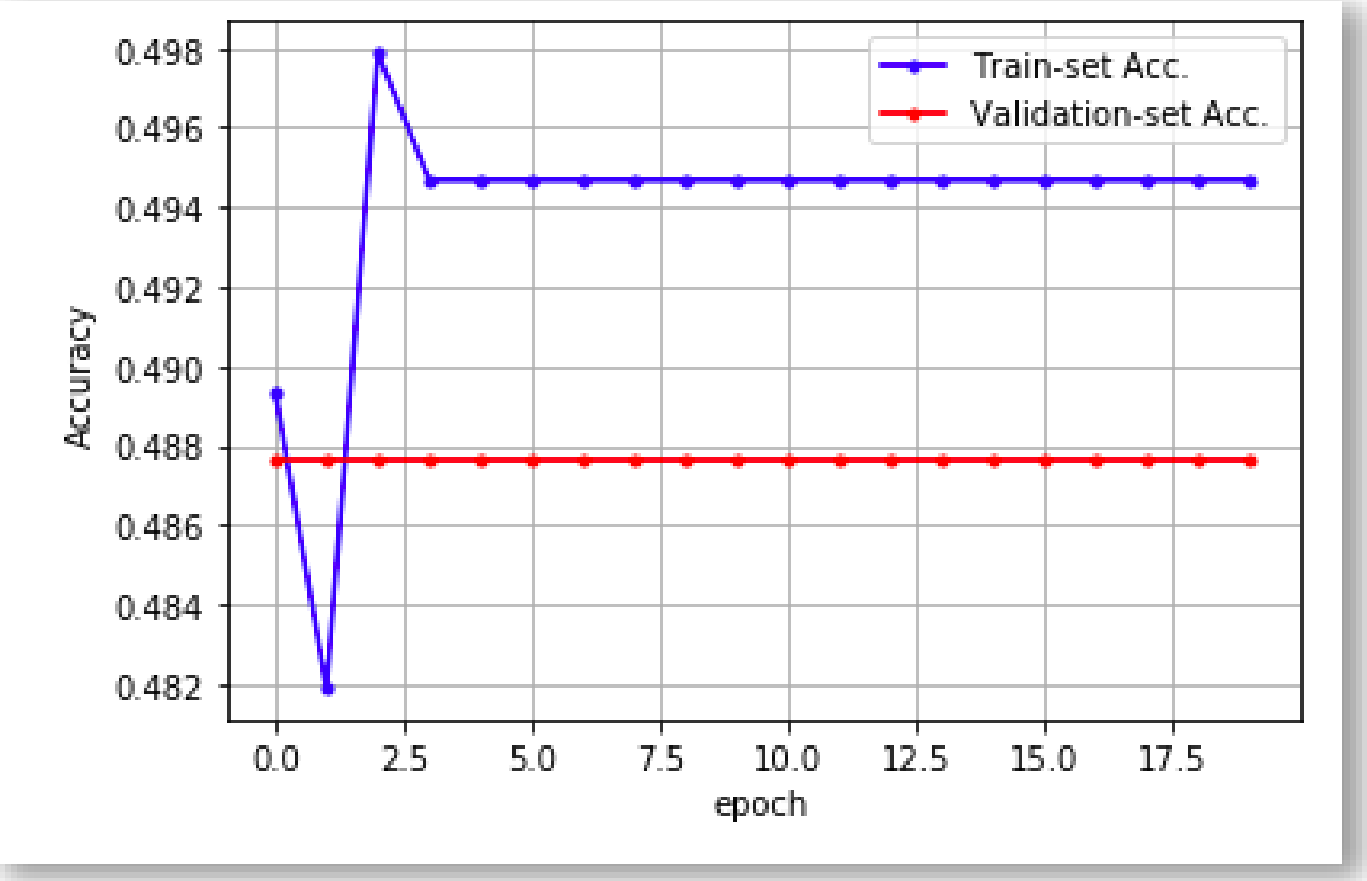
Data training & fit (With Keras)

```
Epoch 17/20
940/940 [==============================] - 0s 204us/sample - loss: 8.1448 - categorical_accuracy: 0.4947 - val_loss:
8.2585 - val_categorical_accuracy: 0.4876
Epoch 18/20
940/940 [==============================] - 0s 197us/sample - loss: 8.1448 - categorical_accuracy: 0.4947 - val_loss:
8.2585 - val_categorical_accuracy: 0.4876
Epoch 19/20
940/940 [==============================] - 0s 309us/sample - loss: 8.1448 - categorical_accuracy: 0.4947 - val_loss:
8.2585 - val_categorical_accuracy: 0.4876
Epoch 20/20
940/940 [==============================] - 0s 198us/sample - loss: 8.1448 - categorical_accuracy: 0.4947 - val_loss:
8.2585 - val_categorical_accuracy: 0.4876
```

```
result = model.evaluate(test_data, test_label, batch_size=100)

print('loss (cross-entropy) :', result[0])
print('test accuracy :', result[1])
```

```
576/576 [==============================] - 0s 83us/sample - loss: 7.7792 - categorical_accuracy: 0.5174
loss (cross-entropy) : 7.779220008187824
test accuracy : 0.5173611
```

