Limpieza y tranformación de datos

Acerca del conjunto de datos

EL conjunto de datos (dataset) a utilizar se encuentra en Kaggle

1. Importar las librerias a utilizar

In [ ]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

2. Cargar o leer el conjunto de datos

In [ ]:
```
datos = pd.read_csv('./fifa21_raw_data.csv')
df = datos.copy()
pd.set_option('display.max_columns', None)
df.head()
```

Out[ ]:

	photoUrl	LongName	playerUrl	Nationality	Po
0	https://cdn.sofifa.com/players/158/023/21_60.png	Lionel Messi	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	RW
1	https://cdn.sofifa.com/players/020/801/21_60.png	C. Ronaldo dos Santos Aveiro	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	
2	https://cdn.sofifa.com/players/200/389/21_60.png	Jan Oblak	http://sofifa.com/player/200389/jan-oblak/210005/	Slovenia	
3	https://cdn.sofifa.com/players/192/985/21_60.png	Kevin De Bruyne	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	CA
4	https://cdn.sofifa.com/players/190/871/21_60.png	Neymar da Silva Santos Jr.	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	LW

3. Limpieza de datos

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18979 entries, 0 to 18978
Data columns (total 77 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   photoUrl          18979 non-null  object
 1   LongName          18979 non-null  object
 2   playerUrl         18979 non-null  object
 3   Nationality       18979 non-null  object
 4   Positions         18979 non-null  object
 5   Name              18979 non-null  object
 6   Age               18979 non-null  int64
 7   ↓OVA              18979 non-null  int64
 8   POT               18979 non-null  int64
 9   Team & Contract   18979 non-null  object
 10  ID                18979 non-null  int64
 11  Height            18979 non-null  object
 12  Weight            18979 non-null  object
 13  foot              18979 non-null  object
 14  BOV               18979 non-null  int64
 15  BP                18979 non-null  object
 16  Growth            18979 non-null  int64
 17  Joined            18979 non-null  object
 18  Loan Date End     1013 non-null   object
 19  Value             18979 non-null  object
 20  Wage              18979 non-null  object
 21  Release Clause    18979 non-null  object
 22  Attacking         18979 non-null  int64
 23  Crossing          18979 non-null  int64
 24  Finishing         18979 non-null  int64
 25  Heading Accuracy  18979 non-null  int64
 26  Short Passing     18979 non-null  int64
 27  Volleys           18979 non-null  int64
 28  Skill             18979 non-null  int64
 29  Dribbling         18979 non-null  int64
 30  Curve             18979 non-null  int64
 31  FK Accuracy       18979 non-null  int64
 32  Long Passing      18979 non-null  int64
 33  Ball Control      18979 non-null  int64
 34  Movement          18979 non-null  int64
 35  Acceleration      18979 non-null  int64
 36  Sprint Speed      18979 non-null  int64
 37  Agility           18979 non-null  int64
 38  Reactions         18979 non-null  int64
 39  Balance           18979 non-null  int64
 40  Power             18979 non-null  int64
 41  Shot Power        18979 non-null  int64
 42  Jumping           18979 non-null  int64
 43  Stamina           18979 non-null  int64
 44  Strength          18979 non-null  int64
 45  Long Shots        18979 non-null  int64
 46  Mentality         18979 non-null  int64
 47  Aggression        18979 non-null  int64
 48  Interceptions     18979 non-null  int64
 49  Positioning       18979 non-null  int64
 50  Vision            18979 non-null  int64
 51  Penalties         18979 non-null  int64
 52  Composure         18979 non-null  int64
 53  Defending         18979 non-null  int64
 54  Marking           18979 non-null  int64
 55  Standing Tackle   18979 non-null  int64
```

