Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Автоматизированные системы обработки информации и управления»

Отчет

Рубежный контроль № 1

По курсу «Технологии машинного обучения»

ИСПОЛНИТЕЛЬ:
Группа ИУ5-65Б
Попов М.А.

"18" апреля 2020 г.

ПРЕПОДАВАТЕЛЬ:
Гапанюк Ю.Е.

_____

"__"_____2021 г.

Москва 2021

1. **Задание**
   Для заданного набора данных произведите масштабирование данных (для одного признака) и преобразование категориальных признаков в количественные двумя способами (label encoding, one hot encoding) для одного признака. Какие методы Вы использовали для решения задачи и почему?
   Для набора данных построить "парные диаграммы".
2. **Скрины jupyter notebook**

Импорт библиотек и загрузка данных

```
[1] import numpy as np
    import pandas as pd
    import seaborn as sns
    import matplotlib.pyplot as plt
    from google.colab import files
    %matplotlib inline
    sns.set(style="ticks")
```

```
uploaded = files.upload()
```

```
Выбрать файлы data.csv
• data.csv(application/vnd.ms-excel) - 9140113 bytes, last modified: 11.10.2019 - 100% done
Saving data.csv to data.csv
```

```
[3] data = pd.read_csv('data.csv', sep=",")
```

Основные характеристики датасета

```
[4] # первые 5 строк датасета
    data.head()
```

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	Club L
0	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/teams/2/light/241.
1	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/teams/2/light/45.
2	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/teams/2/light/73.
3	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/teams/2/light/11.
4	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/teams/2/light/10.

5 rows × 89 columns

```
[6] # размер датасета
    data.shape

    (18207, 89)
```

```
[7] # типы колонок
    data.dtypes

    Unnamed: 0          int64
    ID                  int64
    Name               object
    Age                 int64
    Photo              object
                        ...
    GKHandling        float64
    GKKicking         float64
    GKPositioning     float64
    GKReflexes        float64
    Release Clause     object
    Length: 89, dtype: object
```

[8]
```
# проверка на наличие пустых значений
for col in data.columns:
    temp_null_count = data[data[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
Unnamed: 0 - 0
ID - 0
Name - 0
Age - 0
Photo - 0
Nationality - 0
Flag - 0
Overall - 0
Potential - 0
Club - 241
Club Logo - 0
Value - 0
Wage - 0
Special - 0
Preferred Foot - 48
International Reputation - 48
Weak Foot - 48
Skill Moves - 48
Work Rate - 48
Body Type - 48
Real Face - 48
Position - 60
Jersey Number - 60
Joined - 1553
Loaned From - 16943
Contract Valid Until - 289
Height - 48
Weight - 48
```

[8]
```
LS - 2085
ST - 2085
RS - 2085
LW - 2085
LF - 2085
CF - 2085
RF - 2085
RW - 2085
LAM - 2085
CAM - 2085
RAM - 2085
LM - 2085
LCM - 2085
CM - 2085
RCM - 2085
RM - 2085
LWB - 2085
LDM - 2085
CDM - 2085
RDM - 2085
RWB - 2085
LB - 2085
LCB - 2085
CB - 2085
RCB - 2085
RB - 2085
Crossing - 48
Finishing - 48
HeadingAccuracy - 48
ShortPassing - 48
Volleys - 48
Dribbling - 48
Curve - 48
FKAccuracy - 48
LongPassing - 48
BallControl - 48
Acceleration - 48
SprintSpeed - 48
Agility - 48
Reactions - 48
Balance - 48
ShotPower - 48
Jumping - 48
Stamina - 48
Strength - 48
LongShots - 48
Aggression - 48
Interceptions - 48
Positioning - 48
Vision - 48
Penalties - 48
Composure - 48
Marking - 48
StandingTackle - 48
SlidingTackle - 48
GKDiving - 48
GKHandling - 48
GKKicking - 48
GKPositioning - 48
GKReflexes - 48
Release Clause - 1564
```