Result


Train-set Acc.
Validation-set Acc.
Accuracy
0.498
0.496
0.494
0.492
0.490
0.488
0.486
0.484
0.482
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
epoch

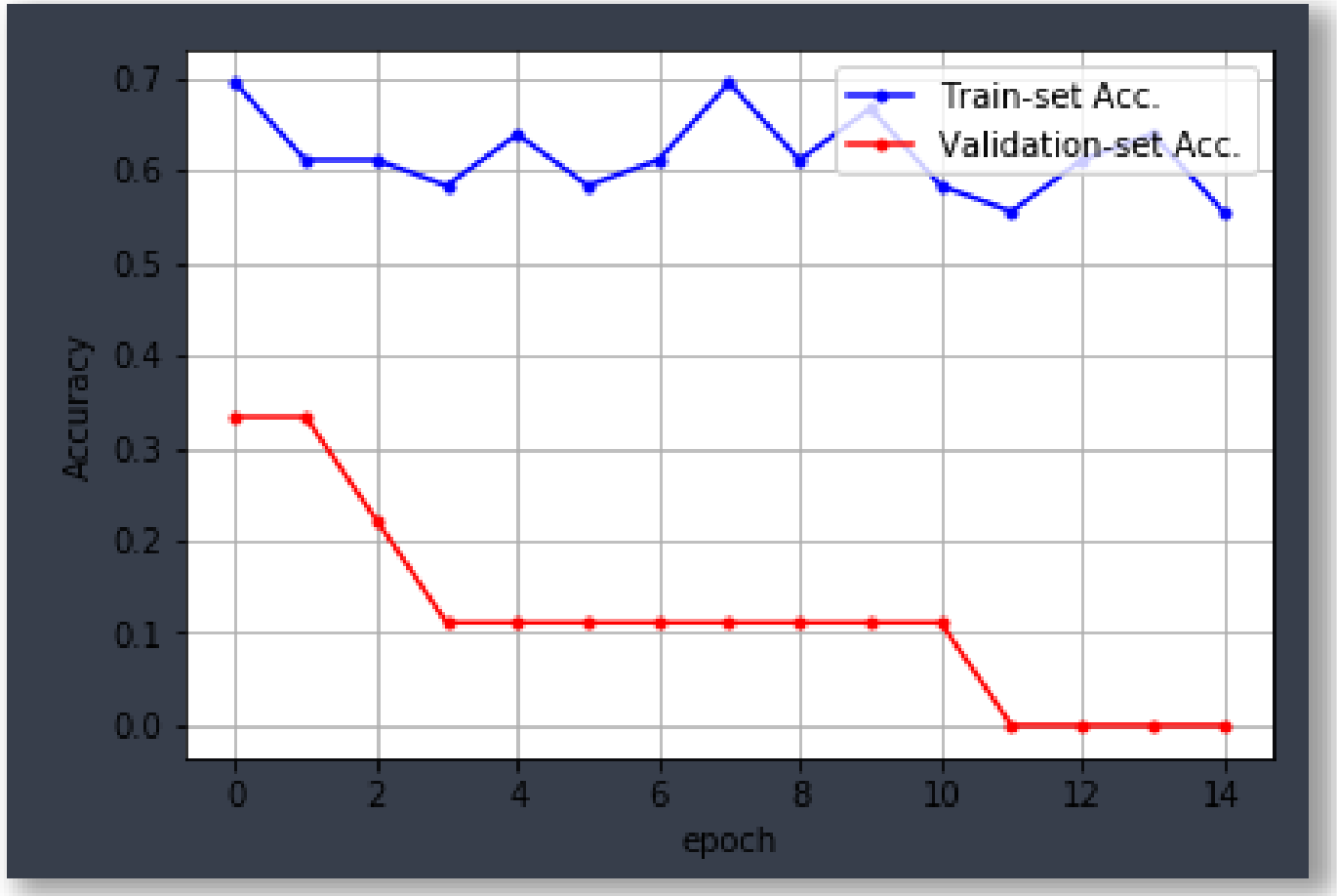
training with 1 team

	Team	Game	Date	Home	Opponent	WINorLOSS	TeamPoints	OpponentPoints	FieldGoals	FieldGoalsAttempted
5460	ATL	54	10/02/2017	Away	SAC	L	107	108	36	71
6560	ATL	65	10/03/2017	Home	TOR	W	105	99	39	75
6660	ATL	66	11/03/2017	Away	MEM	W	107	90	36	72
8161	ATL	81	11/04/2017	Home	CHO	W	103	76	41	79
8260	ATL	82	12/04/2017	Away	IND	L	86	104	30	72
...	...	...	...	...	...	...	...	...	...	...
7590	ATL	75	28/03/2018	Away	MIN	L	114	126	41	77
7690	ATL	76	30/03/2018	Home	PHI	L	91	101	36	101
7790	ATL	77	01/04/2018	Home	ORL	W	94	88	41	85
7890	ATL	78	03/04/2018	Away	MIA	L	98	101	37	88
7990	ATL	79	04/04/2018	Home	MIA	L	86	115	33	90

```
64 rows × 41 columns
Epoch 15/15
36/36 [==============================] - 0s 970us/sample - loss: 0.6916 - categorical_accuracy: 0.5000 - val_loss: 0.9786 - val_categorical_acc
uracy: 0.4444
```

uracy: 0.4444

Result


Train-set Acc.
Validation-set Acc.
Accuracy
epoch
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
2
4
6
8
10
12
14

데이터 전처리

```
del data['new_dates']
del data['Unnamed: 0']
del data['Date']
del data['Game']
del data['TeamPoints']
del data['OpponentPoints']
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1920 entries, 4973 to 9835
Data columns (total 36 columns):
```

