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

Лабораторная работа №3
по дисциплине «Методы машинного обучения»
«Обработка признаков (часть 2)»

ИСПОЛНИТЕЛЬ:

Алехин С.С.
Группа ИУ5-23М

_____

ПРОВЕРИЛ:

Балашов А.М.

_____

Москва, 2023

Задание

1. Выбрать один или несколько наборов данных (датасетов) для решения следующих задач. Каждая задача может быть решена на отдельном датасете, или несколько задач могут быть решены на одном датасете. Просьба не использовать датасет, на котором данная задача решалась в лекции.
2. Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:
    i. масштабирование признаков (не менее чем тремя способами);
    ii. обработку выбросов для числовых признаков (по одному способу для удаления выбросов и для замены выбросов);
    iii. обработку по крайней мере одного нестандартного признака (который не является числовым или категориальным);
    iv. отбор признаков:
        - один метод из группы методов фильтрации (filter methods);
        - один метод из группы методов обертывания (wrapper methods);
        - один метод из группы методов вложений (embedded methods).

Lab3

June 20, 2023

[ ]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
import scipy.stats as stats
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.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
```

[ ]:
```
data = pd.read_csv("datasets/Accident_Information.csv")
```

```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_64500/2004933154.py:1
: DtypeWarning: Columns (0) have mixed types. Specify dtype option on import or
set low_memory=False.
  data = pd.read_csv("Accident_Information.csv")
```

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

[ ]:

	Accident_Index	1st_Road_Class	1st_Road_Number	2nd_Road_Class	\
0	200501BS00001	A	3218.0	NaN	
1	200501BS00002	B	450.0	C	
2	200501BS00003	C	0.0	NaN	
3	200501BS00004	A	3220.0	NaN	
4	200501BS00005	Unclassified	0.0	NaN	

	2nd_Road_Number	Accident_Severity	Carriageway_Hazards	Date	\
0	0.0	Serious	NaN	2005-01-04	
1	0.0	Slight	NaN	2005-01-05	
2	0.0	Slight	NaN	2005-01-06	
3	0.0	Slight	NaN	2005-01-07	
4	0.0	Slight	NaN	2005-01-10	

```
  Day_of_Week  Did_Police_Officer_Attend_Scene_of_Accident  …  \
0     Tuesday                                         1.0  …
1   Wednesday                                         1.0  …
2    Thursday                                         1.0  …
3      Friday                                         1.0  …
4      Monday                                         1.0  …

          Police_Force Road_Surface_Conditions          Road_Type  \
0  Metropolitan Police             Wet or damp  Single carriageway
1  Metropolitan Police                     Dry    Dual carriageway
2  Metropolitan Police                     Dry  Single carriageway
3  Metropolitan Police                     Dry  Single carriageway
4  Metropolitan Police             Wet or damp  Single carriageway

  Special_Conditions_at_Site Speed_limit   Time  Urban_or_Rural_Area  \
0                        NaN        30.0  17:42                Urban
1                        NaN        30.0  17:36                Urban
2                        NaN        30.0  00:15                Urban
3                        NaN        30.0  10:35                Urban
4                        NaN        30.0  21:13                Urban

      Weather_Conditions  Year InScotland
0  Raining no high winds  2005         No
1     Fine no high winds  2005         No
2     Fine no high winds  2005         No
3     Fine no high winds  2005         No
4     Fine no high winds  2005         No

[5 rows x 34 columns]
```

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

[ ]:
```
  Accident_Index 1st_Road_Class  1st_Road_Number 2nd_Road_Class  \
0  200501BS00001              A           3218.0            NaN
1  200501BS00002              B            450.0              C
2  200501BS00003              C              0.0            NaN
3  200501BS00004              A           3220.0            NaN
4  200501BS00005   Unclassified              0.0            NaN

   2nd_Road_Number Accident_Severity Carriageway_Hazards        Date  \
0             0.0           Serious                 NaN  2005-01-04
1             0.0            Slight                 NaN  2005-01-05
2             0.0            Slight                 NaN  2005-01-06
3             0.0            Slight                 NaN  2005-01-07
4             0.0            Slight                 NaN  2005-01-10
```

```
  Day_of_Week  Did_Police_Officer_Attend_Scene_of_Accident  …  \
0     Tuesday                                         1.0  …
1   Wednesday                                         1.0  …
2    Thursday                                         1.0  …
3      Friday                                         1.0  …
4      Monday                                         1.0  …

          Police_Force Road_Surface_Conditions          Road_Type  \
0  Metropolitan Police             Wet or damp  Single carriageway
1  Metropolitan Police                     Dry    Dual carriageway
2  Metropolitan Police                     Dry  Single carriageway
3  Metropolitan Police                     Dry  Single carriageway
4  Metropolitan Police             Wet or damp  Single carriageway

  Special_Conditions_at_Site Speed_limit   Time  Urban_or_Rural_Area  \
0                        NaN        30.0  17:42                Urban
1                        NaN        30.0  17:36                Urban
2                        NaN        30.0  00:15                Urban
3                        NaN        30.0  10:35                Urban
4                        NaN        30.0  21:13                Urban

      Weather_Conditions  Year InScotland
0  Raining no high winds  2005         No
1     Fine no high winds  2005         No
2     Fine no high winds  2005         No
3     Fine no high winds  2005         No
4     Fine no high winds  2005         No

[5 rows x 34 columns]
```

[ ]:
```
#                              (    25%)
data.dropna(axis=1, thresh=1535442)
```

[ ]:
```
        Accident_Index 1st_Road_Class  1st_Road_Number  2nd_Road_Number  \
0         200501BS00001              A           3218.0              0.0
1         200501BS00002              B            450.0              0.0
2         200501BS00003              C              0.0              0.0
3         200501BS00004              A           3220.0              0.0
4         200501BS00005   Unclassified              0.0              0.0
…                   …              …                …                …
2047251  2017984121017           A(M)             74.0              0.0
2047252  2017984121217              C             69.0              0.0
2047253  2017984121717           A(M)             74.0              0.0
2047254  2017984122317              A            708.0              0.0
2047255  2017984122617              B            721.0            724.0

        Accident_Severity        Date Day_of_Week  \
```

```
0                 Serious  2005-01-04     Tuesday
1                  Slight  2005-01-05   Wednesday
2                  Slight  2005-01-06    Thursday
3                  Slight  2005-01-07      Friday
4                  Slight  2005-01-10      Monday
…                   …         …          …
2047251            Slight  2017-12-17      Sunday
2047252            Slight  2017-12-15      Friday
2047253            Slight  2017-12-18      Monday
2047254            Slight  2017-07-18     Tuesday
2047255           Serious  2017-12-20   Wednesday