[10]
```
# поиск колонок, в которых нет пустых значений
data.columns[data.notna().all()]
```

```
Index(['Unnamed: 0', 'ID', 'Name', 'Age', 'Photo', 'Nationality', 'Flag',
       'Overall', 'Potential', 'Club Logo', 'Value', 'Wage', 'Special'],
      dtype='object')
```

Подбор колонок для преобразования категориальных признаков в количественные

[13]
```
# поиск категориальных признаков, в которых мало уникальных значений
unique_objects = data.select_dtypes(include=['object']).nunique().sort_values().head(5)
print(unique_objects)
```

```
Preferred Foot     2
Real Face          2
Work Rate          9
Body Type         10
Height            21
dtype: int64
```

[16]
```
# Вывод категориальных признаков с указанием уникальных значений
category_cols = unique_objects.index.tolist()
for col in data[category_cols]:
  print(col, '-', data[col].unique(), ', кол-во пустых значений: ', data[col].isnull().sum())
```

```
Preferred Foot - ['Left' 'Right' nan] , кол-во пустых значений:  48
Real Face - ['Yes' 'No' nan] , кол-во пустых значений:  48
Work Rate - ['Medium/ Medium' 'High/ Low' 'High/ Medium' 'High/ High' 'Medium/ High'
 'Medium/ Low' 'Low/ High' 'Low/ Medium' 'Low/ Low' nan] , кол-во пустых значений:  48
Body Type - ['Messi' 'C. Ronaldo' 'Neymar' 'Lean' 'Normal' 'Courtois' 'Stocky'
 'PLAYER_BODY_TYPE_25' 'Shaqiri' 'Akinfenwa' nan] , кол-во пустых значений:  48
Height - ["5'7" "6'2" "5'9" "6'4" "5'11" "5'8" "6'0" "5'6" "5'10" "6'6" "6'1" "5'4"
 "6'3" "5'5" "6'5" "6'7" "5'3" "5'2" "6'8" "5'1" "6'9" nan] , кол-во пустых значений:  48
```

[19]
```
# проверка на связь строк, в которых есть пустые значения, чтобы удалить их из датасета
data[data[category_cols[0]].isnull()][category_cols]
```

	Preferred Foot	Real Face	Work Rate	Body Type	Height
13236	NaN	NaN	NaN	NaN	NaN
13237	NaN	NaN	NaN	NaN	NaN
13238	NaN	NaN	NaN	NaN	NaN
13239	NaN	NaN	NaN	NaN	NaN
13240	NaN	NaN	NaN	NaN	NaN
13241	NaN	NaN	NaN	NaN	NaN
13242	NaN	NaN	NaN	NaN	NaN
13243	NaN	NaN	NaN	NaN	NaN
13244	NaN	NaN	NaN	NaN	NaN
13245	NaN	NaN	NaN	NaN	NaN
13246	NaN	NaN	NaN	NaN	NaN
13247	NaN	NaN	NaN	NaN	NaN
13248	NaN	NaN	NaN	NaN	NaN
13249	NaN	NaN	NaN	NaN	NaN
13250	NaN	NaN	NaN	NaN	NaN
13251	NaN	NaN	NaN	NaN	NaN
13252	NaN	NaN	NaN	NaN	NaN

[19]

