Курсовая работа

"Исследование факторов, влияющих на успешность завершения обучения студентом"

по дисциплине:

"Обработка и анализ данных"

Выполнила: Саросек Мария

Группа: ИУ5-32М

Описание задачи

Любое учебное заведение заинтересовано в том, чтобы как можно меньшее количество студентов покинуло его до окончания всего периода обучения. Вступительные испытания служат средством, помогающим предсказать, справится ли студент с образовательной программой. Анализ с помощью методов машинного обучения результатов вступительных испытаний - как и любой другой информации о студенте - может помочь предсказать, дойдет ли потенциальный студент до конца курса. Необходимо разработать модель, цель которой состоит в предсказании, будет ли студент отчислен до окончания срока обучения.

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
from sklearn.impute import KNNImputer
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Lasso
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
%matplotlib inline
sns.set(style="ticks")
```

Выгрузим данные для анализа. В них входит информации о дате рождения, родном городе, гендере, id волны, уровне, жизненном статусе и результатах вступительных испытаний каждого студента.

In [2]:
```
data_loaded = pd.read_csv('train.csv', sep=",")
```

In [3]:
```
data_loaded.head()
```

Out[3]:

	id	Birth date	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_01	..
0	129212391	1990-01-18	Пермь	male	3	7.463235	google_ads	work	5	NaN	..
1	566688420	1996-07-31	Волгоград	female	1	8.785714	NaN	NaN	0	NaN	..
2	242300495	1992-06-27	Новосибирск	female	2	6.775000	other	work	0	NaN	..
3	715424753	1996-06-15	Тула	female	1	8.146104	NaN	NaN	5	0.0	..
4	382884118	1968-07-18	Голицыно	male	3	4.272727	other	other	0	0.0	..

5 rows × 36 columns

In [4]:
```
data_loaded = data_loaded.drop(['Contract termination date', 'Birth date'], axis=1)
data_loaded.head()
```

Out[4]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_01	day_02
0	129212391	Пермь	male	3	7.463235	google_ads	work	5	NaN	0.0
1	566688420	Волгоград	female	1	8.785714	NaN	NaN	0	NaN	10.0
2	242300495	Новосибирск	female	2	6.775000	other	work	0	NaN	20.0
3	715424753	Тула	female	1	8.146104	NaN	NaN	5	0.0	35.0
4	382884118	Голицыно	male	3	4.272727	other	other	0	0.0	NaN

5 rows × 34 columns

In [5]:
```
data_loaded.shape
```

Out[5]:
```
(1060, 34)
```

In [6]:
```
def check_null_values(data):
    return list(zip([i for i in data.columns], zip(
    #типы колонок
    [str(i) for i in data.dtypes],
    #проверка, есть ли пропущенные значения
    [i for i in data.isnull().sum()])))
```

In [7]:
```
data_features = check_null_values(data_loaded)
data_features
```

Out[7]:
```
[('id', ('int64', 0)),
 ('Native city', ('object', 13)),
 ('Gender', ('object', 0)),
 ('Wave id', ('int64', 0)),
 ('Level', ('float64', 0)),
```

```
('Heard about school from', ('object', 407)),
('Life status', ('object', 405)),
('day_00', ('int64', 0)),
('day_01', ('float64', 414)),
('day_02', ('float64', 16)),
('day_03', ('float64', 15)),
('day_04', ('float64', 23)),
('day_05', ('float64', 4)),
('day_06', ('float64', 12)),
('day_07', ('float64', 14)),
('day_08', ('float64', 15)),
('day_09', ('float64', 13)),
('day_10', ('float64', 32)),
('day_11', ('float64', 13)),
('day_12', ('float64', 10)),
('day_13', ('float64', 31)),
('evalexpr', ('float64', 57)),
('match_n_match', ('float64', 109)),
('bsq', ('float64', 33)),
('rush_00', ('float64', 64)),
('rush_01', ('float64', 116)),
('rush_02', ('float64', 158)),
('exam_00', ('float64', 13)),
('exam_01', ('float64', 19)),
('exam_02', ('float64', 13)),
('exam_final', ('float64', 4)),
('Memory entrance game', ('int64', 0)),
('Logic entrance game', ('int64', 0)),
('contract_status', ('int64', 0))]
```

Устранение пропусков в данных

Заполнение значений для одного признака

In [8]:
```
def impute_column(dataset, column, strategy_param, fill_value_param=None):
    """
    Заполнение пропусков в одном признаке
    """
    temp_data = dataset[[column]].values
    size = temp_data.shape[0]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imputer = SimpleImputer(strategy=strategy_param,
                            fill_value=fill_value_param)
    all_data = imputer.fit_transform(temp_data)

    missed_data = temp_data[mask_missing_values_only]
    filled_data = all_data[mask_missing_values_only]

    return all_data.reshape((size,)), filled_data, missed_data
```

In [9]:
```
data_na_replaced = data_loaded.copy()
for column in data_loaded.columns:
    data_na_replaced[column], filled_data, missed_data = impute_column(data_loaded,
```

In [10]:
```
check_null_values(data_na_replaced)
```

Out[10]:
```
[('id', ('int64', 0)),
 ('Native city', ('object', 0)),
 ('Gender', ('object', 0)),
 ('Wave id', ('int64', 0)),
 ('Level', ('float64', 0)),
```

```
('Heard about school from', ('object', 0)),
('Life status', ('object', 0)),
('day_00', ('int64', 0)),
('day_01', ('float64', 0)),
('day_02', ('float64', 0)),
('day_03', ('float64', 0)),
('day_04', ('float64', 0)),
('day_05', ('float64', 0)),
('day_06', ('float64', 0)),
('day_07', ('float64', 0)),
('day_08', ('float64', 0)),
('day_09', ('float64', 0)),
('day_10', ('float64', 0)),
('day_11', ('float64', 0)),
('day_12', ('float64', 0)),
('day_13', ('float64', 0)),
('evalexpr', ('float64', 0)),
('match_n_match', ('float64', 0)),
('bsq', ('float64', 0)),
('rush_00', ('float64', 0)),
('rush_01', ('float64', 0)),
('rush_02', ('float64', 0)),
('exam_00', ('float64', 0)),
('exam_01', ('float64', 0)),
('exam_02', ('float64', 0)),
('exam_final', ('float64', 0)),
('Memory entrance game', ('int64', 0)),
('Logic entrance game', ('int64', 0)),
('contract_status', ('int64', 0))]
```

In [11]:
```
data_na_replaced
```

Out[11]:

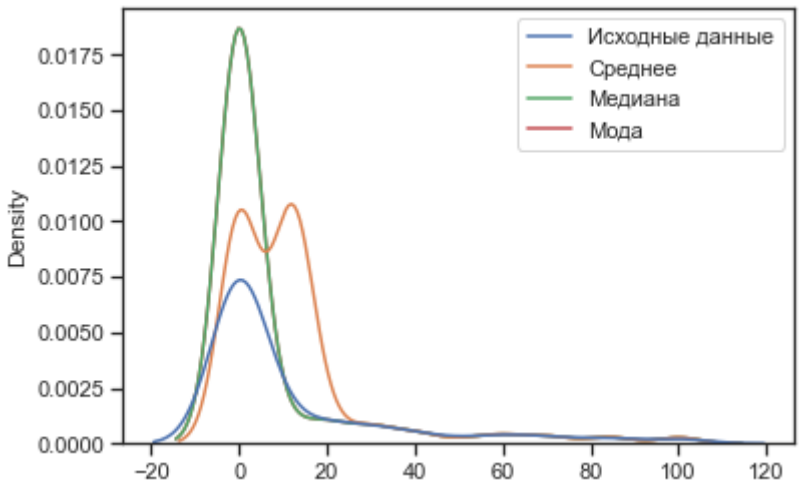
	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_01
0	129212391	Пермь	male	3	7.463235	google_ads	work	5	0.0
1	566688420	Волгоград	female	1	8.785714	other	study	0	0.0
2	242300495	Новосибирск	female	2	6.775000	other	work	0	0.0
3	715424753	Тула	female	1	8.146104	other	study	5	0.0
4	382884118	Голицыно	male	3	4.272727	other	other	0	0.0
...	...	...	...	...	...	...	...	...	...
1055	1913448776	Тверь	male	3	4.149733	vk_ads	study	10	0.0
1056	765964407	Р-Н ВОСКРЕСЕНСКИЙ, НП СЛОБОДКА АЛЕШИНО	male	3	4.320856	from_friends	study	5	0.0
1057	1546808331	Москва	male	2	6.675000	from_friends	study	30	30.0
1058	1918072182	Москва	male	3	6.483333	other	study	50	0.0
1059	1162137757	Арамиль	male	1	4.000000	other	study	5	0.0

1060 rows × 34 columns

In [12]:
```
def research_impute_numeric_column(dataset, num_column, const_value=None):
    strategy_params = ['mean', 'median', 'most_frequent', 'constant']
```

```
    strategy_params_names = ['Среднее', 'Медиана', 'Мода']
    strategy_params_names.append('Константа = ' + str(const_value))

    original_temp_data = dataset[[num_column]].values
    size = original_temp_data.shape[0]
    original_data = original_temp_data.reshape((size,))

    new_df = pd.DataFrame({'Исходные данные':original_data})

    for i in range(len(strategy_params)):
        strategy = strategy_params[i]
        col_name = strategy_params_names[i]
        if (strategy!='constant') or (strategy == 'constant' and const_value!=None):
            if strategy == 'constant':
                temp_data, _, _ = impute_column(dataset, num_column, strategy, fill_
            else:
                temp_data, _, _ = impute_column(dataset, num_column, strategy)
            new_df[col_name] = temp_data

    sns.kdeplot(data=new_df)
```

In [13]:
```
#Сравнение заполнения различными показателями распределения
research_impute_numeric_column(data_loaded, 'day_01')
```



Кодирование категориальных признаков

1. Label Encoding

In [14]:
```
from sklearn.preprocessing import LabelEncoder
```

In [15]:
```
data_encoded = data_na_replaced.copy()
data_encoded
```

Out[15]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_01
0	129212391	Пермь	male	3	7.463235	google_ads	work	5	0.0
1	566688420	Волгоград	female	1	8.785714	other	study	0	0.0
2	242300495	Новосибирск	female	2	6.775000	other	work	0	0.0
3	715424753	Тула	female	1	8.146104	other	study	5	0.0

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_01
4	382884118	Голицыно	male	3	4.272727	other	other	0	0.0
...	...	...	...	...	...	...	...	...	...
1055	1913448776	Тверь	male	3	4.149733	vk_ads	study	10	0.0
1056	765964407	Р-Н ВОСКРЕСЕНСКИЙ, НП СЛОБОДКА АЛЕШИНО	male	3	4.320856	from_friends	study	5	0.0
1057	1546808331	Москва	male	2	6.675000	from_friends	study	30	30.0
1058	1918072182	Москва	male	3	6.483333	other	study	50	0.0
1059	1162137757	Арамиль	male	1	4.000000	other	study	5	0.0

1060 rows × 34 columns

In [16]:
```
#le = LabelEncoder()
category_columns = ['Native city', 'Gender', 'Heard about school from', 'Life status
for col in category_columns:
    le = LabelEncoder()
    data_encoded[col] = le.fit_transform(data_encoded[col])
data_encoded
```

Out[16]:

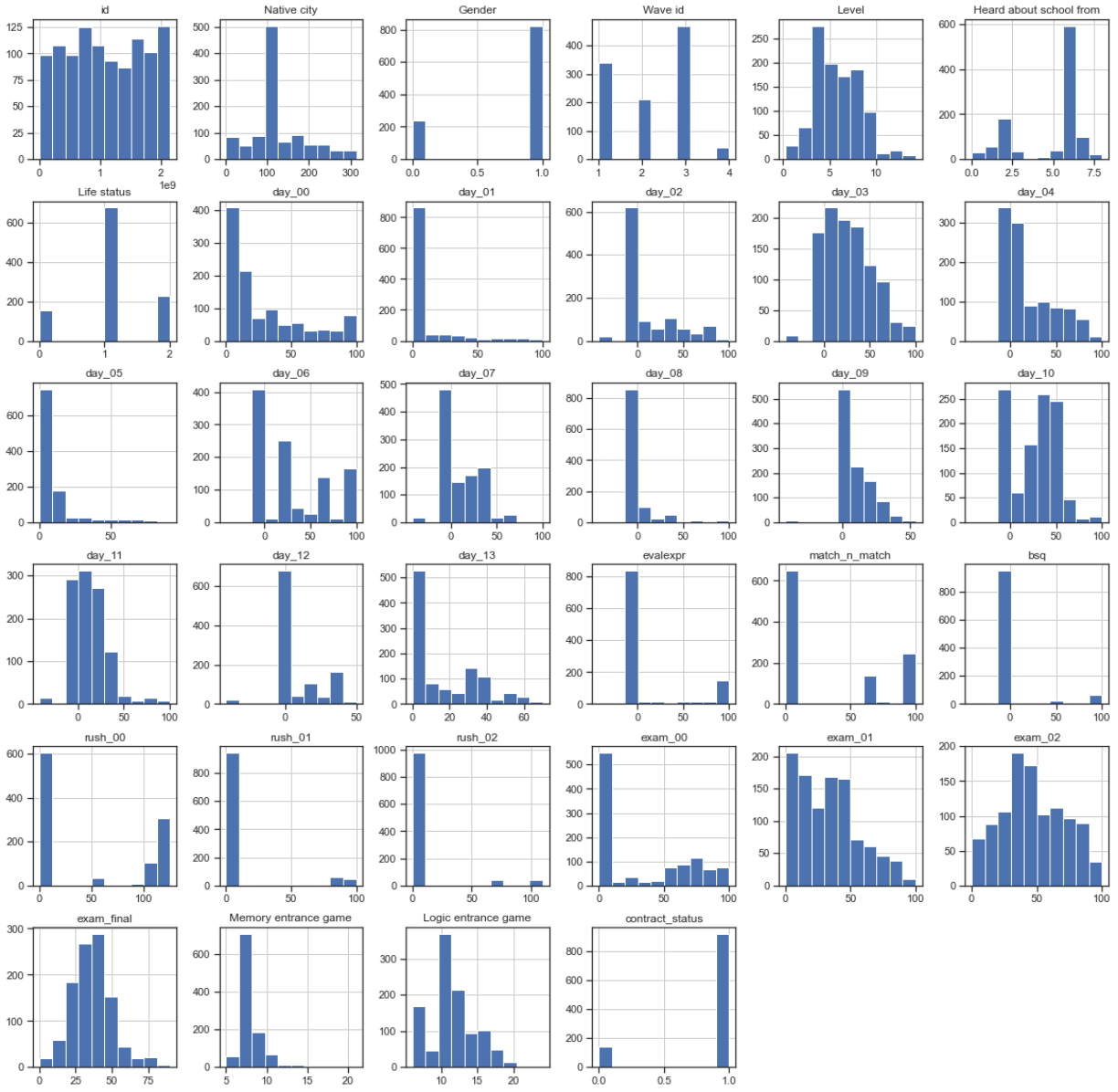
	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_01	day_02	...	ru
0	129212391	159	1	3	7.463235	5	2	5	0.0	0.0	...	
1	566688420	41	0	1	8.785714	6	1	0	0.0	10.0	...	
2	242300495	143	0	2	6.775000	6	2	0	0.0	20.0	...	
3	715424753	214	0	1	8.146104	6	1	5	0.0	35.0	...	
4	382884118	48	1	3	4.272727	6	0	0	0.0	0.0	...	
...	...	...	...	...	...	...	...	...	...	...	...	...
1055	1913448776	208	1	3	4.149733	7	1	10	0.0	0.0	...	
1056	765964407	168	1	3	4.320856	2	1	5	0.0	10.0	...	
1057	1546808331	120	1	2	6.675000	2	1	30	30.0	10.0	...	
1058	1918072182	119	1	3	6.483333	6	1	50	0.0	80.0	...	
1059	1162137757	21	1	1	4.000000	6	1	5	0.0	80.0	...	

1060 rows × 34 columns

Нормализация числовых признаков

In [17]:
```
import scipy.stats as stats
```

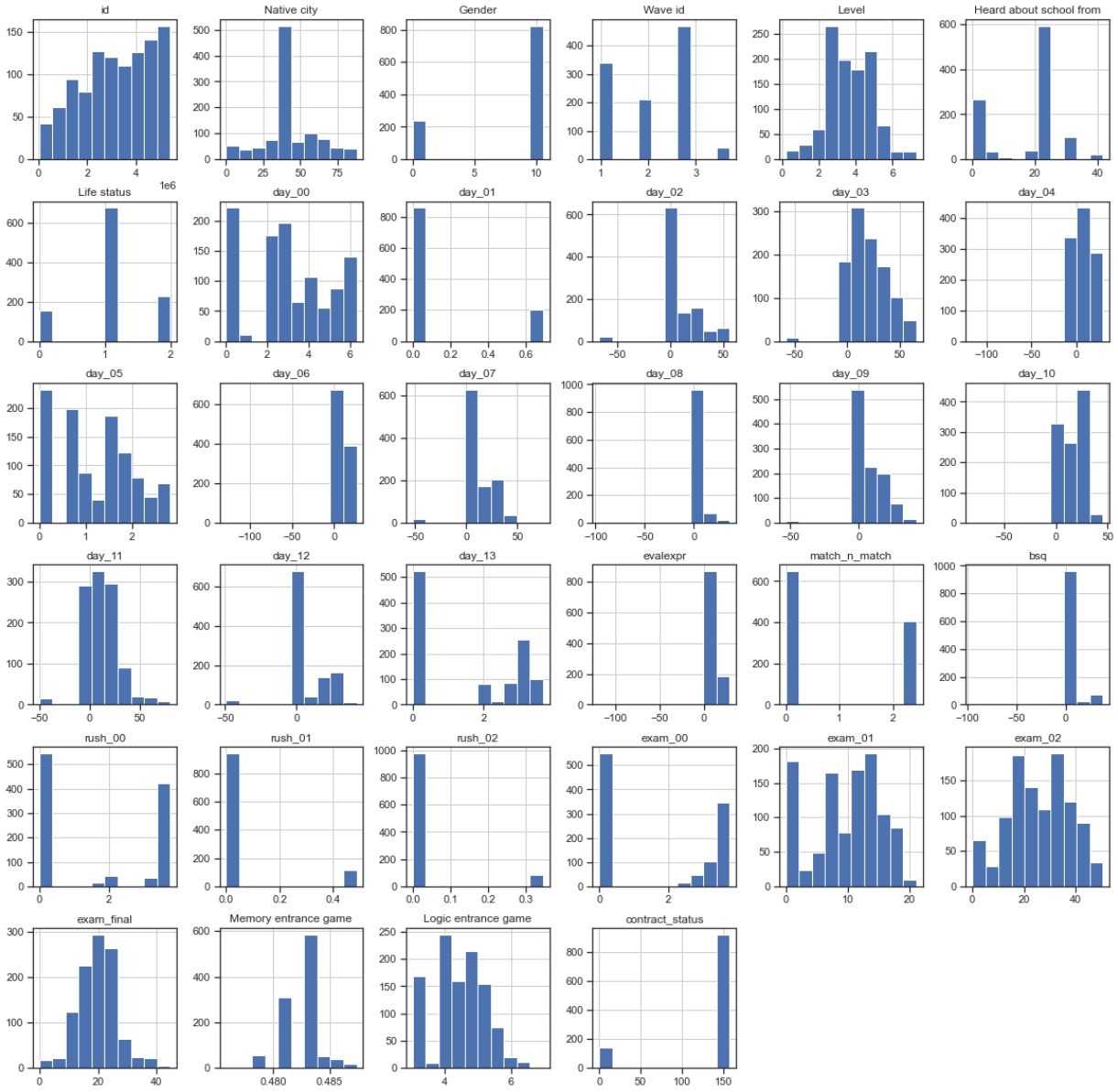
In [18]:
```
data_encoded.hist(figsize=(20, 20))
plt.show()
```



In [19]:
```
def diagnostic_plots(df, variable):
    plt.figure(figsize=(15,6))
    # гистограмма
    plt.subplot(1, 2, 1)
    df[variable].hist(bins=30)
    ## Q-Q plot
    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    plt.show()
```

In [20]:
```
data_normalized = data_encoded.copy()
# Необходимо преобразовать данные к действительному типу
for col in data_normalized.columns:
    data_normalized[col] = data_normalized[col].astype('float')
    data_normalized[col], param = stats.yeojohnson(data_normalized[col])
```

In [21]:
```
data_normalized.hist(figsize=(20,20))
plt.show()
```



In [22]:
```
data_normalized
```

Out[22]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
0	7.440813e+05	53.142349	10.509352	2.817426	4.535441	18.008774	1.988700	2.015112	-0.000(
1	2.110445e+06	19.312977	0.000000	0.971741	5.125305	24.773476	0.996627	0.000000	-0.000(
2	1.159231e+06	49.136182	0.000000	1.906271	4.215184	24.773476	1.988700	0.000000	-0.000(
3	2.487437e+06	66.153741	0.000000	0.971741	4.843920	24.773476	0.996627	2.015112	-0.000(
4	1.600657e+06	21.757480	10.509352	2.817426	2.950189	24.773476	0.000000	0.000000	-0.000(
...	...	...	...	...	...	...	...	...	
1055	4.977851e+06	64.783291	10.509352	2.817426	2.882900	32.602702	0.996627	2.808948	-0.000(
1056	2.610098e+06	55.346994	10.509352	2.817426	2.976363	4.063310	0.996627	2.015112	-0.000(
1057	4.284471e+06	43.153499	10.509352	1.906271	4.167804	4.063310	0.996627	4.317815	0.686(
1058	4.986330e+06	42.886603	10.509352	2.817426	4.076357	24.773476	0.996627	5.117555	-0.000(
1059	3.502103e+06	11.573992	10.509352	0.971741	2.800185	24.773476	0.996627	2.015112	-0.000(

1060 rows × 34 columns

Масштабирование признаков

На основе Z-оценки

In [23]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import RobustScaler
from sklearn.preprocessing import MaxAbsScaler
```

In [24]:
```
data_scaled = data_normalized.copy()
data_scaled.describe()
```

Out[24]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status
count	1.060000e+03	1060.000000	1060.000000	1060.000000	1060.000000	1060.000000	1060.000000
mean	3.192016e+06	44.400394	8.149705	2.078666	3.709458	19.450846	1.064283
std	1.443290e+06	17.464263	4.387321	0.860580	1.184935	10.712010	0.593881
min	4.089877e+04	0.000000	0.000000	0.971741	0.213095	0.000000	0.000000
25%	2.130037e+06	42.283774	10.509352	0.971741	2.897578	4.063310	0.996627
50%	3.280196e+06	42.886603	10.509352	1.906271	3.577479	24.773476	0.996627
75%	4.449189e+06	52.957067	10.509352	2.817426	4.576638	24.773476	0.996627
max	5.393494e+06	87.677070	10.509352	3.711586	7.280049	41.499496	1.988700

8 rows × 34 columns

In [25]:
```
X_ALL = data_scaled.drop('contract_status', axis=1)
X_ALL
```

Out[25]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
0	7.440813e+05	53.142349	10.509352	2.817426	4.535441	18.008774	1.988700	2.015112	-0.000(
1	2.110445e+06	19.312977	0.000000	0.971741	5.125305	24.773476	0.996627	0.000000	-0.000(
2	1.159231e+06	49.136182	0.000000	1.906271	4.215184	24.773476	1.988700	0.000000	-0.000(
3	2.487437e+06	66.153741	0.000000	0.971741	4.843920	24.773476	0.996627	2.015112	-0.000(
4	1.600657e+06	21.757480	10.509352	2.817426	2.950189	24.773476	0.000000	0.000000	-0.000(
...	...	...	...	...	...	...	...	...	
1055	4.977851e+06	64.783291	10.509352	2.817426	2.882900	32.602702	0.996627	2.808948	-0.000(
1056	2.610098e+06	55.346994	10.509352	2.817426	2.976363	4.063310	0.996627	2.015112	-0.000(

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
1057	4.284471e+06	43.153499	10.509352	1.906271	4.167804	4.063310	0.996627	4.317815	0.686(
1058	4.986330e+06	42.886603	10.509352	2.817426	4.076357	24.773476	0.996627	5.117555	-0.000(
1059	3.502103e+06	11.573992	10.509352	0.971741	2.800185	24.773476	0.996627	2.015112	-0.000(

1060 rows × 33 columns

In [26]:
```
# Функция восстановления датафрейма на основе масштабированных данных
def arr_to_df(arr_scaled):
    res = pd.DataFrame(arr_scaled, columns=X_ALL.columns)
    return res
```

In [27]:
```
# Разделим выборку на обучающую и тестовую
X_train, X_test, y_train, y_test = train_test_split(X_ALL, data_scaled['contract_sta
                                                    test_size=0.2,
                                                    random_state=1)
# Преобразуем массивы в DataFrame
X_train_df = arr_to_df(X_train)
X_test_df = arr_to_df(X_test)

X_train_df.shape, X_test_df.shape
```

Out[27]: ((848, 33), (212, 33))

In [28]:
```
# Обучаем StandardScaler на всей выборке и масштабируем
cs11 = StandardScaler()
data_cs11_scaled_temp = cs11.fit_transform(X_ALL)
# формируем DataFrame на основе массива
data_cs11_scaled =  arr_to_df(data_cs11_scaled_temp)
data_cs11_scaled
```

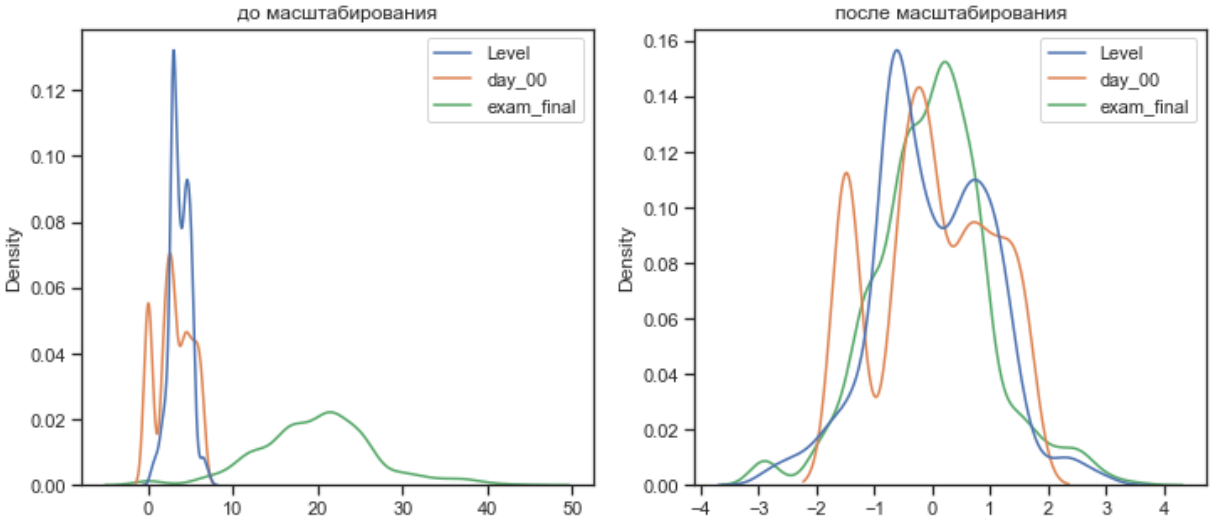
Out[28]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_0
0	-1.696880	0.500799	0.538087	0.858850	0.697399	-0.134685	1.557304	-0.493445	-0.48663
1	-0.749732	-1.437178	-1.858435	-1.286862	1.195437	0.497119	-0.113976	-1.484201	-0.48663
2	-1.409103	0.271298	-1.858435	-0.200419	0.426997	0.497119	1.557304	-1.484201	-0.48663
3	-0.488406	1.246180	-1.858435	-1.286862	0.957856	0.497119	-0.113976	-0.493445	-0.48663
4	-1.103112	-1.297140	0.538087	0.858850	-0.641071	0.497119	-1.792927	-1.484201	-0.48663
...	...	...	...	...	...	...	...	...	
1055	1.237920	1.167671	0.538087	0.858850	-0.697885	1.228347	-0.113976	-0.103145	-0.48663
1056	-0.403379	0.627096	0.538087	0.858850	-0.618971	-1.437153	-0.113976	-0.493445	-0.48663
1057	0.757277	-0.071431	0.538087	-0.200419	0.386993	-1.437153	-0.113976	0.638709	2.06706
1058	1.243798	-0.086720	0.538087	0.858850	0.309782	0.497119	-0.113976	1.031912	-0.48663
1059	0.214949	-1.880520	0.538087	-1.286862	-0.767724	0.497119	-0.113976	-0.493445	-0.48663

1060 rows × 33 columns

In [29]:
```
# Построение плотности распределения
def draw_kde(col_list, df1, df2, label1, label2):
    fig, (ax1, ax2) = plt.subplots(
        ncols=2, figsize=(12, 5))
    # первый график
    ax1.set_title(label1)
    sns.kdeplot(data=df1[col_list], ax=ax1)
    # второй график
    ax2.set_title(label2)
    sns.kdeplot(data=df2[col_list], ax=ax2)
    plt.show()
```

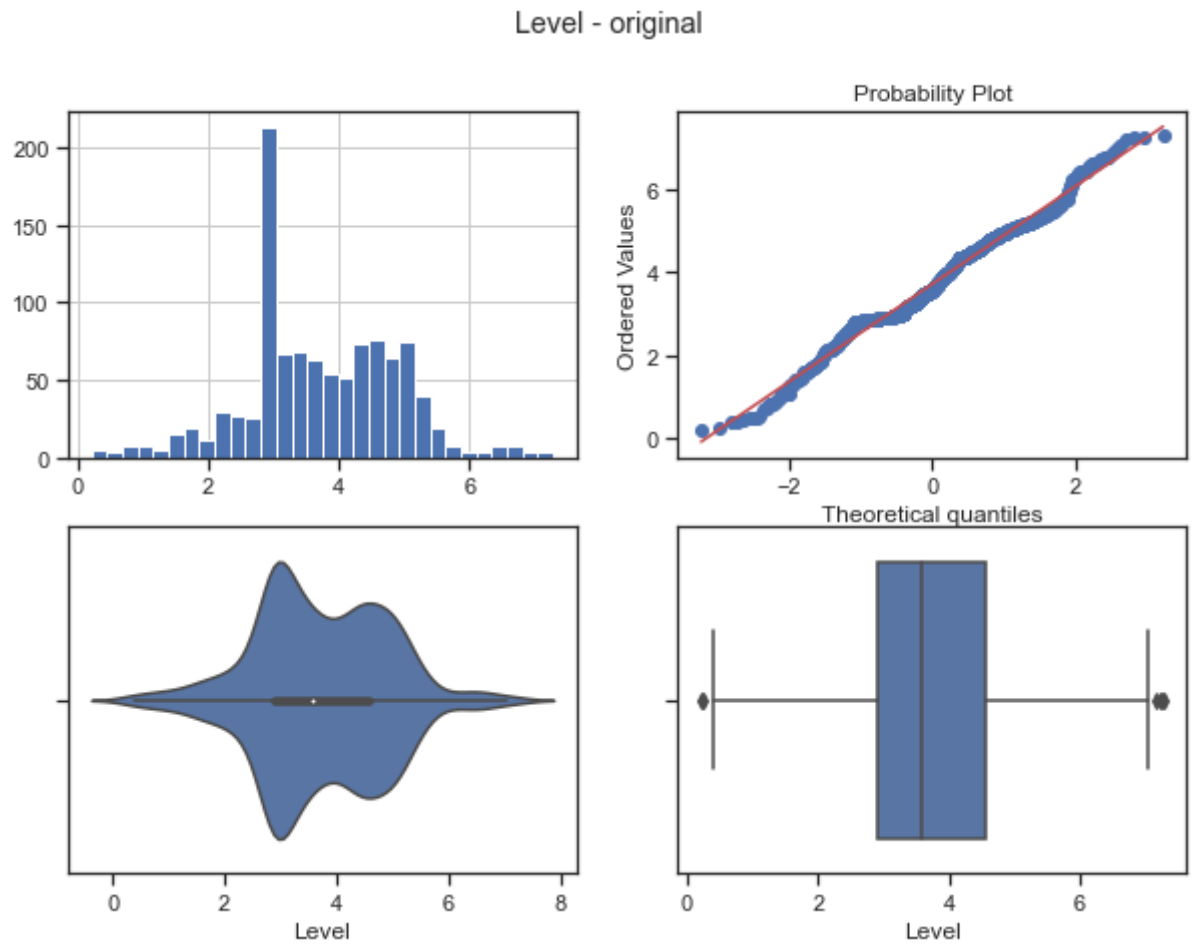
In [30]:
```
draw_kde(['Level', 'day_00', 'exam_final'], data_scaled, data_cs11_scaled, 'до масшт
```



Обработка выбросов

In [31]:
```
def diagnostic_plots(df, variable, title):
    fig, ax = plt.subplots(figsize=(10,7))
    # гистограмма
    plt.subplot(2, 2, 1)
    df[variable].hist(bins=30)
    ## Q-Q plot
    plt.subplot(2, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    # ящик с усами
    plt.subplot(2, 2, 3)
    sns.violinplot(x=df[variable])
    # ящик с усами
    plt.subplot(2, 2, 4)
    sns.boxplot(x=df[variable])
    fig.suptitle(title)
    plt.show()
```

In [32]:
```
diagnostic_plots(data_scaled, 'Level', 'Level - original')
```



Обнаружение выбросов

In [33]:
```
# Тип вычисления верхней и нижней границы выбросов
from enum import Enum
class OutlierBoundaryType(Enum):
    #SIGMA = 1
    QUANTILE = 2
    #IRQ = 3
```