         Did_Police_Officer_Attend_Scene_of_Accident  \
0                                                1.0
1                                                1.0
2                                                1.0
3                                                1.0
4                                                1.0
…                                                …
2047251                                          1.0
2047252                                          2.0
2047253                                          1.0
2047254                                          1.0
2047255                                          1.0

                    Junction_Control                      Junction_Detail  \
0        Data missing or out of range  Not at junction or within 20 metres
1                Auto traffic signal                             Crossroads
2        Data missing or out of range  Not at junction or within 20 metres
3        Data missing or out of range  Not at junction or within 20 metres
4        Data missing or out of range  Not at junction or within 20 metres
…                                …                                     …
2047251  Data missing or out of range  Not at junction or within 20 metres
2047252  Data missing or out of range  Not at junction or within 20 metres
2047253      Give way or uncontrolled                             Slip road
2047254  Data missing or out of range  Not at junction or within 20 metres
2047255      Give way or uncontrolled              T or staggered junction

         …  Pedestrian_Crossing-Physical_Facilities           Police_Force  \
0        …                                        1.0    Metropolitan Police
1        …                                        5.0    Metropolitan Police
2        …                                        0.0    Metropolitan Police
3        …                                        0.0    Metropolitan Police
4        …                                        0.0    Metropolitan Police
…      …                                          …                     …
2047251  …                                        0.0  Dumfries and Galloway
2047252  …                                        0.0  Dumfries and Galloway
```

```
2047253  …                                    0.0  Dumfries and Galloway
2047254  …                                    0.0  Dumfries and Galloway
2047255  …                                    0.0  Dumfries and Galloway

        Road_Surface_Conditions          Road_Type  Speed_limit   Time  \
0                   Wet or damp  Single carriageway         30.0  17:42
1                           Dry    Dual carriageway         30.0  17:36
2                           Dry  Single carriageway         30.0  00:15
3                           Dry  Single carriageway         30.0  10:35
4                   Wet or damp  Single carriageway         30.0  21:13
…                             …                  …          …     …
2047251             Frost or ice    Dual carriageway         70.0  11:30
2047252                     Dry  Single carriageway         20.0  13:00
2047253             Wet or damp    Dual carriageway         70.0  13:30
2047254                     Dry  Single carriageway         60.0  18:00
2047255             Wet or damp  Single carriageway         40.0  13:00

         Urban_or_Rural_Area     Weather_Conditions  Year  InScotland
0                      Urban  Raining no high winds  2005          No
1                      Urban     Fine no high winds  2005          No
2                      Urban     Fine no high winds  2005          No
3                      Urban     Fine no high winds  2005          No
4                      Urban     Fine no high winds  2005          No
…                          …                      …   …         …
2047251                Rural                  Other  2017         Yes
2047252                Urban     Fine no high winds  2017         Yes
2047253                Rural     Fine no high winds  2017         Yes
2047254                Rural     Fine no high winds  2017         Yes
2047255                Rural            Fog or mist  2017         Yes

[2047256 rows x 31 columns]
```

[ ]:
```
#
def impute_na(df, variable, value):
    df[variable].fillna(value, inplace=True)
impute_na(data, 'Number_of_Vehicles', data['Number_of_Vehicles'].mean())
```

[ ]:
```
data.describe()
```

[ ]:
```
       1st_Road_Number  2nd_Road_Number  \
count     2.047254e+06     2.029663e+06
mean      9.921051e+02     3.728153e+02
std       1.809408e+03     1.287796e+03
min       0.000000e+00     0.000000e+00
25%       0.000000e+00     0.000000e+00
50%       1.180000e+02     0.000000e+00
75%       7.020000e+02     0.000000e+00
```

```
max       9.999000e+03     9.999000e+03

       Did_Police_Officer_Attend_Scene_of_Accident      Latitude  \
count                                 2.046978e+06  2.047082e+06
mean                                  1.202319e+00  5.255978e+01
std                                   4.081935e-01  1.445506e+00
min                                   1.000000e+00  4.991294e+01
25%                                   1.000000e+00  5.148540e+01
50%                                   1.000000e+00  5.223758e+01
75%                                   1.000000e+00  5.345590e+01
max                                   3.000000e+00  6.075754e+01

       Location_Easting_OSGR  Location_Northing_OSGR     Longitude  \
count           2.047092e+06            2.047092e+06  2.047081e+06
mean            4.414462e+05            2.968855e+05 -1.410155e+00
std             9.549620e+04            1.605273e+05  1.403532e+00
min             6.495000e+04            1.029000e+04 -7.516225e+00
25%             3.780635e+05            1.777568e+05 -2.329610e+00
50%             4.430500e+05            2.611835e+05 -1.362233e+00
75%             5.242982e+05            3.956100e+05 -2.052600e-01
max             6.555400e+05            1.208800e+06  1.762010e+00

       Number_of_Casualties  Number_of_Vehicles  \
count          2.047256e+06        2.047256e+06
mean           1.345843e+00        1.833525e+00
std            8.179627e-01        7.150543e-01
min            1.000000e+00        1.000000e+00
25%            1.000000e+00        1.000000e+00
50%            1.000000e+00        2.000000e+00
75%            1.000000e+00        2.000000e+00
max            9.300000e+01        6.700000e+01

       Pedestrian_Crossing-Human_Control  \
count                       2.044336e+06
mean                        1.041707e-02
std                         1.351126e-01
min                         0.000000e+00
25%                         0.000000e+00
50%                         0.000000e+00
75%                         0.000000e+00
max                         2.000000e+00

       Pedestrian_Crossing-Physical_Facilities   Speed_limit          Year
count                             2.043696e+06  2.047219e+06  2.047256e+06
mean                              7.518021e-01  3.884360e+01  2.010524e+03
std                               1.835289e+00  1.414791e+01  3.765624e+00
min                               0.000000e+00  0.000000e+00  2.005000e+03
```

