Калашников Артем ИУ5-22М РК1

Вариант №5

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

Задача 5

```
data = pd.read_csv('online_store_customer_data.csv', sep=",")

data.head()
```

```
  Transaction_date  Transaction_ID  Gender   Age Marital_status
State_names  \
0         1/1/2019          151200  Female  19.0         Single
Kansas
1         1/1/2019          151201    Male  49.0         Single
Illinois
2         1/1/2019          151202    Male  63.0        Married   New
Mexico
3         1/1/2019          151203     NaN  18.0         Single
Virginia
4         1/1/2019          151204    Male  27.0         Single
Connecticut

    Segment Employees_status Payment_method  Referal  Amount_spent
0     Basic     Unemployment          Other      1.0       2051.36
1     Basic    self-employed           Card      0.0        544.04
2     Basic          workers         PayPal      1.0       1572.60
3  Platinum          workers           Card      1.0       1199.79
4     Basic    self-employed           Card      0.0           NaN
```

```
data_features = list(zip(
# признаки
[i for i in data.columns],
zip(
    # типы колонок
    [str(i) for i in data.dtypes],
    # проверим есть ли пропущенные значения
    [i for i in data.isnull().sum()]
)))
# Признаки с типом данных и количеством пропусков
data_features
```

```
[('Transaction_date', ('object', 0)),
 ('Transaction_ID', ('int64', 0)),
```

```
 ('Gender', ('object', 28)),
 ('Age', ('float64', 42)),
 ('Marital_status', ('object', 0)),
 ('State_names', ('object', 0)),
 ('Segment', ('object', 0)),
 ('Employees_status', ('object', 26)),
 ('Payment_method', ('object', 0)),
 ('Referal', ('float64', 155)),
 ('Amount_spent', ('float64', 242))]
```

```
pd.get_dummies(data[['Employees_status']]).head()
```

```
   Employees_status_Employees  Employees_status_Unemployment  \
0                           0                              1
1                           0                              0
2                           0                              0
3                           0                              0
4                           0                              0

   Employees_status_self-employed  Employees_status_workers
0                               0                         0
1                               1                         0
2                               0                         1
3                               0                         1
4                               1                         0
```

Задача 25

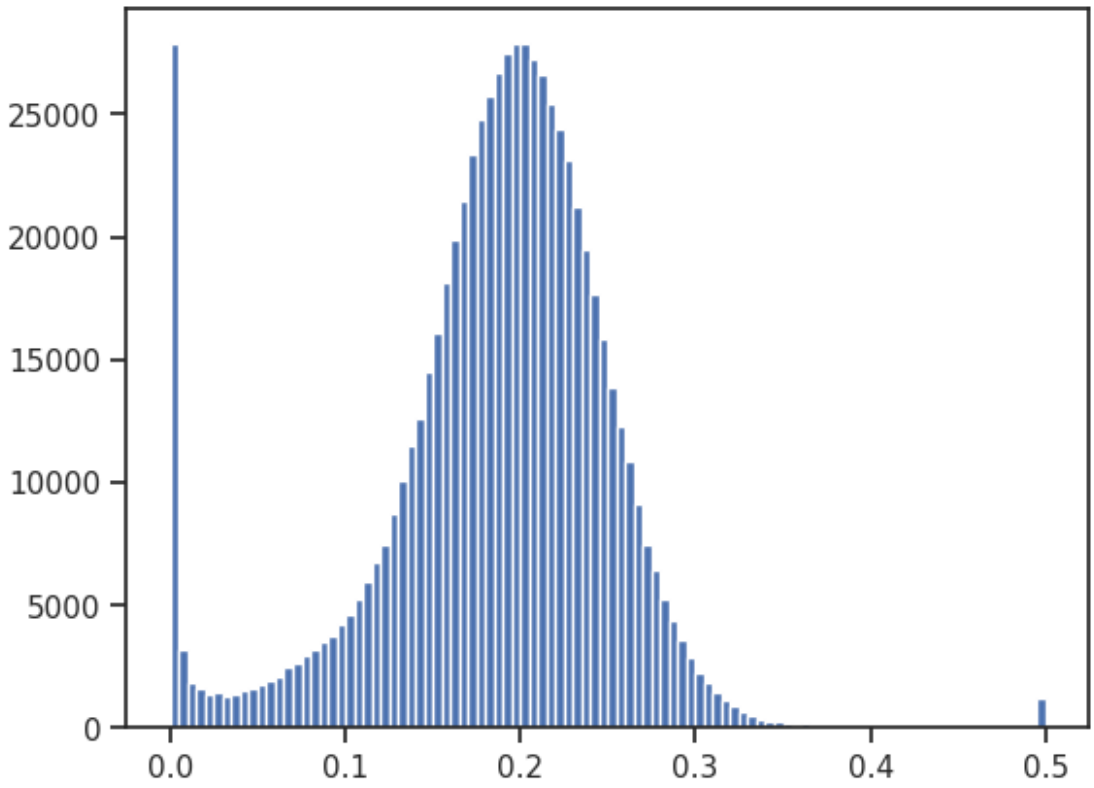
```
data2 = pd.read_csv('bikes.csv', sep=',')
data2 = data2.dropna(axis=0, how='any')
display(data2.shape)
data2.head()
```