In [34]:
```
# Функция вычисления верхней и нижней границы выбросов
def get_outlier_boundaries(df, col, outlier_boundary_type: OutlierBoundaryType):
    #if outlier_boundary_type == OutlierBoundaryType.SIGMA:
    #   K1 = 3
    #    lower_boundary = df[col].mean() - (K1 * df[col].std())
    #    upper_boundary = df[col].mean() + (K1 * df[col].std())

    #elif outlier_boundary_type == OutlierBoundaryType.QUANTILE:
    lower_boundary = df[col].quantile(0.05)
    upper_boundary = df[col].quantile(0.95)

    #elif outlier_boundary_type == OutlierBoundaryType.IRQ:
    #    K2 = 1.5
    #    IQR = df[col].quantile(0.75) - df[col].quantile(0.25)
    #    lower_boundary = df[col].quantile(0.25) - (K2 * IQR)
    #    upper_boundary = df[col].quantile(0.75) + (K2 * IQR)

    #else:
     #   raise NameError('Unknown Outlier Boundary Type')

    return lower_boundary, upper_boundary
```

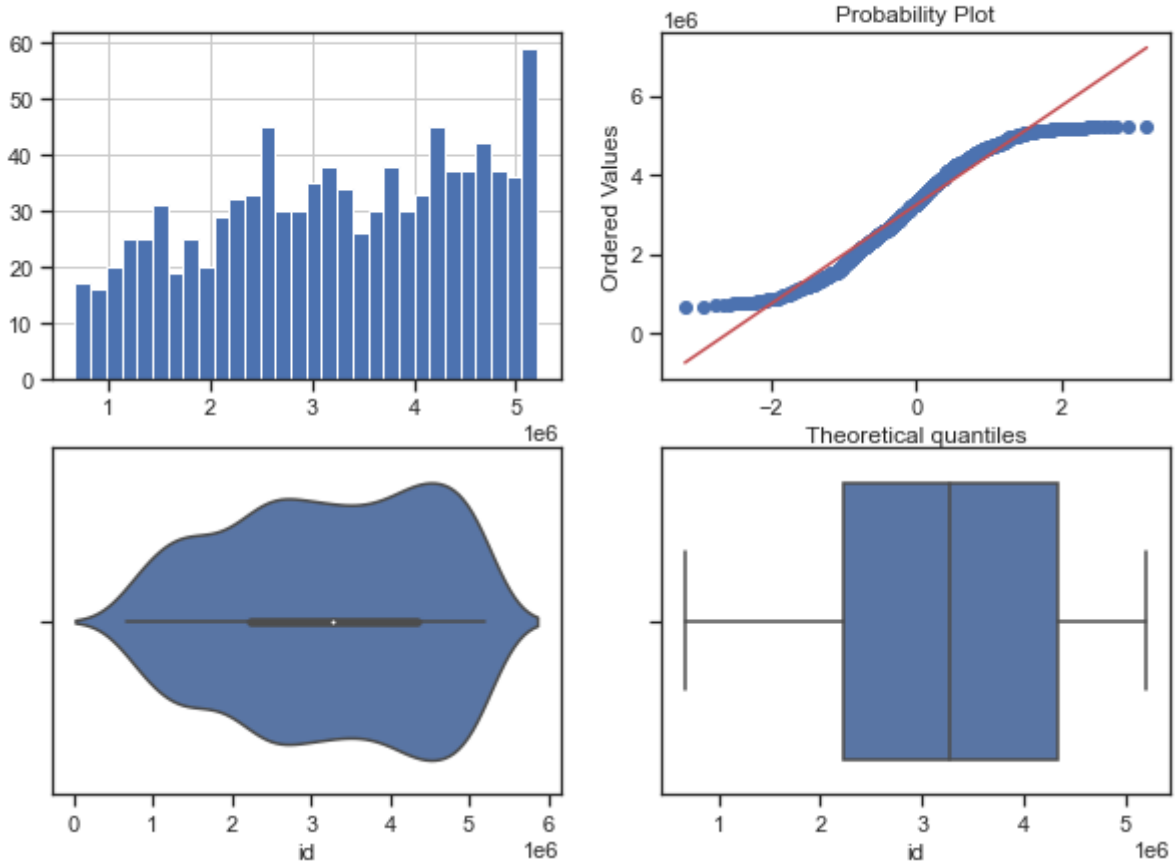
In [35]:
```
outliers_treatment_cols = ['id', 'Level', 'day_00', 'day_01', 'day_02',
```

```
        'day_03', 'day_04', 'day_05', 'day_06', 'day_07', 'day_08', 'day_09',
        'day_10', 'day_11', 'day_12', 'day_13', 'evalexpr', 'match_n_match',
        'bsq', 'rush_00', 'rush_01', 'rush_02', 'exam_00', 'exam_01', 'exam_02',
        'exam_final', 'Memory entrance game', 'Logic entrance game']
outliers_treatment_cols
```

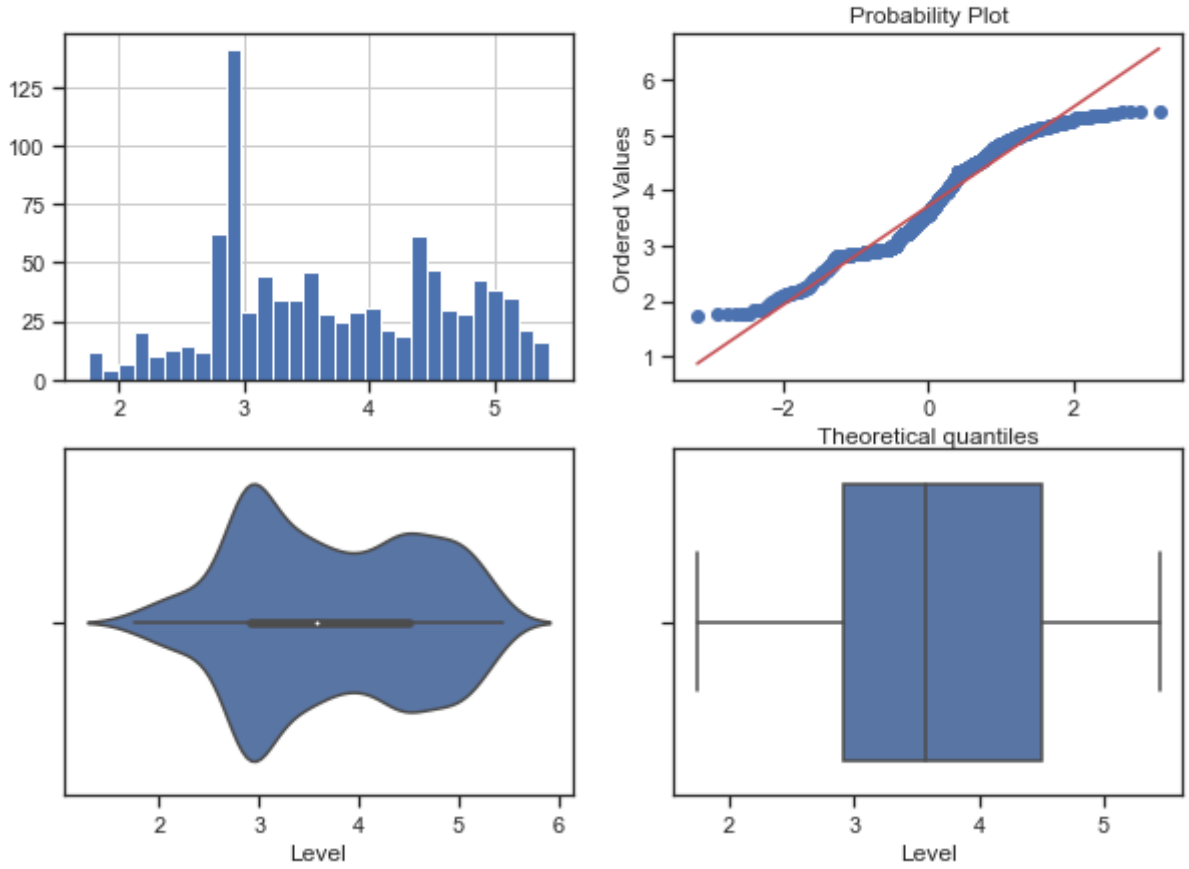
Out[35]:
```
['id',
 'Level',
 'day_00',
 'day_01',
 'day_02',
 'day_03',
 'day_04',
 'day_05',
 'day_06',
 'day_07',
 'day_08',
 'day_09',
 'day_10',
 'day_11',
 'day_12',
 'day_13',
 'evalexpr',
 'match_n_match',
 'bsq',
 'rush_00',
 'rush_01',
 'rush_02',
 'exam_00',
 'exam_01',
 'exam_02',
 'exam_final',
 'Memory entrance game',
 'Logic entrance game']
```

In [36]:
```
data_outliers_deleted = data_scaled.copy()
for col in outliers_treatment_cols:
    for obt in OutlierBoundaryType:
        # Вычисление верхней и нижней границы
        lower_boundary, upper_boundary = get_outlier_boundaries(data_scaled, col, ob
        # Флаги для удаления выбросов
        outliers_temp = np.where(data_scaled[col] > upper_boundary, True,
                                 np.where(data_scaled[col] < lower_boundary, True, F
        # Удаление данных на основе флага
        data_trimmed = data_scaled.loc[~(outliers_temp), ]
        title = 'Поле-{}, метод-{}, строк-{}'.format(col, obt, data_trimmed.shape[0]
        diagnostic_plots(data_trimmed, col, title)
```

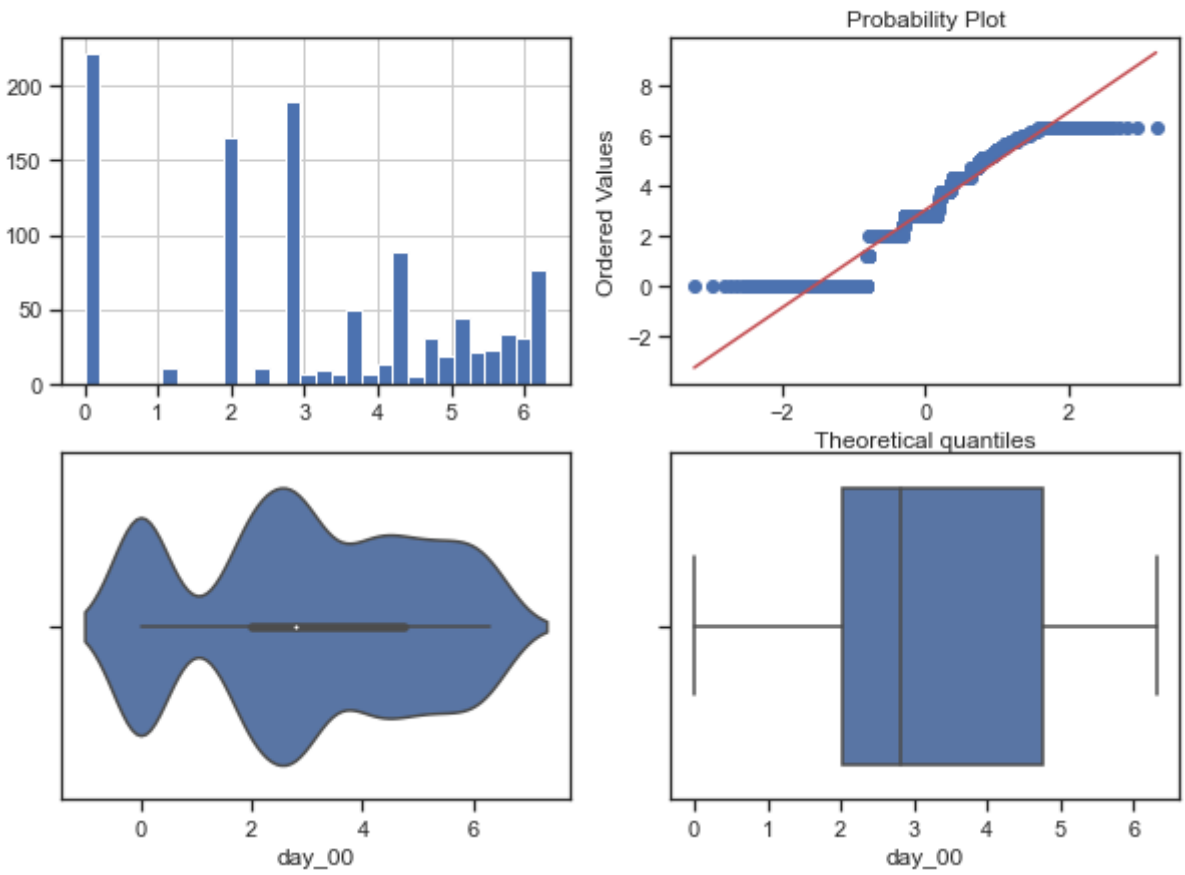
Поле-id, метод-OutlierBoundaryType.QUANTILE, строк-954



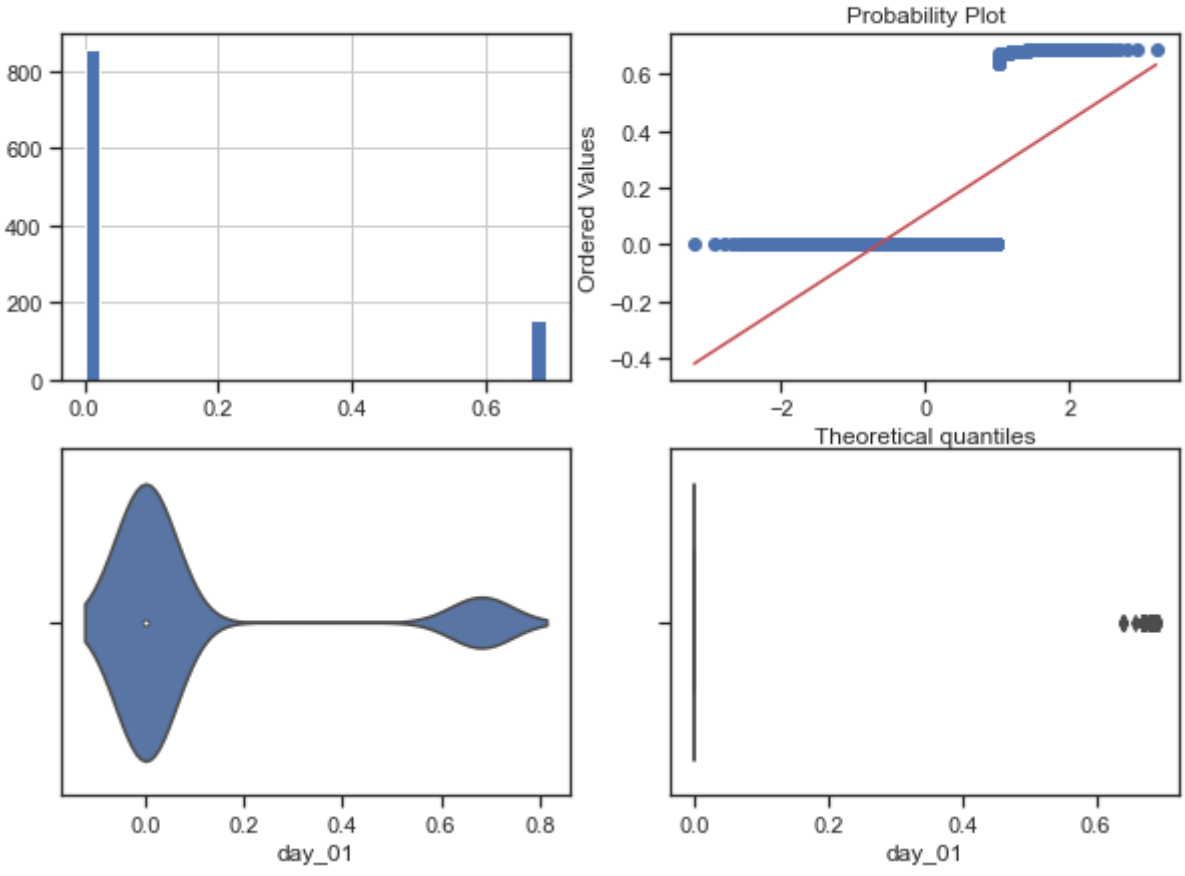
Поле-Level, метод-OutlierBoundaryType.QUANTILE, строк-954



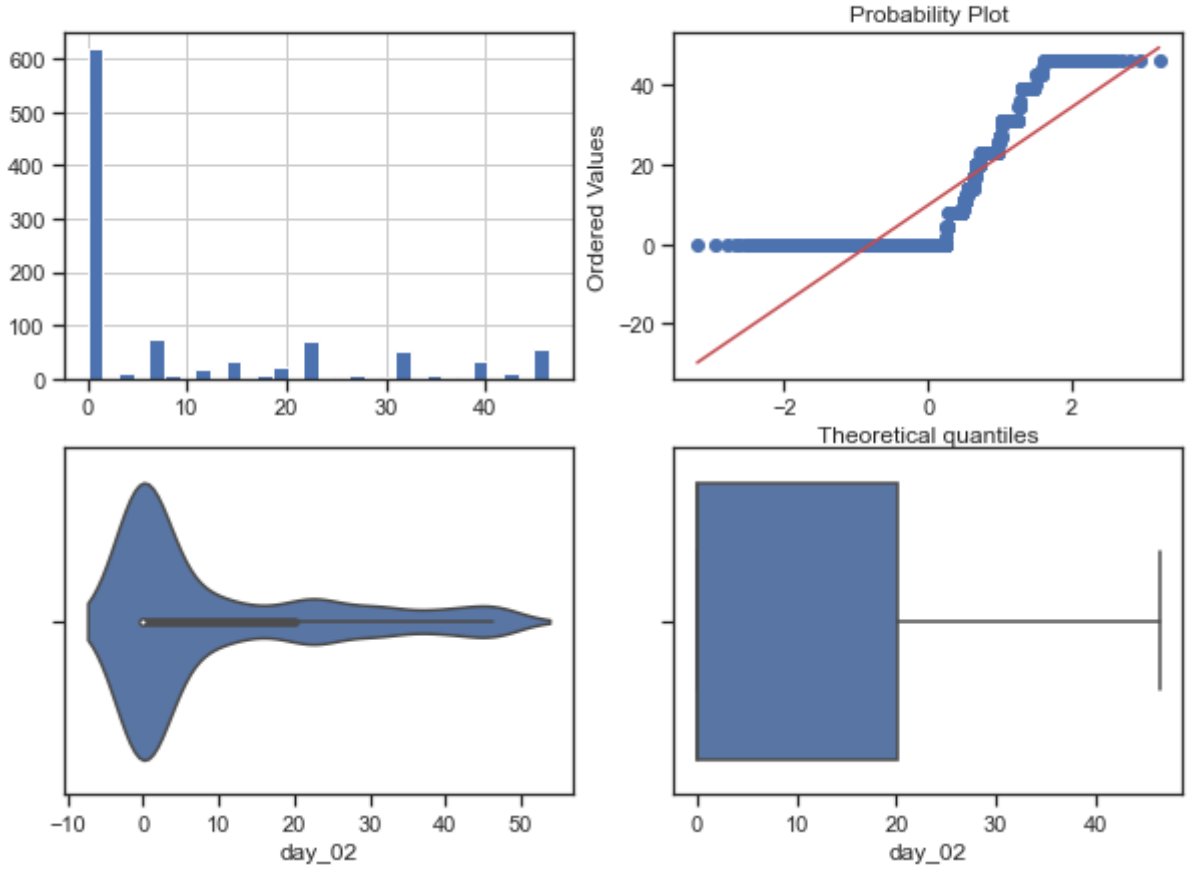
Поле-day_00, метод-OutlierBoundaryType.QUANTILE, строк-1060



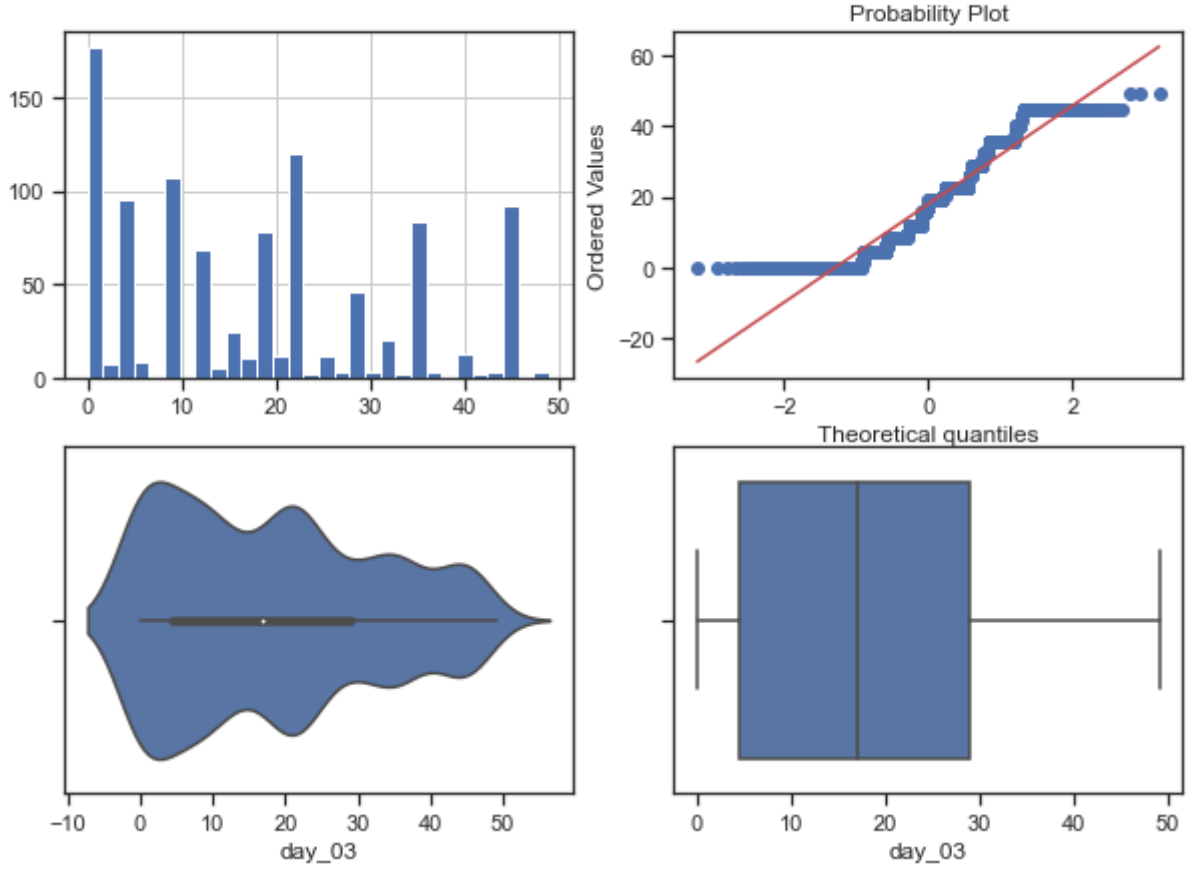
Поле-day_01, метод-OutlierBoundaryType.QUANTILE, строк-1016



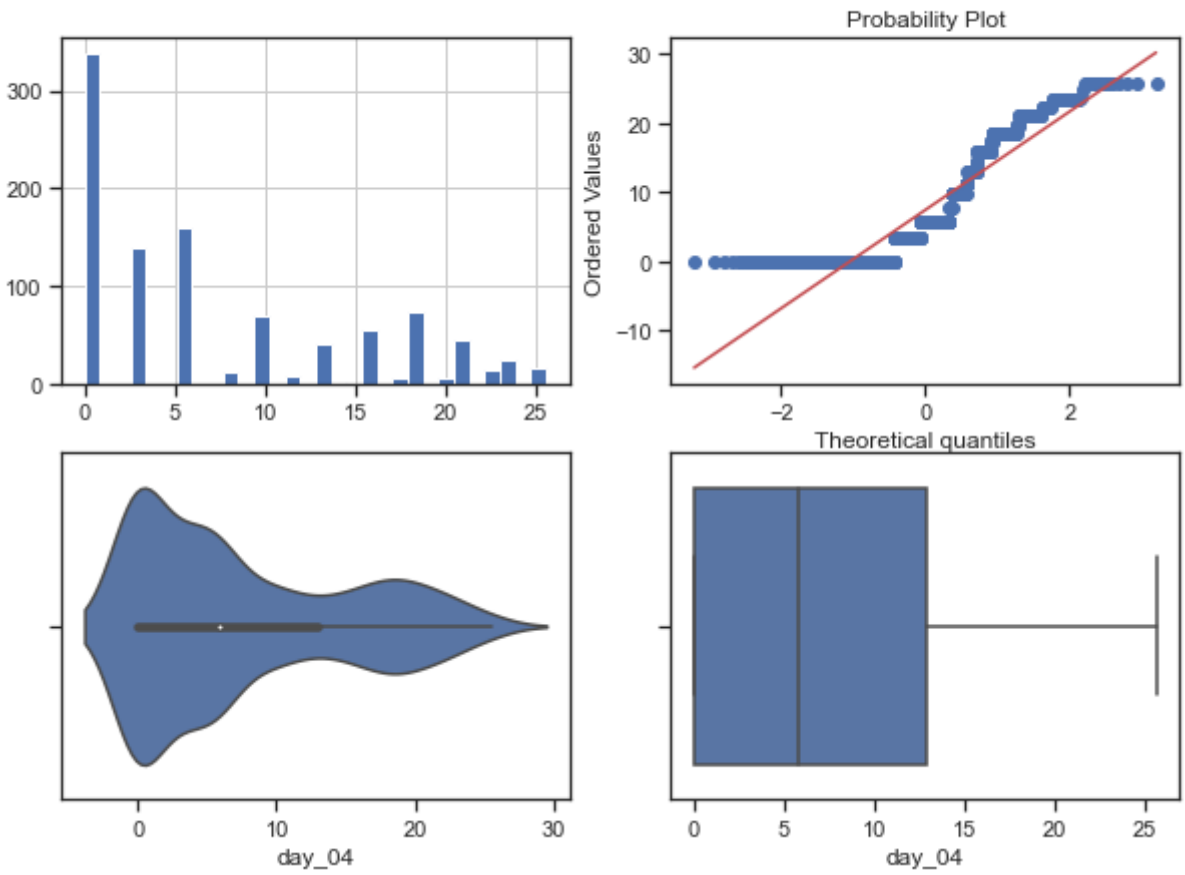
Поле-day_02, метод-OutlierBoundaryType.QUANTILE, строк-1034



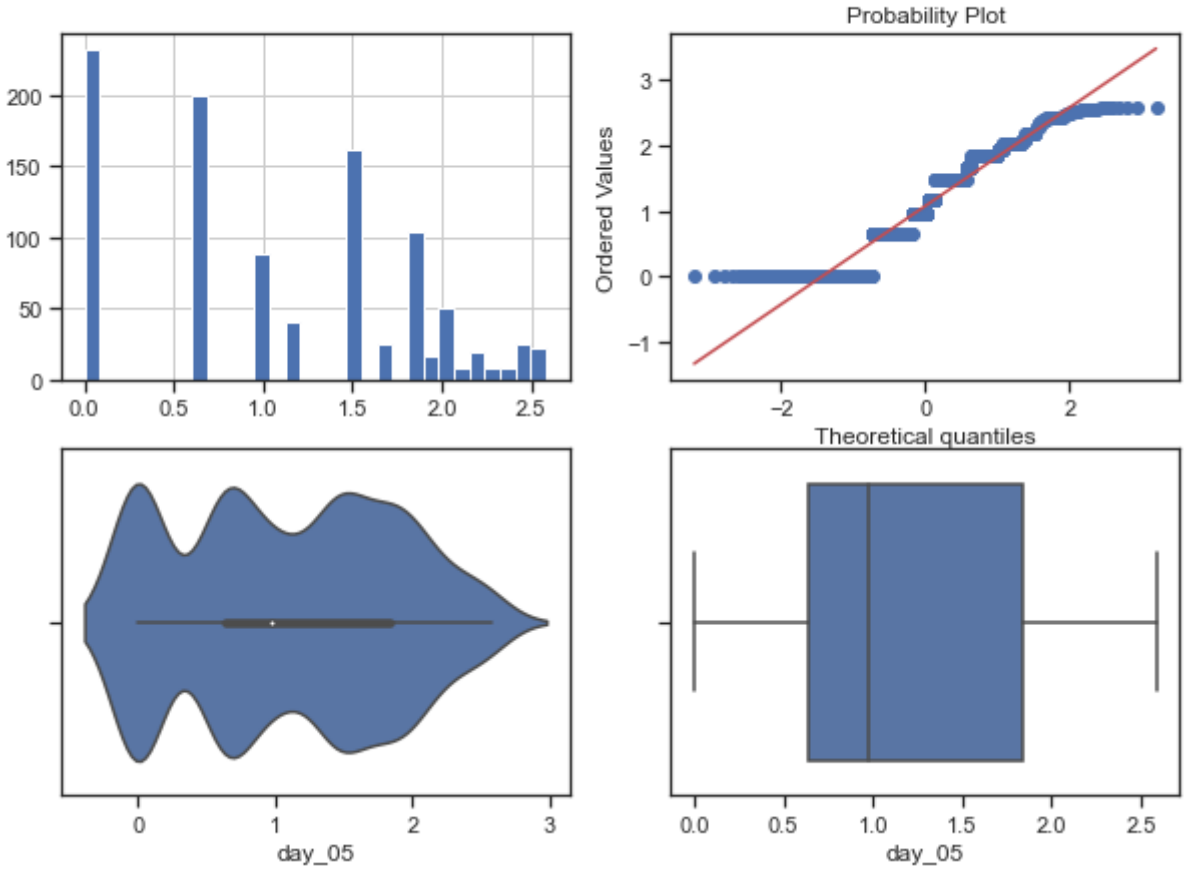
Поле-day_03, метод-OutlierBoundaryType.QUANTILE, строк-1001



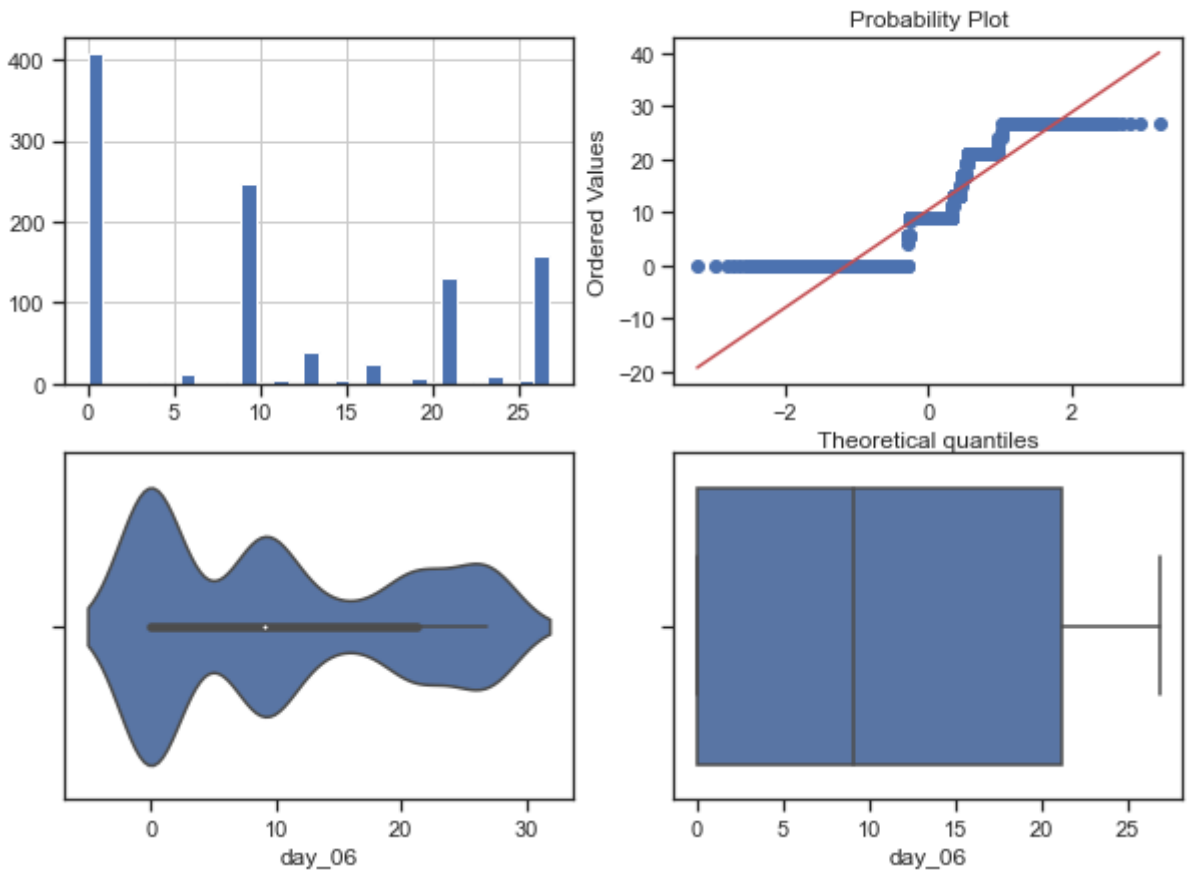
Поле-day_04, метод-OutlierBoundaryType.QUANTILE, строк-1009