```
25%                                0.000000e+00  3.000000e+01  2.007000e+03
50%                                0.000000e+00  3.000000e+01  2.010000e+03
75%                                0.000000e+00  5.000000e+01  2.014000e+03
max                                8.000000e+00  7.000000e+01  2.017000e+03
```

[ ]:
```
def obj_col(column):
    return column[1] == 'object'

col_names = []
for col in list(filter(obj_col, list(zip(list(data.columns), list(data.
 ↪dtypes))))):
  col_names.append(col[0])
col_names.append('Speed_limit')
```

[ ]:
```
X_ALL = data.drop(col_names, axis=1)
```

[ ]:
```
#
#
def arr_to_df(arr_scaled):
    res = pd.DataFrame(arr_scaled, columns=X_ALL.columns)
    return res
```

0.1 StandardScaler

[ ]:
```
#
X_train, X_test, y_train, y_test = train_test_split(X_ALL, data['Speed_limit'],
                                                    test_size=0.2,
                                                    random_state=1)
#              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
```

[ ]:
```
#      StandardScaler
cs11 = StandardScaler()
data_cs11_scaled_temp = cs11.fit_transform(X_ALL)
#       DataFrame
data_cs11_scaled =  arr_to_df(data_cs11_scaled_temp)
data_cs11_scaled
```

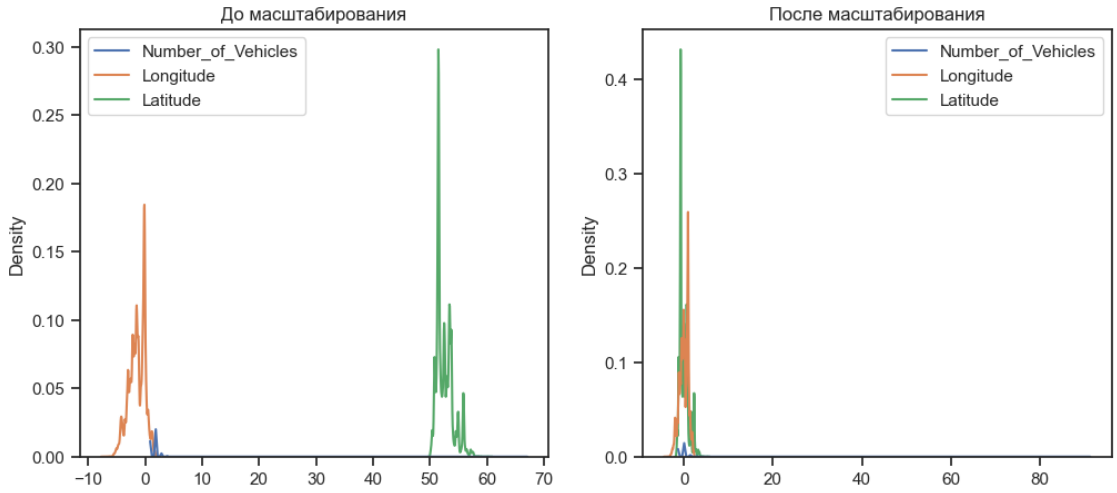
[ ]:
```
          1st_Road_Number  2nd_Road_Number  \
0                1.230179        -0.289499
1               -0.299604        -0.289499
2               -0.548304        -0.289499
3                1.231284        -0.289499
4               -0.548304        -0.289499
```

```
…                 …              …
2047251       -0.507406       -0.289499
2047252       -0.510170       -0.289499
2047253       -0.507406       -0.289499
2047254       -0.157016       -0.289499
2047255       -0.149831        0.272702

         Did_Police_Officer_Attend_Scene_of_Accident  Latitude  \
0                                          -0.495644 -0.740699
1                                          -0.495644 -0.719267
2                                          -0.495644 -0.715652
3                                          -0.495644 -0.745302
4                                          -0.495644 -0.736094
…                                                …       …
2047251                                    -0.495644  1.903840
2047252                                     1.954175  1.684189
2047253                                    -0.495644  1.725338
2047254                                    -0.495644  1.913940
2047255                                    -0.495644  1.681159

         Location_Easting_OSGR  Location_Northing_OSGR  Longitude  \
0                     0.882065               -0.739099   0.868512
1                     0.866252               -0.717856   0.853879
2                     0.869917               -0.714181   0.857620
3                     0.894840               -0.743522   0.880844
4                     0.906987               -0.734115   0.893130
…                        …                      …         …
2047251              -1.397147                1.906471  -1.453825
2047252              -1.147902                1.683631  -1.180565
2047253              -1.252796                1.725898  -1.293416
2047254              -1.371565                1.916283  -1.426770
2047255              -1.285300                1.681961  -1.326595

         Number_of_Casualties  Number_of_Vehicles  \
0                   -0.422811           -1.165682
1                   -0.422811           -1.165682
2                   -0.422811            0.232814
3                   -0.422811           -1.165682
4                   -0.422811           -1.165682
…                      …                  …
2047251             -0.422811           -1.165682
2047252             -0.422811           -1.165682
2047253             -0.422811            0.232814
2047254             -0.422811           -1.165682
2047255              0.799739            0.232814

         Pedestrian_Crossing-Human_Control  \
```

```
0                                  -0.077099
1                                  -0.077099
2                                  -0.077099
3                                  -0.077099
4                                  -0.077099
…                                        …
2047251                            -0.077099
2047252                            -0.077099
2047253                            -0.077099
2047254                            -0.077099
2047255                            -0.077099

         Pedestrian_Crossing-Physical_Facilities      Year
0                                       0.135236 -1.466904
1                                       2.314730 -1.466904
2                                      -0.409637 -1.466904
3                                      -0.409637 -1.466904
4                                      -0.409637 -1.466904
…                                            …        …
2047251                                -0.409637  1.719820
2047252                                -0.409637  1.719820
2047253                                -0.409637  1.719820
2047254                                -0.409637  1.719820
2047255                                -0.409637  1.719820

