Mini Project 1

Kyungsik Han

본 영상에서 다룰 내용

CODING

- 머신러닝 Mini project 실습
  - 축구데이터 분석
  - 데이터 살펴보기
  - 데이터 정제
  - 데이터 상관관계
  - 데이터 시각화
  - 데이터 모델링

Kaggle 데이터를 활용한 데이터 분석 개론

Soccer Dataset

CODING

- 본 예제에서는 현재까지 배운 내용을 바탕으로 전반적인 데이터 분석 과정에 대해서 살펴본다

Kaggle 축구 데이터

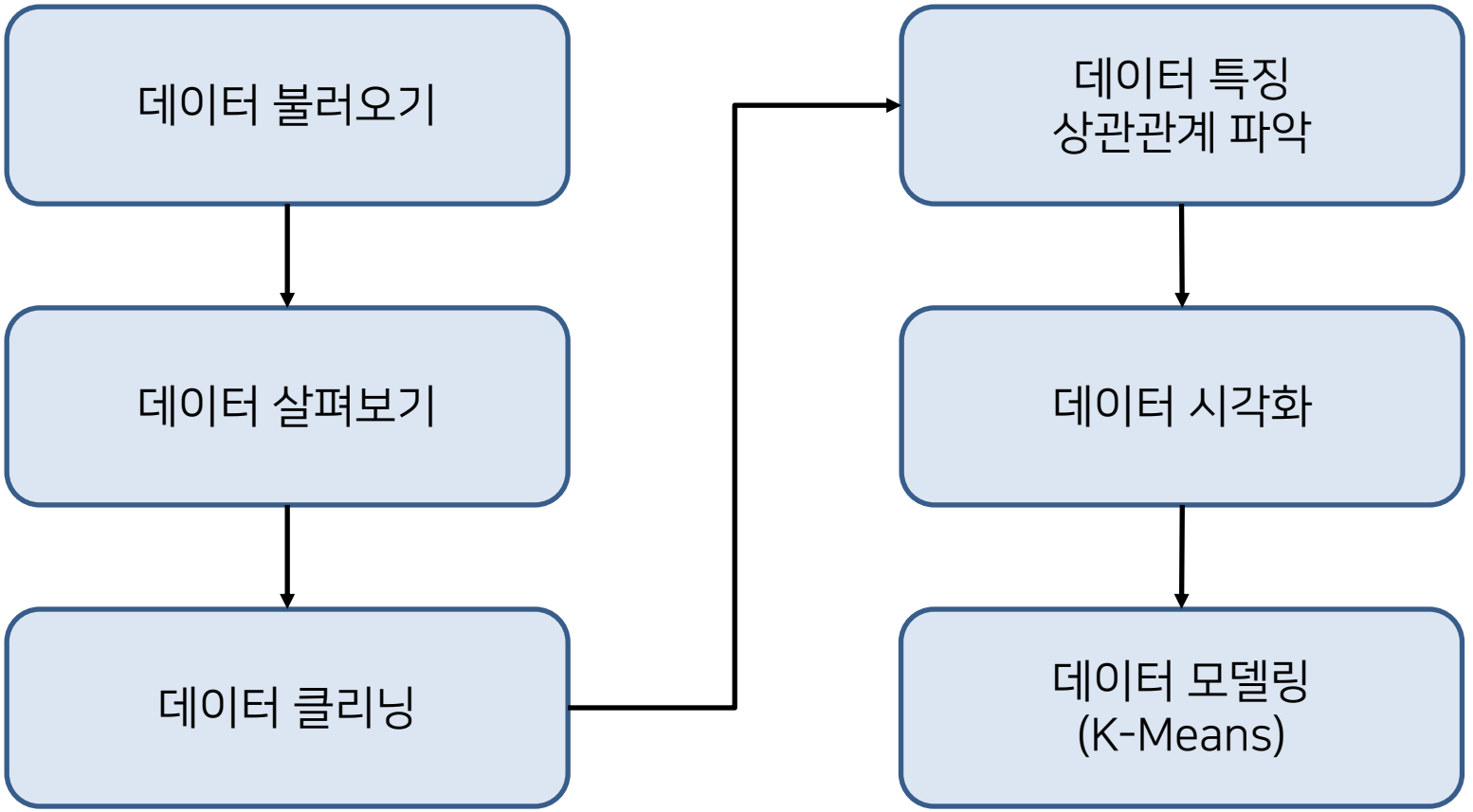
본 데이터에서는 2008-2016년 동안의
유럽 축구 25,000개 경기와 10,000 이상의 선수를 포함