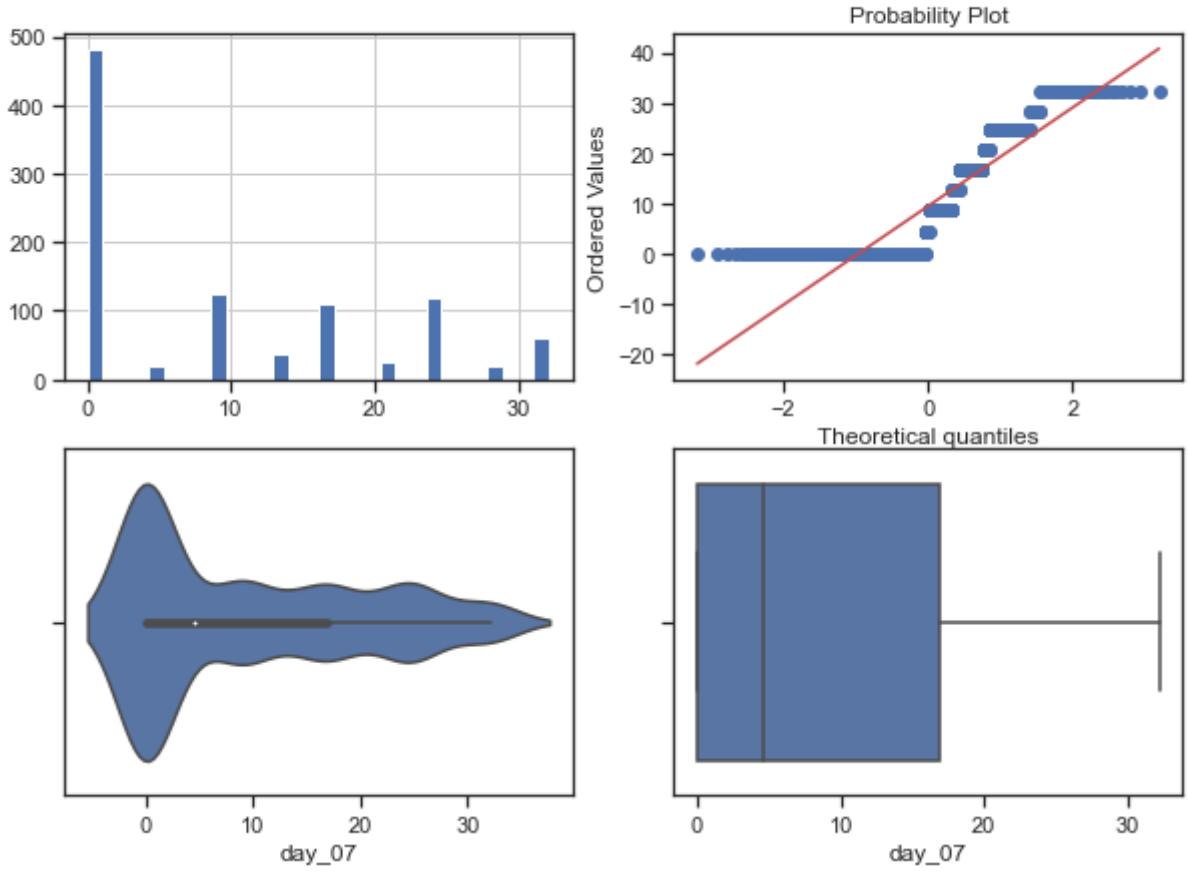
Поле-day_05, метод-OutlierBoundaryType.QUANTILE, строк-1010



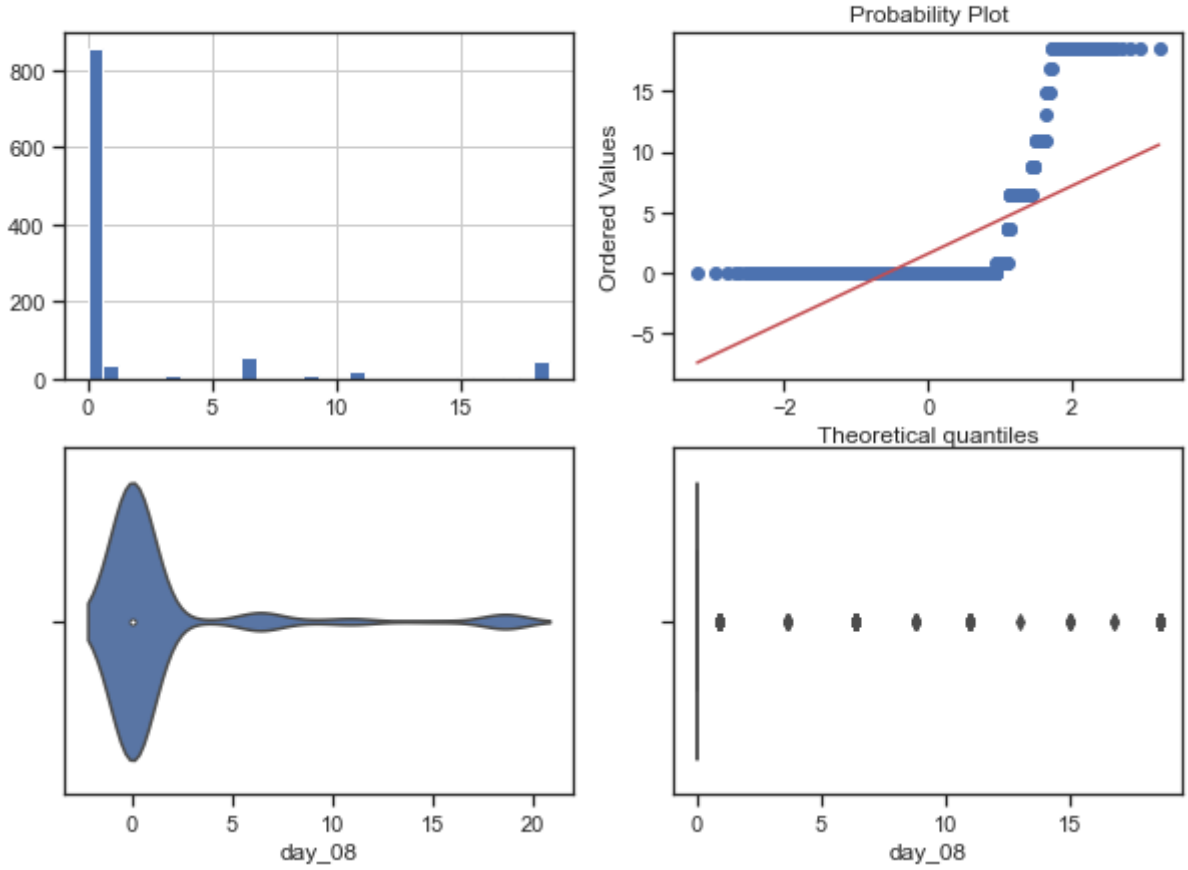
Поле-day_06, метод-OutlierBoundaryType.QUANTILE, строк-1059



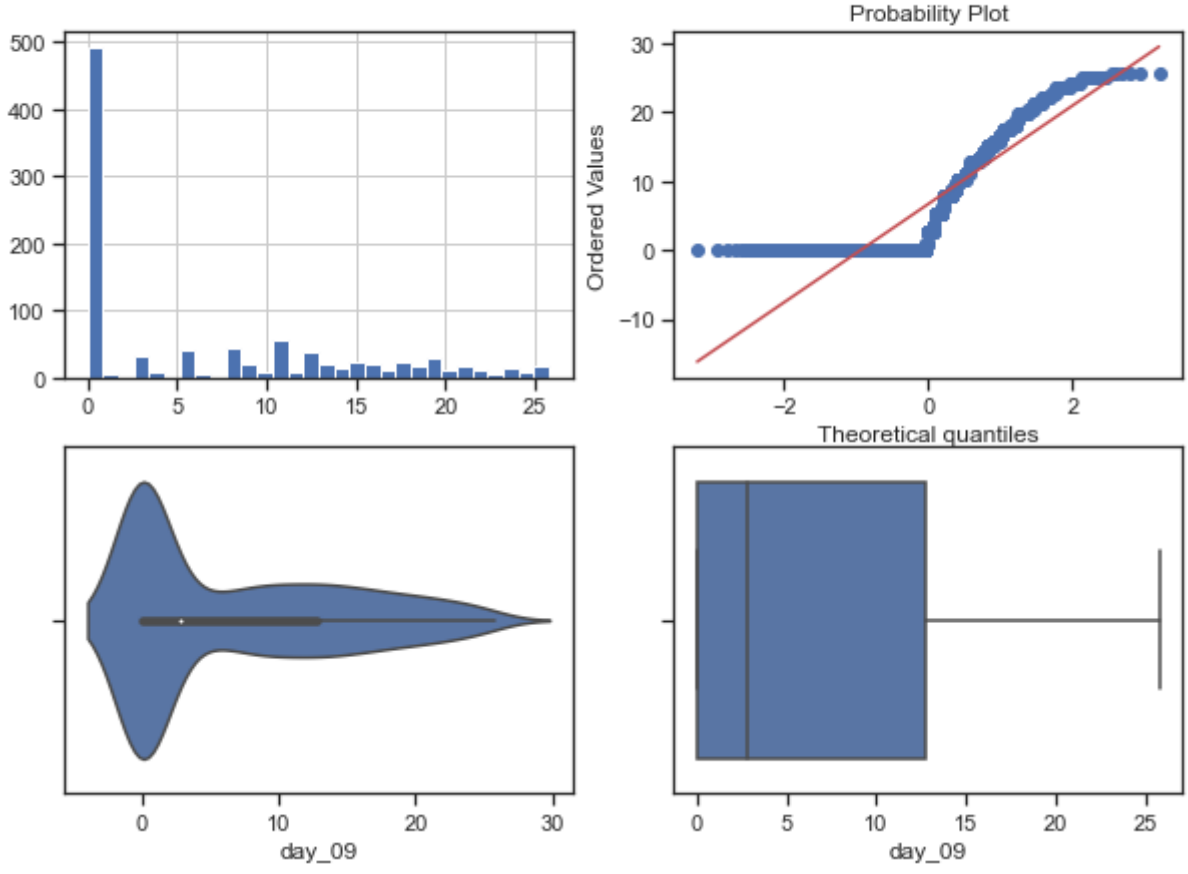
Поле-day_07, метод-OutlierBoundaryType.QUANTILE, строк-996



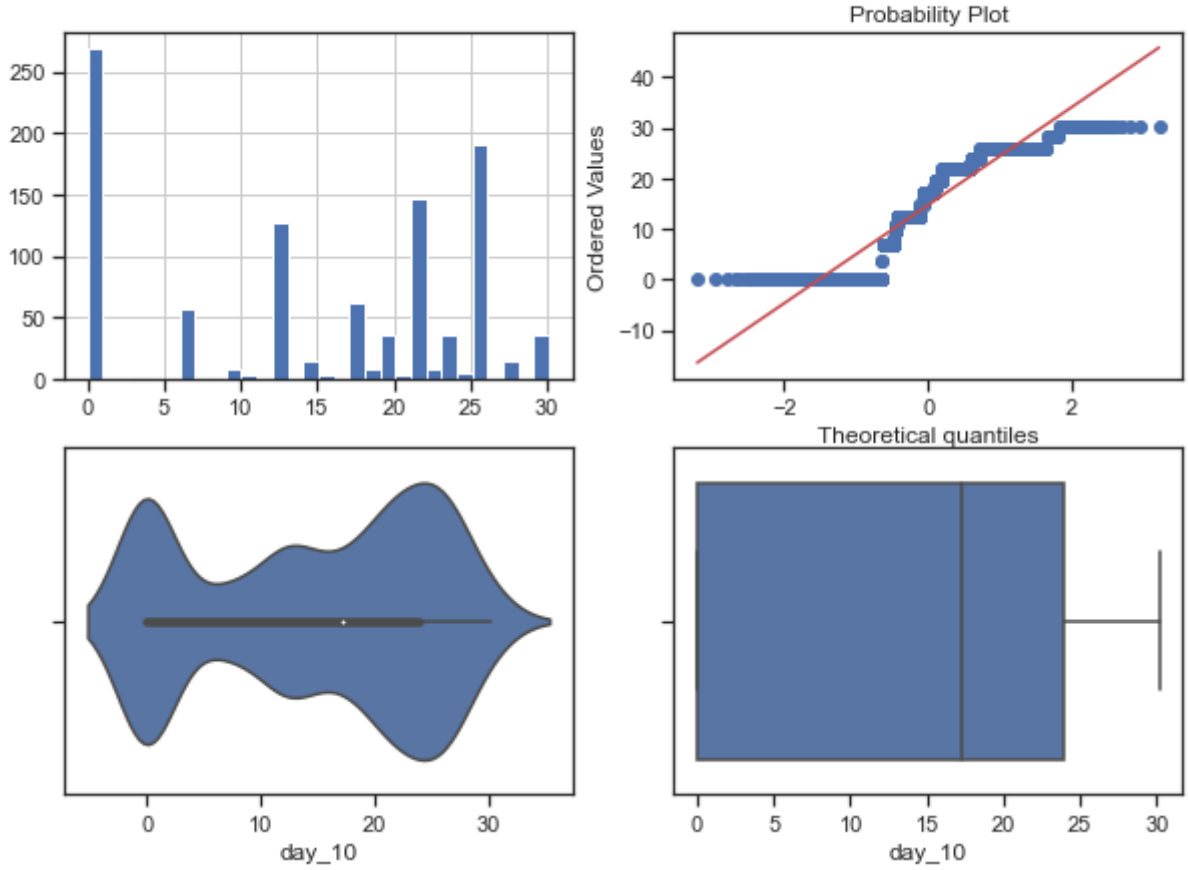
Поле-day_08, метод-OutlierBoundaryType.QUANTILE, строк-1033



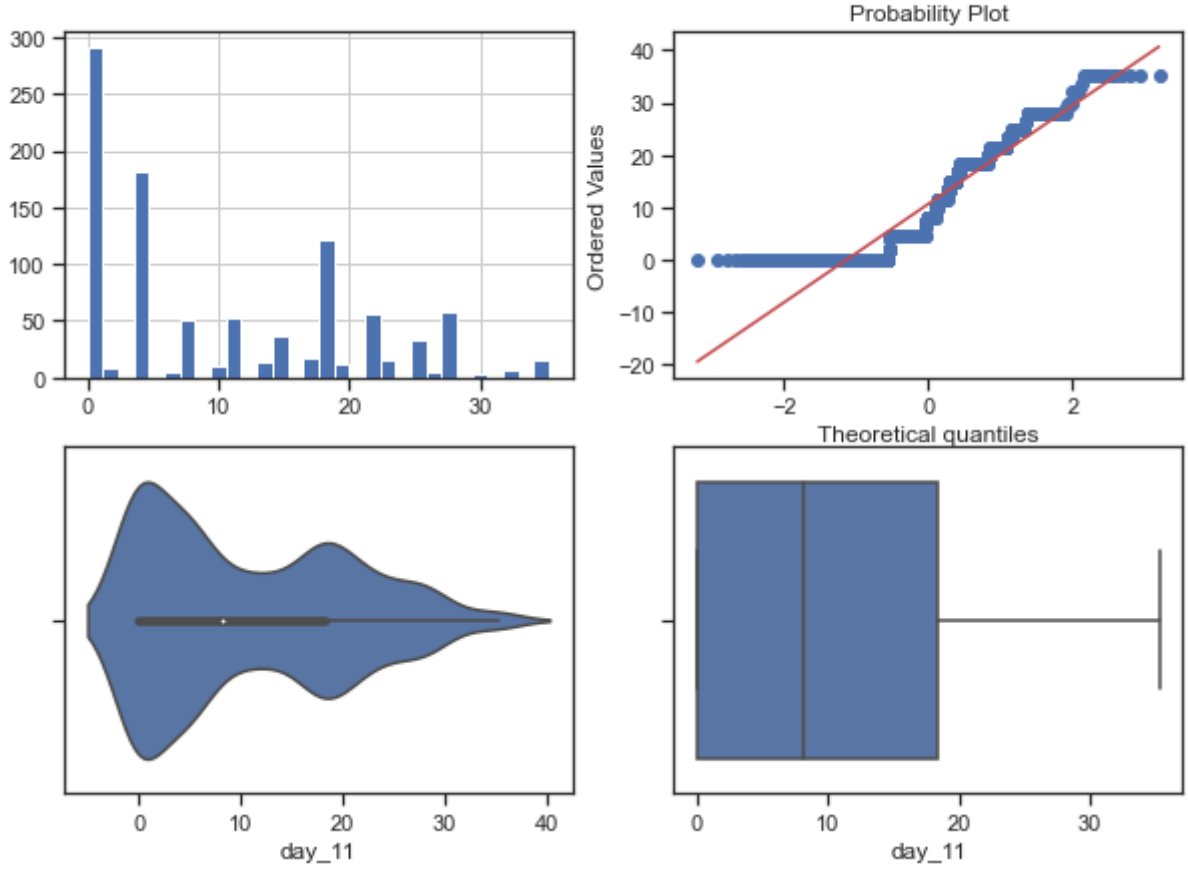
Поле-day_09, метод-OutlierBoundaryType.QUANTILE, строк-1004



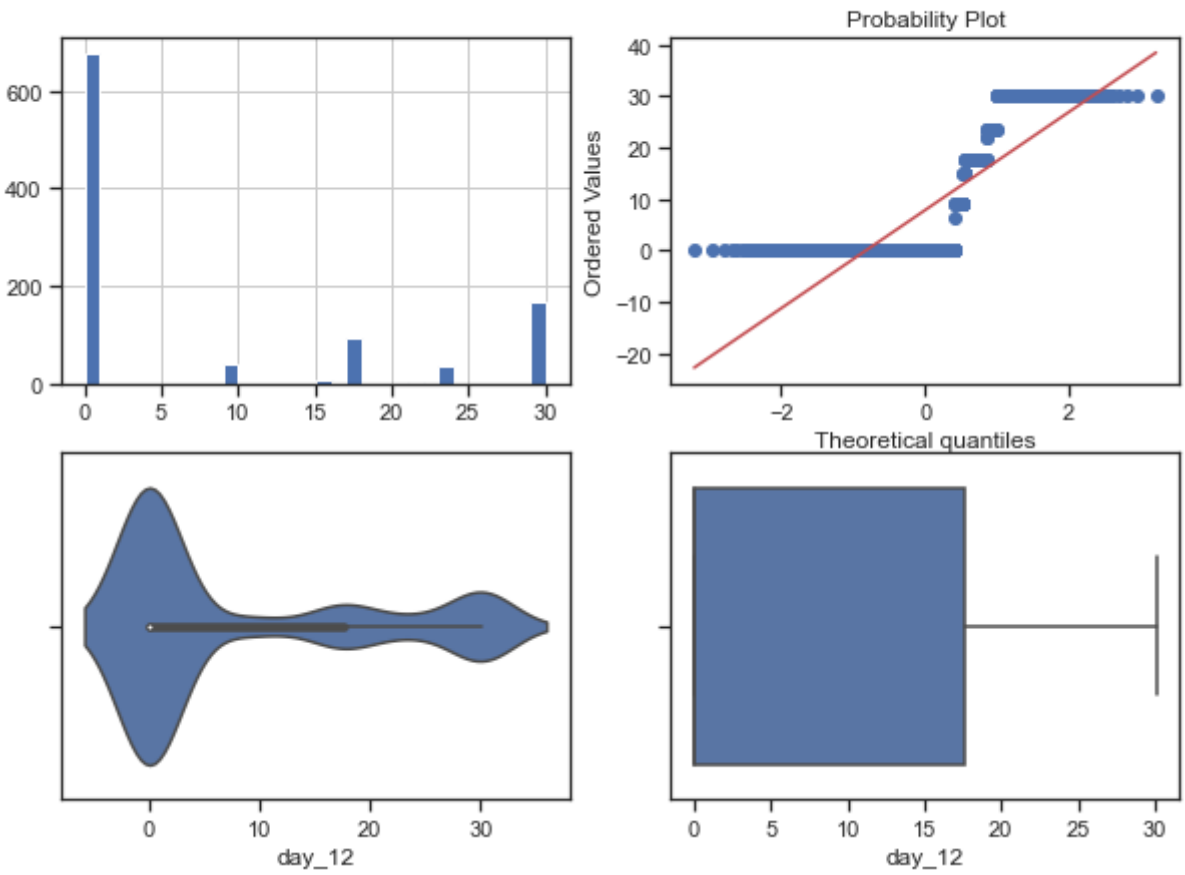
Поле-day_10, метод-OutlierBoundaryType.QUANTILE, строк-1029



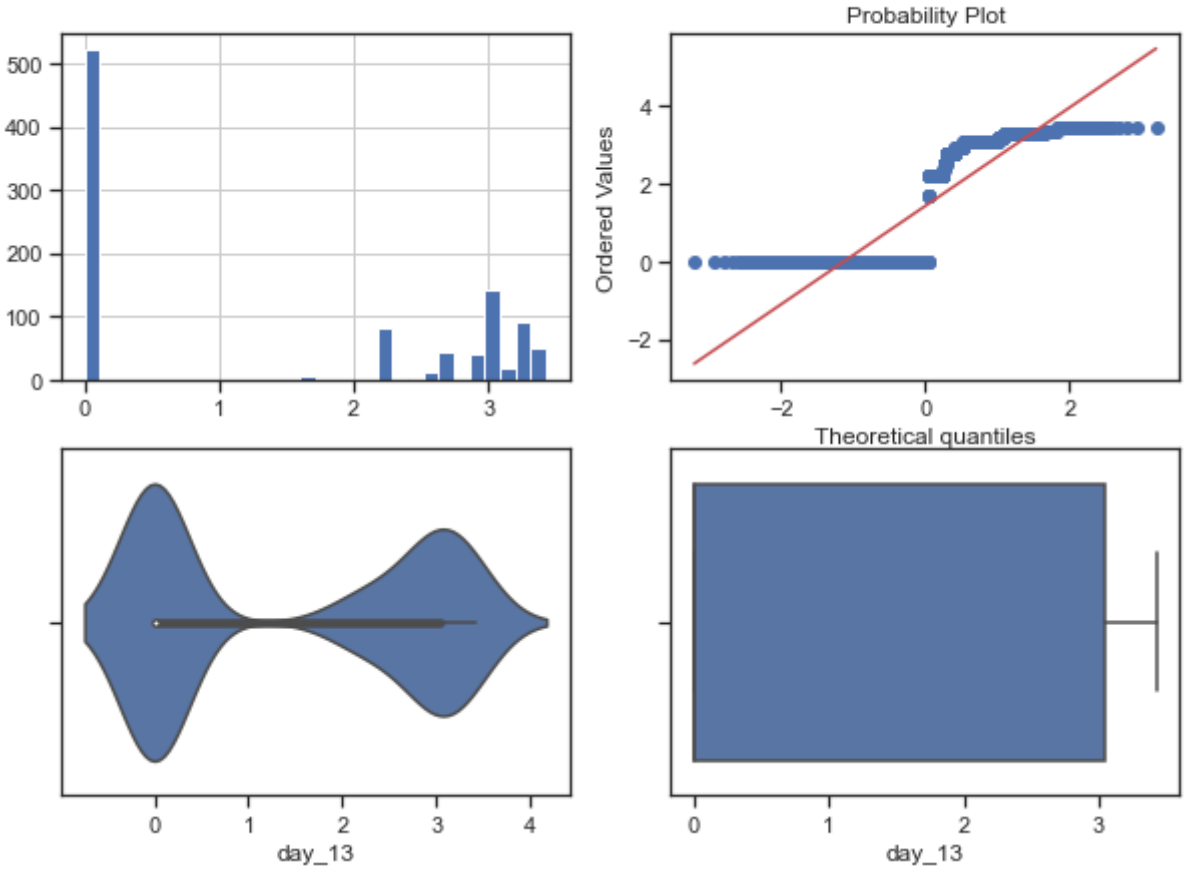
Поле-day_11, метод-OutlierBoundaryType.QUANTILE, строк-997



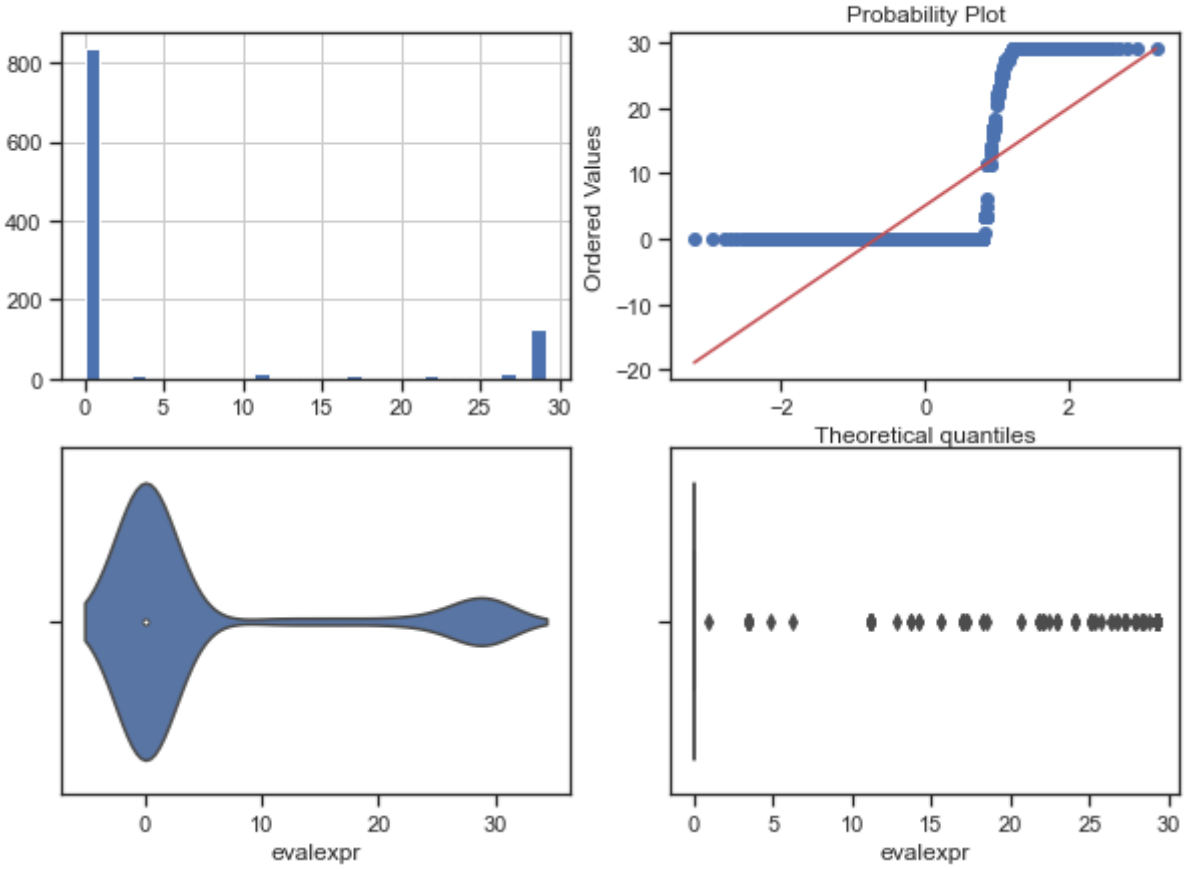
Поле-day_12, метод-OutlierBoundaryType.QUANTILE, строк-1025



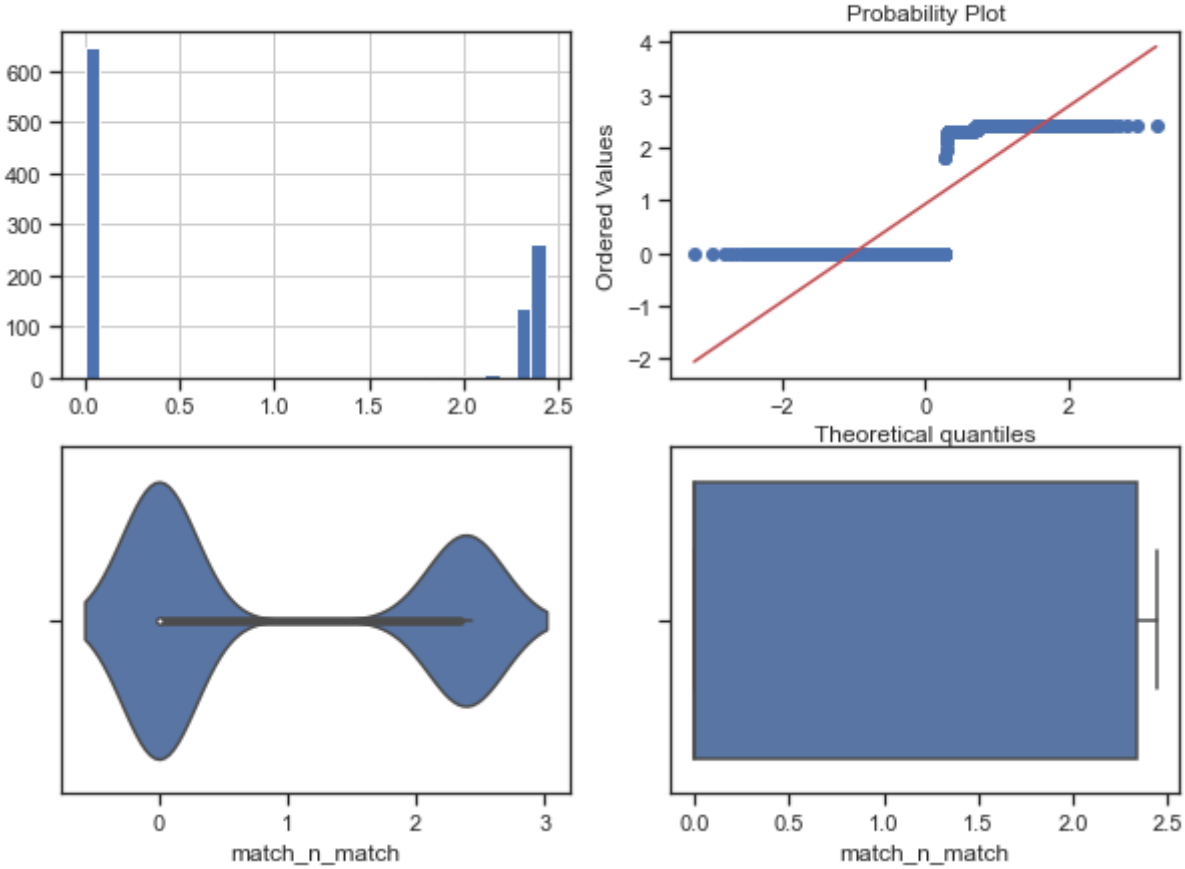
Поле-day_13, метод-OutlierBoundaryType.QUANTILE, строк-1011



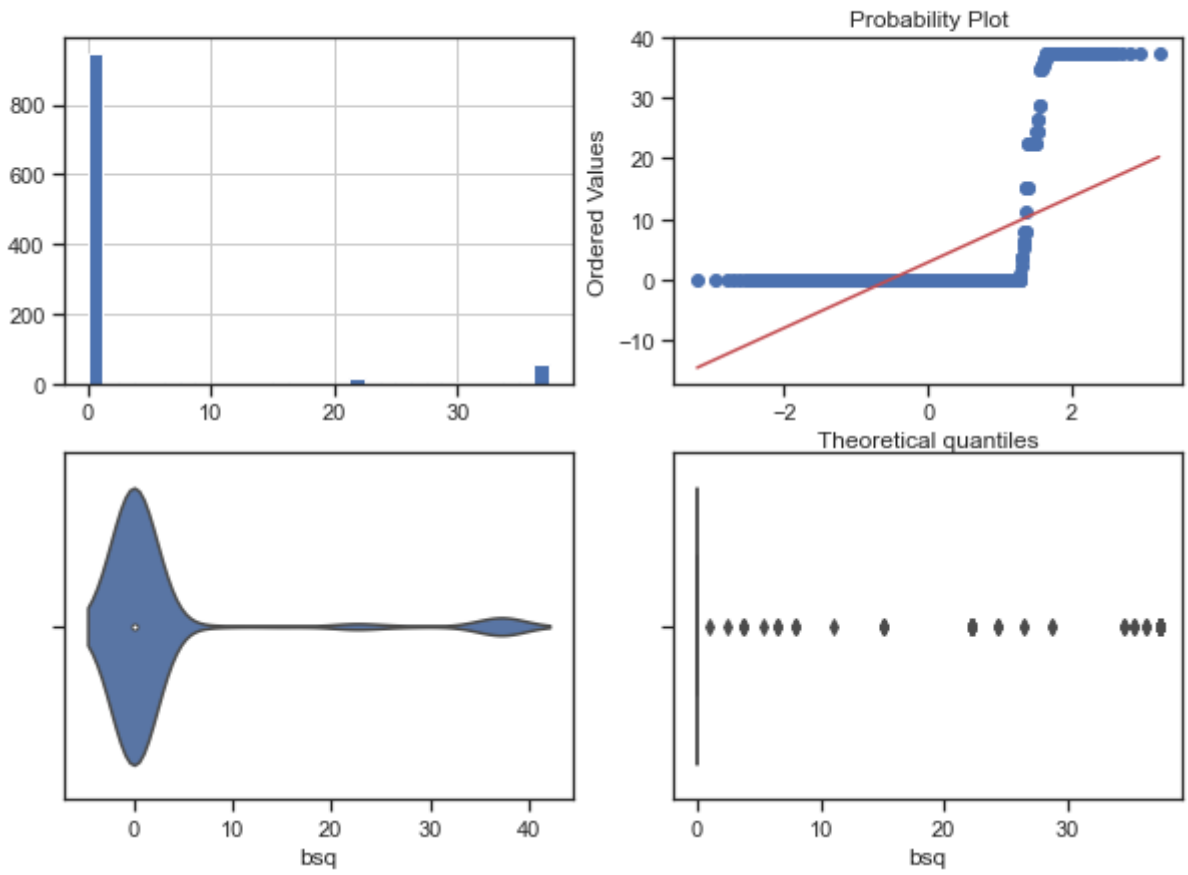
Поле-evalexpr, метод-OutlierBoundaryType.QUANTILE, строк-1055