13252	NaN	NaN	NaN	NaN	NaN
13253	NaN	NaN	NaN	NaN	NaN
13254	NaN	NaN	NaN	NaN	NaN
13255	NaN	NaN	NaN	NaN	NaN
13256	NaN	NaN	NaN	NaN	NaN
13257	NaN	NaN	NaN	NaN	NaN
13258	NaN	NaN	NaN	NaN	NaN
13259	NaN	NaN	NaN	NaN	NaN
13260	NaN	NaN	NaN	NaN	NaN
13261	NaN	NaN	NaN	NaN	NaN
13262	NaN	NaN	NaN	NaN	NaN
13263	NaN	NaN	NaN	NaN	NaN
13264	NaN	NaN	NaN	NaN	NaN
13265	NaN	NaN	NaN	NaN	NaN
13266	NaN	NaN	NaN	NaN	NaN
13267	NaN	NaN	NaN	NaN	NaN
13268	NaN	NaN	NaN	NaN	NaN
13269	NaN	NaN	NaN	NaN	NaN
13270	NaN	NaN	NaN	NaN	NaN
13271	NaN	NaN	NaN	NaN	NaN
13272	NaN	NaN	NaN	NaN	NaN
13273	NaN	NaN	NaN	NaN	NaN
13274	NaN	NaN	NaN	NaN	NaN
13275	NaN	NaN	NaN	NaN	NaN
13276	NaN	NaN	NaN	NaN	NaN
13277	NaN	NaN	NaN	NaN	NaN
13278	NaN	NaN	NaN	NaN	NaN
13279	NaN	NaN	NaN	NaN	NaN
13280	NaN	NaN	NaN	NaN	NaN
13281	NaN	NaN	NaN	NaN	NaN
13282	NaN	NaN	NaN	NaN	NaN
13283	NaN	NaN	NaN	NaN	NaN

```
[22] # удаляем строки, т.к. значения не определены во всех колонках
     data = data[data[category_cols[0]].notna()]
     # проверяем
     data[data[category_cols[0]].isnull()][category_cols]
```

	Preferred Foot	Real Face	Work Rate	Body Type	Height

Преобразование категориальных признаков методами: label encoding и one hot encoding

```
[23] # импорт
     from sklearn.preprocessing import LabelEncoder
```

```
[25] # label encdoing для колонки Real Face
     le = LabelEncoder()
     face_le = le.fit_transform(data['Real Face'])
     print('Real Face, label encoded -', face_le)
     print('Real Face, unique values -', np.unique(face_le))
     print('Real Face, source values -', le.inverse_transform(face_le))
```

```
Real Face, label encoded - [1 1 1 ... 0 0 0]
Real Face, unique values - [0 1]
Real Face, source values - ['Yes' 'Yes' 'Yes' ... 'No' 'No' 'No']
```

```
[27] # one hot encdoging для колонки Work Rate
     pd.get_dummies(data['Work Rate'], dummy_na=True, prefix='Rate').head(n=15)
```

	Rate_High/ High	Rate_High/ Low	Rate_High/ Medium	Rate_Low/ High	Rate_Low/ Low	Rate_Low/ Medium	Rate_Medium/ High	Rate_Medium/ Low	Rate_Medium/ Medium	Rate_nan
0	0	0	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	0
4	1	0	0	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	0
8	0	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	0
10	0	0	1	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	0
12	0	0	0	0	0	0	1	0	0	0
13	0	0	1	0	0	0	0	0	0	0
14	0	0	0	0	0	0	1	0	0	0