In this project, we will use

CODING

- 데이터베이스 사용: sqlite3
- 데이터 관리: pandas and numpy
- 데이터 시각화: matplotlib
- 머신러닝 알고리즘: scikit-learn (sklearn)

프로젝트 진행 과정



데이터 불러오기

CODING

데이터 불러오기

In [4]:
```
# Create your connection.
cnx = sqlite3.connect('database.sqlite')
df = pd.read_sql_query("SELECT * FROM Player_Attributes", cnx)
```

데이터 살펴보기 (1/2)

CODING

Pandas의 dataframe을 활용하여 데이터의 column을 살펴본다

```
In [5]: df.columns
```

```
Out[5]: Index(['id', 'player_fifa_api_id', 'player_api_id', 'date', 'overall_rating',
       'potential', 'preferred_foot', 'attacking_work_rate',
       'defensive_work_rate', 'crossing', 'finishing', 'heading_accuracy',
       'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_accuracy',
       'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
       'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'stamina',
       'strength', 'long_shots', 'aggression', 'interceptions', 'positioning',
       'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle',
       'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
       'gk_reflexes'],
      dtype='object')
```

데이터 살펴보기 (2/2)

CODING

데이터의 간단한 통계를 살펴본다

```
In [8]: df.describe().transpose()
```

Out[8]:

	count	mean	std	min	25%	50%	75%	max
id	183978.0	91989.500000	53110.018250	1.0	45995.25	91989.5	137983.75	183978.0
player_fifa_api_id	183978.0	165671.524291	53851.094769	2.0	155798.00	183488.0	199848.00	234141.0
player_api_id	183978.0	135900.617324	136927.840510	2625.0	34763.00	77741.0	191080.00	750584.0
overall_rating	183142.0	68.600015	7.041139	33.0	64.00	69.0	73.00	94.0
potential	183142.0	73.460353	6.592271	39.0	69.00	74.0	78.00	97.0
crossing	183142.0	55.086883	17.242135	1.0	45.00	59.0	68.00	95.0
finishing	183142.0	49.921078	19.038705	1.0	34.00	53.0	65.00	97.0
heading_accuracy	183142.0	57.266023	16.488905	1.0	49.00	60.0	68.00	98.0
short_passing	183142.0	62.429672	14.194068	3.0	57.00	65.0	72.00	97.0

데이터 클리닝 (1/2)

CODING

실제 데이터는 노이즈가 심한 경우가 많기 때문에 데이터 정제 작업이 필요하다.
아래 코드에서는 전체 데이터에서 누락된 데이터가 얼마나 있는지 확인해본다.

In [5]:
```
#is any row NULL ?
df.isnull().any().any(), df.shape
```

Out[5]:
```
(True, (183978, 42))
```

In [11]:
```
# Fix it

# Take initial # of rows
rows = df.shape[0]

# Drop the NULL rows
df = df.dropna()
```

null 값이 삭제된 후, 총 갯수가 줄어들었는지 확인

In [13]:
```
#Check if all NULLS are gone ?
print(rows)
df.isnull().any().any(), df.shape
```

```
183978
```

Out[13]:
```
(False, (180354, 42))
```

데이터 클리닝 (2/2)

CODING

Null 값을 삭제한 후 데이터 살펴보기

```
In [15]: df.head(5)
```

Out[15]:

	id	player_fifa_api_id	player_api_id	date	overall_rating	potential	preferred_foot	attacking_work_rate	defensive_work_rate	crossing	...	visio
182972	182973	138155	33076	2014-09-18 00:00:00	68.0	68.0	left	medium	medium	32.0	...	39.(
11280	11281	164299	78541	2008-08-30 00:00:00	73.0	88.0	left	high	medium	78.0	...	79.(
106981	106982	205988	362694	2014-02-14 00:00:00	74.0	86.0	left	high	medium	75.0	...	57.(
74956	74957	199027	202181	2012-02-22 00:00:00	63.0	81.0	left	medium	medium	60.0	...	52.(
2968	2969	187850	112976	2015-07-03 00:00:00	66.0	72.0	right	medium	medium	63.0	...	56.(