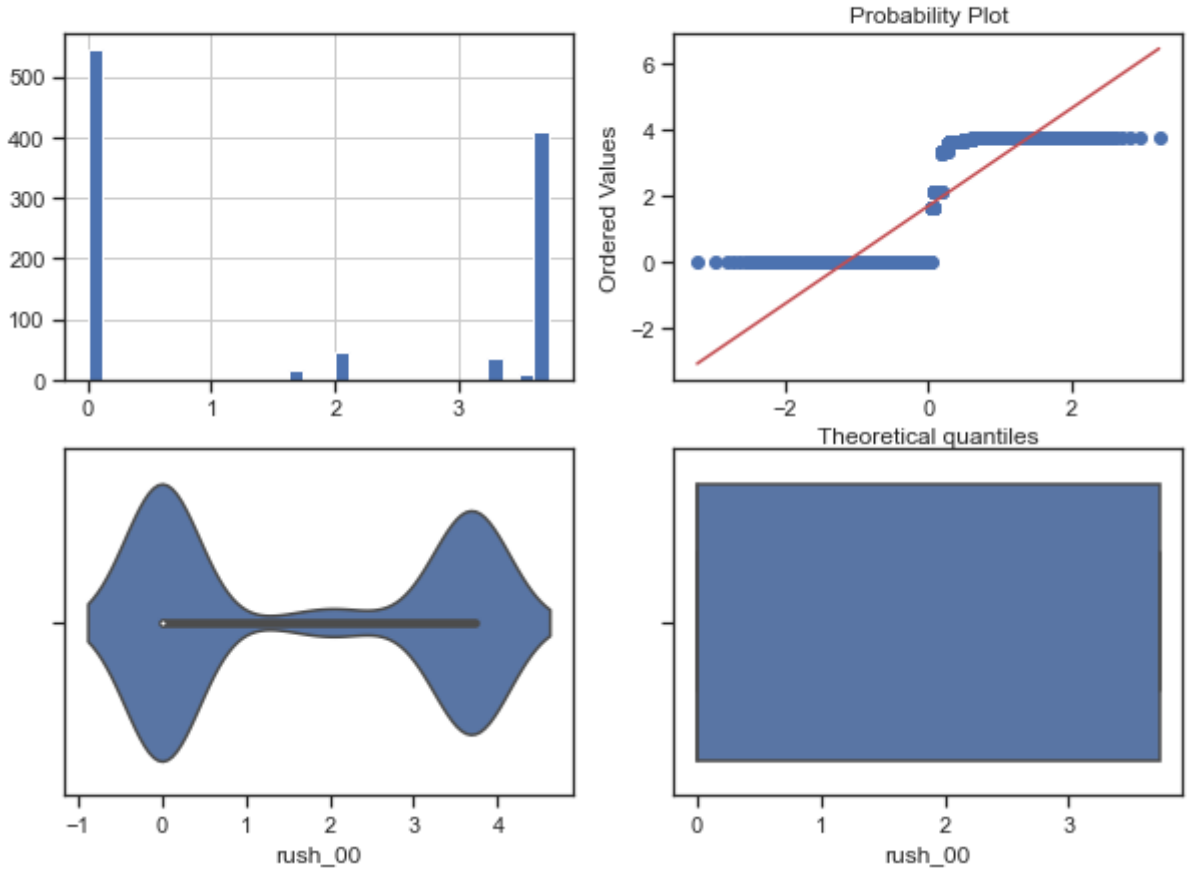
Поле-match_n_match, метод-OutlierBoundaryType.QUANTILE, строк-1060



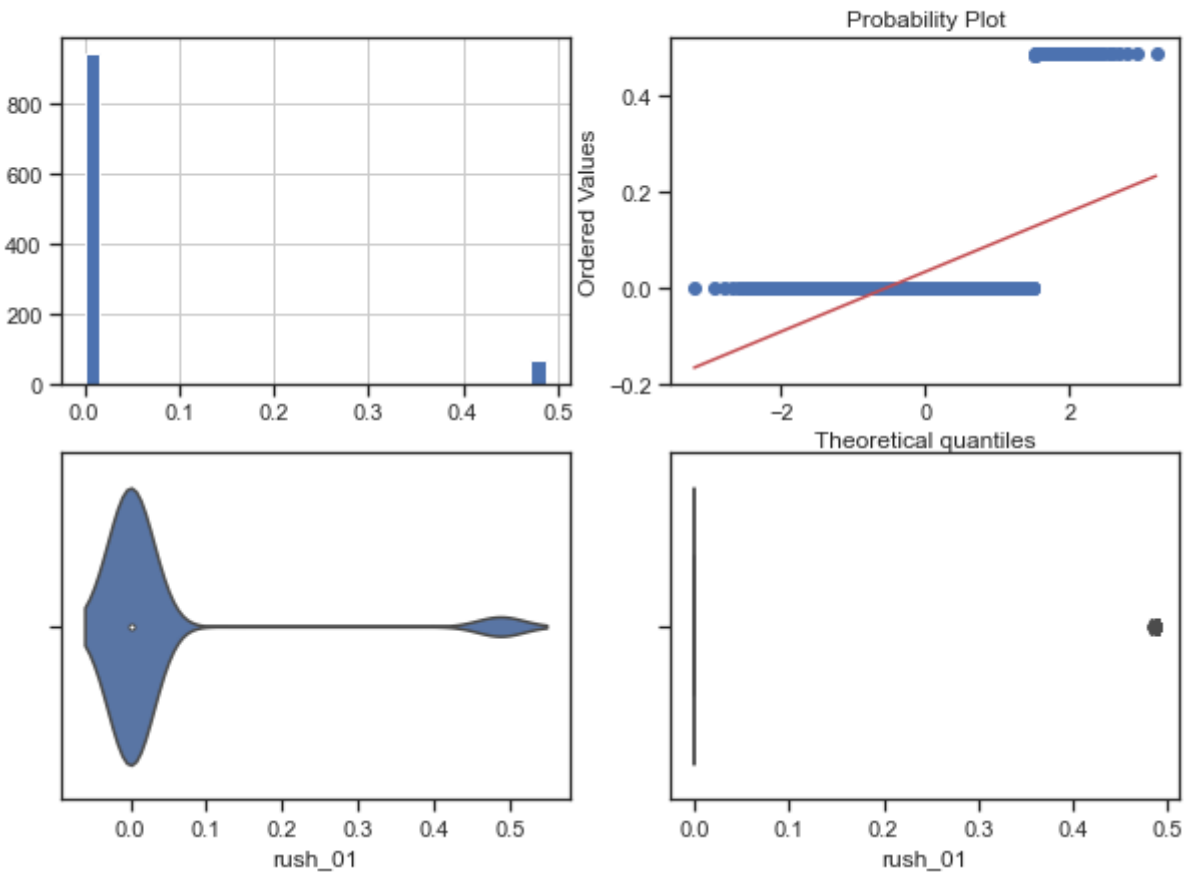
Поле-bsq, метод-OutlierBoundaryType.QUANTILE, строк-1051



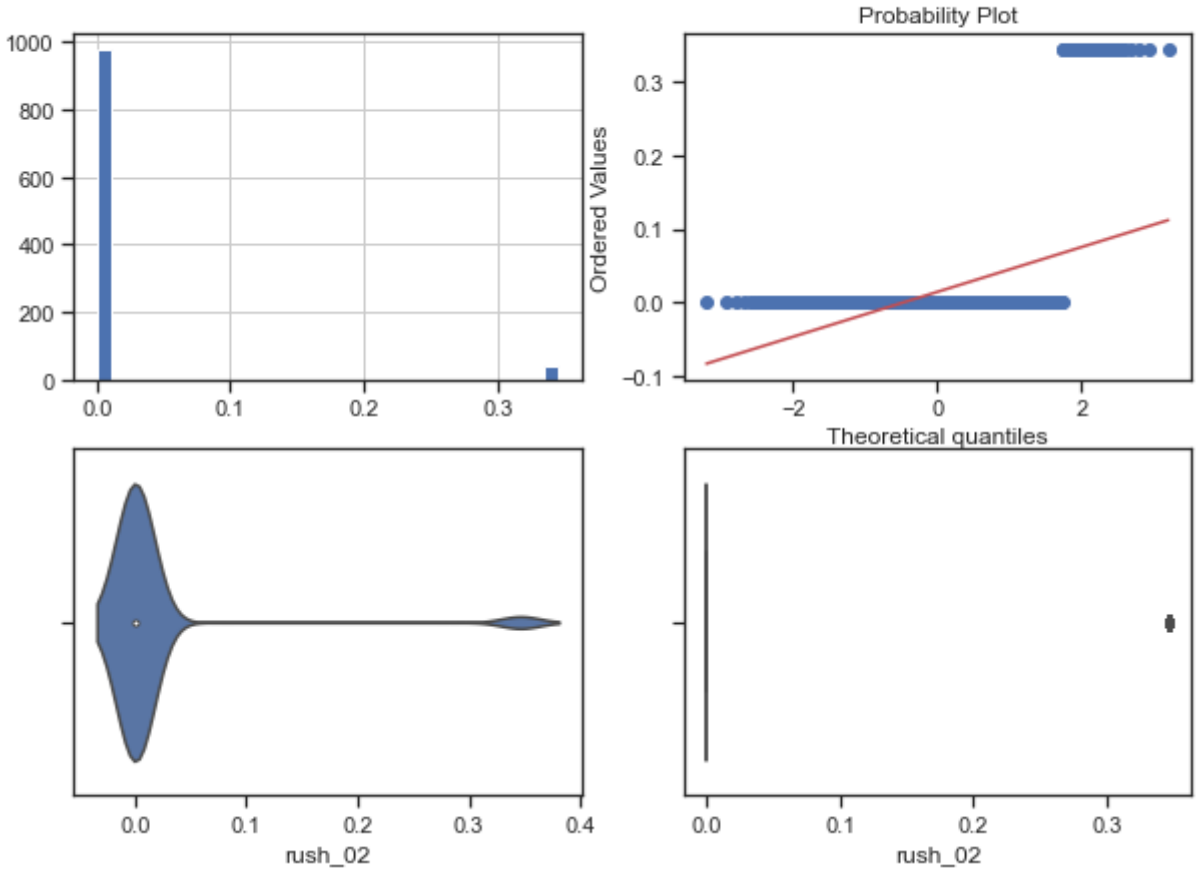
Поле-rush_00, метод-OutlierBoundaryType.QUANTILE, строк-1058



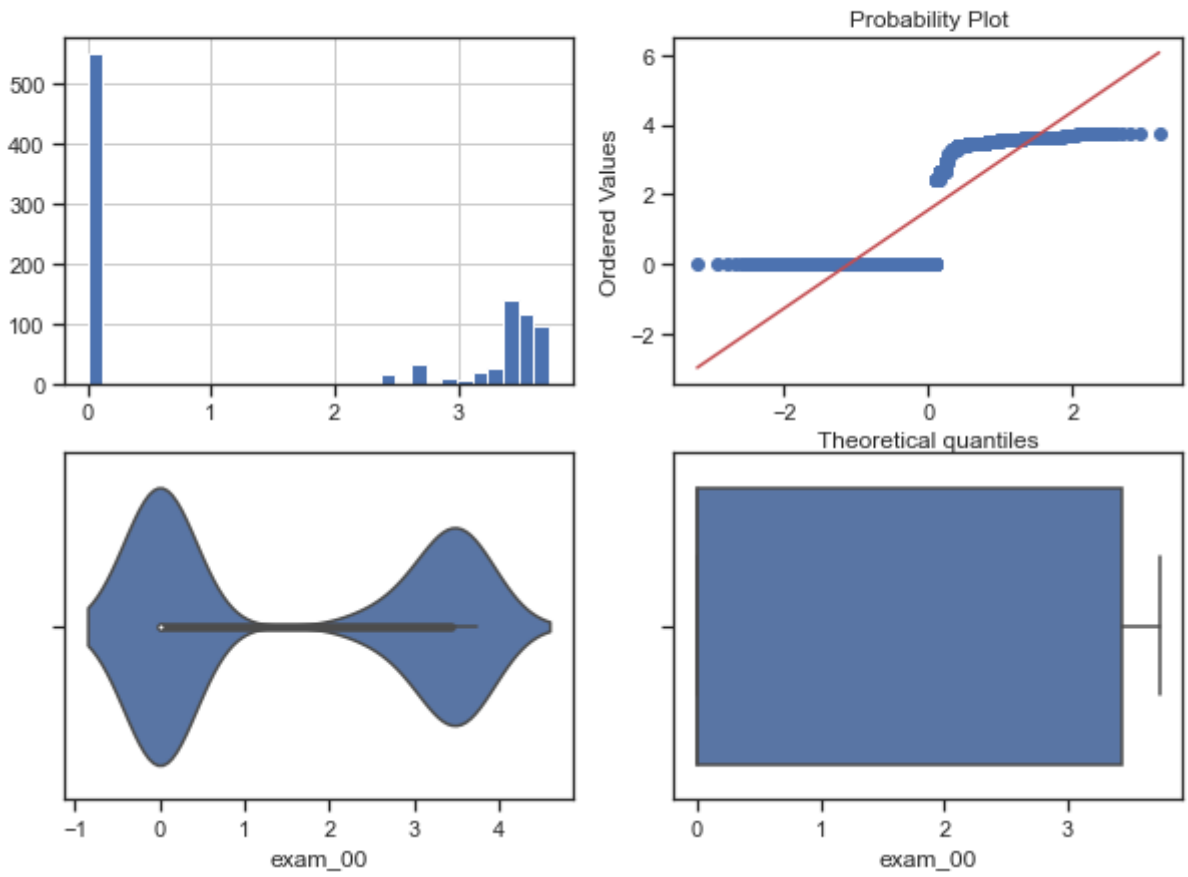
Поле-rush_01, метод-OutlierBoundaryType.QUANTILE, строк-1008



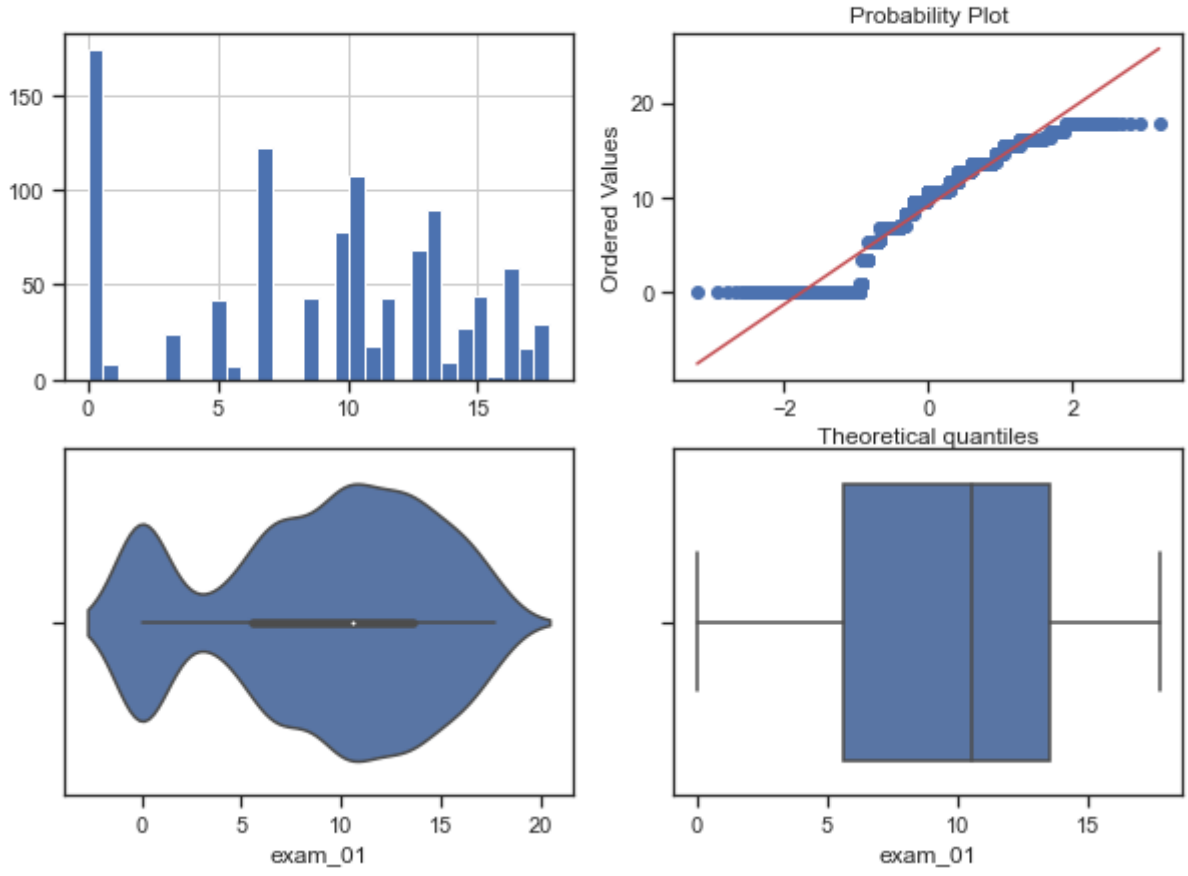
Поле-rush_02, метод-OutlierBoundaryType.QUANTILE, строк-1020



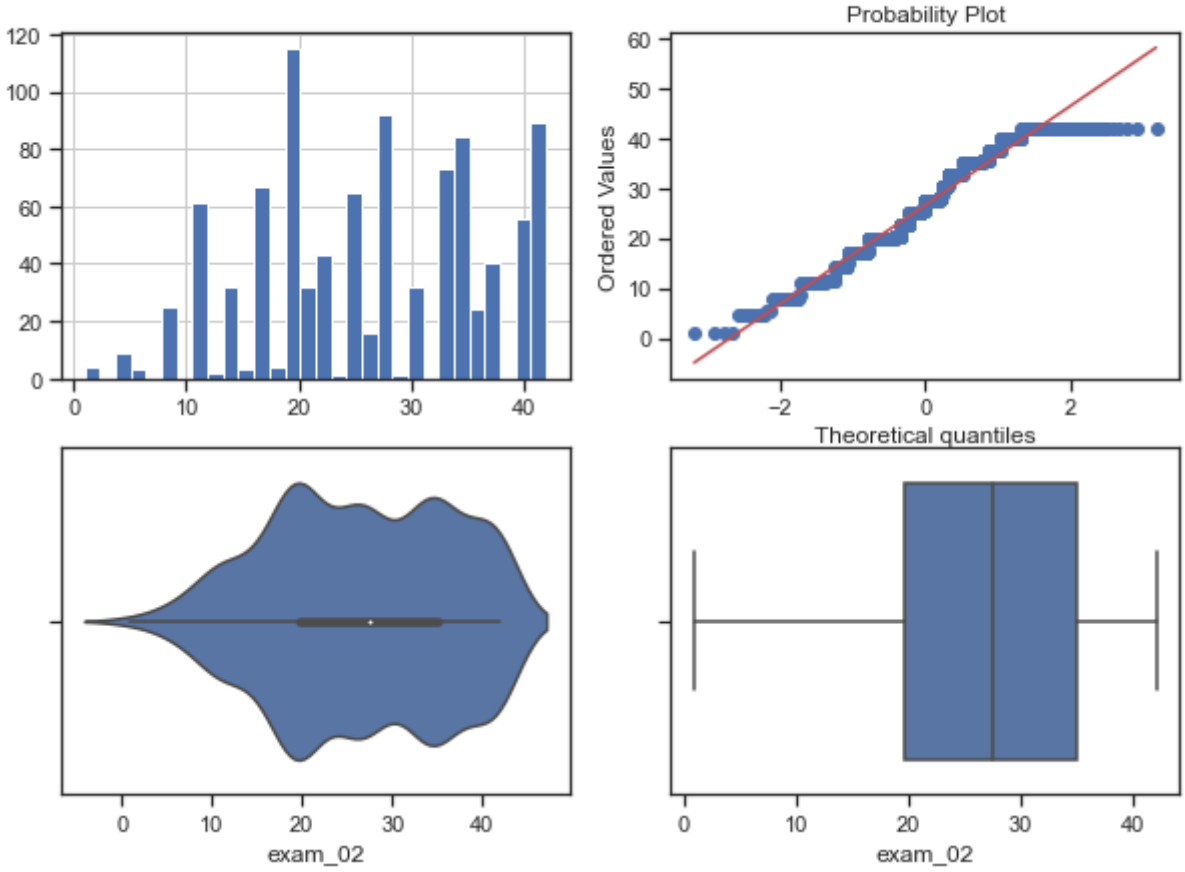
Поле-exam_00, метод-OutlierBoundaryType.QUANTILE, строк-1015



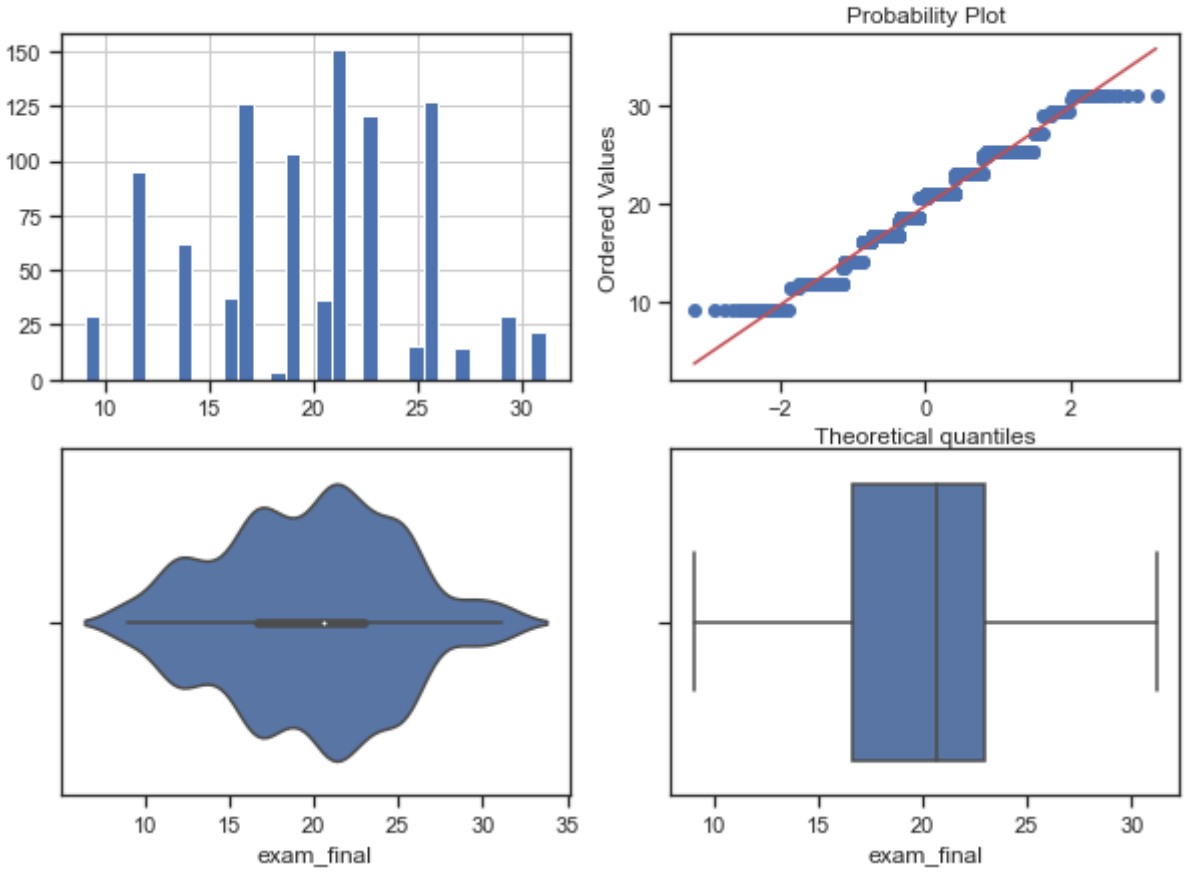
Поле-exam_01, метод-OutlierBoundaryType.QUANTILE, строк-1011

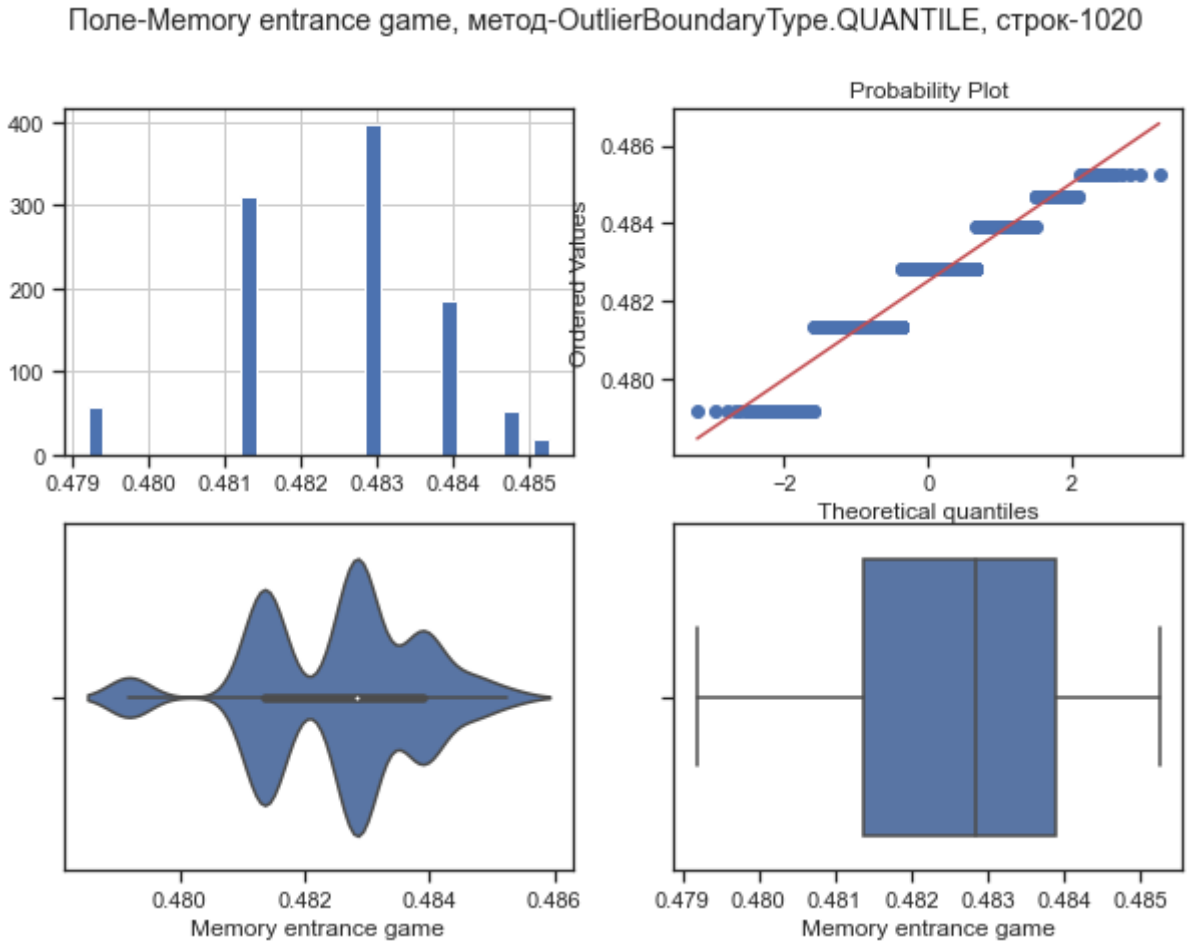


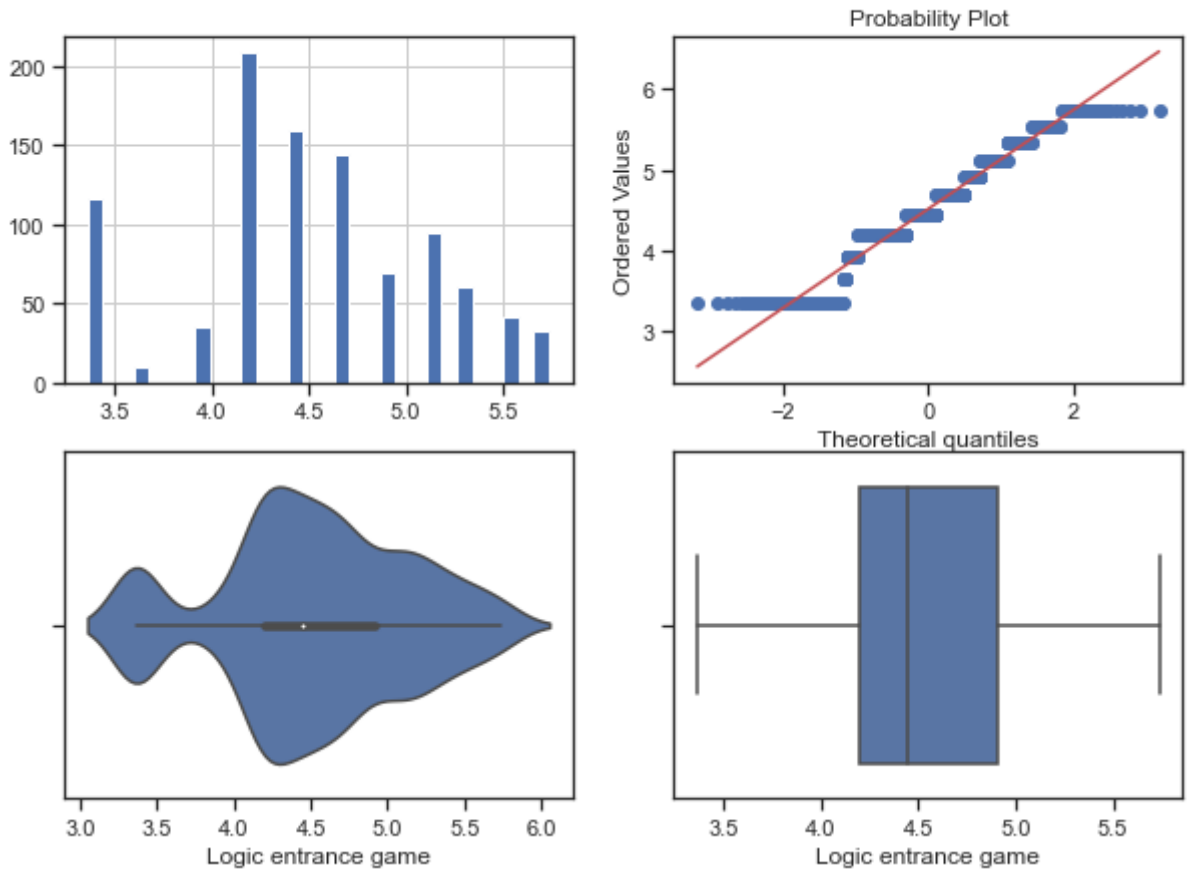
Поле-exam_02, метод-OutlierBoundaryType.QUANTILE, строк-973



