In [229]:
```
import numpy as np
import random
import pandas as pd
import seaborn as sns
import scipy.stats as stats
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
from IPython.display import Image
%matplotlib inline
sns.set(style="ticks")
```

In [230]:
```
data = pd.read_csv('WineQT.csv', sep = ',')
```

In [231]:
```
data.head()
```

Out[231]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4

In [232]:
```
data.shape
```

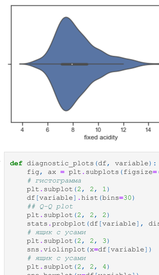
Out[232]:
```
(1143, 13)
```

In [233]:
```
data.dtypes
```

Out[233]:
```
fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
total sulfur dioxide    float64
density                 float64
pH                      float64
sulphates               float64
alcohol                 float64
quality                   int64
Id                        int64
dtype: object
```

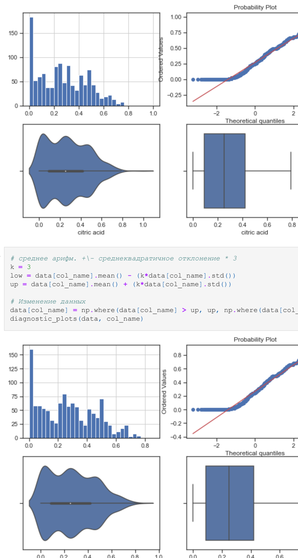
In [234]:
```
# Для студентов группы ЮУБ-24М, ЮУБФ-24М - для произвольной колонки данных построить
#"Скрипичная диаграмма (violin plot)

sns.violinplot(x = data['fixed acidity'])
```

Out[234]:
```
<AxesSubplot:xlabel="fixed acidity">
```

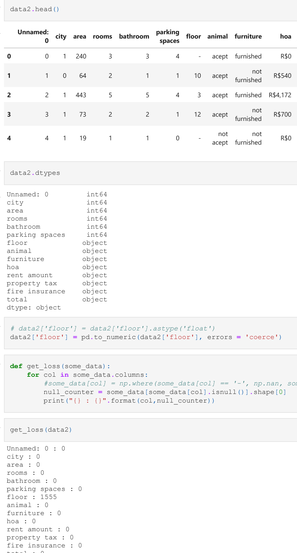


In [235]:
```
def diagnostic_plots(df, variable):
    fig, ax = plt.subplots(figsize=(10,7))
    # гистограмма
    plt.subplot(2, 2, 1)
    df[variable].hist(bins=30)
    ## Q-Q plot
    plt.subplot(2, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    # ящик с усами
    plt.subplot(2, 2, 3)
    sns.violinplot(x=df[variable])
    # ящик с усами
    plt.subplot(2, 2, 4)
    sns.boxplot(x=df[variable])
    plt.show()
```

In [236]:
```
# Задача 26
# Для набора данных для одного (произвольного) числового признака проведите
# обнаружение и замену (найденными верхними и нижними границами) выбросов на основе п
col_name = 'citric acid'
diagnostic_plots(data, col_name)
```



In [238]:
```
# среднее арифм. +\- среднеквадратичное отклонение * 3
k = 3
low = data[col_name].mean() - (k*data[col_name].std())
up = data[col_name].mean() + (k*data[col_name].std())

# Изменение данных
data[col_name] = np.where(data[col_name] > up, up, np.where(data[col_name] < low, low,
diagnostic_plots(data, col_name)
```



In [219]:
```
# задача 6
# Для набора данных проведите устранение пропусков для одного (произвольного) числово
# использованием метода заполнения средним значением.

data2 = pd.read_csv('houses_to_rent.csv', sep = ',')
```

In [220]:
```
data2.head()
```

Out[220]:

	Unnamed: 0	city	area	rooms	bathroom	parking spaces	floor	animal	furniture	hoa	rent amount	property tax	ins
0	0	1	240	3	3	4	-	acept	furnished	R$0	R$8,000	R$1,000	
1	1	0	64	2	1	1	10	acept	not furnished	R$540	R$820	R$122	
2	2	1	443	5	5	4	3	acept	furnished	R$4,172	R$7,000	R$1,417	
3	3	1	73	2	2	1	12	acept	not furnished	R$700	R$1,250	R$150	
4	4	1	19	1	1	0	-	not acept	not furnished	R$0	R$1,200	R$41	

In [221]:
```
data2.dtypes
```

Out[221]:
```
Unnamed: 0          int64
city                int64
area                int64
rooms               int64
bathroom            int64
parking spaces      int64
floor              object
animal             object
furniture          object
hoa                object
rent amount        object
property tax       object
fire insurance     object
total              object
dtype: object
```

In [222]:
```
# data2['floor'] = data2['floor'].astype('float')
data2['floor'] = pd.to_numeric(data2['floor'], errors = 'coerce')
```

In [223]:
```
def get_loss(some_data):
    for col in some_data.columns:
        #some_data[col] = np.where(some_data[col] == '-', np.nan, some_data[col])
        null_counter = some_data[some_data[col].isnull()].shape[0]
        print("{} : {}".format(col,null_counter))
```

In [224]:
```
get_loss(data2)
```

Out[224]:
```
Unnamed: 0 : 0
city : 0
area : 0
rooms : 0
bathroom : 0
parking spaces : 0
floor : 1555
animal : 0
furniture : 0
hoa : 0
rent amount : 0
property tax : 0
fire insurance : 0
total : 0
```

In [225]:
```
data3 = data2.copy()
data3['floor'].fillna(data2['floor'].mean(), inplace = True)
```

In [226]:
```
get_loss(data3)
```

Out[226]:
```
Unnamed: 0 : 0
city : 0
area : 0
rooms : 0
bathroom : 0
parking spaces : 0
floor : 0
animal : 0
furniture : 0
hoa : 0
rent amount : 0
property tax : 0
fire insurance : 0
total : 0
```

In [227]:
```
data3.head()
```

Out[227]:

	Unnamed: 0	city	area	rooms	bathroom	parking spaces	floor	animal	furniture	hoa	rent amount	property tax
0	0	1	240	3	3	4	7.621436	acept	furnished	R$0	R$8,000	R$1,000
1	1	0	64	2	1	1	10.000000	acept	not furnished	R$540	R$820	R$122
2	2	1	443	5	5	4	3.000000	acept	furnished	R$4,172	R$7,000	R$1,417
3	3	1	73	2	2	1	12.000000	acept	not furnished	R$700	R$1,250	R$150
4	4	1	19	1	1	0	7.621436	not acept	not furnished	R$0	R$1,200	R$41

In [ ]:
```
```