[2047256 rows x 12 columns]
```

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

[ ]:
```
draw_kde(['Number_of_Vehicles', 'Longitude', 'Latitude'], data,␣
 ↪data_cs11_scaled, '          ', '            ')
```



0.2 "Mean Normalisation"

[ ]:
```
#
X_train, X_test, y_train, y_test = train_test_split(X_ALL, data['Speed_limit'],
                                                    test_size=0.2,
                                                    random_state=1)
#               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
```

[ ]:
```
((1637804, 12), (409452, 12))
```

[ ]:
```
class MeanNormalisation:

    def fit(self, param_df):
        self.means = X_train.mean(axis=0)
        maxs = X_train.max(axis=0)
        mins = X_train.min(axis=0)
        self.ranges = maxs - mins

    def transform(self, param_df):
        param_df_scaled = (param_df - self.means) / self.ranges
        return param_df_scaled

    def fit_transform(self, param_df):
        self.fit(param_df)
        return self.transform(param_df)
```

[ ]:
```
sc21 = MeanNormalisation()
data_cs21_scaled = sc21.fit_transform(X_ALL)
data_cs21_scaled.describe()
```

[ ]:

	1st_Road_Number	2nd_Road_Number	\
count	2.047254e+06	2.029663e+06	
mean	2.035487e-05	2.013229e-05	
std	1.809589e-01	1.287925e-01	
min	-9.920008e-02	-3.726513e-02	
25%	-9.920008e-02	-3.726513e-02	
50%	-8.739890e-02	-3.726513e-02	
75%	-2.899306e-02	-3.726513e-02	
max	9.007999e-01	9.627349e-01	

	Did_Police_Officer_Attend_Scene_of_Accident	Latitude	\
count	2.046978e+06	2.047082e+06	
mean	-3.447226e-06	5.075726e-05	
std	2.040968e-01	1.333117e-01	
min	-1.011628e-01	-2.440538e-01	
25%	-1.011628e-01	-9.903352e-02	
50%	-1.011628e-01	-2.966383e-02	
75%	-1.011628e-01	8.269546e-02	
max	8.988372e-01	7.560889e-01	

	Location_Easting_OSGR	Location_Northing_OSGR	Longitude	\
count	2.047092e+06	2.047092e+06	2.047081e+06	
mean	-1.083523e-04	5.057594e-05	-1.013670e-04	
std	1.616963e-01	1.339647e-01	1.512715e-01	
min	-6.376000e-01	-2.391218e-01	-6.582082e-01	
25%	-1.074294e-01	-9.936585e-02	-9.919943e-02	
50%	2.607220e-03	-2.974377e-02	5.063607e-03	
75%	1.401785e-01	8.243911e-02	1.297611e-01	
max	3.624000e-01	7.610701e-01	3.417918e-01	

	Number_of_Casualties	Number_of_Vehicles	\
count	2.047256e+06	2.047256e+06	
mean	-1.030520e-06	-2.124674e-06	
std	8.890899e-03	1.083416e-02	
min	-3.760198e-03	-1.263130e-02	
25%	-3.760198e-03	-1.263130e-02	
50%	-3.760198e-03	2.520217e-03	
75%	-3.760198e-03	2.520217e-03	
max	9.962398e-01	9.873687e-01	

	Pedestrian_Crossing-Human_Control	\
count	2.044336e+06	
mean	-5.390953e-05	

```
std                          6.755628e-02
min                         -5.262447e-03
25%                         -5.262447e-03
50%                         -5.262447e-03
75%                         -5.262447e-03
max                          9.947376e-01

       Pedestrian_Crossing-Physical_Facilities          Year
count                             2.043696e+06  2.047256e+06
mean                              1.115384e-05  6.689755e-05
std                               2.294112e-01  3.138020e-01
min                              -9.396411e-02 -4.602502e-01
25%                              -9.396411e-02 -2.935836e-01
50%                              -9.396411e-02 -4.358356e-02
75%                              -9.396411e-02  2.897498e-01
max                               9.060359e-01  5.397498e-01
```

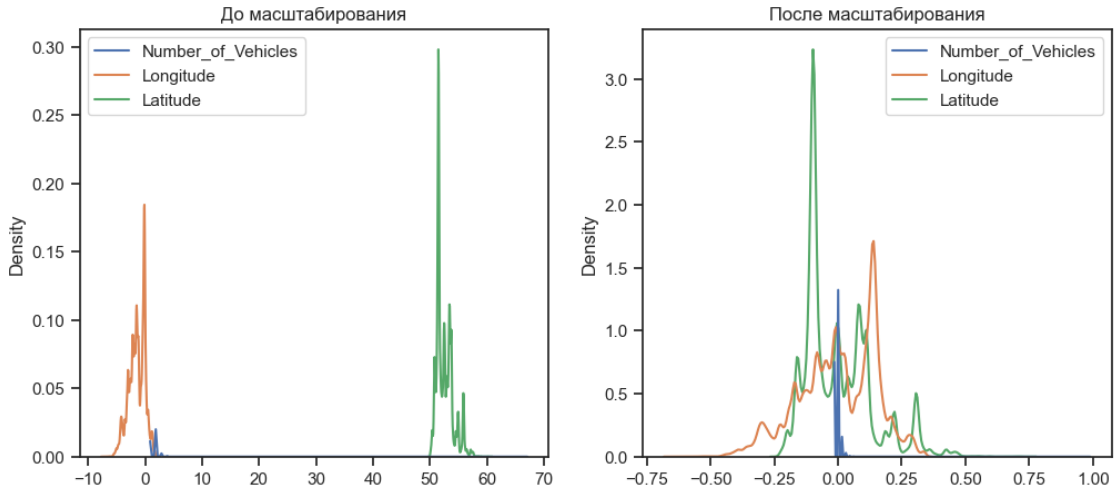
[ ]:
```
cs22 = MeanNormalisation()
cs22.fit(X_train)
data_cs22_scaled_train = cs22.transform(X_train)
data_cs22_scaled_test = cs22.transform(X_test)
```

[ ]:
```
data_cs22_scaled_train.describe()
```

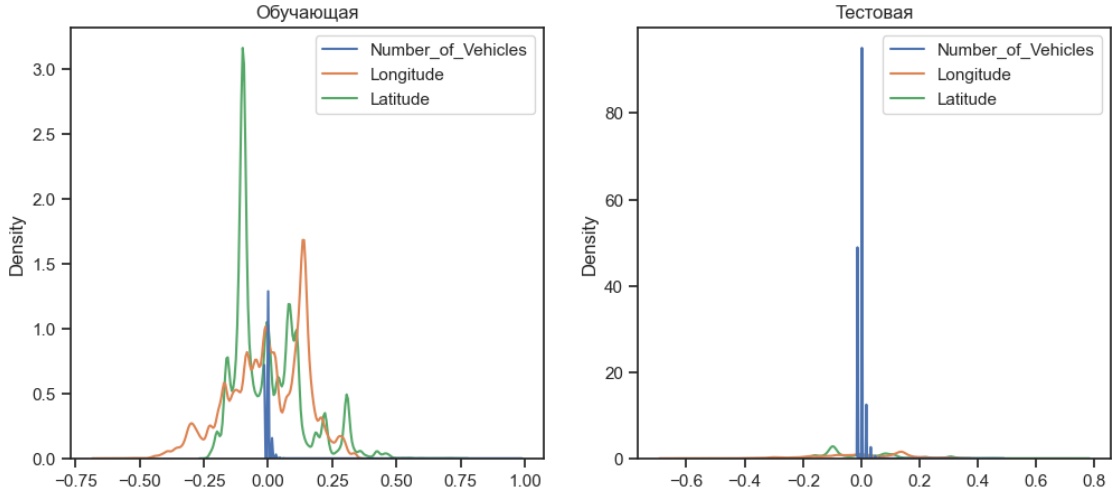
[ ]:
```
       1st_Road_Number  2nd_Road_Number  \
count     1.637802e+06     1.623664e+06
mean     -4.365507e-20    -4.233943e-19
std       1.809421e-01     1.287702e-01
min      -9.920008e-02    -3.726513e-02
25%      -9.920008e-02    -3.726513e-02
50%      -8.739890e-02    -3.726513e-02
75%      -2.899306e-02    -3.726513e-02
max       9.007999e-01     9.627349e-01