```
 56  Sliding Tackle    18979 non-null  int64
 57  Goalkeeping       18979 non-null  int64
 58  GK Diving         18979 non-null  int64
 59  GK Handling       18979 non-null  int64
 60  GK Kicking        18979 non-null  int64
 61  GK Positioning    18979 non-null  int64
 62  GK Reflexes       18979 non-null  int64
 63  Total Stats       18979 non-null  int64
 64  Base Stats        18979 non-null  int64
 65  W/F               18979 non-null  object
 66  SM                18979 non-null  object
 67  A/W               18979 non-null  object
 68  D/W               18979 non-null  object
 69  IR                18979 non-null  object
 70  PAC               18979 non-null  int64
 71  SHO               18979 non-null  int64
 72  PAS               18979 non-null  int64
 73  DRI               18979 non-null  int64
 74  DEF               18979 non-null  int64
 75  PHY               18979 non-null  int64
 76  Hits              18979 non-null  object
dtypes: int64(55), object(22)
memory usage: 11.1+ MB
```

3.1 Columnas innecesarias

In [ ]:
```
# Eliminar columnas innecesarias
cols_eliminar = ['photoUrl','playerUrl']
df.drop(columns=cols_eliminar, inplace=True)
```

3.2 Columnas a limpiar

Estas columnas son:

- Team & Contract
- Height
- Weight
- Joined
- Value
- Release Clause
- W/F
- SM
- IR
- Hits
- Positions

3.2.1 Columna `Team & Contract`

In [ ]:
```
df['Team & Contract'].head()
```

Out[ ]:
```
0           \n\n\n\nFC Barcelona\n2004 ~ 2021\n\n
1               \n\n\n\nJuventus\n2018 ~ 2022\n\n
2        \n\n\n\nAtlético Madrid\n2014 ~ 2023\n\n
3        \n\n\n\nManchester City\n2015 ~ 2023\n\n
4    \n\n\n\nParis Saint-Germain\n2017 ~ 2022\n\n
Name: Team & Contract, dtype: object
```

In [ ]:
```
# Quitar los valores '\n' de la columna 'Tema & Contract'
df['Team & Contract'].replace('\n', '', regex=True, inplace=True)
df['Team & Contract'].head()
```

Out[ ]:
```
0           FC Barcelona2004 ~ 2021
1               Juventus2018 ~ 2022
2        Atlético Madrid2014 ~ 2023
3        Manchester City2015 ~ 2023
4    Paris Saint-Germain2017 ~ 2022
Name: Team & Contract, dtype: object
```

In [ ]:
```
# Extraer el nombre del equipo y los años de duración del contrato
df['Team'] = df['Team & Contract'].str.split('\d+ ~ \d+').str[0].str.strip()
df['Contract'] = df['Team & Contract'].str.extract(r'(\d+ ~ \d+)')

# Eliminar la columna 'Team & Contract'
df.drop(columns='Team & Contract', inplace=True)

print(df['Team'].head())
print('-'*50)
print(df['Contract'].head())
```

```
0           FC Barcelona
1               Juventus
2        Atlético Madrid
3        Manchester City
4    Paris Saint-Germain
Name: Team, dtype: object
--------------------------------------------------
0    2004 ~ 2021
1    2018 ~ 2022
2    2014 ~ 2023
3    2015 ~ 2023
4    2017 ~ 2022
Name: Contract, dtype: object

--------------------------------------------------
0    2004 ~ 2021
1    2018 ~ 2022
2    2014 ~ 2023
3    2015 ~ 2023
4    2017 ~ 2022
Name: Contract, dtype: object
```

3.2.2 Columna `Height`

In [ ]:
```
df['Height'].head()
```

Out[ ]:
```
0     5'7"
1     6'2"
2     6'2"
3    5'11"
4     5'9"
Name: Height, dtype: object
```

In [ ]:
```
# Pasar 'Height' a pulgadas

# Definir la función para la conversión de altura a pulgadas
def convertir_a_pulgadas(Height):
    pies, pulgadas = Height.split("'")
    pies = int(pies)
    pulgadas = int(pulgadas.replace("\"", ""))
    altura_pulgadas = pies * 12 + pulgadas
    return altura_pulgadas

# Aplicar la función utilizando operaciones vectorizadas
df['Height_inches'] = df['Height'].apply(convertir_a_pulgadas)

df['Height_inches'].head()
```

Out[ ]:
```
0    67
1    74
2    74
3    71
4    69
Name: Height_inches, dtype: int64
```

In [ ]:
```
# Eliminar la columna 'Height'
df.drop(columns='Height', inplace=True)