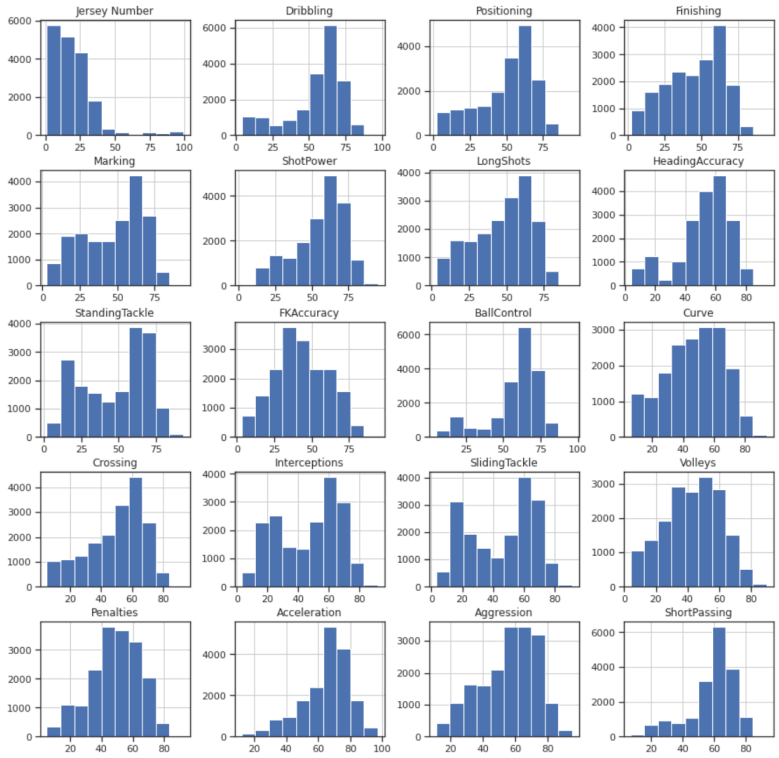
Поиск колонки для масштабирования данных

```
[30] # выборка 20 количественных признаков с количеством уникальных значений меньше 1000
     quantity_cols = data.select_dtypes(exclude=['object']).nunique().where(lambda x:x < 1000).sort_values(ascending=False).head(20)
     print(quantity_cols)
```

```
Jersey Number      99.0
Dribbling          94.0
Positioning        94.0
Finishing          93.0
Marking            92.0
ShotPower          92.0
LongShots          92.0
HeadingAccuracy    91.0
StandingTackle     90.0
FKAccuracy         90.0
BallControl        90.0
Curve              89.0
Crossing           89.0
Interceptions      89.0
SlidingTackle      88.0
Volleys            87.0
Penalties          87.0
Acceleration       86.0
Aggression         85.0
ShortPassing       85.0
dtype: float64
```

```
[32] # гистограммы для этих признаков
     data[quantity_cols.index.tolist()].hist(figsize=(15, 15))
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3f2322d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3df976d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3debed50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3de81410>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3deb4a90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3de75150>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3de2a850>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dde2e10>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dde2e50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dda1610>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dd19250>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dcce8d0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dc84f50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dc45610>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dbfefd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dbbf950>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3dbf8fd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3db39790>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3db70990>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3c3db26fd0>]],
      dtype=object)
```

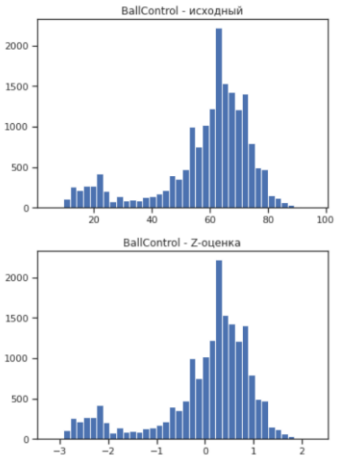


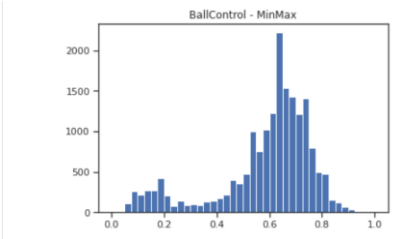
Масштабирование данных на основе Z-оценки и MinMax

[33]
```
# импорт
from sklearn.preprocessing import MinMaxScaler, StandardScaler
```

[35]
```
# масштабируем для BallControl

# Z-оценка
st_s = StandardScaler()
st_s_ball_control = st_s.fit_transform(data[['BallControl']])

# MinMax
mm_s = MinMaxScaler()
mm_s_ball_control = mm_s.fit_transform(data[['BallControl']])

# построение гистограмм
plt.hist(data['BallControl'], 40)
plt.title('BallControl - исходный')
plt.show()

plt.hist(st_s_ball_control, 40)
plt.title('BallControl - Z-оценка')
plt.show()

plt.hist(mm_s_ball_control, 40)
plt.title('BallControl - MinMax')
plt.show()
```

[35]