       Did_Police_Officer_Attend_Scene_of_Accident      Latitude  \
count                                 1.637578e+06  1.637663e+06
mean                                -4.107284e-17  8.399485e-16
std                                  2.041291e-01  1.332812e-01
min                                 -1.011628e-01 -2.439111e-01
25%                                 -1.011628e-01 -9.904219e-02
50%                                 -1.011628e-01 -2.972363e-02
75%                                 -1.011628e-01  8.267032e-02
max                                  8.988372e-01  7.560889e-01

       Location_Easting_OSGR  Location_Northing_OSGR     Longitude  \
count           1.637672e+06            1.637672e+06  1.637663e+06
mean            3.879536e-17            1.416353e-17  1.976902e-17
```

```
std              1.617012e-01            1.339335e-01  1.512730e-01
min             -6.376000e-01           -2.389299e-01 -6.582082e-01
25%             -1.072321e-01           -9.937983e-02 -9.908233e-02
50%              2.728286e-03           -2.983014e-02  5.206522e-03
75%              1.402323e-01            8.241867e-02  1.298215e-01
max              3.624000e-01            7.610701e-01  3.417918e-01

       Number_of_Casualties  Number_of_Vehicles  \
count          1.637804e+06        1.637804e+06
mean          -1.445564e-19        1.267080e-18
std            8.920207e-03        1.083473e-02
min           -3.760198e-03       -1.263130e-02
25%           -3.760198e-03       -1.263130e-02
50%           -3.760198e-03        2.520217e-03
75%           -3.760198e-03        2.520217e-03
max            9.962398e-01        9.873687e-01

       Pedestrian_Crossing-Human_Control  \
count                       1.635456e+06
mean                       -1.838180e-18
std                         6.790829e-02
min                        -5.262447e-03
25%                        -5.262447e-03
50%                        -5.262447e-03
75%                        -5.262447e-03
max                         9.947376e-01

       Pedestrian_Crossing-Physical_Facilities          Year
count                             1.634950e+06  1.637804e+06
mean                             -4.925059e-18  6.649712e-15
std                               2.294635e-01  3.139150e-01
min                              -9.396411e-02 -4.602502e-01
25%                              -9.396411e-02 -2.935836e-01
50%                              -9.396411e-02 -4.358356e-02
75%                              -9.396411e-02  2.897498e-01
max                               9.060359e-01  5.397498e-01
```

[ ]:
```
draw_kde(['Number_of_Vehicles', 'Longitude', 'Latitude'], data,␣
↪data_cs21_scaled, '          ', '            ')
```



[ ]:
```
draw_kde(['Number_of_Vehicles', 'Longitude', 'Latitude'],␣
↪data_cs22_scaled_train, data_cs22_scaled_test, '     ', '     ')
```



0.3 MinMax-

[ ]:
```
#      StandardScaler
cs31 = MinMaxScaler()
data_cs31_scaled_temp = cs31.fit_transform(X_ALL)
#       DataFrame
data_cs31_scaled = arr_to_df(data_cs31_scaled_temp)
data_cs31_scaled.describe()
```

[ ]:

	1st_Road_Number	2nd_Road_Number	\
count	2.047254e+06	2.029663e+06	
mean	9.922043e-02	3.728526e-02	
std	1.809589e-01	1.287925e-01	
min	0.000000e+00	0.000000e+00	
25%	0.000000e+00	0.000000e+00	
50%	1.180118e-02	0.000000e+00	
75%	7.020702e-02	0.000000e+00	
max	1.000000e+00	1.000000e+00	

	Did_Police_Officer_Attend_Scene_of_Accident	Latitude	\
count	2.046978e+06	2.047082e+06	
mean	1.011594e-01	2.440697e-01	
std	2.040968e-01	1.332927e-01	
min	0.000000e+00	0.000000e+00	
25%	0.000000e+00	1.449996e-01	
50%	0.000000e+00	2.143594e-01	
75%	0.000000e+00	3.267026e-01	
max	1.000000e+00	1.000000e+00	

	Location_Easting_OSGR	Location_Northing_OSGR	Longitude	\
count	2.047092e+06	2.047092e+06	2.047081e+06	
mean	6.374917e-01	2.391265e-01	6.581069e-01	
std	1.616963e-01	1.339390e-01	1.512715e-01	
min	0.000000e+00	0.000000e+00	0.000000e+00	
25%	5.301707e-01	1.397291e-01	5.590088e-01	
50%	6.402073e-01	2.093378e-01	6.632718e-01	
75%	7.777786e-01	3.214992e-01	7.879694e-01	
max	1.000000e+00	1.000000e+00	1.000000e+00	

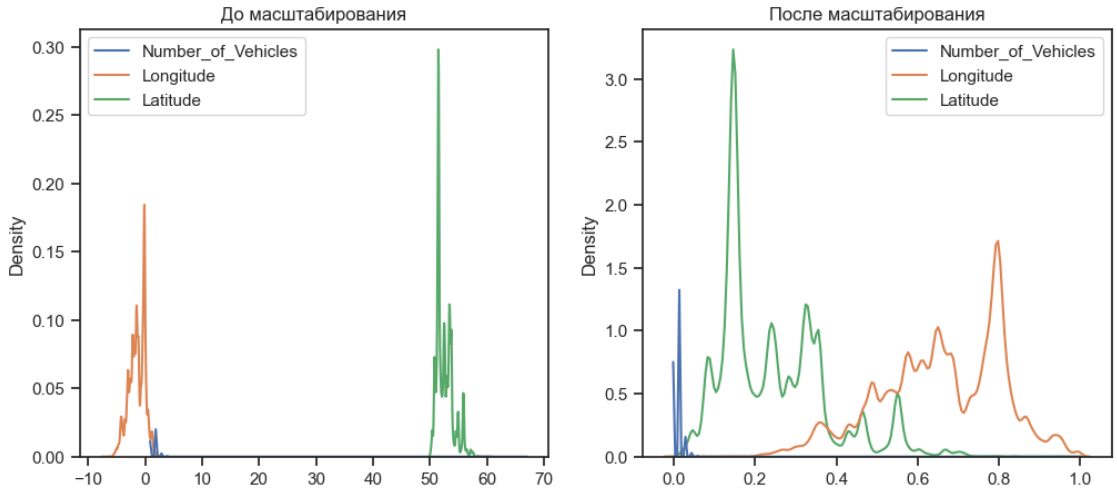
	Number_of_Casualties	Number_of_Vehicles	\
count	2.047256e+06	2.047256e+06	
mean	3.759168e-03	1.262917e-02	
std	8.890899e-03	1.083416e-02	
min	0.000000e+00	0.000000e+00	
25%	0.000000e+00	0.000000e+00	
50%	0.000000e+00	1.515152e-02	
75%	0.000000e+00	1.515152e-02	
max	1.000000e+00	1.000000e+00	

	Pedestrian_Crossing-Human_Control	\
count	2.044336e+06	
mean	5.208537e-03	
std	6.755628e-02	
min	0.000000e+00	
25%	0.000000e+00	
50%	0.000000e+00	

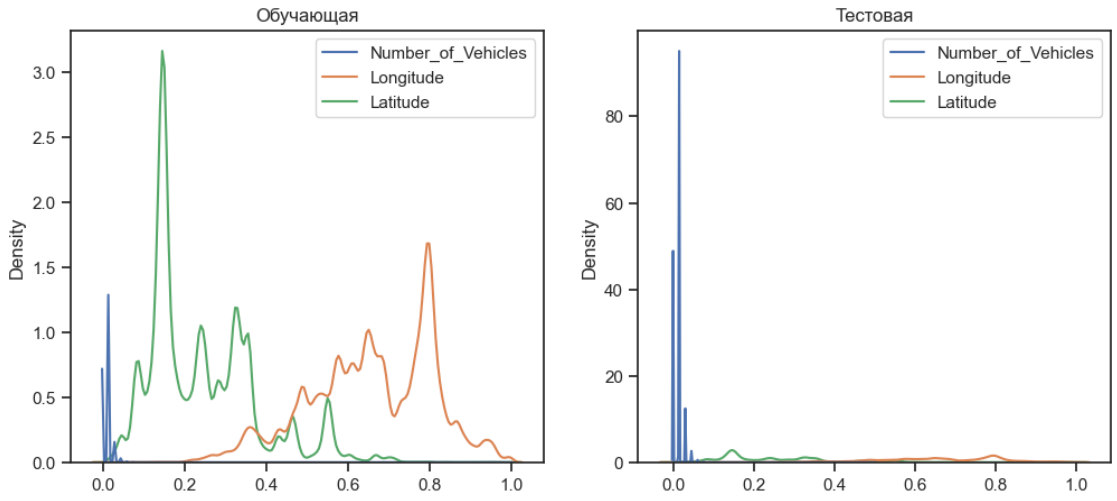
```
75%                                0.000000e+00
max                                1.000000e+00