Поле-exam_final, метод-OutlierBoundaryType.QUANTILE, строк-971



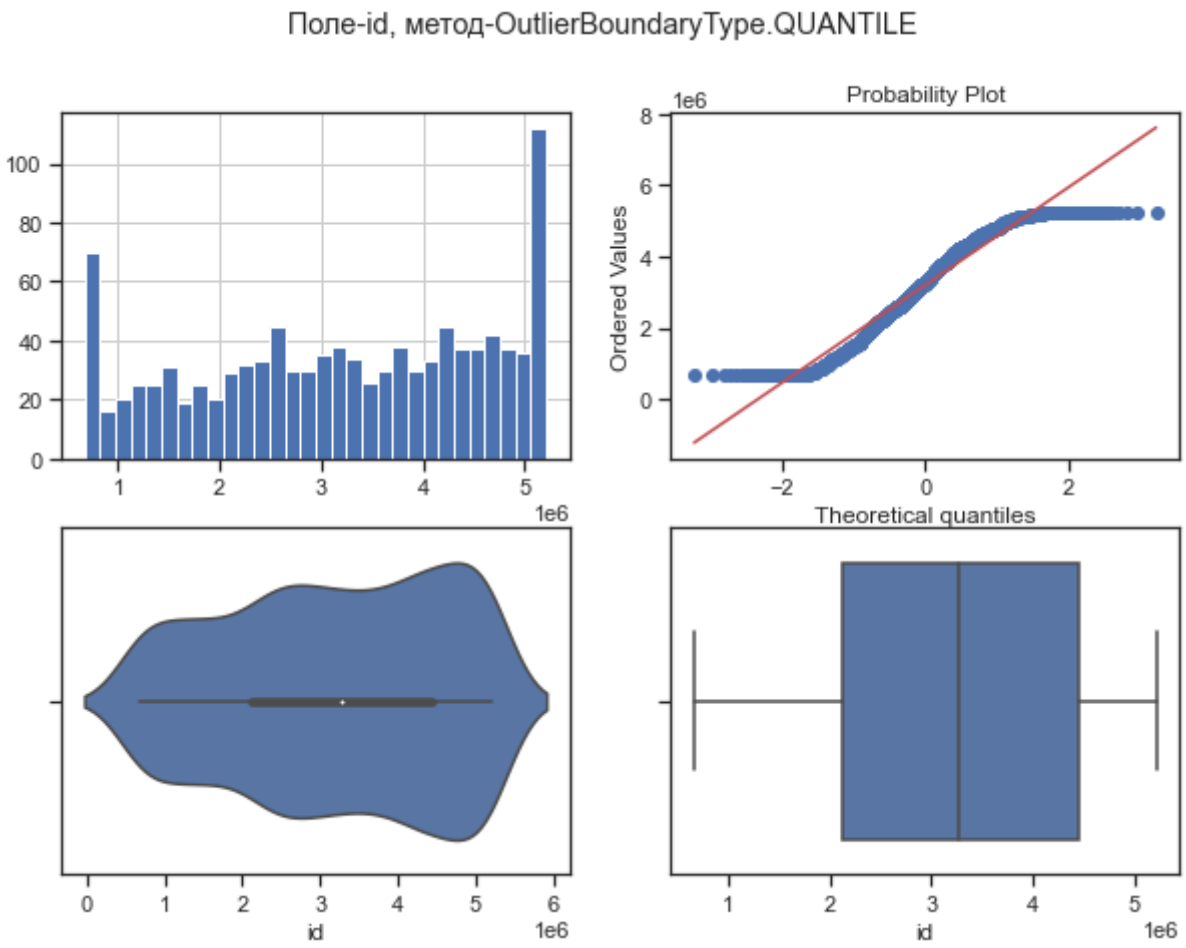




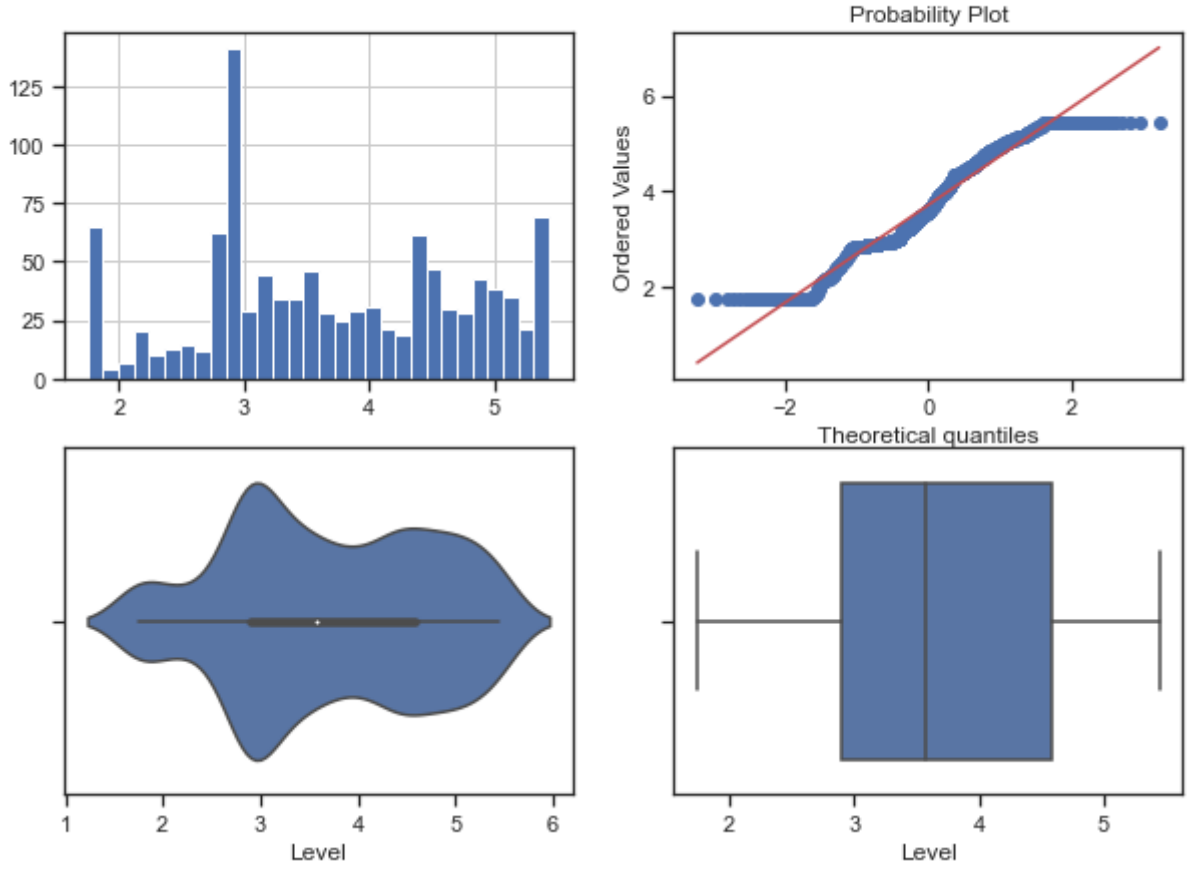
Замена выбросов

In [37]:
```
for col in outliers_treatment_cols:
```

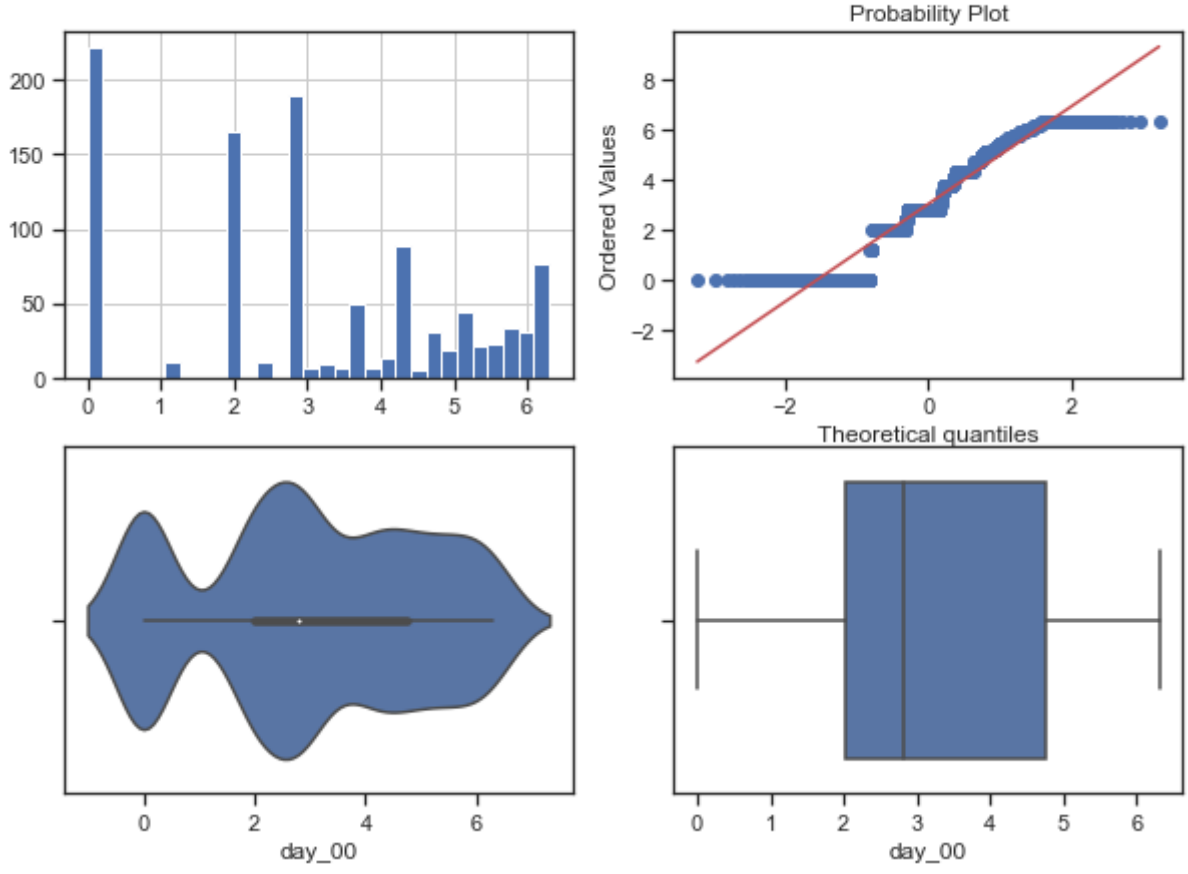
```
    for obt in OutlierBoundaryType:
        # Вычисление верхней и нижней границы
        lower_boundary, upper_boundary = get_outlier_boundaries(data_scaled, col, ob
        # Изменение данных
        data_outliers_deleted[col] = np.where(data_scaled[col] > upper_boundary, upp
                                     np.where(data_scaled[col] < lower_boundary, lower_b
        title = 'Поле-{}, метод-{}'.format(col, obt)
        diagnostic_plots(data_outliers_deleted, col, title)
```



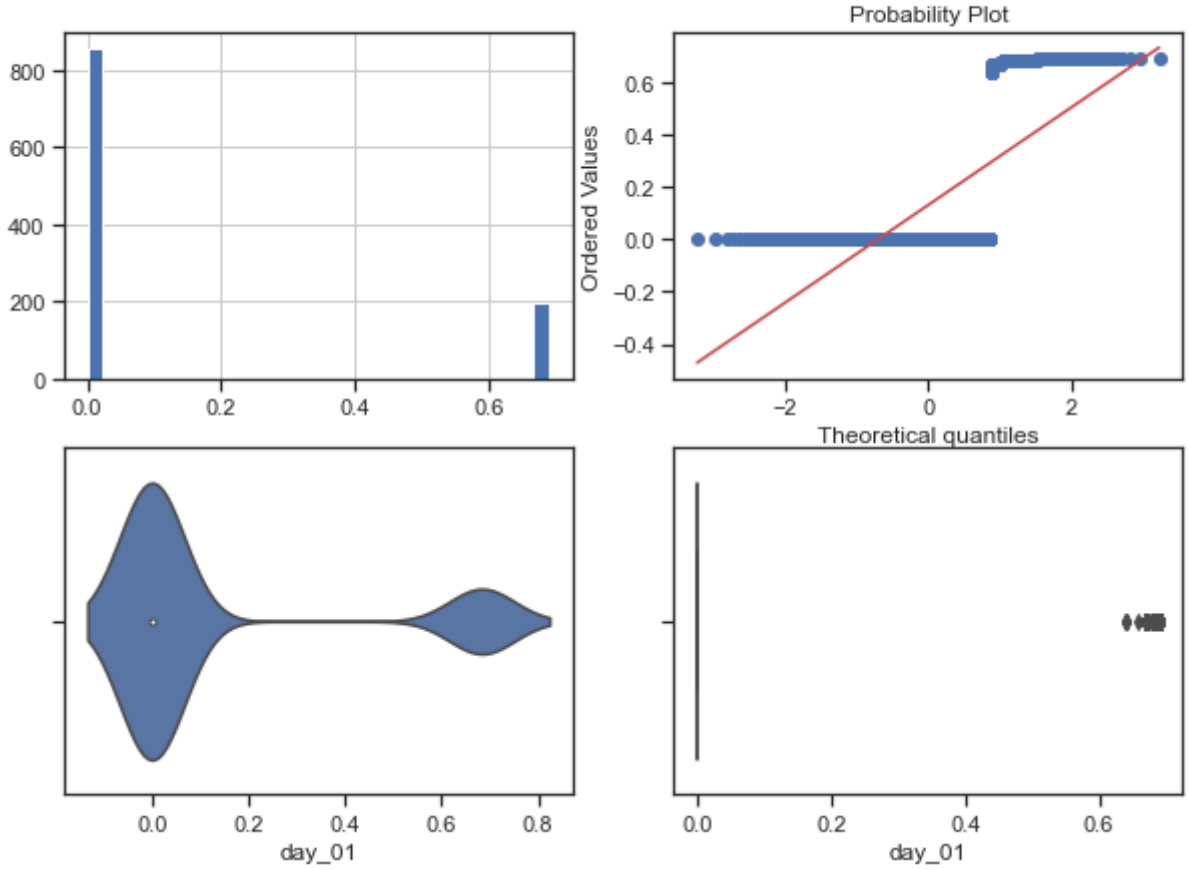
Поле-Level, метод-OutlierBoundaryType.QUANTILE



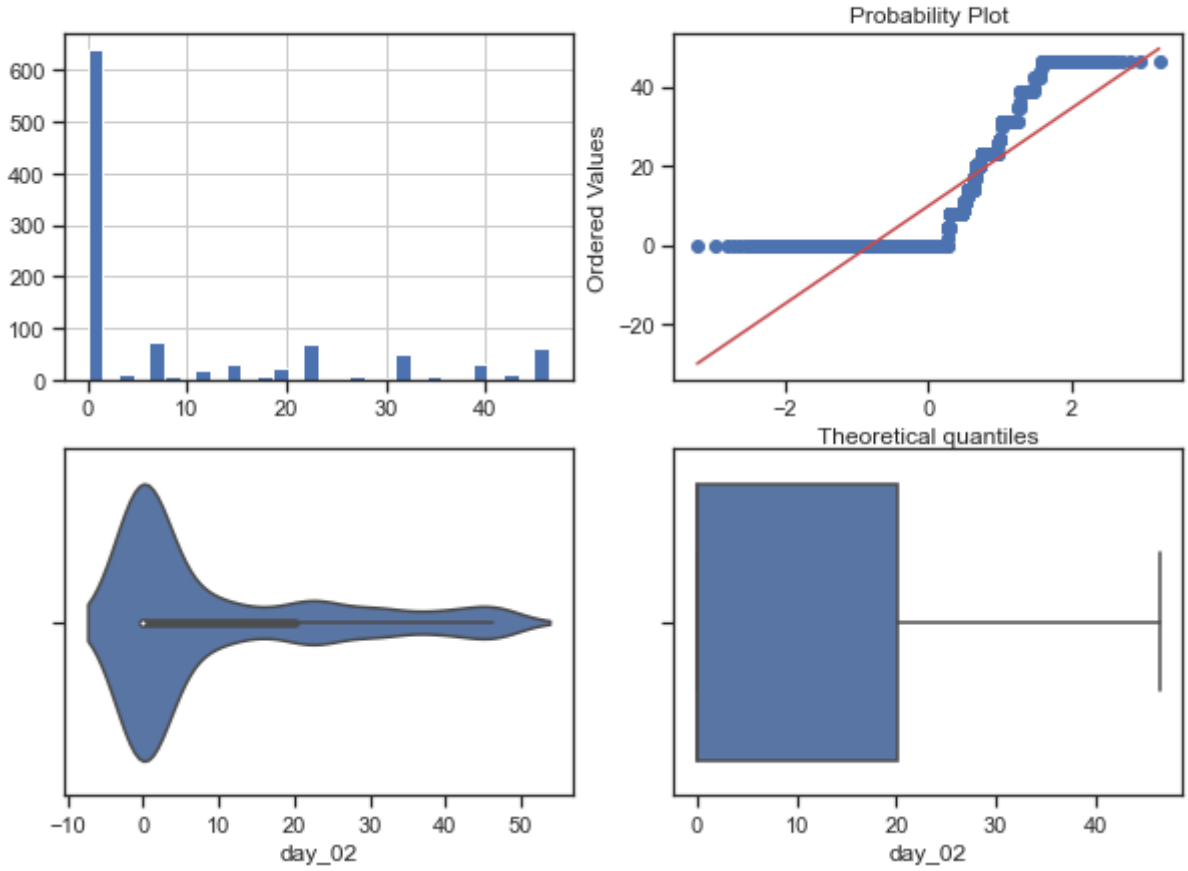
Поле-day_00, метод-OutlierBoundaryType.QUANTILE



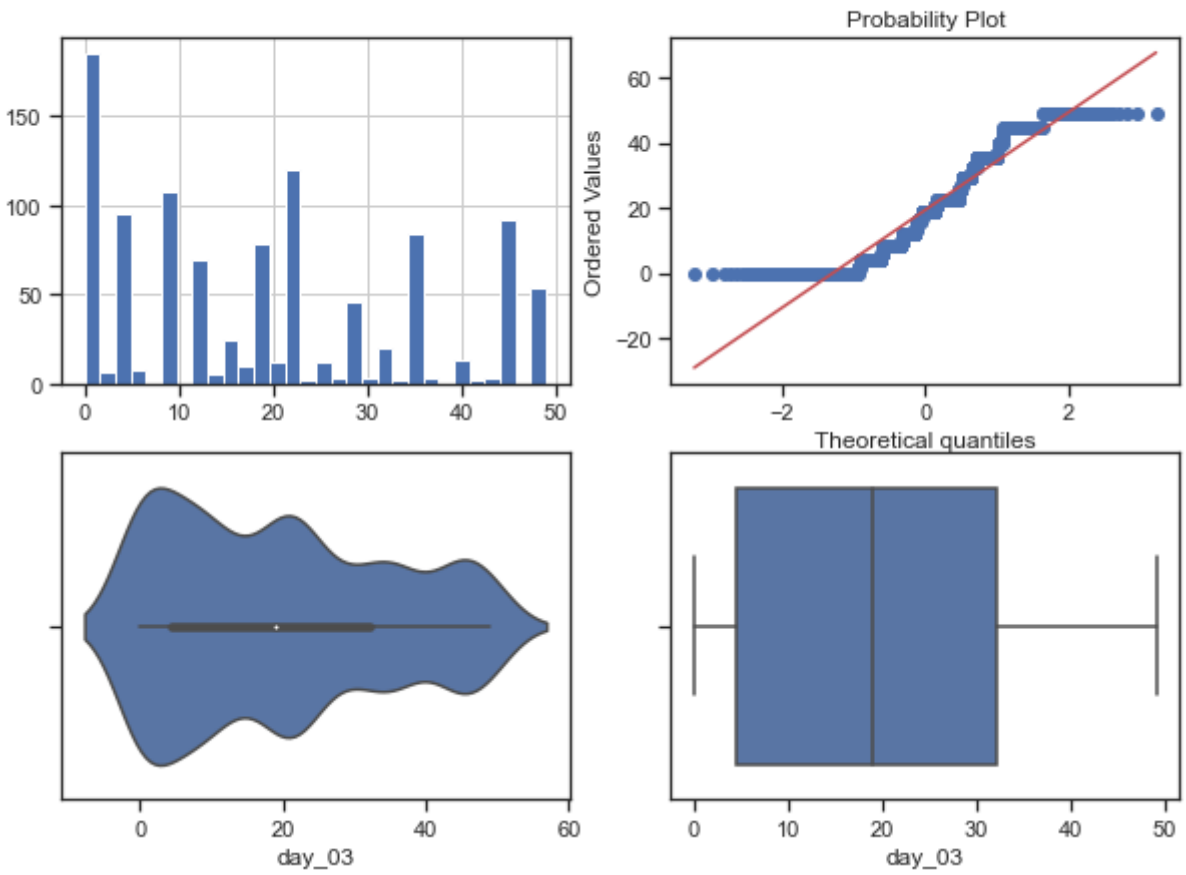
Поле-day_01, метод-OutlierBoundaryType.QUANTILE



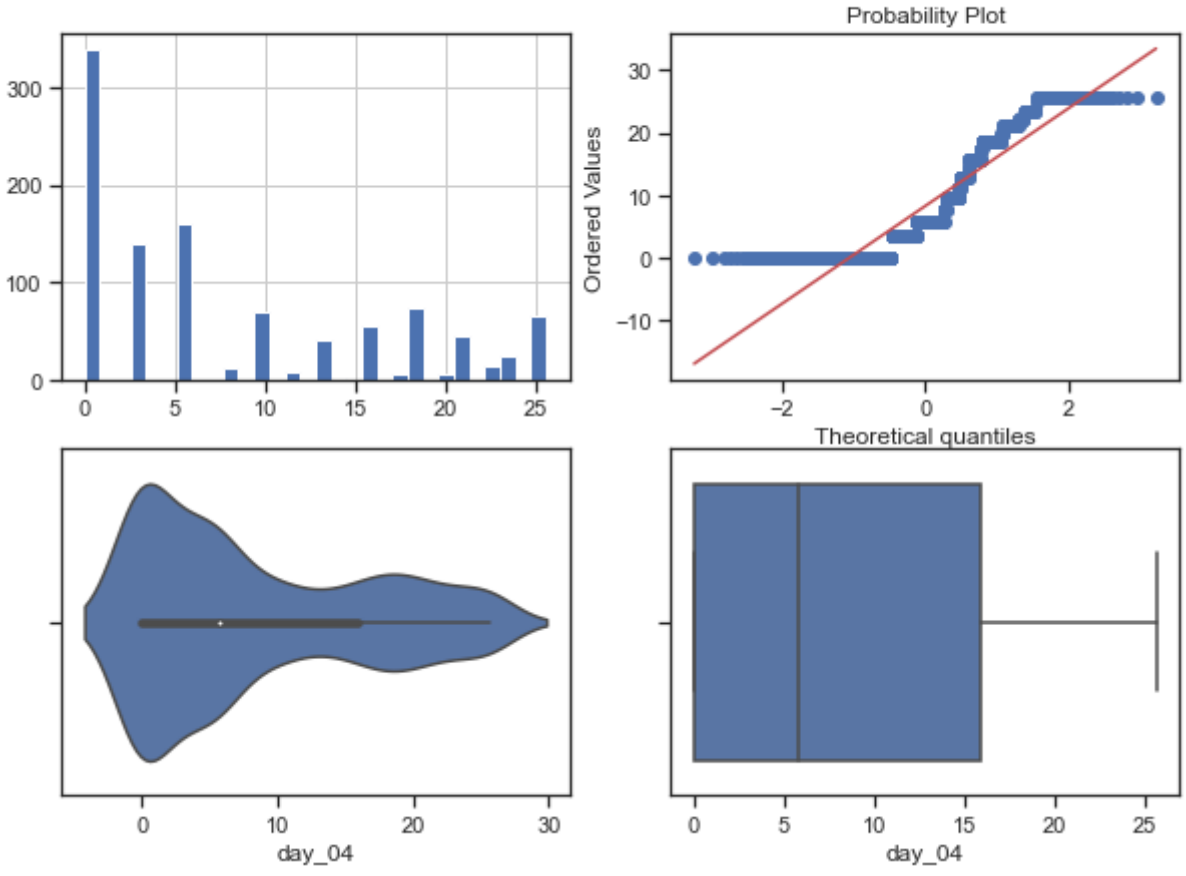
Поле-day_02, метод-OutlierBoundaryType.QUANTILE



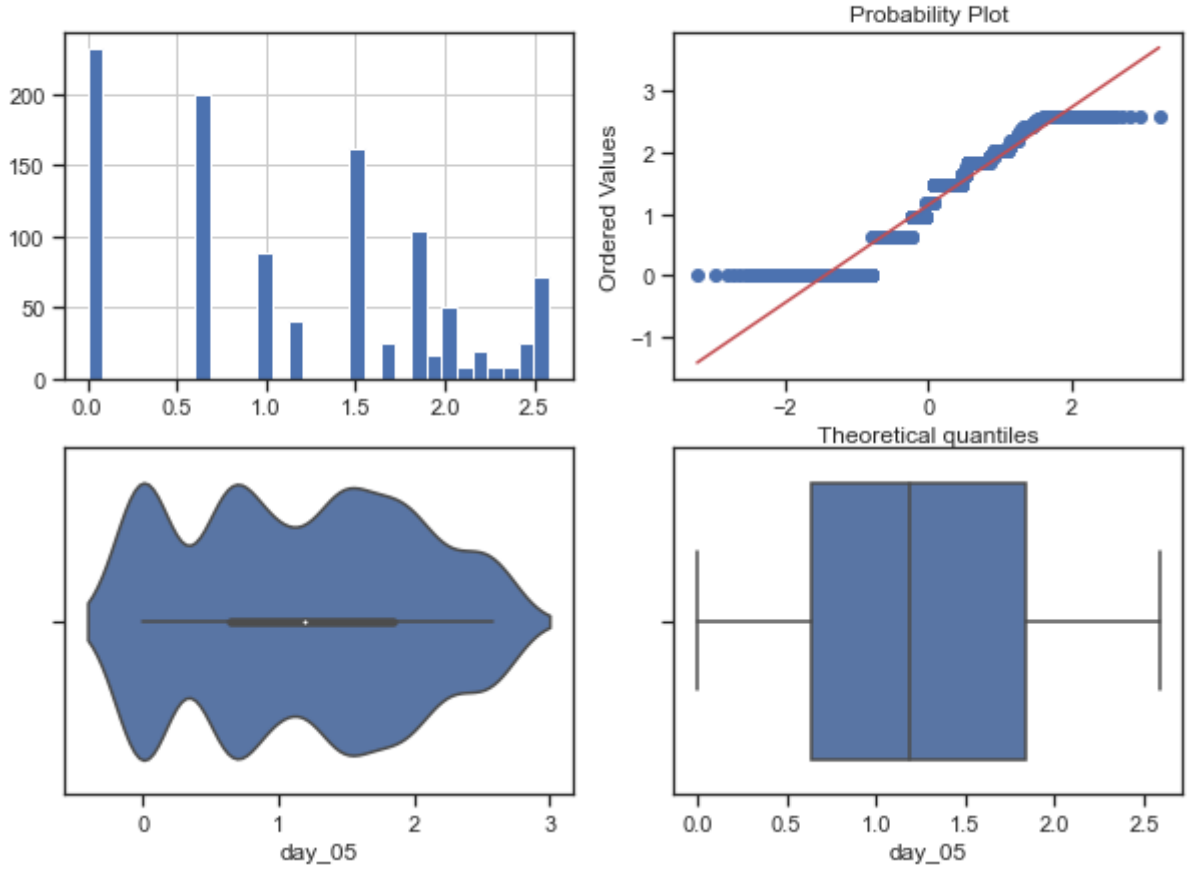
Поле-day_03, метод-OutlierBoundaryType.QUANTILE



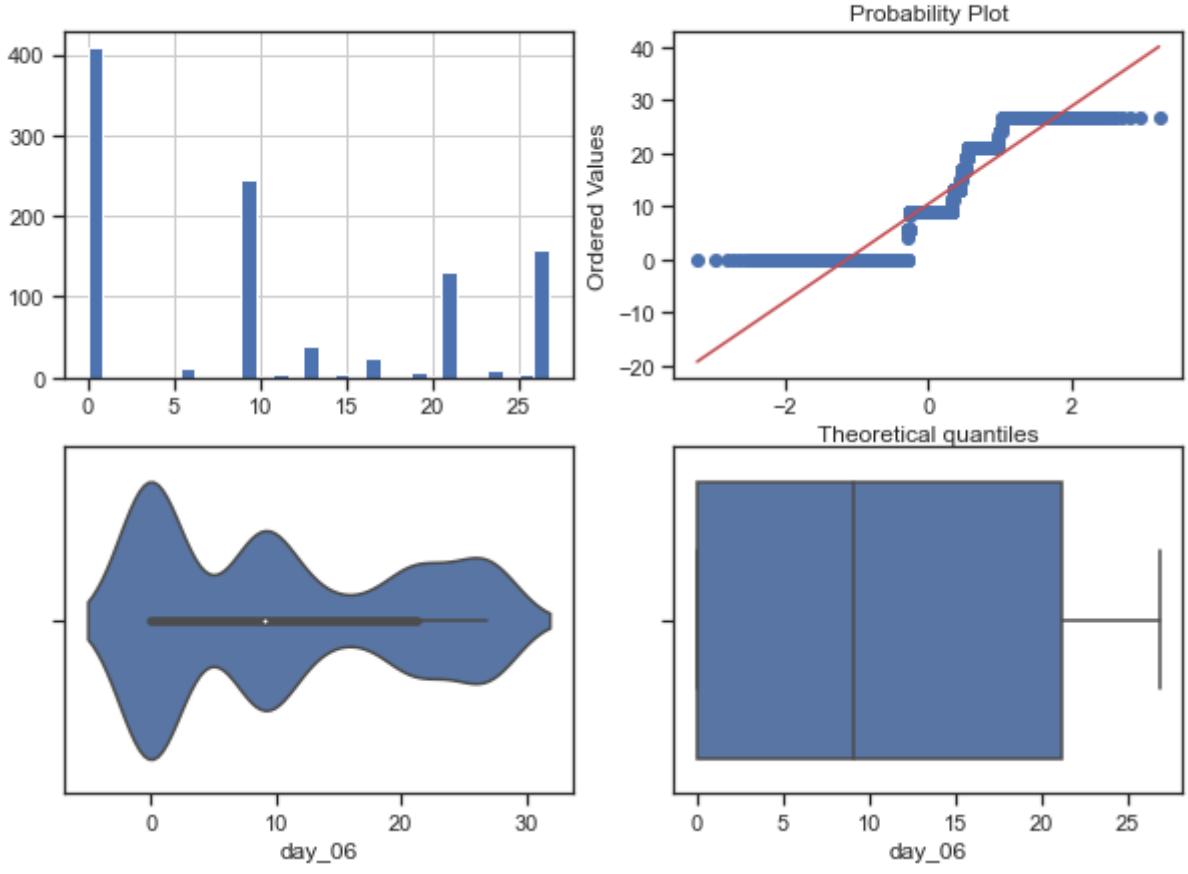
Поле-day_04, метод-OutlierBoundaryType.QUANTILE



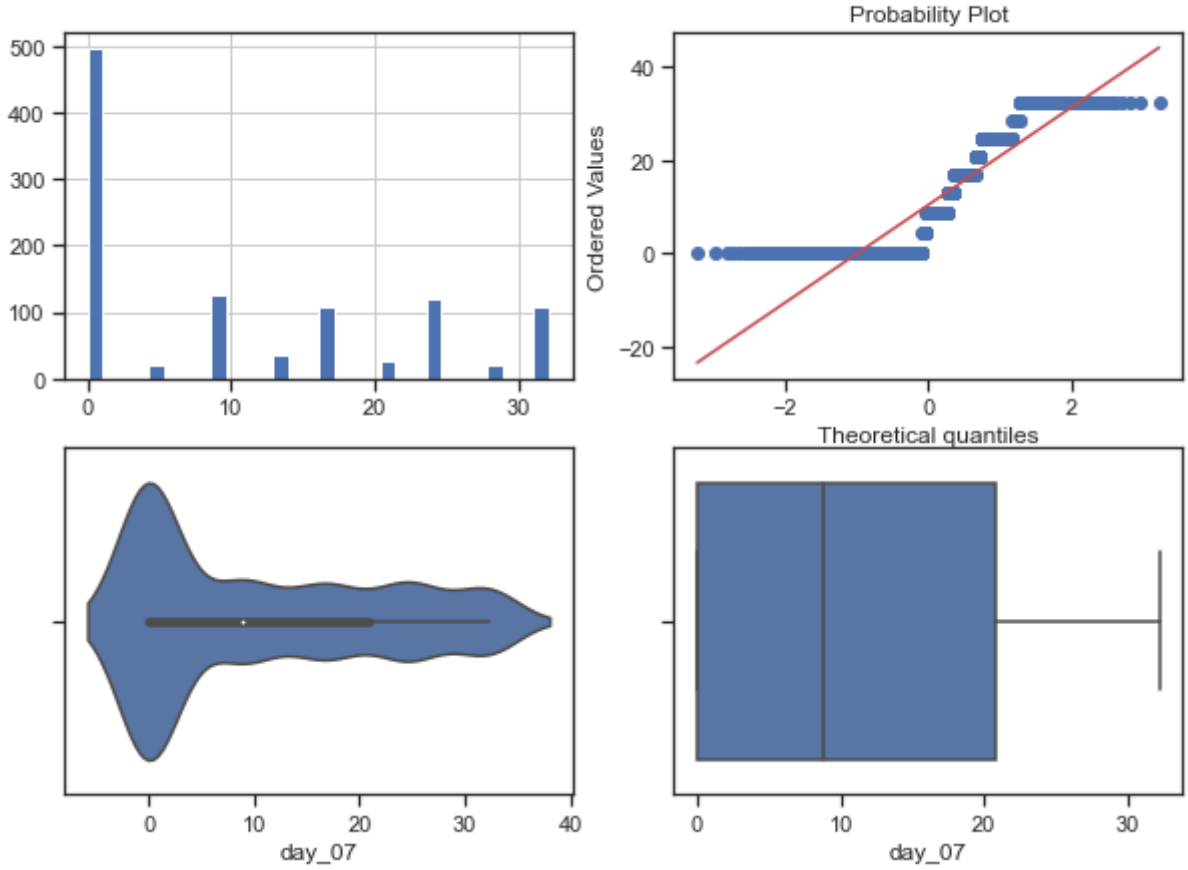
Поле-day_05, метод-OutlierBoundaryType.QUANTILE