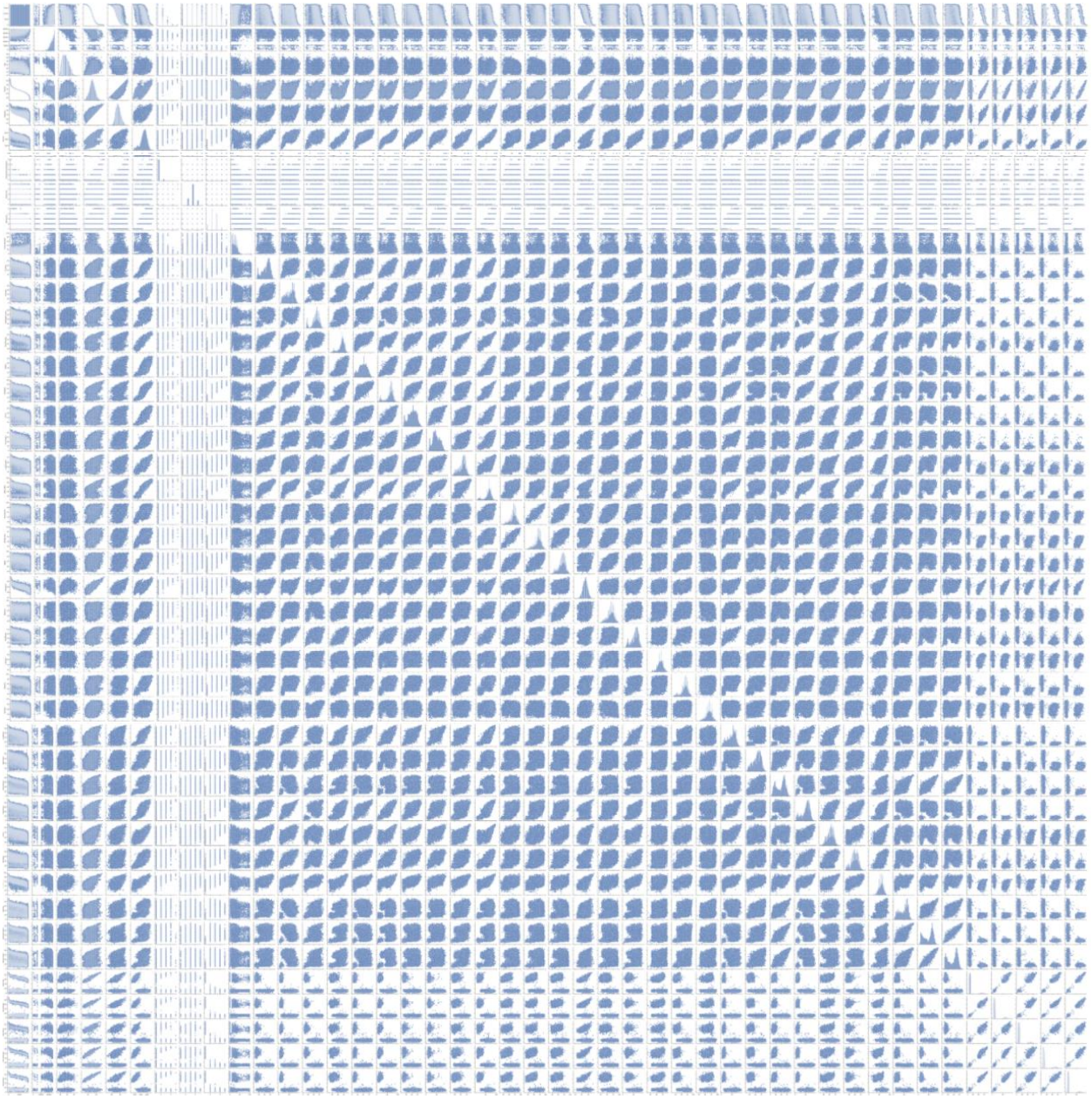




Дополнительное задание: построение "парных диаграмм"

In [20]:
```
sns.pairplot(data)
```

Out[20]:
```
<seaborn.axisgrid.PairGrid at 0x2c602b478b0>
```



Вывод

Масштабирование данных и преобразование категориальных признаков в количественные были проведены с учетом характеристик датасета. Для масштабирования были использованы методы *MinMax* и *на основе Z-оценки*. Для преобразования категориальных признаков в количественные были использованы методы *label encoding* и one *hot encoding*. Были подобраны признаки, которые наглядно продемонстрировали работу этих методов. Для реализации и визуализации были использованы функции из библиотек: *pandas*, *sklearn*, *seaborn*, *matplotlib*.