       Pedestrian_Crossing-Physical_Facilities          Year
count                             2.043696e+06  2.047256e+06
mean                              9.397527e-02  4.603171e-01
std                               2.294112e-01  3.138020e-01
min                               0.000000e+00  0.000000e+00
25%                               0.000000e+00  1.666667e-01
50%                               0.000000e+00  4.166667e-01
75%                               0.000000e+00  7.500000e-01
max                               1.000000e+00  1.000000e+00
```

[ ]:
```
cs32 = MinMaxScaler()
cs32.fit(X_train)
data_cs32_scaled_train_temp = cs32.transform(X_train)
data_cs32_scaled_test_temp = cs32.transform(X_test)
#       DataFrame
data_cs32_scaled_train = arr_to_df(data_cs32_scaled_train_temp)
data_cs32_scaled_test = arr_to_df(data_cs32_scaled_test_temp)
```

[ ]:
```
draw_kde(['Number_of_Vehicles', 'Longitude', 'Latitude'], data,␣
↪data_cs31_scaled, '           ', '            ')
```



[ ]:
```
draw_kde(['Number_of_Vehicles', 'Longitude', 'Latitude'],␣
↪data_cs32_scaled_train, data_cs32_scaled_test, '     ', '     ')
```



0.4

[ ]:
```
data2 = pd.read_csv("datasets/Car_Sales.csv")
```

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

[ ]:
```
  Manufacturer    Model  Sales_in_thousands  __year_resale_value Vehicle_type  \
0        Acura  Integra              16.919               16.360    Passenger
1        Acura       TL              39.384               19.875    Passenger
2        Acura       CL              14.114               18.225    Passenger
3        Acura       RL               8.588               29.725    Passenger
4         Audi       A4              20.397               22.255    Passenger

   Price_in_thousands  Engine_size  Horsepower  Wheelbase  Width  Length  \
0               21.50          1.8       140.0      101.2   67.3   172.4
1               28.40          3.2       225.0      108.1   70.3   192.9
2                 NaN          3.2       225.0      106.9   70.6   192.0
3               42.00          3.5       210.0      114.6   71.4   196.6
4               23.99          1.8       150.0      102.6   68.2   178.0

   Curb_weight  Fuel_capacity  Fuel_efficiency Latest_Launch  \
0        2.639           13.2             28.0      2/2/2012
1        3.517           17.2             25.0      6/3/2011
2        3.470           17.2             26.0      1/4/2012
3        3.850           18.0             22.0     3/10/2011
4        2.998           16.4             27.0     10/8/2011