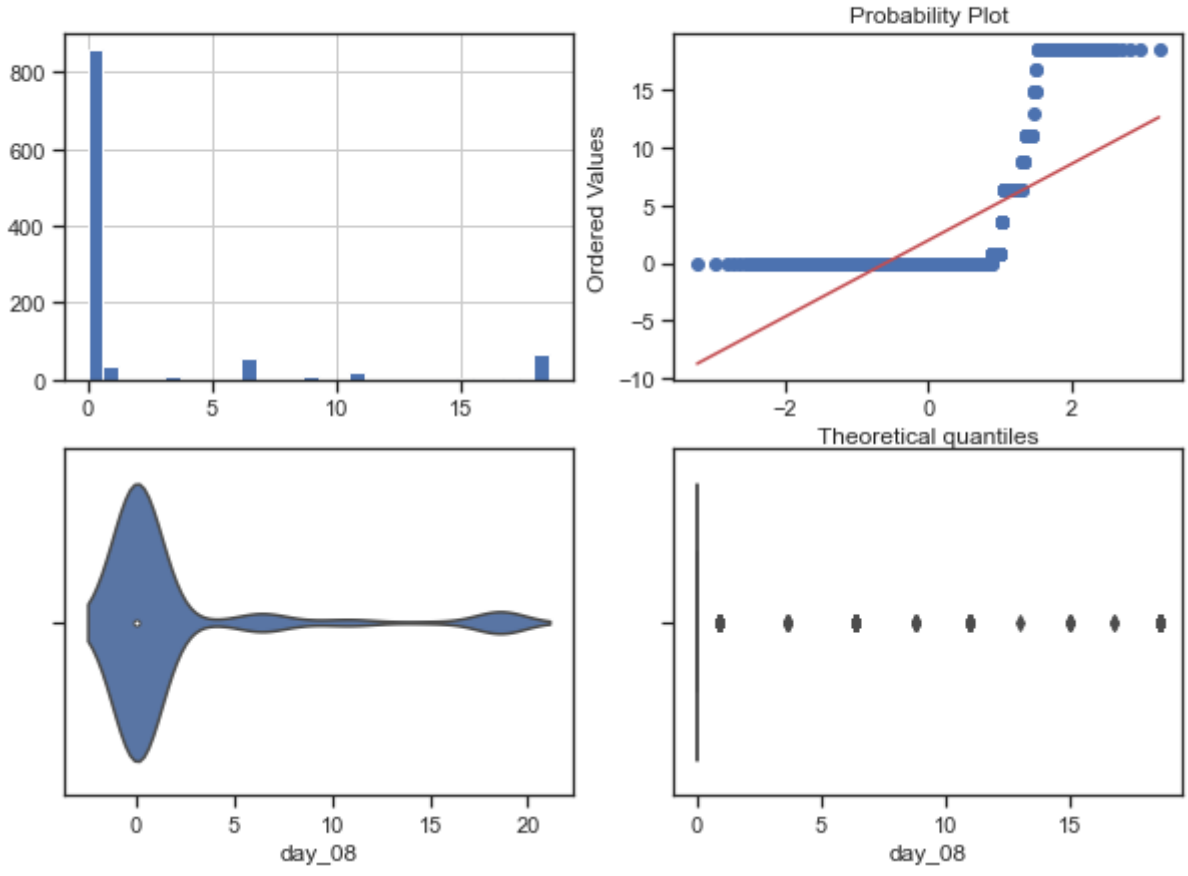
Поле-day_06, метод-OutlierBoundaryType.QUANTILE



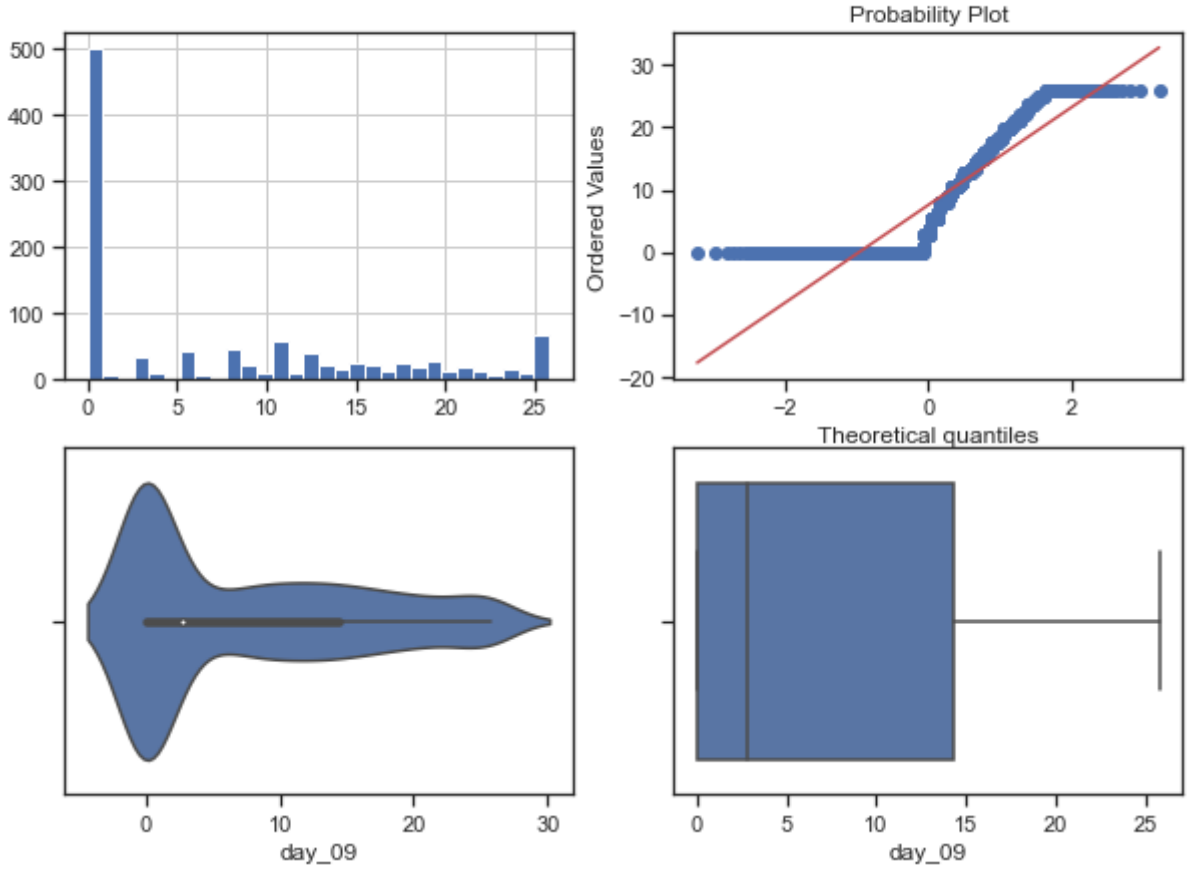
Поле-day_07, метод-OutlierBoundaryType.QUANTILE



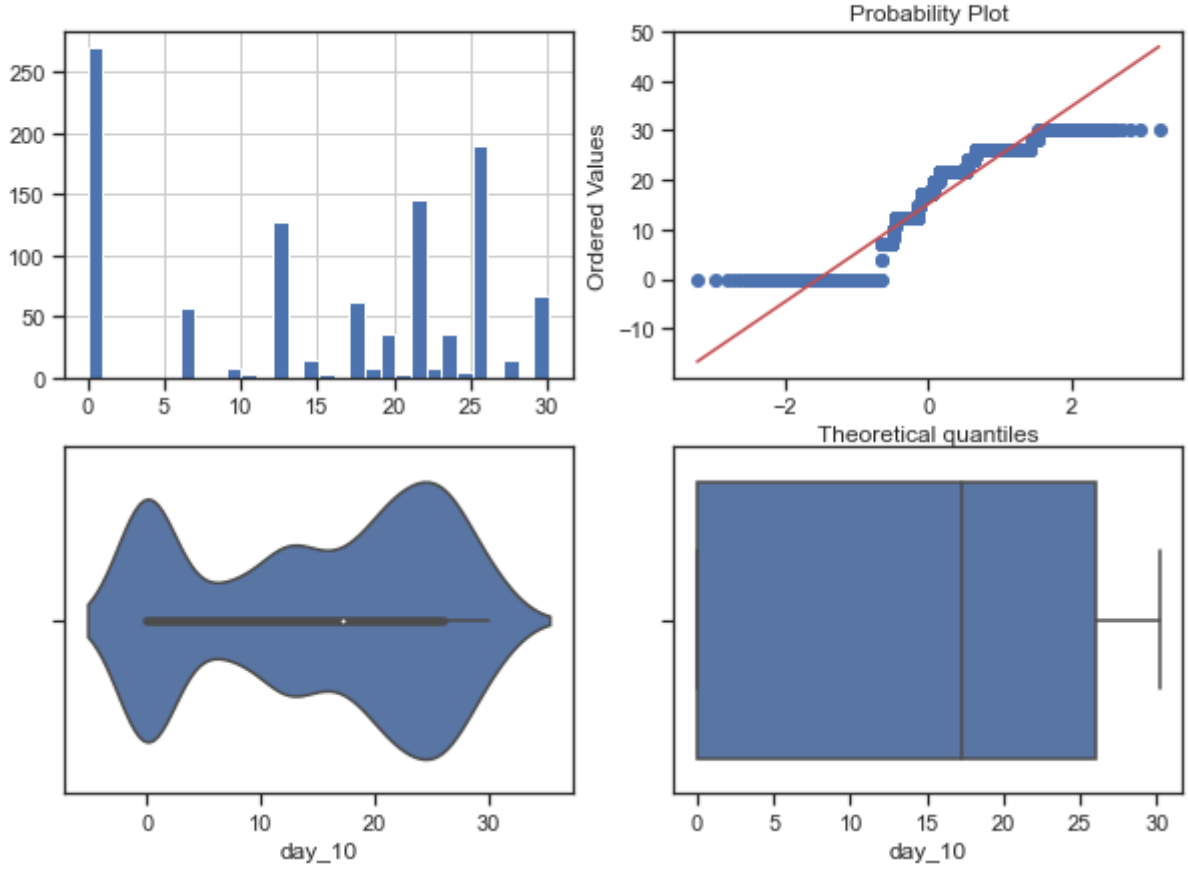
Поле-day_08, метод-OutlierBoundaryType.QUANTILE



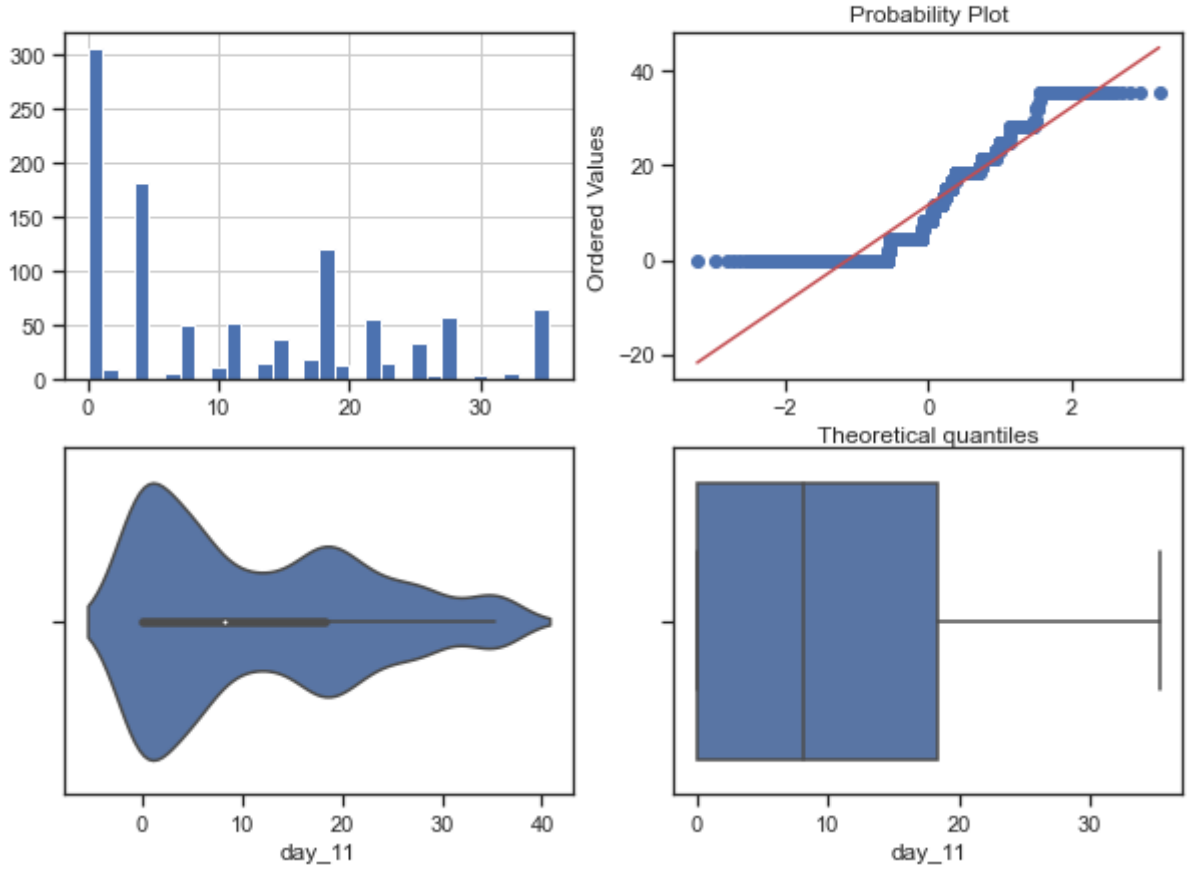
Поле-day_09, метод-OutlierBoundaryType.QUANTILE



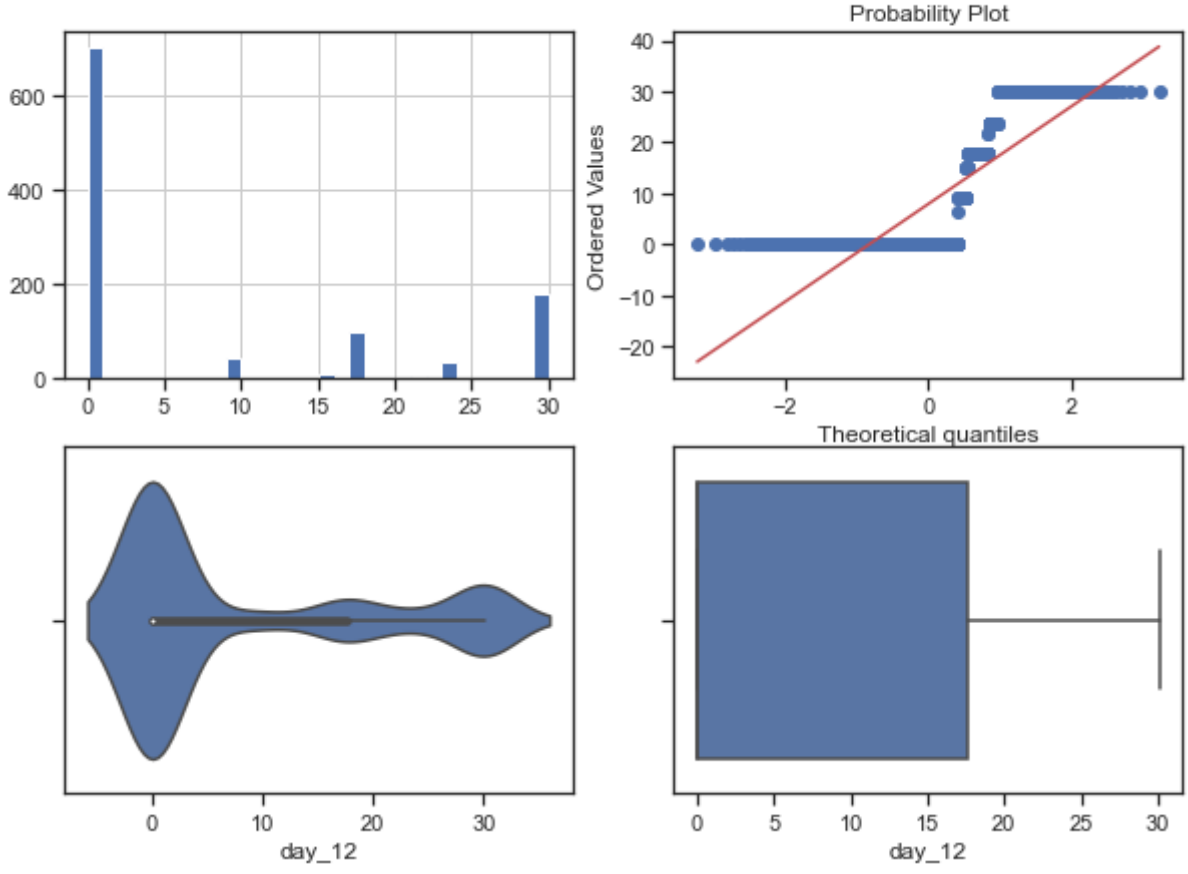
Поле-day_10, метод-OutlierBoundaryType.QUANTILE



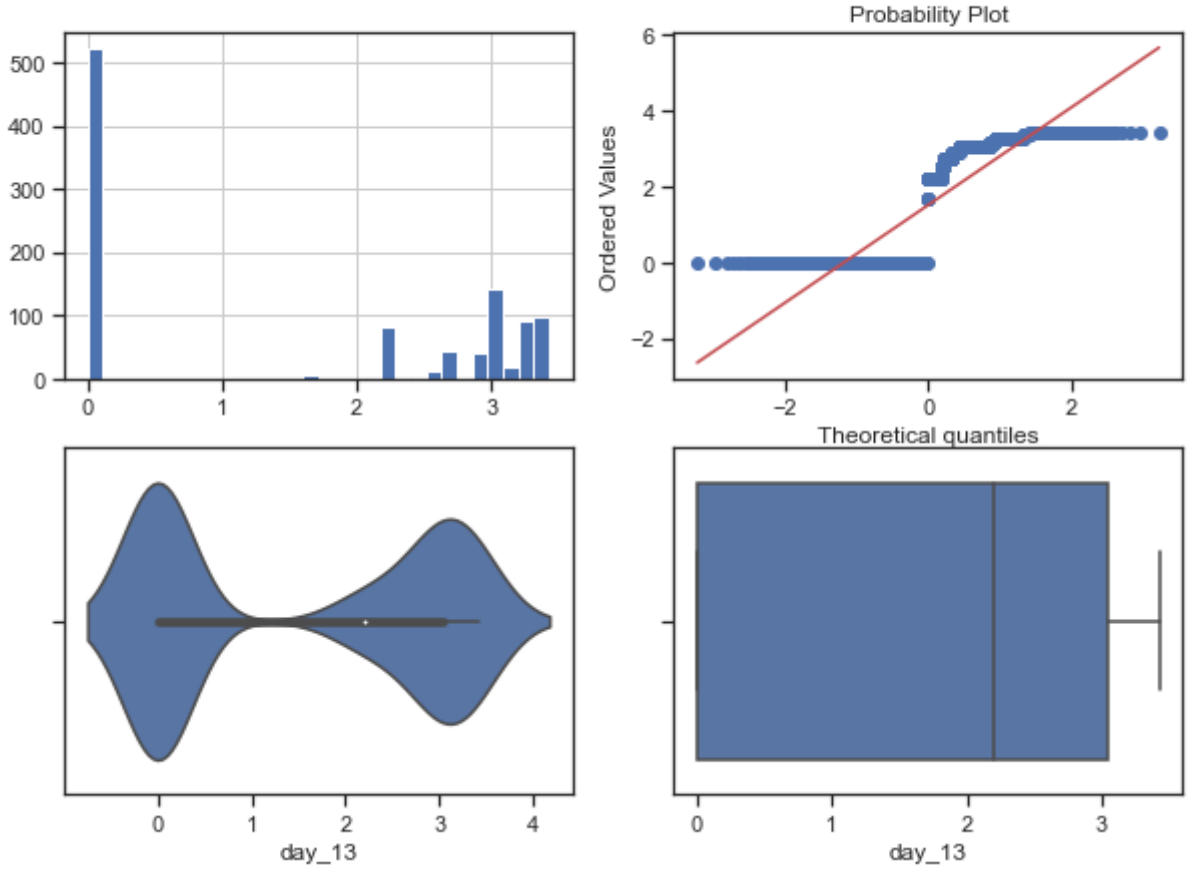
Поле-day_11, метод-OutlierBoundaryType.QUANTILE



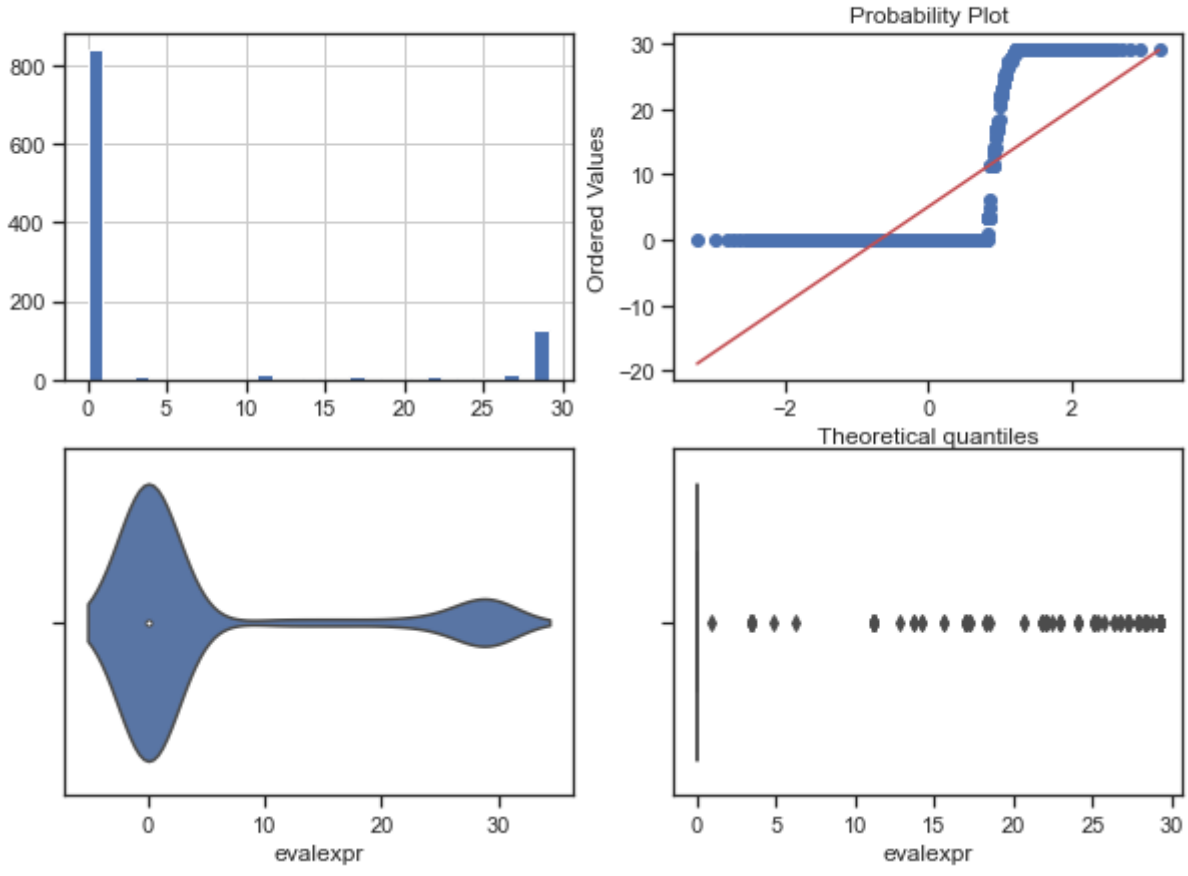
Поле-day_12, метод-OutlierBoundaryType.QUANTILE



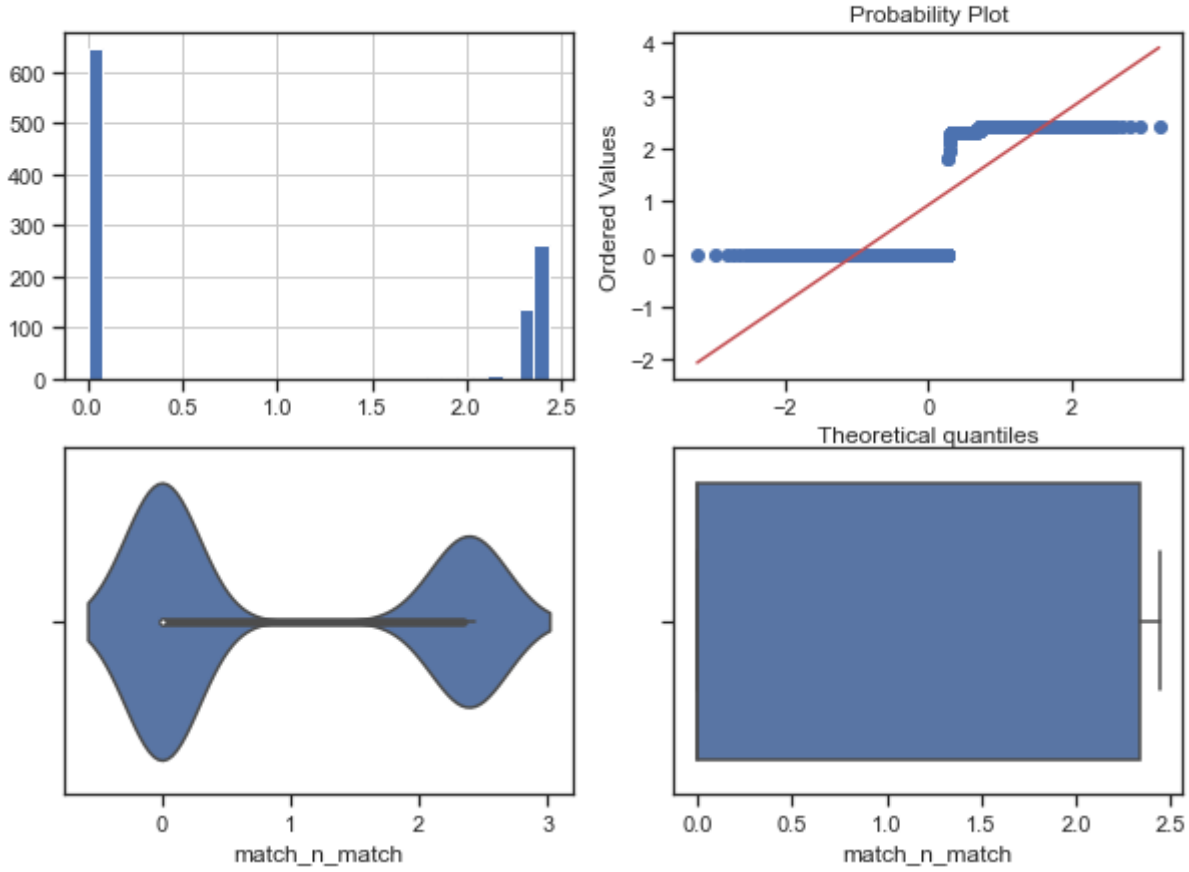
Поле-day_13, метод-OutlierBoundaryType.QUANTILE



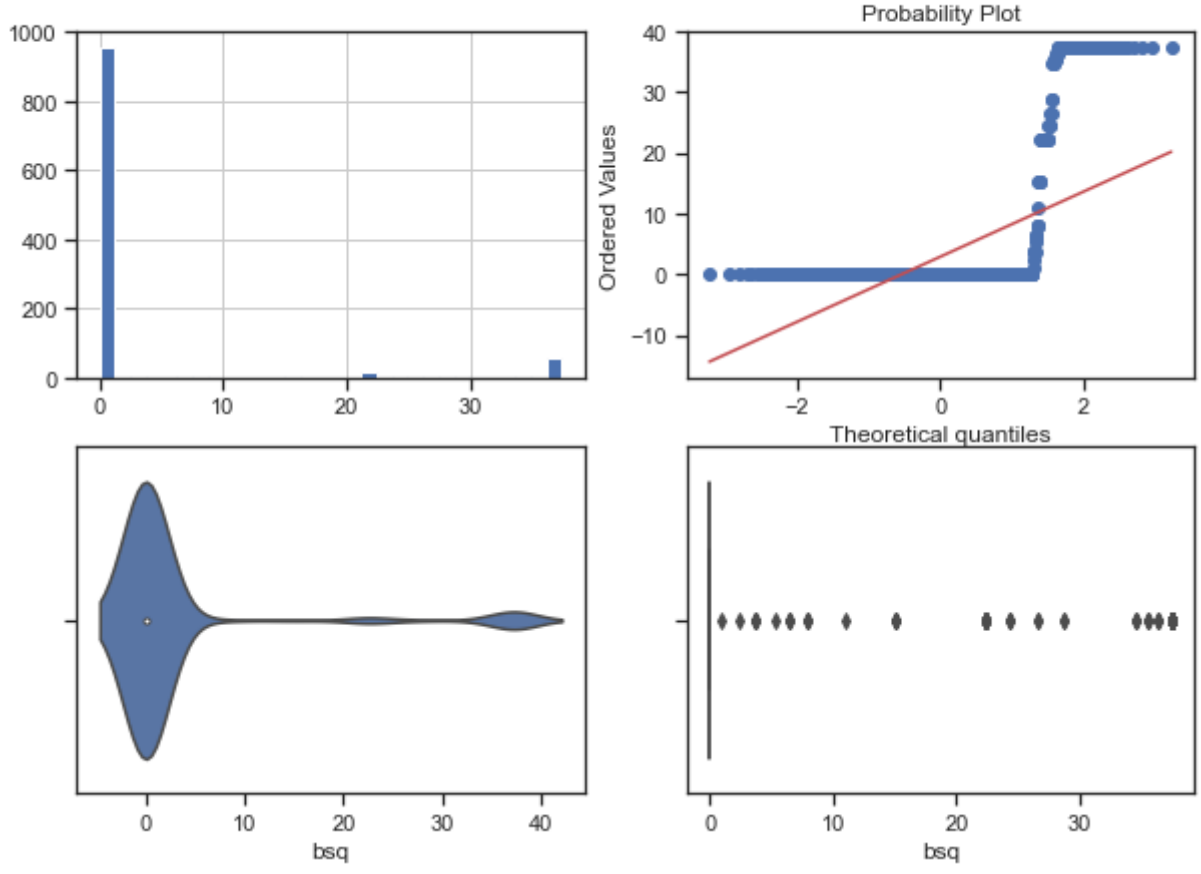
Поле-evalexpr, метод-OutlierBoundaryType.QUANTILE



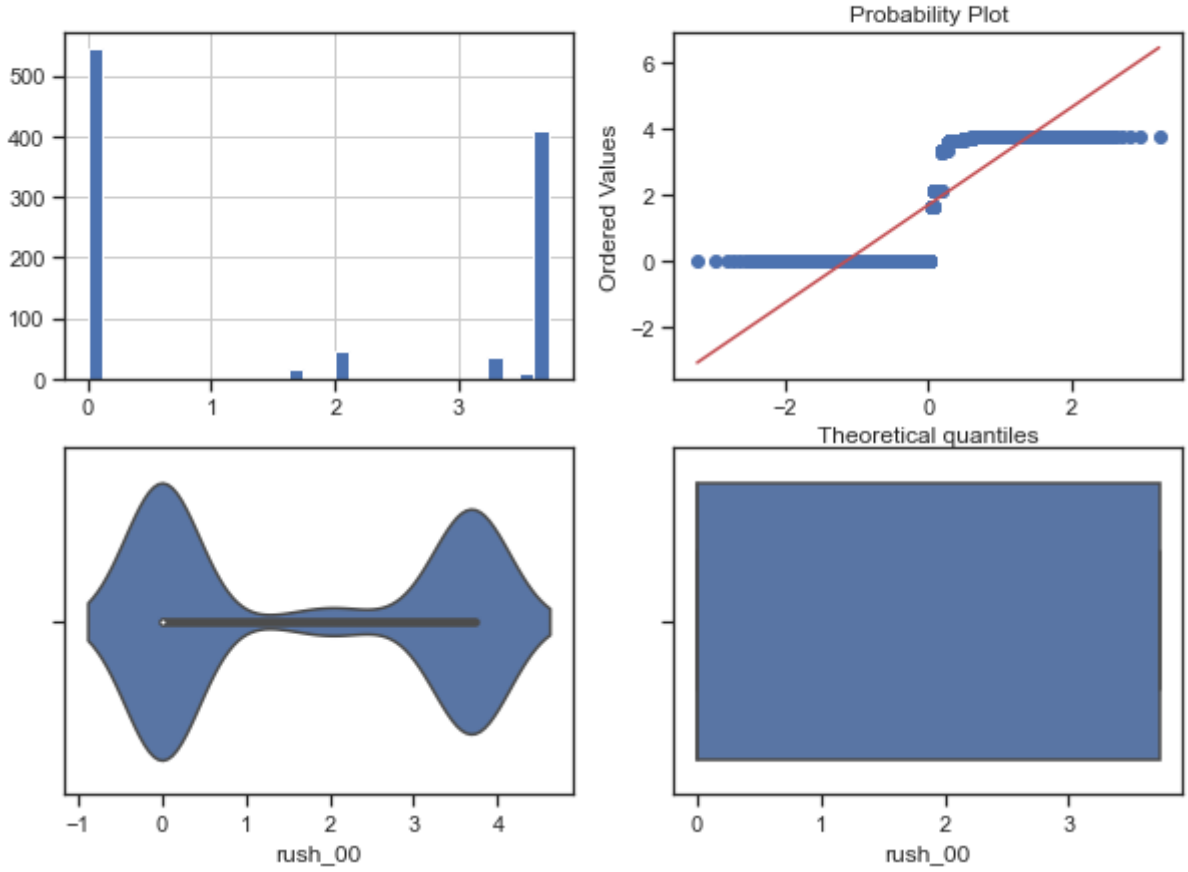
Поле-match_n_match, метод-OutlierBoundaryType.QUANTILE