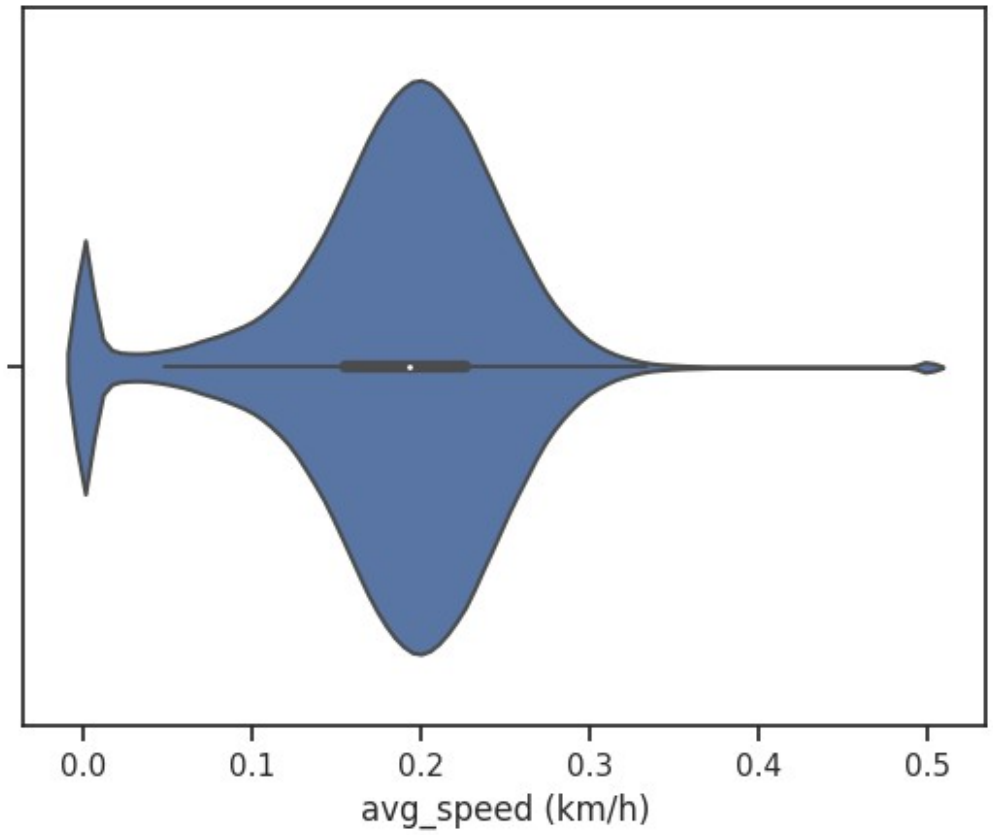
```
(705495, 14)
```

```
             departure               return  departure_id
departure_name  \
0  2020-03-23 06:09:44  2020-03-23 06:16:26            86
Kuusitie
1  2020-03-23 06:11:58  2020-03-23 06:26:31            26
Kamppi (M)
2  2020-03-23 06:16:29  2020-03-23 06:24:23           268  Porolahden
koulu
3  2020-03-23 06:33:53  2020-03-23 07:14:03           751
Vallipolku
4  2020-03-23 06:36:09  2020-03-23 07:04:10            62
Länsisatamankatu

   return_id       return_name  distance (m)  duration (sec.)  \
0      111.0     Esterinportti        1747.0            401.0
1       10.0       Kasarmitori        1447.0            869.0
2      254.0       Agnetankuja        1772.0            469.0
```

```
3      106.0   Korppaanmäentie        7456.0           2406.0
4      121.0  Vilhonvuorenkatu        7120.0           1679.0

   avg_speed (km/h)  departure_latitude  departure_longitude
return_latitude  \
0          0.261397           60.195245            24.901900
60.197572
1          0.099908           60.168610            24.930537
60.165017
2          0.226695           60.195540            25.053581
60.187234
3          0.185935           60.227827            24.819614
60.203474
4          0.254437           60.158928            24.909692
60.186463

   return_longitude  Air temperature (degC)
0         24.926781                     0.9
1         24.949473                     0.9
2         25.036412                     0.9
3         24.898930                     0.9
4         24.967872                     0.9
```

```
col = "avg_speed (km/h)"

plt.hist(data2[[col]], 100)
plt.show()
sns.violinplot(x=data2[col]);
```