   Power_perf_factor
0          58.280150
1          91.370778
```

```
2                NaN
3          91.389779
4          62.777639
```

[ ]:
```
data2.describe()
```

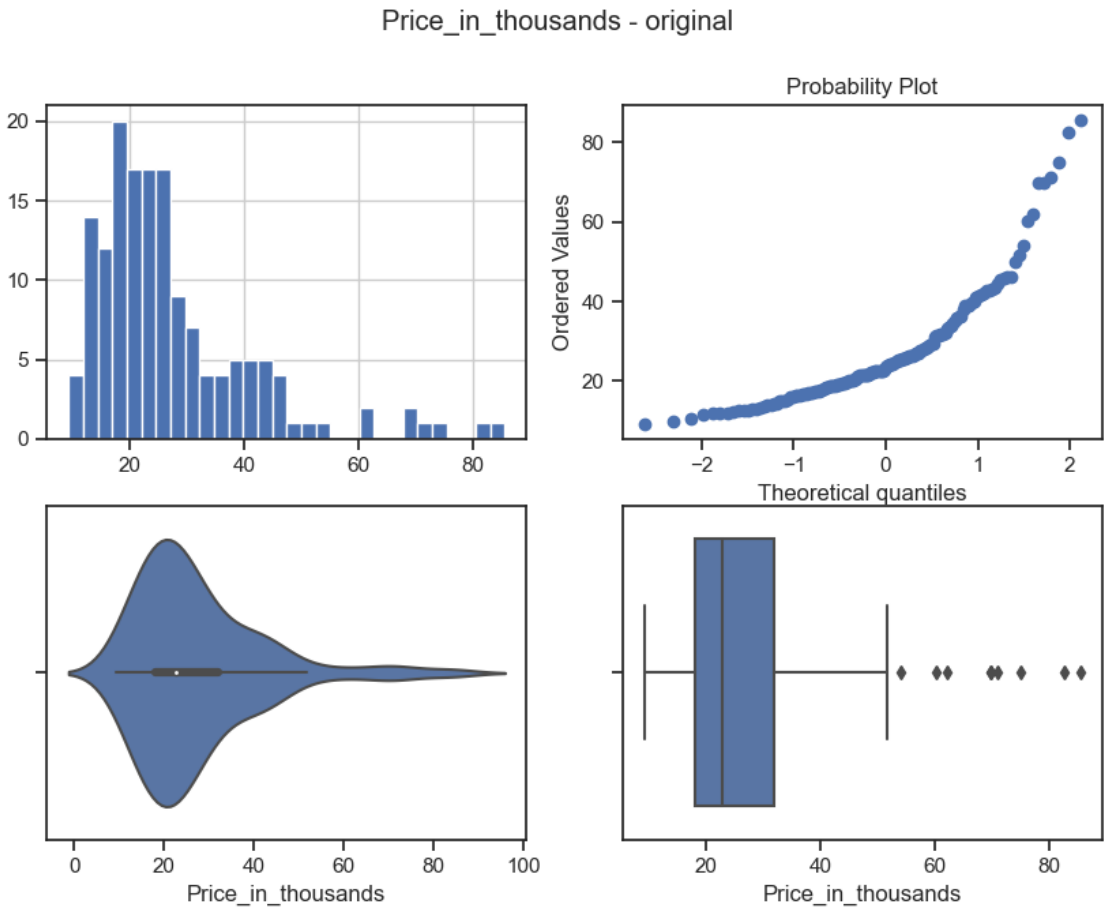
[ ]:
```
       Sales_in_thousands  __year_resale_value  Price_in_thousands  \
count          157.000000           121.000000          155.000000
mean            52.998076            18.072975           27.390755
std             68.029422            11.453384           14.351653
min              0.110000             5.160000            9.235000
25%             14.114000            11.260000           18.017500
50%             29.450000            14.180000           22.799000
75%             67.956000            19.875000           31.947500
max            540.561000            67.550000           85.500000

       Engine_size  Horsepower   Wheelbase       Width      Length  \
count   156.000000  156.000000  156.000000  156.000000  156.000000
mean      3.060897  185.948718  107.487179   71.150000  187.343590
std       1.044653   56.700321    7.641303    3.451872   13.431754
min       1.000000   55.000000   92.600000   62.600000  149.400000
25%       2.300000  149.500000  103.000000   68.400000  177.575000
50%       3.000000  177.500000  107.000000   70.550000  187.900000
75%       3.575000  215.000000  112.200000   73.425000  196.125000
max       8.000000  450.000000  138.700000   79.900000  224.500000

       Curb_weight  Fuel_capacity  Fuel_efficiency  Power_perf_factor
count   155.000000     156.000000       154.000000         155.000000
mean      3.378026      17.951923        23.844156          77.043591
std       0.630502       3.887921         4.282706          25.142664
min       1.895000      10.300000        15.000000          23.276272
25%       2.971000      15.800000        21.000000          60.407707
50%       3.342000      17.200000        24.000000          72.030917
75%       3.799500      19.575000        26.000000          89.414878
max       5.572000      32.000000        45.000000         188.144323
```

[ ]:
```
def diagnostic_plots(df, variable, title):
    fig, ax = plt.subplots(figsize=(10,7))
    #
    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)
    # violinplot
    plt.subplot(2, 2, 3)
    sns.violinplot(x=df[variable])
```