# Renombrar la columna 'Height_inches' a 'Height'
df = df.rename(columns={'Height_inches': 'Height'})

# Cambiar de float a int la columna
df['Height'] = df['Height'].astype(int)
df['Height'].head()
```

Out[ ]:
```
0    67
1    74
2    74
3    71
4    69
Name: Height, dtype: int32
```

3.2.3 Columna `Weight`

In [ ]:
```
df['Weight'].head()
```

Out[ ]:
```
0    159lbs
1    183lbs
2    192lbs
3    154lbs
4    150lbs
Name: Weight, dtype: object
```

In [ ]:
```
# Quitar el string 'lbs'
df['Weight'].replace('lbs', '', regex=True, inplace=True)
df['Weight'].head()
```

Out[ ]:
```
0    159
1    183
2    192
3    154
4    150
Name: Weight, dtype: object
```

3.2.4 Columna `Joined`

In [ ]:
```
df['Joined'].head()
```

Out[ ]:
```
0     Jul 1, 2004
1    Jul 10, 2018
2    Jul 16, 2014
3    Aug 30, 2015
4     Aug 3, 2017
Name: Joined, dtype: object
```

In [ ]:
```
# Convertir a tipo de dato datetime
df['Joined'] = pd.to_datetime(df['Joined'])

# Aplicar el formato deseado a la columna 'Joined'
df['Joined'] = df['Joined'].dt.strftime('%m-%d-%Y')
df['Joined'] = pd.to_datetime(df['Joined'])

df['Joined'].head()
```

Out[ ]:
```
0   2004-07-01
1   2018-07-10
2   2014-07-16
3   2015-08-30
4   2017-08-03
Name: Joined, dtype: datetime64[ns]
```

3.2.5 Columna `Value`

In [ ]:
```
df['Value'].head()
```

Out[ ]:
```
0    €67.5M
1      €46M
2      €75M
3      €87M
4      €90M
Name: Value, dtype: object
```

In [ ]:
```
valor_temporal = []

# Reemplazar caracteres y convertir a valor numérico
for valor in df['Value']:
    valor = str(valor).replace('€', '')
    valor = valor.replace('K', '000')
    valor = valor.replace('M', '000000')
    valor = valor.replace('.', 'F')

    if 'F' in valor:
        valor = valor.replace('F', '')
        valor = int(valor) / 10

    valor_temporal.append(int(valor))
```

```
df['Value'] = valor_temporal
df['Value'].head()
```

Out[ ]:
```
0    67500000
1    46000000
2    75000000
3    87000000
4    90000000
Name: Value, dtype: int64
```

3.2.6 Columna `Release Clause`

In [ ]:
```
df['Release Clause'].head()
```

Out[ ]:
```
0    €138.4M
1     €75.9M
2    €159.4M
3      €161M
4    €166.5M
Name: Release Clause, dtype: object
```

In [ ]:
```
valor_temporal = []

# Reemplazar caracteres y convertir a valor numérico
for valor in df['Release Clause']:
    valor = str(valor).replace('€', '')
    valor = valor.replace('K', '000')
    valor = valor.replace('M', '000000')
    valor = valor.replace('.', 'F')

    if 'F' in valor:
        valor = valor.replace('F', '')
        valor = int(valor) / 10

    valor_temporal.append(int(valor))

df['Release Clause'] = valor_temporal
df['Release Clause'].head()
```

Out[ ]:
```
0    138400000
1     75900000
2    159400000
3    161000000
4    166500000
Name: Release Clause, dtype: int64
```

3.2.7 Columnas `W/F`

In [ ]:
```
df['W/F'].head()
```

Out[ ]:
```
0    4 ★
1    4 ★
2    3 ★
3    5 ★
4    5 ★
Name: W/F, dtype: object
```

In [ ]:
```
# Eliminar los caracteres "★" de la columna
df['W/F'] = df['W/F'].str.replace('★', '')