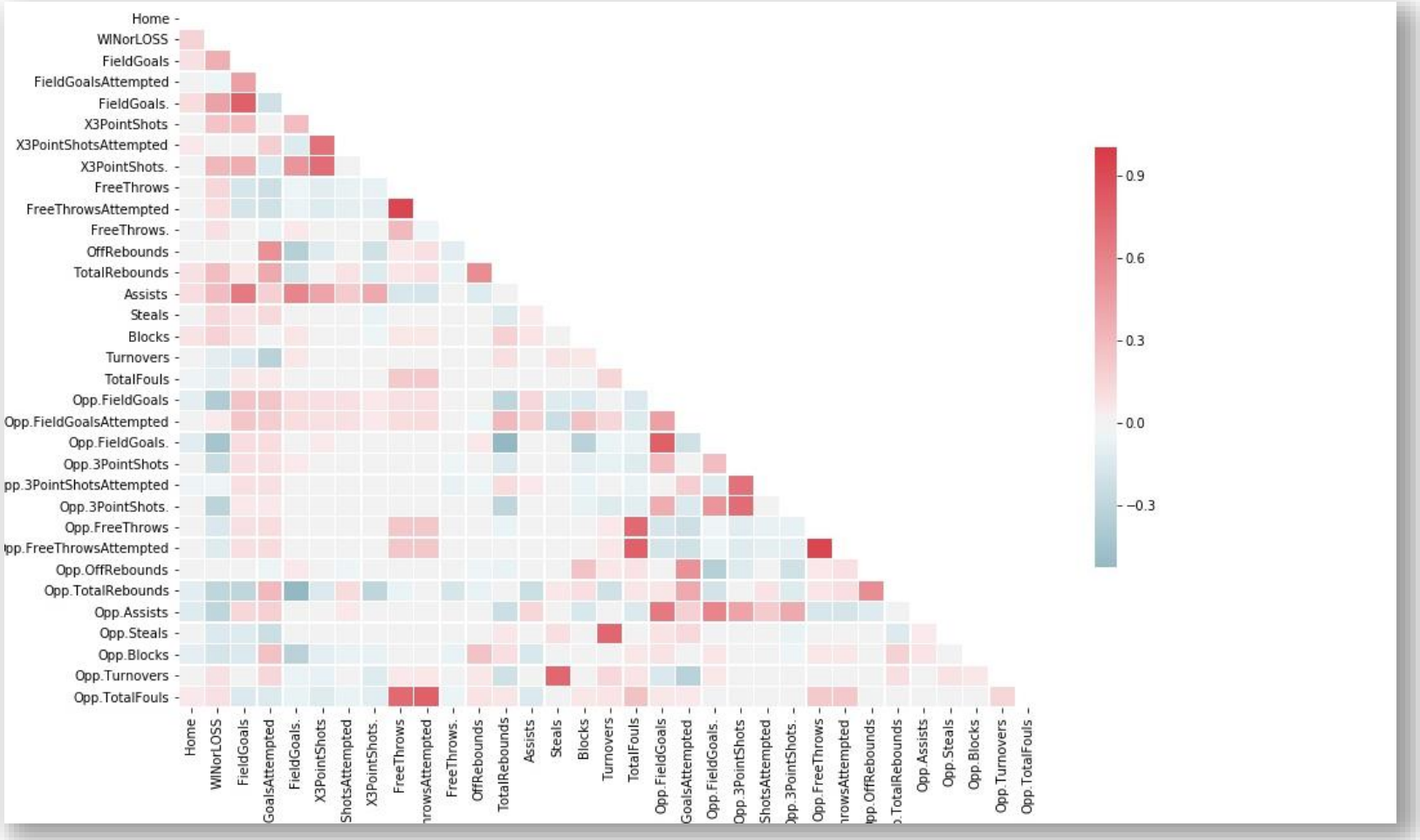
Data columns (total 36 columns)

데이터 전처리

```
target_col=['FieldGoals', 'FieldGoalsAttempted', 'FieldGoals.'
                 , 'X3PointShots','X3PointShotsAttempted', 'X3PointShots.' ,'FreeThrows',
                 'FreeThrowsAttempted', 'FreeThrows.', 'OffRebounds',
                 'TotalRebounds', 'Assists', 'Steals', 'Blocks', 'Turnovers', 'TotalFouls',
                'Opp.FieldGoals', 'Opp.FieldGoalsAttempted', 'Opp.FieldGoals.' ,'Opp.3PointShots',
                 'Opp.3PointShotsAttempted', 'Opp.3PointShots.' ,'Opp.FreeThrows',
                 'Opp.FreeThrowsAttempted',
                'Opp.OffRebounds', 'Opp.TotalRebounds', 'Opp.Assists', 'Opp.Steals', 'Opp.Blocks',
                'Opp.Turnovers', 'Opp.TotalFouls']
weight_col = data[target_col].max()
weight_col
```

```
FieldGoals                   58.000
FieldGoalsAttempted         113.000
FieldGoals.                   0.667
X3PointShots                 24.000
X3PointShotsAttempted        57.000
X3PointShots.                 0.714
FreeThrows                   41.000
FreeThrowsAttempted          64.000
FreeThrows.                   1.000
OffRebounds                  23.000
TotalRebounds                68.000
Assists                      40.000
Steals                       17.000
```