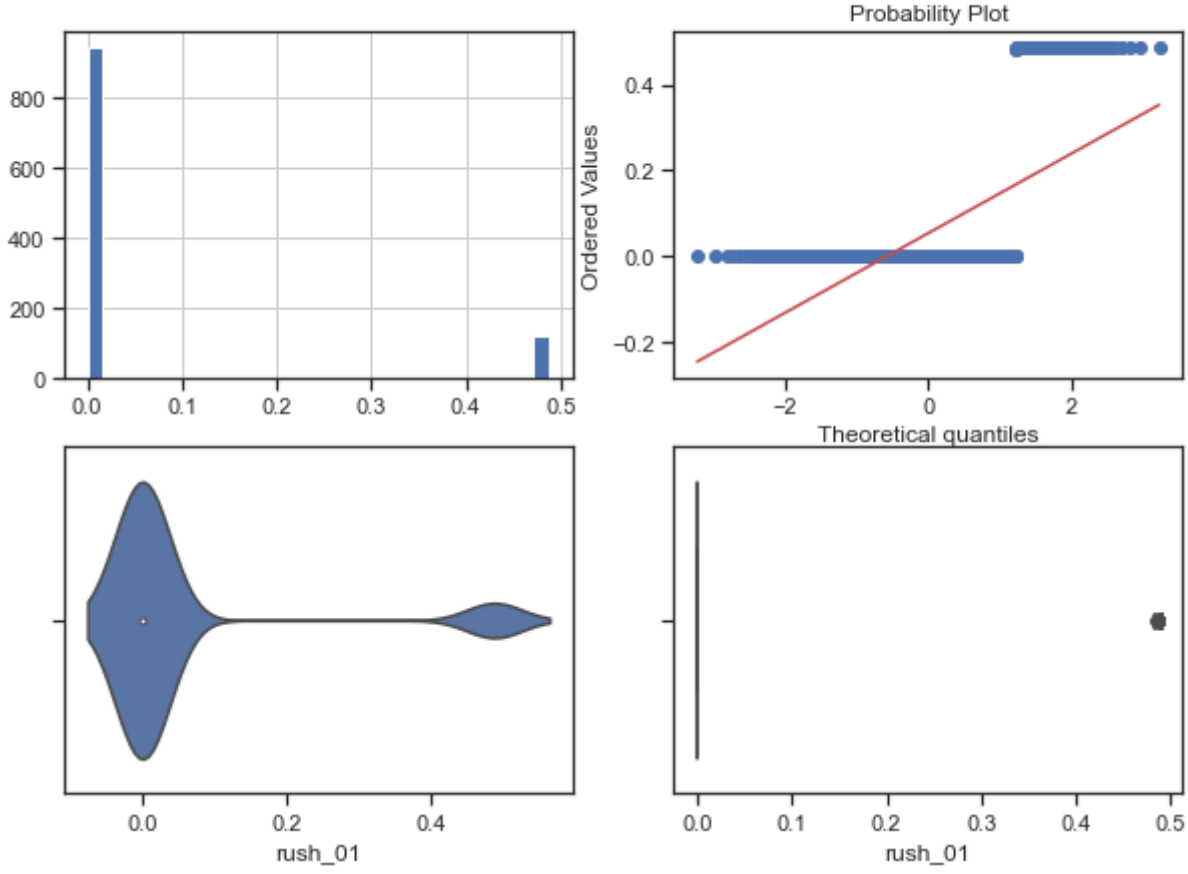
Поле-bsq, метод-OutlierBoundaryType.QUANTILE



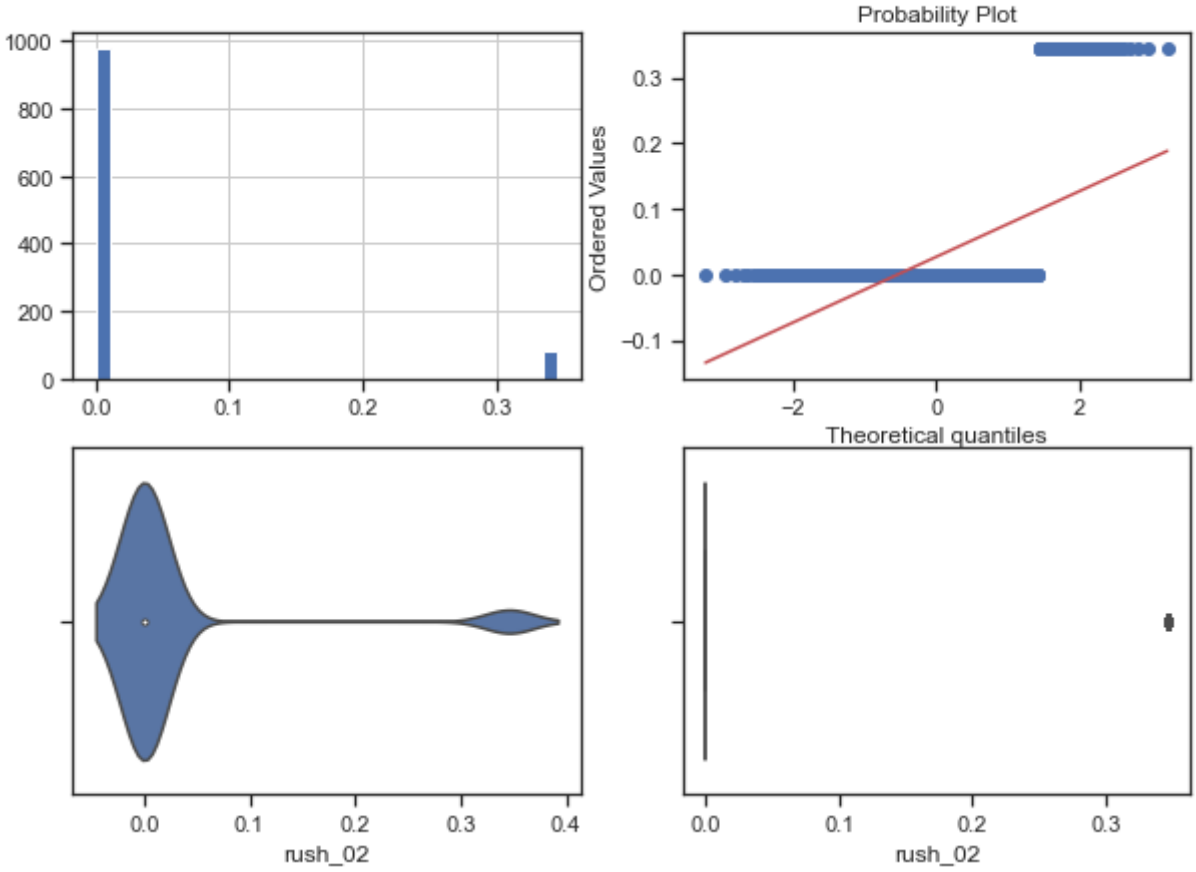
Поле-rush_00, метод-OutlierBoundaryType.QUANTILE



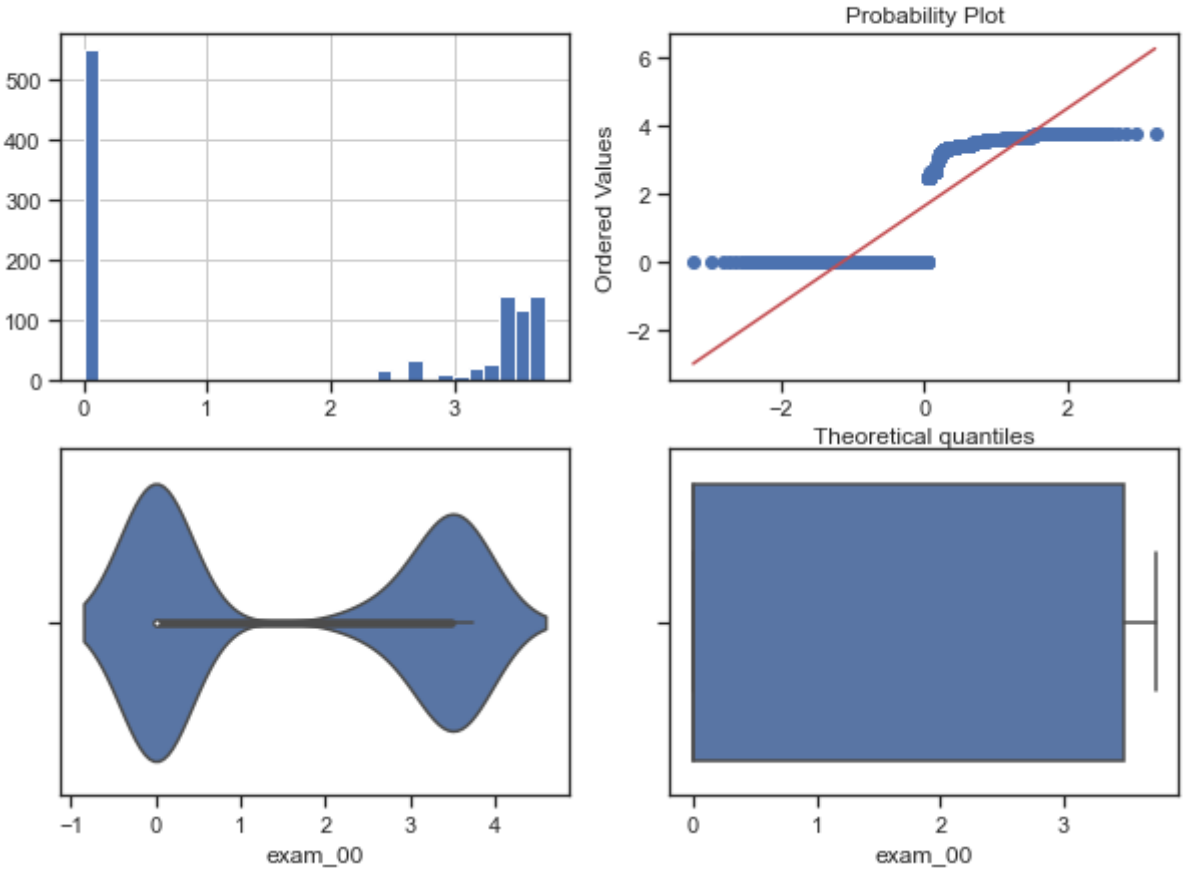
Поле-rush_01, метод-OutlierBoundaryType.QUANTILE



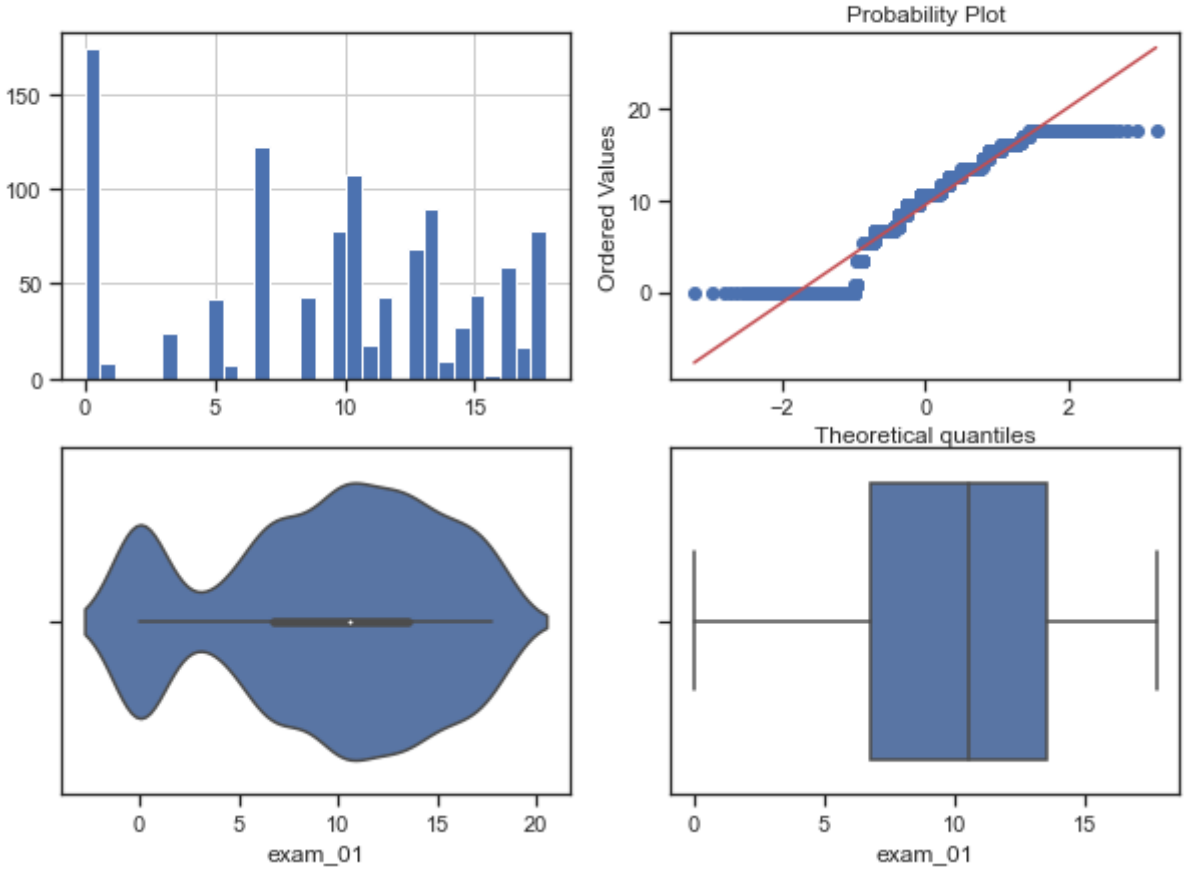
Поле-rush_02, метод-OutlierBoundaryType.QUANTILE



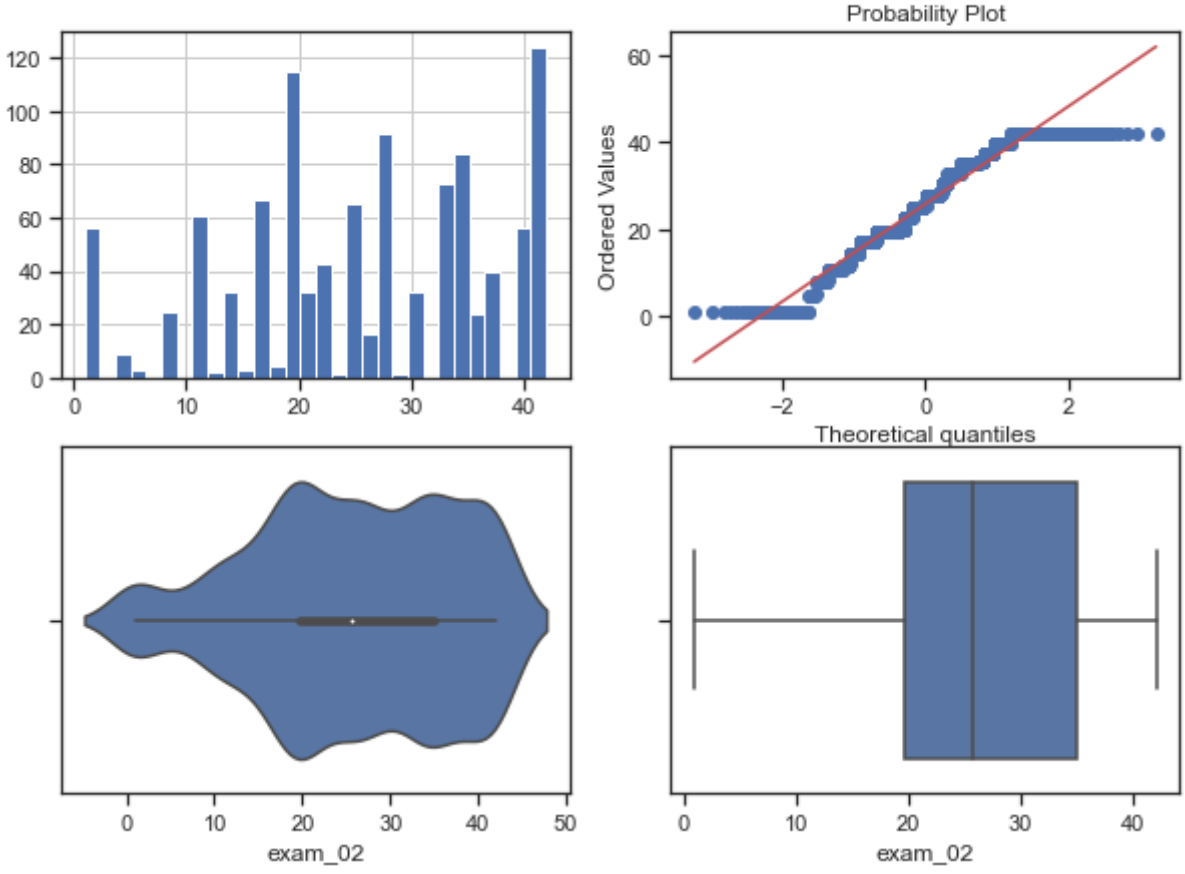
Поле-exam_00, метод-OutlierBoundaryType.QUANTILE



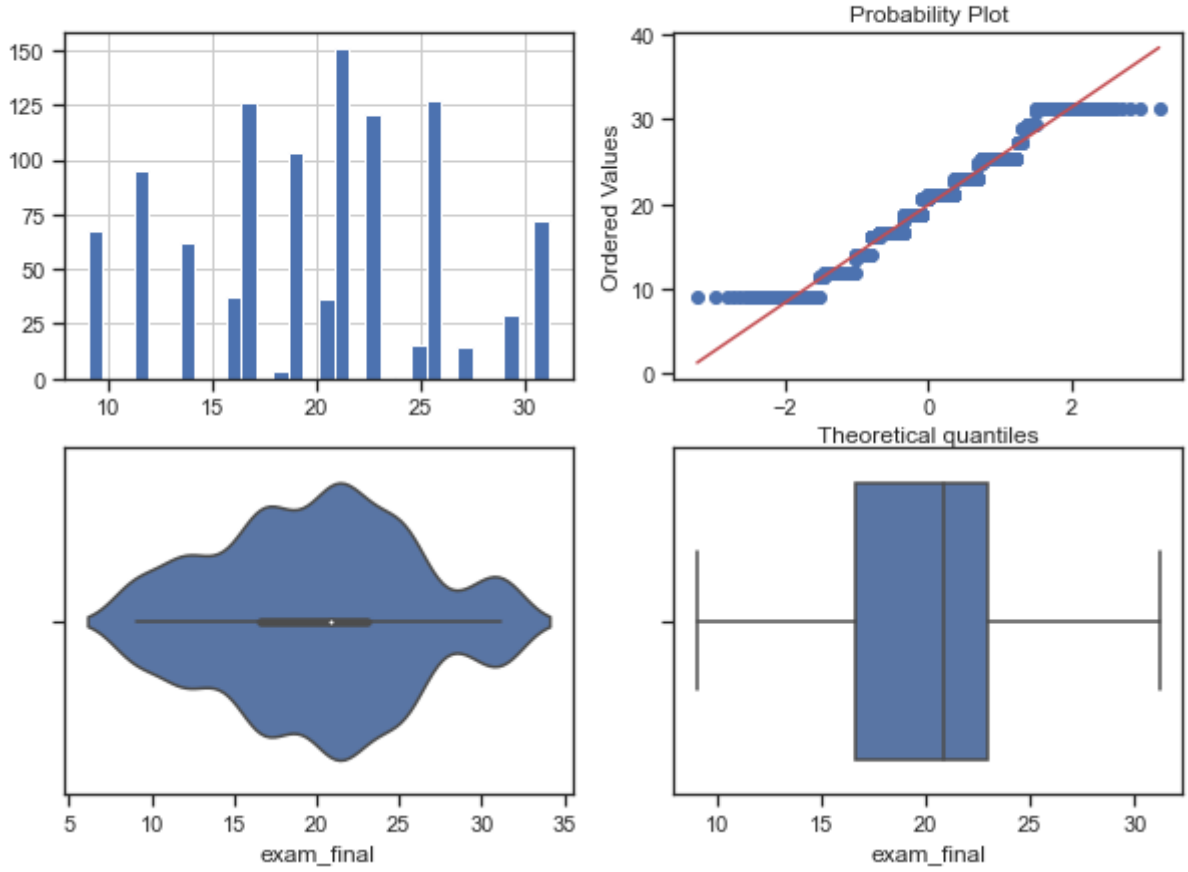
Поле-exam_01, метод-OutlierBoundaryType.QUANTILE



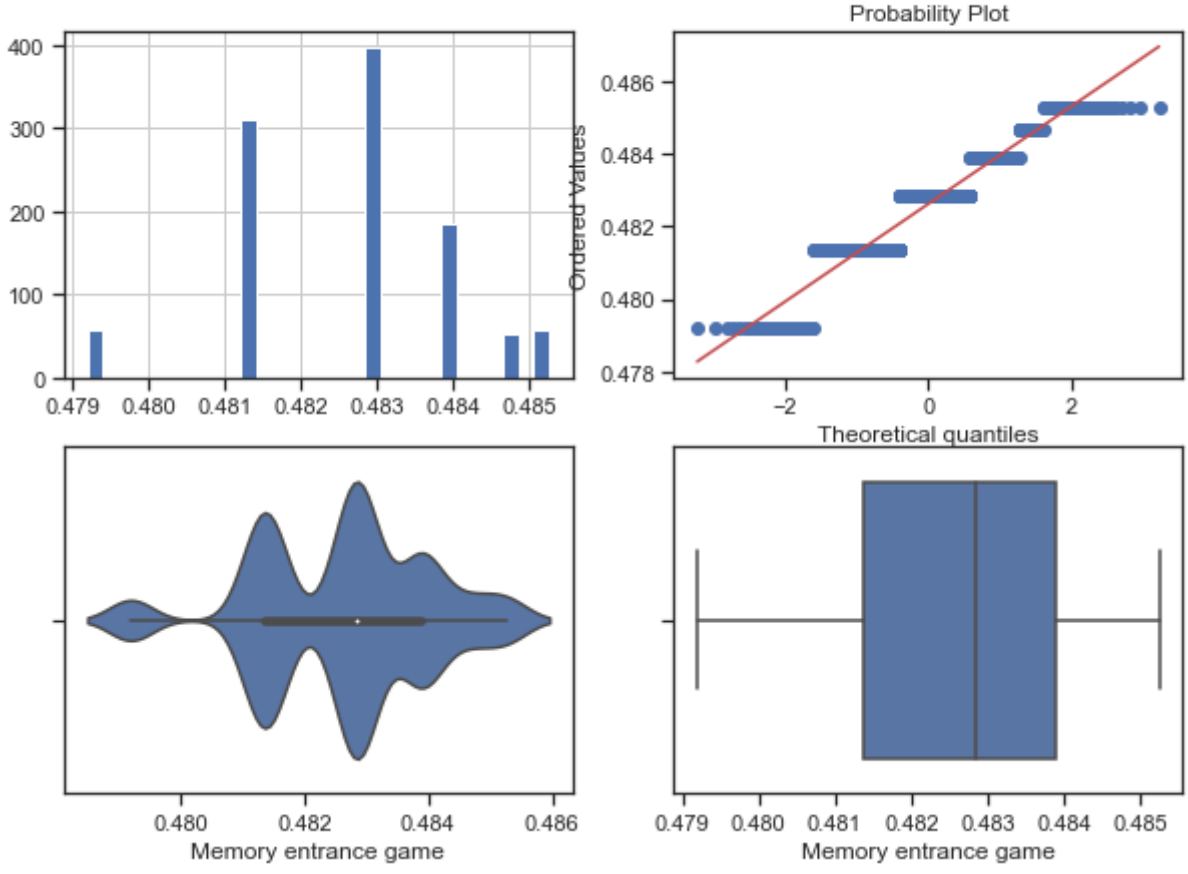
Поле-exam_02, метод-OutlierBoundaryType.QUANTILE

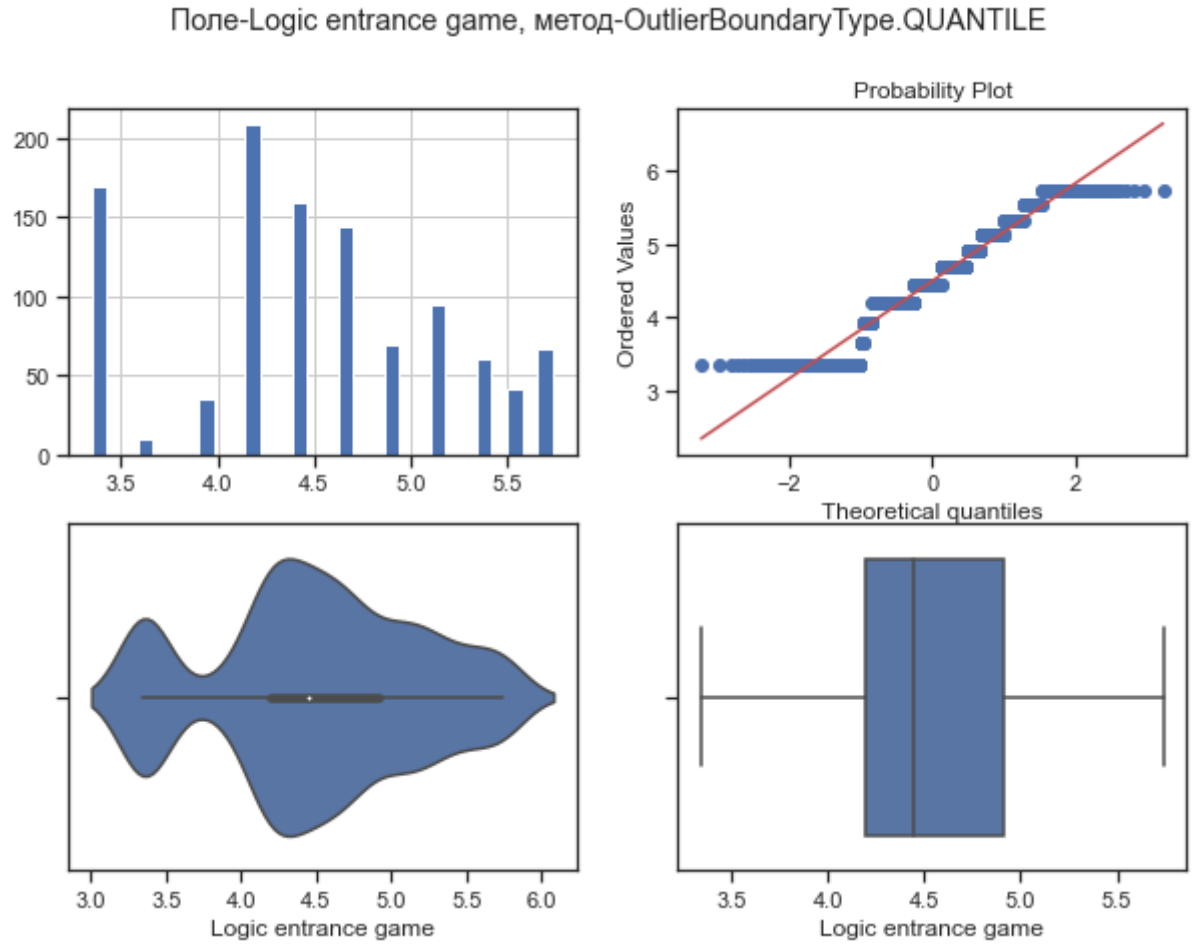


Поле-exam_final, метод-OutlierBoundaryType.QUANTILE



Поле-Memory entrance game, метод-OutlierBoundaryType.QUANTILE





In [38]:
```
data_outliers_deleted
```

Out[38]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
0	7.440813e+05	53.142349	10.509352	2.817426	4.535441	18.008774	1.988700	2.015112	-0.000(
1	2.110445e+06	19.312977	0.000000	0.971741	5.125305	24.773476	0.996627	0.000000	-0.000(
2	1.159231e+06	49.136182	0.000000	1.906271	4.215184	24.773476	1.988700	0.000000	-0.000(
3	2.487437e+06	66.153741	0.000000	0.971741	4.843920	24.773476	0.996627	2.015112	-0.000(
4	1.600657e+06	21.757480	10.509352	2.817426	2.950189	24.773476	0.000000	0.000000	-0.000(
...	...	...	...	...	...	...	...	...	
1055	4.977851e+06	64.783291	10.509352	2.817426	2.882900	32.602702	0.996627	2.808948	-0.000(
1056	2.610098e+06	55.346994	10.509352	2.817426	2.976363	4.063310	0.996627	2.015112	-0.000(
1057	4.284471e+06	43.153499	10.509352	1.906271	4.167804	4.063310	0.996627	4.317815	0.686(
1058	4.986330e+06	42.886603	10.509352	2.817426	4.076357	24.773476	0.996627	5.117555	-0.000(
1059	3.502103e+06	11.573992	10.509352	0.971741	2.800185	24.773476	0.996627	2.015112	-0.000(

1060 rows × 34 columns

Отбор признаков

Метод фильтрации

Метод, основынный на корреляции

In [39]:
```
cols_to_fs = data_outliers_deleted.columns
fs_data = data_outliers_deleted[cols_to_fs].copy()
fs_data.shape
```

Out[39]:
```
(1060, 34)
```

In [40]:
```
fs_data_features = list(zip(
[i for i in fs_data.columns],
zip(
    #типы колонок
    [str(i) for i in fs_data.dtypes],
    #проверка, есть ли пропущенные значения
    [i for i in fs_data.isnull().sum()]
)))
fs_data_features
```

Out[40]:
```
[('id', ('float64', 0)),
 ('Native city', ('float64', 0)),
 ('Gender', ('float64', 0)),
 ('Wave id', ('float64', 0)),
 ('Level', ('float64', 0)),
 ('Heard about school from', ('float64', 0)),
 ('Life status', ('float64', 0)),
 ('day_00', ('float64', 0)),
 ('day_01', ('float64', 0)),
 ('day_02', ('float64', 0)),
 ('day_03', ('float64', 0)),
 ('day_04', ('float64', 0)),
 ('day_05', ('float64', 0)),
 ('day_06', ('float64', 0)),
 ('day_07', ('float64', 0)),
 ('day_08', ('float64', 0)),
 ('day_09', ('float64', 0)),
 ('day_10', ('float64', 0)),
 ('day_11', ('float64', 0)),
 ('day_12', ('float64', 0)),
 ('day_13', ('float64', 0)),
 ('evalexpr', ('float64', 0)),
 ('match_n_match', ('float64', 0)),
 ('bsq', ('float64', 0)),
 ('rush_00', ('float64', 0)),
 ('rush_01', ('float64', 0)),
 ('rush_02', ('float64', 0)),
 ('exam_00', ('float64', 0)),
 ('exam_01', ('float64', 0)),
 ('exam_02', ('float64', 0)),
 ('exam_final', ('float64', 0)),
 ('Memory entrance game', ('float64', 0)),
 ('Logic entrance game', ('float64', 0)),
 ('contract_status', ('float64', 0))]
```

In [41]:
```
fs_data.tail()
```

Out[41]:

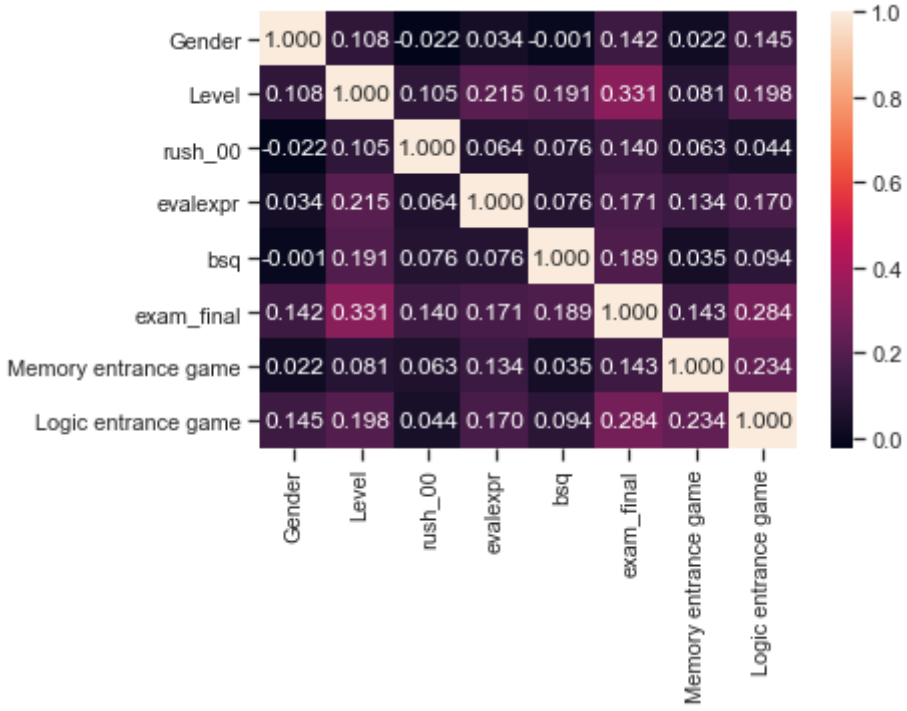
	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
1055	4.977851e+06	64.783291	10.509352	2.817426	2.882900	32.602702	0.996627	2.808948	-0.000(
1056	2.610098e+06	55.346994	10.509352	2.817426	2.976363	4.063310	0.996627	2.015112	-0.000(

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
1057	4.284471e+06	43.153499	10.509352	1.906271	4.167804	4.063310	0.996627	4.317815	0.686(
1058	4.986330e+06	42.886603	10.509352	2.817426	4.076357	24.773476	0.996627	5.117555	-0.000(
1059	3.502103e+06	11.573992	10.509352	0.971741	2.800185	24.773476	0.996627	2.015112	-0.000(