# Convertir la columna a tipo entero
df['W/F'] = df['W/F'].astype(int)
df['W/F'].head()
```

Out[ ]:
```
0    4
1    4
2    3
3    5
4    5
Name: W/F, dtype: int32
```

3.2.8 Columna `SM`

In [ ]:
```
df['SM'].head()
```

Out[ ]:
```
0    4★
1    5★
2    1★
3    4★
4    5★
Name: SM, dtype: object
```

In [ ]:
```
# Eliminar los caracteres "★" de la columna
df['SM'] = df['SM'].str.replace('★', '')

# Convertir la columna a tipo entero
df['SM'] = df['SM'].astype(int)
df['SM'].head()
```

Out[ ]:
```
0    4
1    5
2    1
3    4
4    5
Name: SM, dtype: int32
```

2.3.9 Columna `IR`

In [ ]:
```
df['IR'].head()
```

Out[ ]:
```
0    5 ★
1    5 ★
2    3 ★
3    4 ★
4    5 ★
Name: IR, dtype: object
```

In [ ]:
```
# Eliminar los caracteres "★" de la columna
df['IR'] = df['IR'].str.replace('★', '')
```

```
# Convertir la columna a tipo entero
df['IR'] = df['IR'].astype(int)
df['IR'].head()
```

Out[ ]:
```
0    5
1    5
2    3
3    4
4    5
Name: IR, dtype: int32
```

2.3.10 Columna `Hits`

In [ ]:
```
df['Hits'].head()
```

Out[ ]:
```
0    \n372
1    \n344
2     \n86
3    \n163
4    \n273
Name: Hits, dtype: object
```

In [ ]:
```
valor_temporal = []

# Reemplazar caracteres y convertir a valor numérico
for valor in df['Hits']:
    valor = str(valor).replace('\n', '')
    valor = valor.replace('K', '000')
    valor = valor.replace('.', 'F')

    if 'F' in valor:
        valor = valor.replace('F', '')
        valor = int(valor) / 10

    valor_temporal.append(int(valor))

df['Hits'] = valor_temporal
df['Hits'].head()
```

Out[ ]:
```
0    372
1    344
2     86
3    163
4    273
Name: Hits, dtype: int64
```

2.3.11 Columna `Wage`

In [ ]:
```
df['Wage'].head()
```

Out[ ]:
```
0    €560K
1    €220K
2    €125K
3    €370K
4    €270K
Name: Wage, dtype: object
```

In [ ]:
```
valor_temporal = []

# Reemplazar caracteres y convertir a valor numérico
for valor in df['Wage']:
    valor = str(valor).replace('€', '')
    valor = valor.replace('K', '000')
    valor = valor.replace('M', '000000')
    valor = valor.replace('.', 'F')

    if 'F' in valor:
        valor = valor.replace('F', '')
        valor = int(valor) / 10

    valor_temporal.append(int(valor))

df['Wage'] = valor_temporal
df['Wage'].head()
```

Out[ ]:
```
0    560000
1    220000
2    125000
3    370000
4    270000
Name: Wage, dtype: int64
```

4 Verificación de la limpieza, transformacion de los datos y exportacion

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

Out[ ]:

	LongName	Nationality	Positions	Name	Age	↓OVA	POT	ID	Weight	foot	BOV	BP	Growth	Joi
0	Lionel Messi	Argentina	RW ST CF	L. Messi	33	93	93	158023	159	Left	93	RW	0	20 0
1	C. Ronaldo dos Santos Aveiro	Portugal	ST LW	Cristiano Ronaldo	35	92	92	20801	183	Right	92	ST	0	20 0
2	Jan Oblak	Slovenia	GK	J. Oblak	27	91	93	200389	192	Right	91	GK	2	20 0
3	Kevin De Bruyne	Belgium	CAM CM	K. De Bruyne	29	91	91	192985	154	Right	91	CAM	0	20 08
4	Neymar da Silva Santos Jr.	Brazil	LW CAM	Neymar Jr	28	91	91	190871	150	Right	91	LW	0	20 08

In [ ]:
```
# Exportar archivo a formato .csv

df.to_csv('clean_fifa21_raw_data.csv', index=False)
```