Рубежный контроль номер 1

Мирсонов Вячеслав РТ5-61Б

Тема: Технологии разведочного анализа и обработки данных.

Задача 2.

Для заданного набора данных проведите обработку пропусков в данных для одного категориального и одного количественного признака. Какие способы обработки пропусков в данных для категориальных и количественных признаков Вы использовали? Какие признаки Вы будете использовать для дальнейшего построения моделей машинного обучения и почему?

Для анализа будем использовать датасет, содержащий полный набор данных игроков FIFA 19 (https://www.kaggle.com/karangadiya/fifa19).

In [108]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [109]:

```
# Будем использовать только обучающую выборку
data = pd.read_csv('data1.csv', sep=",", quoting=3)
```

```
Z:\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3146: DtypeWarning: Columns (29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54) have mixed types.Specify dtype option on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [110]:

```
# размер набора данных
data.shape
```

Out[110]:

```
(18179, 89)
```

In [111]:

```
# типы колонок
data.dtypes
```

Out[111]:

```
NumID              int64
ID                object
Name               int64
Age               object
Photo             object
                   ...
GKHandling       float64
GKKicking        float64
GKPositioning    float64
GKReflexes       float64
Release Clause    object
Length: 89, dtype: object
```

In [112]:

```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

Out[112]:

```
NumID              0
ID                 0
Name               0
Age                0
Photo              0
                  ...
GKHandling        48
GKKicking         48
GKPositioning     48
GKReflexes       289
Release Clause   289
Length: 89, dtype: int64
```

In [113]:

```
# Первые 5 строк датасета
data.head()
```

Out[113]:

	NumID	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	
"0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	https://cdn.sofifa.org/te
"1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	https://cdn.sofifa.org/
"2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	https://cdn.sofifa.org/
"3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	https://cdn.sofifa.org/
"4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	https://cdn.sofifa.org/

5 rows × 89 columns

In [114]:

```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 18179
```

Заполним пропуски столбца категориального признака Позиционирование (GKPositioning)

In [115]:

```
data['GKPositioning']
```

Out[115]:

```
"0       14.0
"1       14.0
"2       15.0
"3       88.0
"4       10.0
        ...
"18202    8.0
"18203    5.0
"18204    6.0
"18205    8.0
"18206   12.0
Name: GKPositioning, Length: 18179, dtype: float64
```

In [116]:

```
#Проверим значения
data['GKPositioning'].unique()
```

Out[116]:

```
array([14., 15., 88., 10.,  8., 33.,  7.,  5.,  6., 13., 85., 86.,  9.,
       87., 11.,  4., 12., 83., 89., 90., 82.,  2., 16., 84., 81., 79.,
       80., 19.,  3., 78., 77.,  1., 75., 76., 74., 72., 32., 17., 73.,
       65., 70., 18., 71., 69., 20., 68., 67., 62., 24., 66., 64., 63.,
       30., 55., 59., 61., 60., 23., 57., 58., 50., 54., 51., 49., nan,
       56., 53., 52., 27., 38., 47., 48., 46., 44., 45., 41., 42., 43.,
       39., 40.])
```

In [117]:

```
from sklearn.impute import SimpleImputer
```

In [118]:

```
#Количество пустых значений
data[data['GKPositioning'].isnull()].shape[0]
```

Out[118]:

```
48
```

In [119]:

```
imp = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data['GKPositioning'] = imp.fit_transform(data[['GKPositioning']])
```

In [120]:

```
#проверка столбца на пустые значения
data[data['GKPositioning'].isnull()].shape[0]
```

Out[120]:

```
0
```

In [121]:

```
#Проверим значения
data['GKPositioning'].unique()
```

Out[121]:

```
array([14., 15., 88., 10.,  8., 33.,  7.,  5.,  6., 13., 85., 86.,  9.,
       87., 11.,  4., 12., 83., 89., 90., 82.,  2., 16., 84., 81., 79.,
       80., 19.,  3., 78., 77.,  1., 75., 76., 74., 72., 32., 17., 73.,
       65., 70., 18., 71., 69., 20., 68., 67., 62., 24., 66., 64., 63.,
       30., 55., 59., 61., 60., 23., 57., 58., 50., 54., 51., 49., 56.,
       53., 52., 27., 38., 47., 48., 46., 44., 45., 41., 42., 43., 39.,
       40.])
```

Как видно, значения "nan" уже нет. Значит ход выполнения правильный

Заполним пропуски столбца количественного признака "Вес".

In [130]:

```
from sklearn.impute import MissingIndicator
```

In [131]:

```
data['Weight']
```

Out[131]:

```
"0       159.0
"1       183.0
"2       150.0
"3       168.0
"4       154.0
         ...
"18202   134.0
"18203   170.0
"18204   148.0
"18205   154.0
"18206   176.0
Name: Weight, Length: 18179, dtype: float64
```

In [132]:

```
# Выведем номера строк, в которых значения
empty_index = data[data['Weight'].isnull()].index
data[data.index.isin(empty_index)]['Weight']
```

Out[132]:

```
Series([], Name: Weight, dtype: float64)
```

In [133]:

```
temp_data = data[['Weight']]
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data[['Weight']])
imp_num = SimpleImputer(strategy = 'median')
data[['Weight']] = imp_num.fit_transform(data[['Weight']])
```

In [135]:

```
#проверка столбца на пустые значения
data[data['Weight'].isnull()].shape[0]
```

Out[135]:

```
0
```

In [136]:

```
data['Weight']
```

Out[136]:

```
"0       159.0
"1       183.0
"2       150.0
"3       168.0
"4       154.0
         ...
"18202   134.0
"18203   170.0
"18204   148.0
"18205   154.0
"18206   176.0
Name: Weight, Length: 18179, dtype: float64
```

Join-plot

Для двух колонок: [GKPositioning] и [Weight], построим joinplot-диаграмму.

In [138]:

```
sns.jointplot(x='GKPositioning', y='Weight', data=data)
```

Out[138]:

<seaborn.axisgrid.JointGrid at 0x18acac80160>