Heatmap


Home
WINorLOSS
FieldGoals
FieldGoalsAttempted
FieldGoals.
X3PointShots
X3PointShotsAttempted
X3PointShots.
FreeThrows
FreeThrowsAttempted
FreeThrows.
OffRebounds
TotalRebounds
Assists
Steals
Blocks
Turnovers
TotalFouls
Opp.FieldGoals
Opp.FieldGoalsAttempted
Opp.FieldGoals.
Opp.3PointShots
pp.3PointShotsAttempted
Opp.3PointShots.
Opp.FreeThrows
pp.FreeThrowsAttempted
Opp.OffRebounds
Opp.TotalRebounds
Opp.Assists
Opp.Steals
Opp.Blocks
Opp.Turnovers
Opp.TotalFouls
0.9
0.6
0.3
0.0
−0.3

Data training & fit (With Keras)

```
from sklearn import model_selection

train_data, test_data, train_label, test_label = model_selection.train_test_split(x_data, y_data, test_size=0.3, random

print(train_data.shape)
print(test_data.shape)
print(train_label.shape)
print(test_label.shape)
```

Data training & fit (With Keras)

```
model = models.Sequential()

model.add(layers.Dense(input_dim=31, units=128, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.BatchNormalization()) # Use this line as if needed
model.add(layers.Activation('elu')) # layers.ELU or layers.LeakyReLU

model.add(layers.Dense(units=256, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.BatchNormalization())
model.add(layers.Activation('elu'))

model.add(layers.Dense(units=256, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.BatchNormalization())
model.add(layers.Activation('elu'))
model.add(layers.Dropout(rate=0.3))

model.add(layers.Dense(units=128, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.BatchNormalization())
model.add(layers.Activation('elu'))



model.add(layers.Dense(units=2, activation='softmax')) # One-hot vector for 0 & 1
```

Data training & fit (With Keras)