5 rows × 42 columns

데이터 특징 간의 상관 관계 파악

CODING

상관 관계 분석: Pearson Correlation 을 많이 사용

- -1 과 +1 사이의 값을 가짐
- Python에서는 corr() 함수를 사용

```
In [17]: df['overall_rating'].corr(df['penalties'])
Out[17]: 0.392715107911188 13
```

이 두 컬럼에 대한 Pearson의 상관 계수는 0.39 이다.

피어슨은 -1 ~ +1 사이의 값을 갖는데, 0.39면 약하게 긍정적인 관계를 가지고 있다고 볼 수 있다.

데이터 특징 간의 상관 관계 파악 (Correlation)

CODING ✖

Create a list of potential Features that you want to measure correlation with

In [15]:
```
potentialFeatures = ['acceleration', 'curve', 'free_kick_accuracy', 'ball_control', 'shot_power', 'stamina']
```

아래의 for 루프는 플레이어의 "overall_rating"의 상관 계수를 목록에 추가 한 각 기능을 잠재성으로 출력한다.

In [16]:
```
# check how the features are correlated with the overall ratings

for f in potentialFeatures:
    related = df['overall_rating'].corr(df[f])
    print("%s: %f" % (f,related))
```

```
acceleration: 0.243998
curve: 0.357566
free_kick_accuracy: 0.349800
ball_control: 0.443991
shot_power: 0.428053
stamina: 0.325606
```

데이터 시각화 (1/2)

CODING ✖

‘overall_rating’ 의 특징과 다른 특징들 간의 상관 관계를 파악

다음으로 "overall_rating"을 사용하여 각 피쳐의 상관 계수를 plotting 한다. 먼저 열을 선택하고 "상관 관계"라는 상관 계수가있는 목록을 만든다.

In [18]:
```
cols = ['potential',  'crossing', 'finishing', 'heading_accuracy',
       'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_accuracy',
       'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
       'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'stamina',
       'strength', 'long_shots', 'aggression', 'interceptions', 'positioning',
       'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle',
       'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
       'gk_reflexes']
```

In [24]:
```
# create a list containing Pearson's correlation between 'overall_rating' with each column in cols
correlations = [ df['overall_rating'].corr(df[f]) for f in cols ]
```

데이터 시각화 (2/2)

CODING ✖

상관 관계 결과 값을 그림으로 표현
→ 결과는 직접 코드로 확인 ☺

In [28]:
```
# create a function for plotting a dataframe with string columns and numeric values

def plot_dataframe(df, y_label):
    color='coral'
    fig = plt.gcf()
    fig.set_size_inches(20, 12)
    plt.ylabel(y_label)

    ax = df.correlation.plot(linewidth=3.3, color=color)
    ax.set_xticks(df.index)
    ax.set_xticklabels(df.attributes, rotation=75)
    plt.show()
```

In [29]:
```
# create a dataframe using cols and correlations

df2 = pd.DataFrame({'attributes': cols, 'correlation': correlations})
```

In [30]:
```
# let's plot above dataframe using the function we created

plot_dataframe(df2, 'Player\'s Overall Rating')
```

데이터 모델링 (k-Means)

CODING ✖

In [31]:
```
# Perform scaling on the dataframe containing the features

data = scale(df_select)

# Define number of clusters
noOfClusters = 4

# Train a model
model = KMeans(init='k-means++', n_clusters=noOfClusters, n_init=20).fit(data)
```

In [32]:
```
print(90*'_')
print("\nCount of players in each cluster")
print(90*'_')

pd.value_counts(model.labels_, sort=False)
```

```
__________________________________________________________________________________________

Count of players in each cluster
__________________________________________________________________________________________
```

Out[32]:
```
0    55898
1    50125
2    23778
3    50553
dtype: int64
```

Let’s see the code

다음 영상에서 배울 내용

CODING X

- 지도학습 회귀(regression) 개념 및 실습

수고하셨습니다