5 rows × 34 columns

In [42]:
```
heatmap_cols = ['Gender', 'Level', 'rush_00', 'evalexpr', 'bsq', 'exam_final',
        'Memory entrance game', 'Logic entrance game']
sns.heatmap(fs_data[heatmap_cols].corr(), annot=True, fmt='.3f')
```

Out[42]: <AxesSubplot:>



In [43]:
```
# Формирование DataFrame с сильными корреляциями
def make_corr_df(df):
    cr = df.corr() # !!!вот здесь был недочет - data.corr -> df.corr
    cr = cr.abs().unstack()
    cr = cr.sort_values(ascending=False)
    cr = cr[cr >= 0.3]
    cr = cr[cr < 1]
    cr = pd.DataFrame(cr).reset_index()
    cr.columns = ['f1', 'f2', 'corr']
    return cr
```

In [44]:
```
make_corr_df(fs_data)
```

Out[44]:

	f1	f2	corr
0	exam_02	exam_final	0.594318
1	exam_final	exam_02	0.594318
2	exam_01	exam_final	0.583808

	f1	f2	corr
3	exam_final	exam_01	0.583808
4	exam_01	exam_02	0.553440
5	exam_02	exam_01	0.553440
6	Wave id	Level	0.542252
7	Level	Wave id	0.542252
8	exam_01	exam_00	0.410065
9	exam_00	exam_01	0.410065
10	exam_00	exam_02	0.381366
11	exam_02	exam_00	0.381366
12	exam_final	exam_00	0.364109
13	exam_00	exam_final	0.364109
14	exam_final	day_03	0.355106
15	day_03	exam_final	0.355106
16	exam_02	day_03	0.354631
17	day_03	exam_02	0.354631
18	exam_01	day_03	0.348999
19	day_03	exam_01	0.348999
20	day_11	day_13	0.336402
21	day_13	day_11	0.336402
22	Level	contract_status	0.335277
23	contract_status	Level	0.335277
24	Level	exam_final	0.331369
25	exam_final	Level	0.331369
26	exam_final	day_11	0.330181
27	day_11	exam_final	0.330181
28	exam_02	Level	0.329313
29	Level	exam_02	0.329313
30	day_02	day_03	0.328753
31	day_03	day_02	0.328753
32	day_12	day_13	0.313258
33	day_13	day_12	0.313258
34	Heard about school from	Wave id	0.312613
35	Wave id	Heard about school from	0.312613
36	exam_final	day_13	0.311530
37	day_13	exam_final	0.311530
38	Level	exam_01	0.306495

	f1	f2	corr
39	exam_01	Level	0.306495
40	day_13	exam_02	0.302562
41	exam_02	day_13	0.302562

In [45]:
```
# Обнаружение групп коррелирующих признаков
def corr_groups(cr):
    grouped_feature_list = []
    correlated_groups = []

    for feature in cr['f1'].unique():
        if feature not in grouped_feature_list:
            # находим коррелирующие признаки
            correlated_block = cr[cr['f1'] == feature]
            cur_dups = list(correlated_block['f2'].unique()) + [feature]
            grouped_feature_list = grouped_feature_list + cur_dups
            correlated_groups.append(cur_dups)
    return correlated_groups
```

In [46]:
```
# Группы коррелирующих признаков
corr_groups(make_corr_df(fs_data))
```

Out[46]:
```
[['exam_final', 'exam_01', 'exam_00', 'day_03', 'Level', 'day_13', 'exam_02'],
 ['Level', 'Heard about school from', 'Wave id'],
 ['day_13', 'exam_final', 'day_11'],
 ['Level', 'contract_status'],
 ['day_03', 'day_02'],
 ['day_13', 'day_12']]
```

Метод, основанный на статистических характеристиках

In [47]:
```
from sklearn.feature_selection import mutual_info_classif, mutual_info_regression
from sklearn.feature_selection import SelectKBest, SelectPercentile
```

In [48]:
```
x = fs_data.drop('contract_status', axis=1)
x
```

Out[48]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
0	7.440813e+05	53.142349	10.509352	2.817426	4.535441	18.008774	1.988700	2.015112	-0.000(
1	2.110445e+06	19.312977	0.000000	0.971741	5.125305	24.773476	0.996627	0.000000	-0.000(
2	1.159231e+06	49.136182	0.000000	1.906271	4.215184	24.773476	1.988700	0.000000	-0.000(
3	2.487437e+06	66.153741	0.000000	0.971741	4.843920	24.773476	0.996627	2.015112	-0.000(
4	1.600657e+06	21.757480	10.509352	2.817426	2.950189	24.773476	0.000000	0.000000	-0.000(
...	...	...	...	...	...	...	...	...	
1055	4.977851e+06	64.783291	10.509352	2.817426	2.882900	32.602702	0.996627	2.808948	-0.000(
1056	2.610098e+06	55.346994	10.509352	2.817426	2.976363	4.063310	0.996627	2.015112	-0.000(
1057	4.284471e+06	43.153499	10.509352	1.906271	4.167804	4.063310	0.996627	4.317815	0.686(
1058	4.986330e+06	42.886603	10.509352	2.817426	4.076357	24.773476	0.996627	5.117555	-0.000(

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
1059	3.502103e+06	11.573992	10.509352	0.971741	2.800185	24.773476	0.996627	2.015112	-0.000(

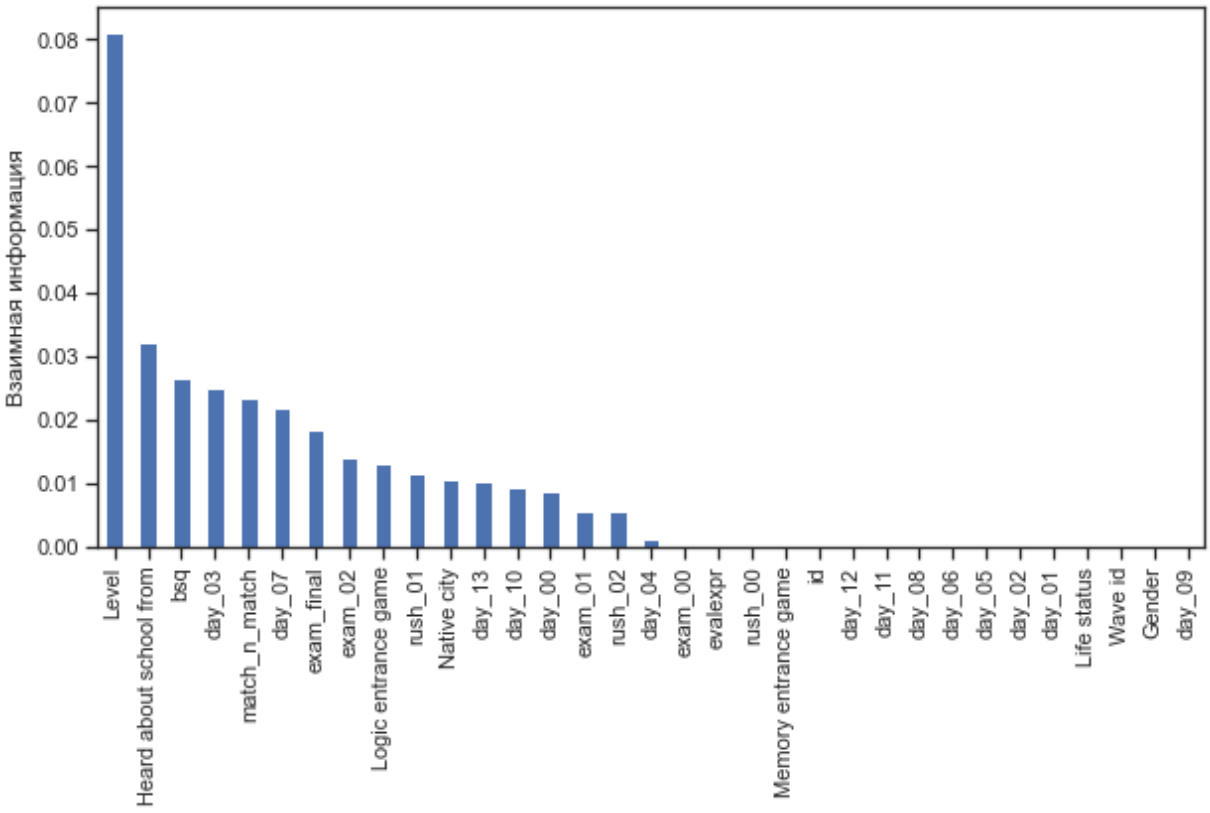
1060 rows × 33 columns

In [49]:
```
y = fs_data['contract_status']
y
```

Out[49]:
```
0       157.438327
1       157.438327
2       157.438327
3       157.438327
4       157.438327
           ...
1055    157.438327
1056    157.438327
1057    157.438327
1058    157.438327
1059      0.000000
Name: contract_status, Length: 1060, dtype: float64
```

In [50]:
```
mi = mutual_info_regression(x, y)
mi = pd.Series(mi)
mi.index = x.columns
mi.sort_values(ascending=False).plot.bar(figsize=(10,5))
plt.ylabel('Взаимная информация')
```

Out[50]:
```
Text(0, 0.5, 'Взаимная информация')
```



In [51]:
```
sel_mi = SelectKBest(mutual_info_regression, k=5).fit(x, y)

list(zip(x.columns, sel_mi.get_support()))
```

Out[51]:
```
[('id', False),
 ('Native city', False),
 ('Gender', False),
 ('Wave id', False),
 ('Level', True),
 ('Heard about school from', False),
 ('Life status', False),
 ('day_00', False),
 ('day_01', False),
 ('day_02', True),
 ('day_03', False),
 ('day_04', False),
 ('day_05', True),
 ('day_06', False),
 ('day_07', False),
 ('day_08', False),
 ('day_09', False),
 ('day_10', False),
 ('day_11', True),
 ('day_12', False),
 ('day_13', False),
 ('evalexpr', False),
 ('match_n_match', False),
 ('bsq', True),
 ('rush_00', False),
 ('rush_01', False),
 ('rush_02', False),
 ('exam_00', False),
 ('exam_01', False),
 ('exam_02', False),
 ('exam_final', False),
 ('Memory entrance game', False),
 ('Logic entrance game', False)]
```

Обучение моделей с различными вариантами

In [52]:
```
data_encoded['contract_status']
```

Out[52]:
```
0       1
1       1
2       1
3       1
4       1
       ..
1055    1
1056    1
1057    1
1058    1
1059    0
Name: contract_status, Length: 1060, dtype: int64
```

In [53]:
```
# данные с выполненными 1-4 пунктами предобработки
#(устранение пропусков в данных, кодирование категориальных признаков, нормализацию
#масштабирование признаков
data_cs11_scaled
```

Out[53]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_0
0	-1.696880	0.500799	0.538087	0.858850	0.697399	-0.134685	1.557304	-0.493445	-0.48663
1	-0.749732	-1.437178	-1.858435	-1.286862	1.195437	0.497119	-0.113976	-1.484201	-0.48663

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_0
2	-1.409103	0.271298	-1.858435	-0.200419	0.426997	0.497119	1.557304	-1.484201	-0.48663
3	-0.488406	1.246180	-1.858435	-1.286862	0.957856	0.497119	-0.113976	-0.493445	-0.48663
4	-1.103112	-1.297140	0.538087	0.858850	-0.641071	0.497119	-1.792927	-1.484201	-0.48663
...	...	...	...	...	...	...	...	...	
1055	1.237920	1.167671	0.538087	0.858850	-0.697885	1.228347	-0.113976	-0.103145	-0.48663
1056	-0.403379	0.627096	0.538087	0.858850	-0.618971	-1.437153	-0.113976	-0.493445	-0.48663
1057	0.757277	-0.071431	0.538087	-0.200419	0.386993	-1.437153	-0.113976	0.638709	2.06706
1058	1.243798	-0.086720	0.538087	0.858850	0.309782	0.497119	-0.113976	1.031912	-0.48663
1059	0.214949	-1.880520	0.538087	-1.286862	-0.767724	0.497119	-0.113976	-0.493445	-0.48663

1060 rows × 33 columns

In [54]:
```
# Разделим выборку на обучающую и тестовую
X_train_basic, X_test_basic, y_train_basic, y_test_basic = train_test_split(data_cs1
                                                    test_size=0.2,
                                                    random_state=1)
# Преобразуем массивы в DataFrame
X_train_basic_df = arr_to_df(X_train_basic)
X_test_basic_df = arr_to_df(X_test_basic)

X_train_basic_df.shape, X_test_basic_df.shape
```

Out[54]: ((848, 33), (212, 33))

In [55]:
```
# данные, с замененными выбросами
data_outliers_deleted
```

Out[55]:

	id	Native city	Gender	Wave id	Level	Heard about school from	Life status	day_00	day_
0	7.440813e+05	53.142349	10.509352	2.817426	4.535441	18.008774	1.988700	2.015112	-0.000(
1	2.110445e+06	19.312977	0.000000	0.971741	5.125305	24.773476	0.996627	0.000000	-0.000(
2	1.159231e+06	49.136182	0.000000	1.906271	4.215184	24.773476	1.988700	0.000000	-0.000(
3	2.487437e+06	66.153741	0.000000	0.971741	4.843920	24.773476	0.996627	2.015112	-0.000(
4	1.600657e+06	21.757480	10.509352	2.817426	2.950189	24.773476	0.000000	0.000000	-0.000(
...	...	...	...	...	...	...	...	...	
1055	4.977851e+06	64.783291	10.509352	2.817426	2.882900	32.602702	0.996627	2.808948	-0.000(
1056	2.610098e+06	55.346994	10.509352	2.817426	2.976363	4.063310	0.996627	2.015112	-0.000(
1057	4.284471e+06	43.153499	10.509352	1.906271	4.167804	4.063310	0.996627	4.317815	0.686(
1058	4.986330e+06	42.886603	10.509352	2.817426	4.076357	24.773476	0.996627	5.117555	-0.000(
1059	3.502103e+06	11.573992	10.509352	0.971741	2.800185	24.773476	0.996627	2.015112	-0.000(

1060 rows × 34 columns

In [56]:
```
# Разделим выборку на обучающую и тестовую
X_train_advanced, X_test_advanced, y_train_advanced, y_test_advanced = train_test_sp
                                                    data_encoded['contract_status'],
                                                    test_size=0.2,
                                                    random_state=1)
# Преобразуем массивы в DataFrame
X_train_advanced_df = arr_to_df(X_train_advanced)
X_test_advanced_df = arr_to_df(X_test_advanced)

X_train_advanced_df.shape, X_test_advanced_df.shape
```

Out[56]: ((848, 33), (212, 33))

In [57]:
```
class MetricLogger:

    def __init__(self):
        self.df = pd.DataFrame(
            {'metric': pd.Series([], dtype='str'),
            'alg': pd.Series([], dtype='str'),
            'value': pd.Series([], dtype='float')})

    def add(self, metric, alg, value):
        """
        Добавление значения
        """
        # Удаление значения если оно уже было ранее добавлено
        self.df.drop(self.df[(self.df['metric']==metric)&(self.df['alg']==alg)].inde
        # Добавление нового значения
        temp = [{'metric':metric, 'alg':alg, 'value':value}]
        self.df = self.df.append(temp, ignore_index=True)

    def get_data_for_metric(self, metric, ascending=True):
        """
        Формирование данных с фильтром по метрике
        """
        temp_data = self.df[self.df['metric']==metric]
        temp_data_2 = temp_data.sort_values(by='value', ascending=ascending)
        return temp_data_2['alg'].values, temp_data_2['value'].values

    def plot(self, str_header, metric, ascending=True, figsize=(5, 5)):
        """
        Вывод графика
        """
        array_labels, array_metric = self.get_data_for_metric(metric, ascending)
        fig, ax1 = plt.subplots(figsize=figsize)
        pos = np.arange(len(array_metric))
        rects = ax1.barh(pos, array_metric,
                         align='center',
                         height=0.5,
                         tick_label=array_labels)
        ax1.set_title(str_header)
        for a,b in zip(pos, array_metric):
            plt.text(0.5, a-0.05, str(round(b,3)), color='white')
        plt.show()
```

In [58]:
```
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
```

In [59]:
```
clas_models_dict = {'LinR': LinearRegression(),
                    'SVR': SVR(),
                    'KNN_5':KNeighborsRegressor(n_neighbors=5),
                    'Tree':DecisionTreeRegressor(random_state=1),
                    'GB': GradientBoostingRegressor(random_state=1),
                    'RF':RandomForestRegressor(n_estimators=50, random_state=1)}
```

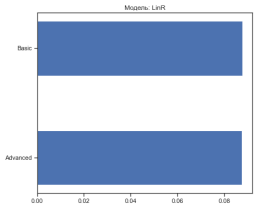
In [60]:
```
X_data_dict = {'Basic': (X_train_basic_df, X_test_basic_df),
               'Advanced': (X_train_advanced_df, X_test_advanced_df)}
```

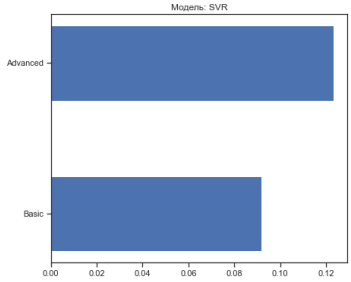
In [61]:
```
def test_models(clas_models_dict, X_data_dict, y_train, y_test):

    logger = MetricLogger()

    for model_name, model in clas_models_dict.items():

        for data_name, data_tuple in X_data_dict.items():

            X_train, X_test = data_tuple

            model.fit(X_train, y_train)
            y_pred = model.predict(X_test)
            mse = mean_squared_error(y_test, y_pred)
            logger.add(model_name, data_name, mse)

    return logger
```

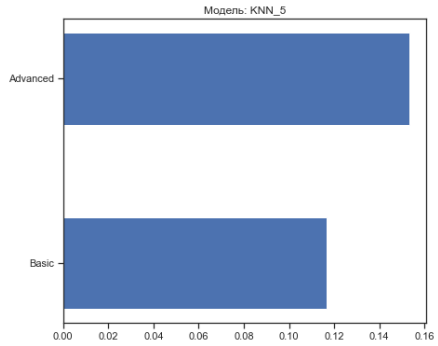
In [62]:
```
%%time
logger = test_models(clas_models_dict, X_data_dict, y_train_basic, y_test_basic)
```

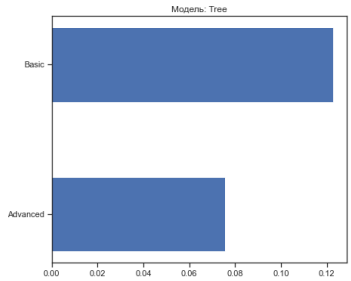
```
Wall time: 1.11 s
```

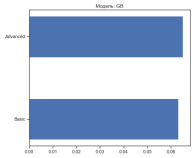
In [63]:
```
# Построим графики метрик качества модели
for model in clas_models_dict:
    logger.plot('Модель: ' + model, model, figsize=(7, 6))
```

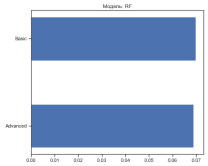












Модель с использованием AutoML библиотеки mljar

In [64]:
```
!pip install --user mljar-supervised
!pip install delayed
```

```
Requirement already satisfied: mljar-supervised in c:\users\masha\anaconda3\lib\site
-packages (0.10.4)
Requirement already satisfied: joblib==1.0.1 in c:\users\masha\anaconda3\lib\site-pa
ckages (from mljar-supervised) (1.0.1)
Requirement already satisfied: pandas==1.2.0 in c:\users\masha\anaconda3\lib\site-pa
ckages (from mljar-supervised) (1.2.0)
Requirement already satisfied: optuna==2.7.0 in c:\users\masha\anaconda3\lib\site-pa
ckages (from mljar-supervised) (2.7.0)
Requirement already satisfied: numpy>=1.20.0 in c:\users\masha\anaconda3\lib\site-pa
ckages (from mljar-supervised) (1.20.3)
Requirement already satisfied: category-encoders==2.2.2 in c:\users\masha\anaconda3
\lib\site-packages (from mljar-supervised) (2.2.2)
Requirement already satisfied: seaborn==0.11.1 in c:\users\masha\anaconda3\lib\site-
packages (from mljar-supervised) (0.11.1)
Requirement already satisfied: scipy==1.6.1 in c:\users\masha\anaconda3\lib\site-pac
kages (from mljar-supervised) (1.6.1)
Requirement already satisfied: lightgbm==3.0.0 in c:\users\masha\anaconda3\lib\site-
packages (from mljar-supervised) (3.0.0)
Requirement already satisfied: scikit-plot==0.3.7 in c:\users\masha\anaconda3\lib\si
te-packages (from mljar-supervised) (0.3.7)
Requirement already satisfied: xgboost==1.3.3 in c:\users\masha\anaconda3\lib\site-p
ackages (from mljar-supervised) (1.3.3)
Requirement already satisfied: pyarrow>=2.0.0 in c:\users\masha\anaconda3\lib\site-p
ackages (from mljar-supervised) (4.0.0)
Requirement already satisfied: shap==0.36.0 in c:\users\masha\anaconda3\lib\site-pac
kages (from mljar-supervised) (0.36.0)
Requirement already satisfied: wordcloud==1.8.1 in c:\users\masha\anaconda3\lib\site
```

```
-packages (from mljar-supervised) (1.8.1)
Requirement already satisfied: dtreeviz==1.3 in c:\users\masha\anaconda3\lib\site-pa
ckages (from mljar-supervised) (1.3)
Requirement already satisfied: markdown in c:\users\masha\anaconda3\lib\site-package
s (from mljar-supervised) (3.3.4)
Requirement already satisfied: matplotlib>=3.2.2 in c:\users\masha\anaconda3\lib\sit
e-packages (from mljar-supervised) (3.3.2)
Requirement already satisfied: catboost==0.24.4 in c:\users\masha\anaconda3\lib\site
-packages (from mljar-supervised) (0.24.4)
Requirement already satisfied: cloudpickle==1.3.0 in c:\users\masha\anaconda3\lib\si
te-packages (from mljar-supervised) (1.3.0)
Requirement already satisfied: tabulate==0.8.7 in c:\users\masha\anaconda3\lib\site-
packages (from mljar-supervised) (0.8.7)
Requirement already satisfied: scikit-learn==0.24.2 in c:\users\masha\anaconda3\lib
\site-packages (from mljar-supervised) (0.24.2)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\masha\anaconda3\li
b\site-packages (from pandas==1.2.0->mljar-supervised) (2.8.1)
Requirement already satisfied: pytz>=2017.3 in c:\users\masha\anaconda3\lib\site-pac
kages (from pandas==1.2.0->mljar-supervised) (2020.1)
Requirement already satisfied: colorlog in c:\users\masha\anaconda3\lib\site-package
s (from optuna==2.7.0->mljar-supervised) (5.0.1)
Requirement already satisfied: alembic in c:\users\masha\anaconda3\lib\site-packages
(from optuna==2.7.0->mljar-supervised) (1.6.2)
Requirement already satisfied: cliff in c:\users\masha\anaconda3\lib\site-packages
(from optuna==2.7.0->mljar-supervised) (3.7.0)
Requirement already satisfied: tqdm in c:\users\masha\anaconda3\lib\site-packages (f
rom optuna==2.7.0->mljar-supervised) (4.50.2)
Requirement already satisfied: cmaes>=0.8.2 in c:\users\masha\anaconda3\lib\site-pac
kages (from optuna==2.7.0->mljar-supervised) (0.8.2)
Requirement already satisfied: sqlalchemy>=1.1.0 in c:\users\masha\anaconda3\lib\sit
e-packages (from optuna==2.7.0->mljar-supervised) (1.3.20)
Requirement already satisfied: packaging>=20.0 in c:\users\masha\anaconda3\lib\site-
packages (from optuna==2.7.0->mljar-supervised) (20.4)
Requirement already satisfied: statsmodels>=0.9.0 in c:\users\masha\anaconda3\lib\si
te-packages (from category-encoders==2.2.2->mljar-supervised) (0.12.0)
Requirement already satisfied: patsy>=0.5.1 in c:\users\masha\anaconda3\lib\site-pac
kages (from category-encoders==2.2.2->mljar-supervised) (0.5.1)
Requirement already satisfied: numba in c:\users\masha\anaconda3\lib\site-packages
(from shap==0.36.0->mljar-supervised) (0.51.2)
Requirement already satisfied: slicer in c:\users\masha\anaconda3\lib\site-packages
(from shap==0.36.0->mljar-supervised) (0.0.7)
Requirement already satisfied: pillow in c:\users\masha\anaconda3\lib\site-packages
(from wordcloud==1.8.1->mljar-supervised) (8.0.1)
Requirement already satisfied: colour in c:\users\masha\anaconda3\lib\site-packages
(from dtreeviz==1.3->mljar-supervised) (0.1.5)
Requirement already satisfied: graphviz>=0.9 in c:\users\masha\anaconda3\lib\site-pa
ckages (from dtreeviz==1.3->mljar-supervised) (0.16)
Requirement already satisfied: pytest in c:\users\masha\anaconda3\lib\site-packages
(from dtreeviz==1.3->mljar-supervised) (0.0.0)
Requirement already satisfied: cycler>=0.10 in c:\users\masha\anaconda3\lib\site-pac
kages (from matplotlib>=3.2.2->mljar-supervised) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\masha\anaconda3\lib\sit
e-packages (from matplotlib>=3.2.2->mljar-supervised) (1.3.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users
\masha\anaconda3\lib\site-packages (from matplotlib>=3.2.2->mljar-supervised) (2.4.
7)
Requirement already satisfied: certifi>=2020.06.20 in c:\users\masha\anaconda3\lib\s
ite-packages (from matplotlib>=3.2.2->mljar-supervised) (2020.6.20)
Requirement already satisfied: six in c:\users\masha\anaconda3\lib\site-packages (fr
om catboost==0.24.4->mljar-supervised) (1.15.0)
Requirement already satisfied: plotly in c:\users\masha\anaconda3\lib\site-packages
(from catboost==0.24.4->mljar-supervised) (4.14.3)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\masha\anaconda3\lib
\site-packages (from scikit-learn==0.24.2->mljar-supervised) (2.1.0)
Requirement already satisfied: colorama; sys_platform == "win32" in c:\users\masha\a
naconda3\lib\site-packages (from colorlog->optuna==2.7.0->mljar-supervised) (0.4.4)
Requirement already satisfied: Mako in c:\users\masha\anaconda3\lib\site-packages (f
rom alembic->optuna==2.7.0->mljar-supervised) (1.1.4)
Requirement already satisfied: python-editor>=0.3 in c:\users\masha\anaconda3\lib\si
```

```
te-packages (from alembic->optuna==2.7.0->mljar-supervised) (1.0.4)
Requirement already satisfied: PyYAML>=3.12 in c:\users\masha\anaconda3\lib\site-pac
kages (from cliff->optuna==2.7.0->mljar-supervised) (5.3.1)
Requirement already satisfied: stevedore>=2.0.1 in c:\users\masha\anaconda3\lib\site
-packages (from cliff->optuna==2.7.0->mljar-supervised) (3.3.0)
Requirement already satisfied: pbr!=2.1.0,>=2.0.0 in c:\users\masha\anaconda3\lib\si
te-packages (from cliff->optuna==2.7.0->mljar-supervised) (5.6.0)
Requirement already satisfied: cmd2>=1.0.0 in c:\users\masha\anaconda3\lib\site-pack
ages (from cliff->optuna==2.7.0->mljar-supervised) (1.5.0)
Requirement already satisfied: PrettyTable>=0.7.2 in c:\users\masha\anaconda3\lib\si
te-packages (from cliff->optuna==2.7.0->mljar-supervised) (2.1.0)
Requirement already satisfied: llvmlite<0.35,>=0.34.0.dev0 in c:\users\masha\anacond
a3\lib\site-packages (from numba->shap==0.36.0->mljar-supervised) (0.34.0)
Requirement already satisfied: setuptools in c:\users\masha\anaconda3\lib\site-packa
ges (from numba->shap==0.36.0->mljar-supervised) (50.3.1.post20201107)
Requirement already satisfied: attrs>=17.4.0 in c:\users\masha\anaconda3\lib\site-pa
ckages (from pytest->dtreeviz==1.3->mljar-supervised) (20.3.0)
Requirement already satisfied: iniconfig in c:\users\masha\anaconda3\lib\site-packag
es (from pytest->dtreeviz==1.3->mljar-supervised) (1.1.1)
Requirement already satisfied: pluggy<1.0,>=0.12 in c:\users\masha\anaconda3\lib\sit
e-packages (from pytest->dtreeviz==1.3->mljar-supervised) (0.13.1)
Requirement already satisfied: py>=1.8.2 in c:\users\masha\anaconda3\lib\site-packag
es (from pytest->dtreeviz==1.3->mljar-supervised) (1.9.0)
Requirement already satisfied: toml in c:\users\masha\anaconda3\lib\site-packages (f
rom pytest->dtreeviz==1.3->mljar-supervised) (0.10.1)
Requirement already satisfied: atomicwrites>=1.0 in c:\users\masha\anaconda3\lib\sit
e-packages (from pytest->dtreeviz==1.3->mljar-supervised) (1.4.0)
Requirement already satisfied: retrying>=1.3.3 in c:\users\masha\anaconda3\lib\site-
packages (from plotly->catboost==0.24.4->mljar-supervised) (1.3.3)
Requirement already satisfied: MarkupSafe>=0.9.2 in c:\users\masha\anaconda3\lib\sit
e-packages (from Mako->alembic->optuna==2.7.0->mljar-supervised) (1.1.1)
Requirement already satisfied: wcwidth>=0.1.7 in c:\users\masha\anaconda3\lib\site-p
ackages (from cmd2>=1.0.0->cliff->optuna==2.7.0->mljar-supervised) (0.2.5)
Requirement already satisfied: pyreadline3; sys_platform == "win32" and python_versi
on >= "3.8" in c:\users\masha\anaconda3\lib\site-packages (from cmd2>=1.0.0->cliff->
optuna==2.7.0->mljar-supervised) (3.3)
Requirement already satisfied: pyperclip>=1.6 in c:\users\masha\anaconda3\lib\site-p
ackages (from cmd2>=1.0.0->cliff->optuna==2.7.0->mljar-supervised) (1.8.2)
Requirement already satisfied: delayed in c:\users\masha\anaconda3\lib\site-packages
(0.11.0b1)
Requirement already satisfied: hiredis in c:\users\masha\anaconda3\lib\site-packages
(from delayed) (2.0.0)
Requirement already satisfied: redis in c:\users\masha\anaconda3\lib\site-packages
(from delayed) (3.5.3)
```

In [65]:
```
#from supervised import AutoML
import sklearn
from sklearn.model_selection import train_test_split
from supervised.automl import AutoML
```

In [66]:
```
train = data_loaded
```

In [67]:
```
automl = AutoML()
```

In [68]:
```
automl.fit(train[train.columns[:-1]], train["contract_status"])
```

```
AutoML directory: AutoML_1
The task is binary_classification with evaluation metric logloss
AutoML will use algorithms: ['Baseline', 'Linear', 'Decision Tree', 'Random Forest',
'Xgboost', 'Neural Network']
AutoML will ensemble availabe models
2021-05-16 20:58:37,906 supervised.preprocessing.eda ERROR There was an issue when r
unning EDA. 'charmap' codec can't encode characters in position 11-16: character map
s to <undefined>
AutoML steps: ['simple_algorithms', 'default_algorithms', 'ensemble']
* Step simple_algorithms will try to check up to 3 models
1_Baseline logloss 0.39034 trained in 0.38 seconds
```

```
2_DecisionTree logloss 0.361809 trained in 15.63 seconds
3_Linear logloss 0.24329 trained in 3.41 seconds
* Step default_algorithms will try to check up to 3 models
4_Default_Xgboost logloss 0.239248 trained in 7.1 seconds
5_Default_NeuralNetwork logloss 0.314716 trained in 1.37 seconds
6_Default_RandomForest logloss 0.25748 trained in 4.36 seconds
* Step ensemble will try to check up to 1 model
Ensemble logloss 0.213427 trained in 0.59 seconds
An input array is constant; the correlation coefficent is not defined.
AutoML fit time: 50.29 seconds
AutoML best model: Ensemble
```

Out[68]:
```
AutoML()
```

In [71]:
```
automl_prepared_data = AutoML()
```

In [72]:
```
automl_prepared_data.fit(data_cs11_scaled, data_encoded["contract_status"])
```

```
AutoML directory: AutoML_2
The task is binary_classification with evaluation metric logloss
AutoML will use algorithms: ['Baseline', 'Linear', 'Decision Tree', 'Random Forest',
'Xgboost', 'Neural Network']
AutoML will ensemble availabe models
AutoML steps: ['simple_algorithms', 'default_algorithms', 'ensemble']
* Step simple_algorithms will try to check up to 3 models
1_Baseline logloss 0.39034 trained in 0.41 seconds
2_DecisionTree logloss 0.361809 trained in 5.03 seconds
3_Linear logloss 0.266026 trained in 3.73 seconds
* Step default_algorithms will try to check up to 3 models
4_Default_Xgboost logloss 0.244262 trained in 4.92 seconds
5_Default_NeuralNetwork logloss 0.563712 trained in 1.38 seconds
6_Default_RandomForest logloss 0.265742 trained in 4.64 seconds
* Step ensemble will try to check up to 1 model
Ensemble logloss 0.225113 trained in 0.64 seconds
An input array is constant; the correlation coefficent is not defined.
AutoML fit time: 38.4 seconds
AutoML best model: Ensemble
```

Out[72]:
```
AutoML()
```