25000
20000
15000
10000
5000
0
0.0
0.1
0.2
0.3
0.4
0.5

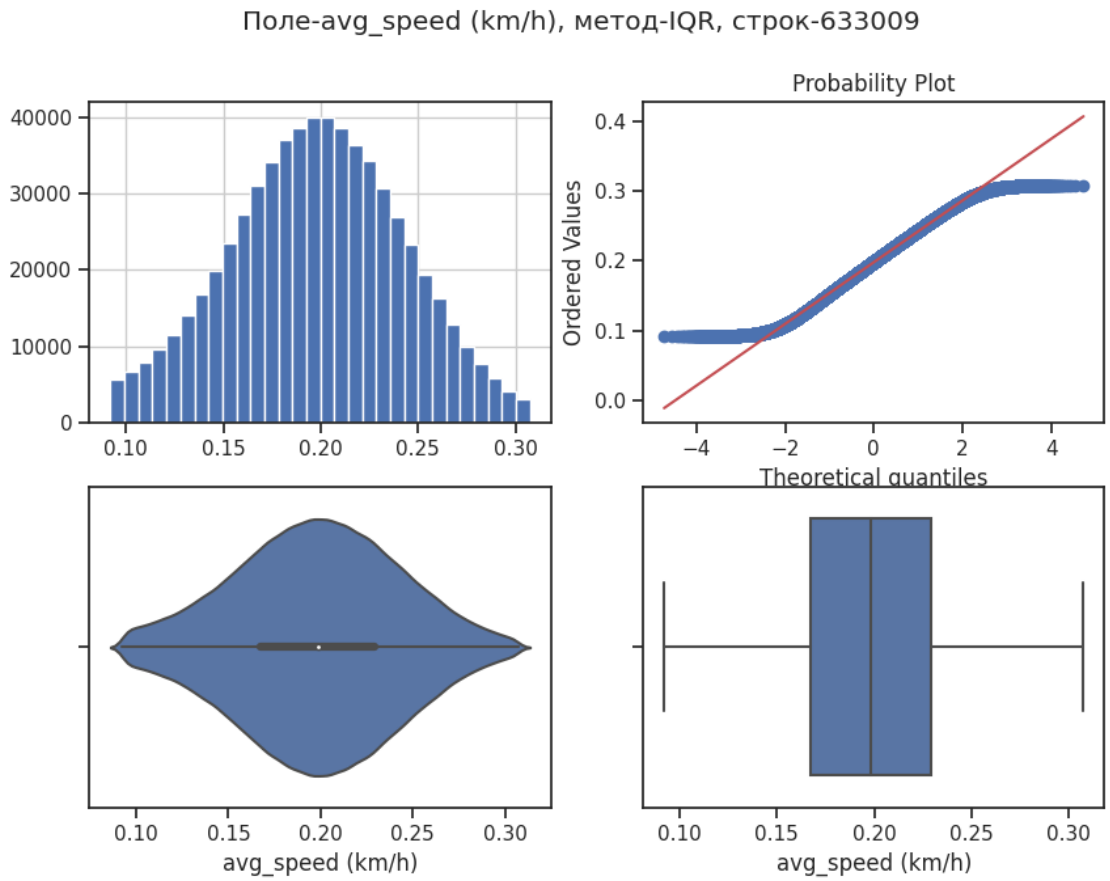

0.0
0.1
0.2
0.3
0.4
0.5
avg_speed (km/h)

```
K = 1.5 #Значение K обычно выбирается равным 1,5
IQR = data2[col].quantile(0.75) - data2[col].quantile(0.35)

# Вычисление верхней и нижней границы
lower_boundary = data2[col].quantile(0.35) - (K * IQR)
upper_boundary = data2[col].quantile(0.75) + (K * IQR)

 # Флаги для удаления выбросов
outliers_temp = np.where(data2[col] > upper_boundary, True,
                         np.where(data2[col] < lower_boundary, True,
False))
outliers_temp
```

```
array([False, False, False, ..., False, False, False])
```

```
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()

# Удаление данных на основе флага
data_trimmed = data2.loc[~(outliers_temp), ]
title = 'Поле-{}, метод-{}, строк-{}'.format(col, 'IQR',
data_trimmed.shape[0])
diagnostic_plots(data_trimmed, col, title)
```

```
<ipython-input-59-1fe78d5d2ee2>:4: MatplotlibDeprecationWarning: Auto-
removal of overlapping axes is deprecated since 3.6 and will be
removed two minor releases later; explicitly call ax.remove() as
needed.
  plt.subplot(2, 2, 1)
```



Дополнительное задание

```
data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 705495 entries, 0 to 705634
Data columns (total 14 columns):
 #   Column                   Non-Null Count   Dtype  
---  ------                   --------------   -----  
 0   departure                705495 non-null  object 
 1   return                   705495 non-null  object 
 2   departure_id             705495 non-null  int64  
 3   departure_name           705495 non-null  object 
 4   return_id                705495 non-null  float64
 5   return_name              705495 non-null  object 
 6   distance (m)             705495 non-null  float64
 7   duration (sec.)          705495 non-null  float64
 8   avg_speed (km/h)         705495 non-null  float64
 9   departure_latitude       705495 non-null  float64
 10  departure_longitude      705495 non-null  float64
 11  return_latitude          705495 non-null  float64
 12  return_longitude         705495 non-null  float64
 13  Air temperature (degC)   705495 non-null  float64
```

```
dtypes: float64(9), int64(1), object(4)
memory usage: 80.7+ MB
```

```
plt.hist(data2[['Air temperature (degC)']], 100)
plt.show()
```