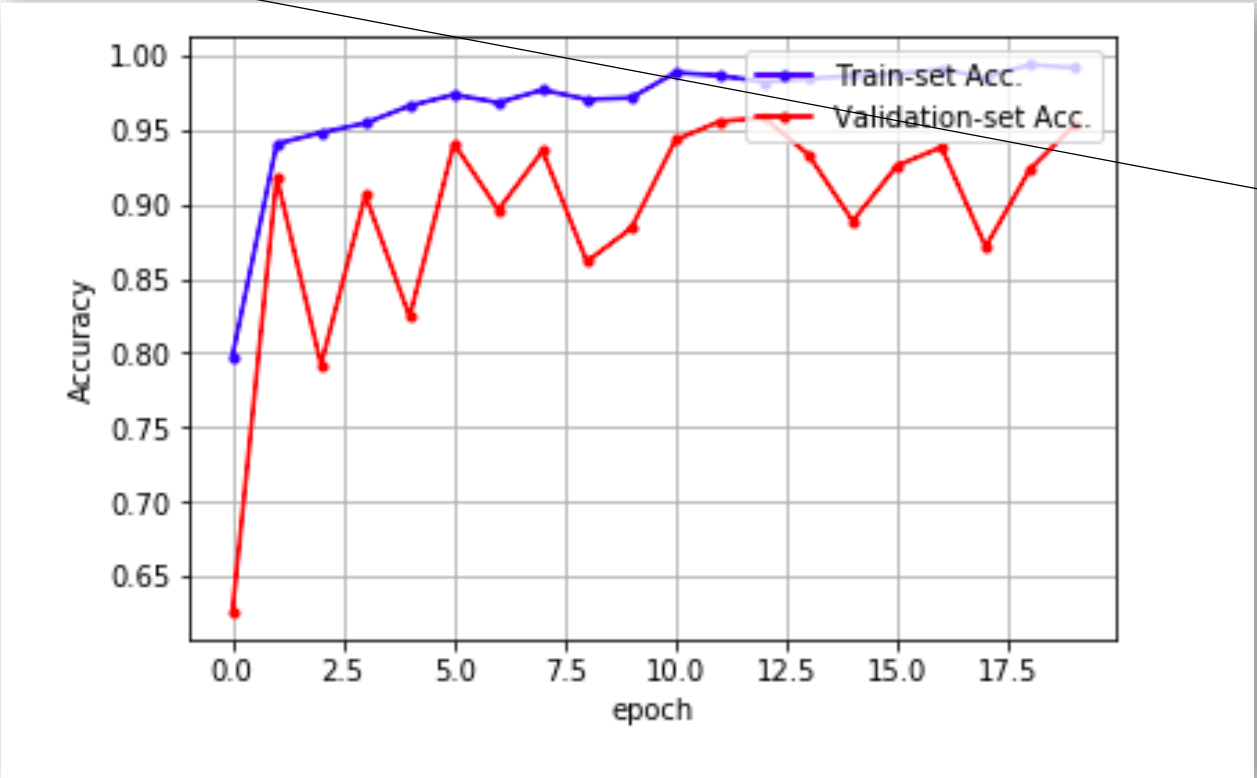
```
model.compile(optimizer=optimizers.Adam(),
              loss=losses.categorical_crossentropy,
              metrics=[metrics.categorical_accuracy])
```

```
history = model.fit(train_data, train_label, batch_size=100, epochs=20, validation_split=0.3)
```

```
Train on 940 samples, validate on 404 samples
Epoch 1/20
940/940 [==============================] - 2s 2ms/sample - loss: 0.4055 - categorical_accuracy: 0.7979 - val_loss: 0.
6024 - val_categorical_accuracy: 0.6262
Epoch 2/20
940/940 [==============================] - 0s 136us/sample - loss: 0.1535 - categorical_accuracy: 0.9404 - val_loss:
0.2604 - val_categorical_accuracy: 0.9183
Epoch 3/20
940/940 [==============================] - 0s 120us/sample - loss: 0.1325 - categorical_accuracy: 0.9479 - val_loss:
0.4208 - val_categorical_accuracy: 0.7921
Epoch 17/20
940/940 [==============================] - 0s 137us/sample - loss: 0.0285 - categorical_accuracy: 0.9904 - val_loss:
0.1610 - val_categorical_accuracy: 0.9381
Epoch 18/20
940/940 [==============================] - 0s 133us/sample - loss: 0.0350 - categorical_accuracy: 0.9851 - val_loss:
0.3097 - val_categorical_accuracy: 0.8713
Epoch 19/20
940/940 [==============================] - 0s 118us/sample - loss: 0.0248 - categorical_accuracy: 0.9936 - val_loss:
0.2219 - val_categorical_accuracy: 0.9233
Epoch 20/20
940/940 [==============================] - 0s 122us/sample - loss: 0.0248 - categorical_accuracy: 0.9915 - val_loss:
0.1117 - val_categorical_accuracy: 0.9530
```

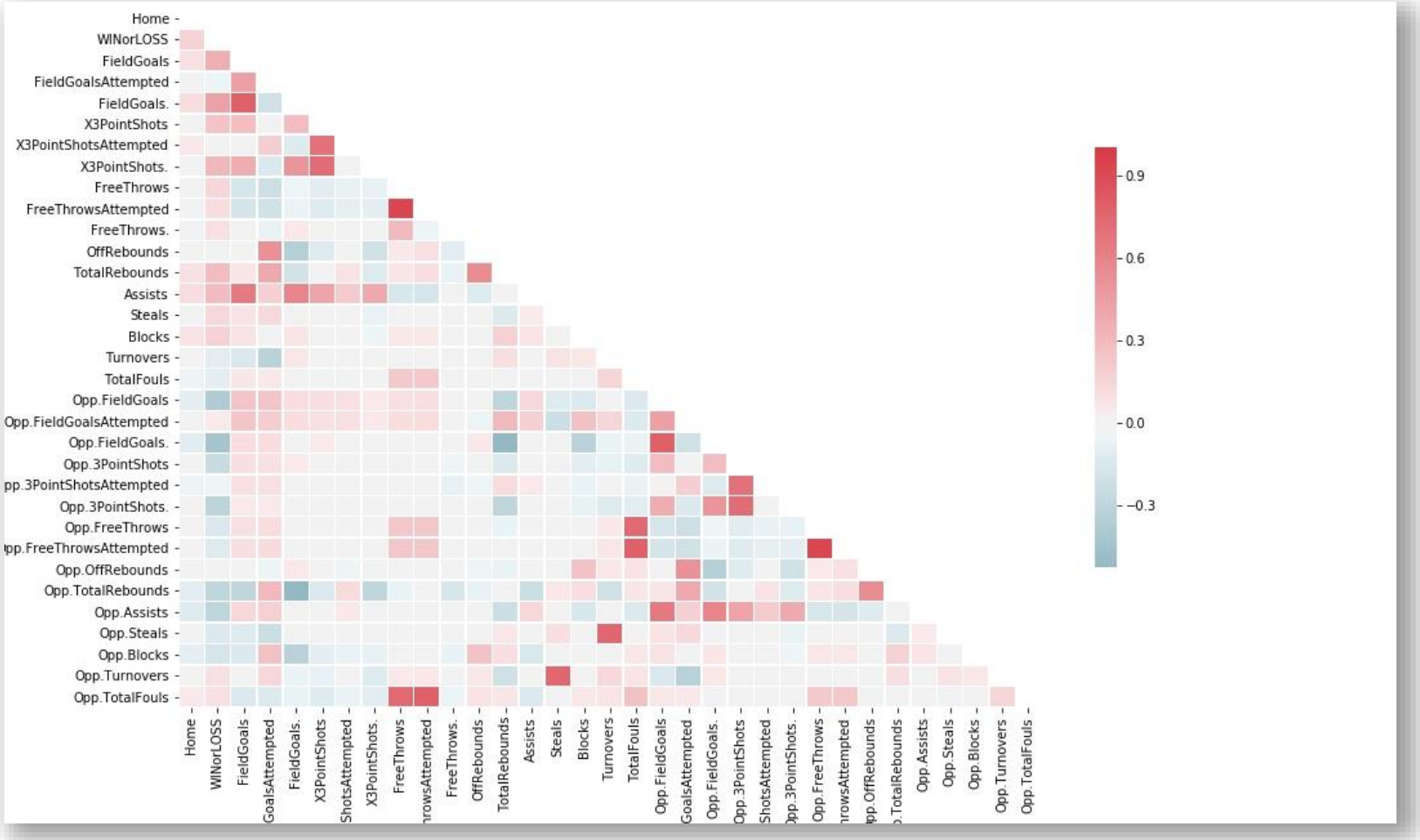
Result

```
576/576 [==============================] - 0s 59us/sample - loss: 0.1279 - categorical_accuracy: 0.9497
loss (cross-entropy) : 0.12793234921991825
test accuracy : 0.9496528
```




```
test accuracy : 0.9496528
```

Result


Home
WINorLOSS
FieldGoals
FieldGoalsAttempted
FieldGoals.
X3PointShots
X3PointShotsAttempted
X3PointShots.
FreeThrows
FreeThrowsAttempted
FreeThrows.
OffRebounds
TotalRebounds
Assists
Steals
Blocks
Turnovers
TotalFouls
Opp.FieldGoals
Opp.FieldGoalsAttempted
Opp.FieldGoals.
Opp.3PointShots
pp.3PointShotsAttempted
Opp.3PointShots.
Opp.FreeThrows
pp.FreeThrowsAttempted
Opp.OffRebounds
Opp.TotalRebounds
Opp.Assists
Opp.Steals
Opp.Blocks
Opp.Turnovers
Opp.TotalFouls
0.9
0.6
0.3
0.0
−0.3

배운 점 및 느낀 점

데이터 전처리의 중요성

데이터 정규화가 결과에 미치는 영향에 대해 다시 한 번 생각해 볼 수 있었음

데이터의 규모

무의미해 보이는 feature도 학습에 영향을 줌


감사합니다.