```
    # boxplot
    plt.subplot(2, 2, 4)
    sns.boxplot(x=df[variable])
    fig.suptitle(title)
    plt.show()
```

[ ]:
```
diagnostic_plots(data2, 'Price_in_thousands', 'Price_in_thousands - original')
```

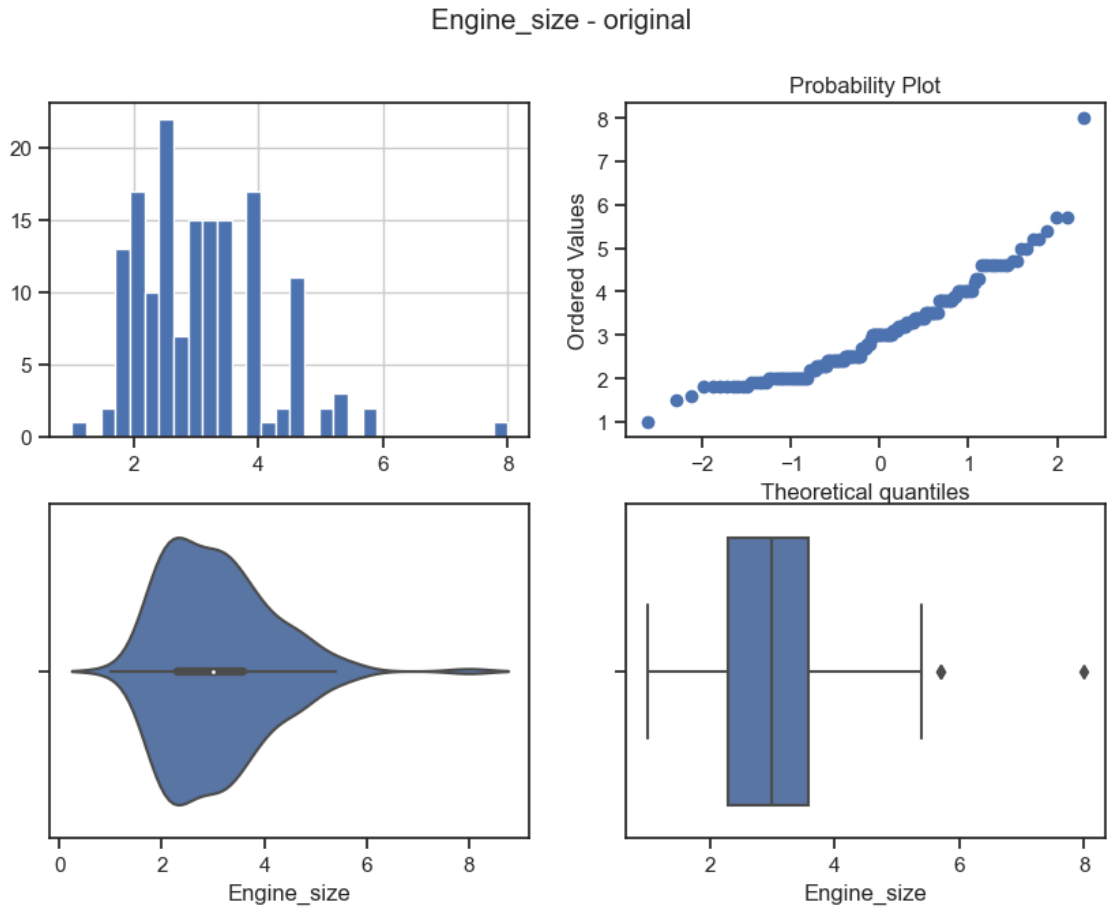
```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_64500/4201870494.py:4
: MatplotlibDeprecationWarning: Auto-removal of overlapping axes is deprecated
since 3.6 and will be removed two minor releases later; explicitly call
ax.remove() as needed.
  plt.subplot(2, 2, 1)
```



[ ]:
```
diagnostic_plots(data2, 'Engine_size', 'Engine_size - original')
```

```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_64500/4201870494.py: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)
```



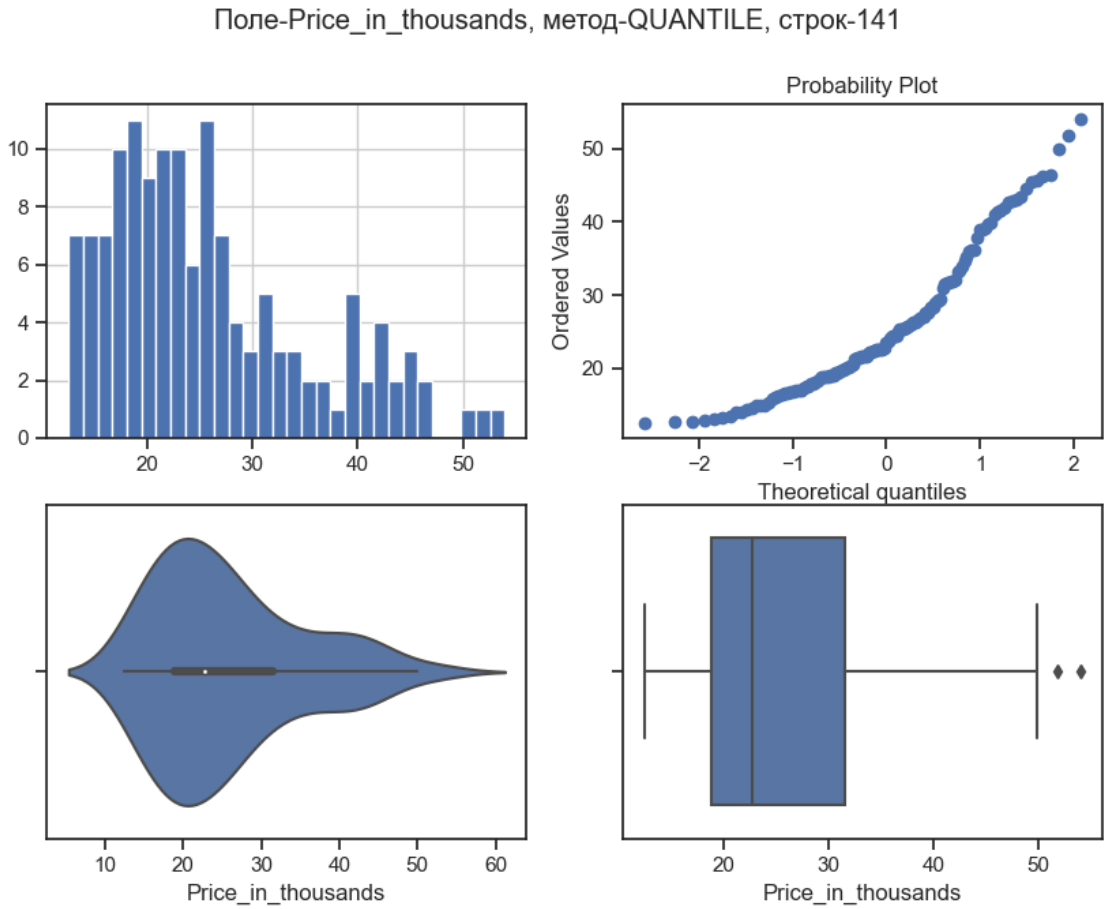
[ ]:
```
#
from enum import Enum
class OutlierBoundaryType(Enum):
    SIGMA = 1
    QUANTILE = 2
    IRQ = 3
```

[ ]:
```
#
def get_outlier_boundaries(df, col):
    lower_boundary = df[col].quantile(0.05)
    upper_boundary = df[col].quantile(0.95)
    return lower_boundary, upper_boundary
```

0.5 (number_of_reviews)

[ ]:
```
#
lower_boundary, upper_boundary = get_outlier_boundaries(data2,␣
 ↪"Price_in_thousands")
#
outliers_temp = np.where(data2["Price_in_thousands"] > upper_boundary, True,
                         np.where(data2["Price_in_thousands"] < lower_boundary,␣
 ↪True, False))
#
data_trimmed = data2.loc[~(outliers_temp), ]
title = '  -{},    -{},    -{}'.format("Price_in_thousands", "QUANTILE",␣
 ↪data_trimmed.shape[0])
diagnostic_plots(data_trimmed, "Price_in_thousands", title)
```

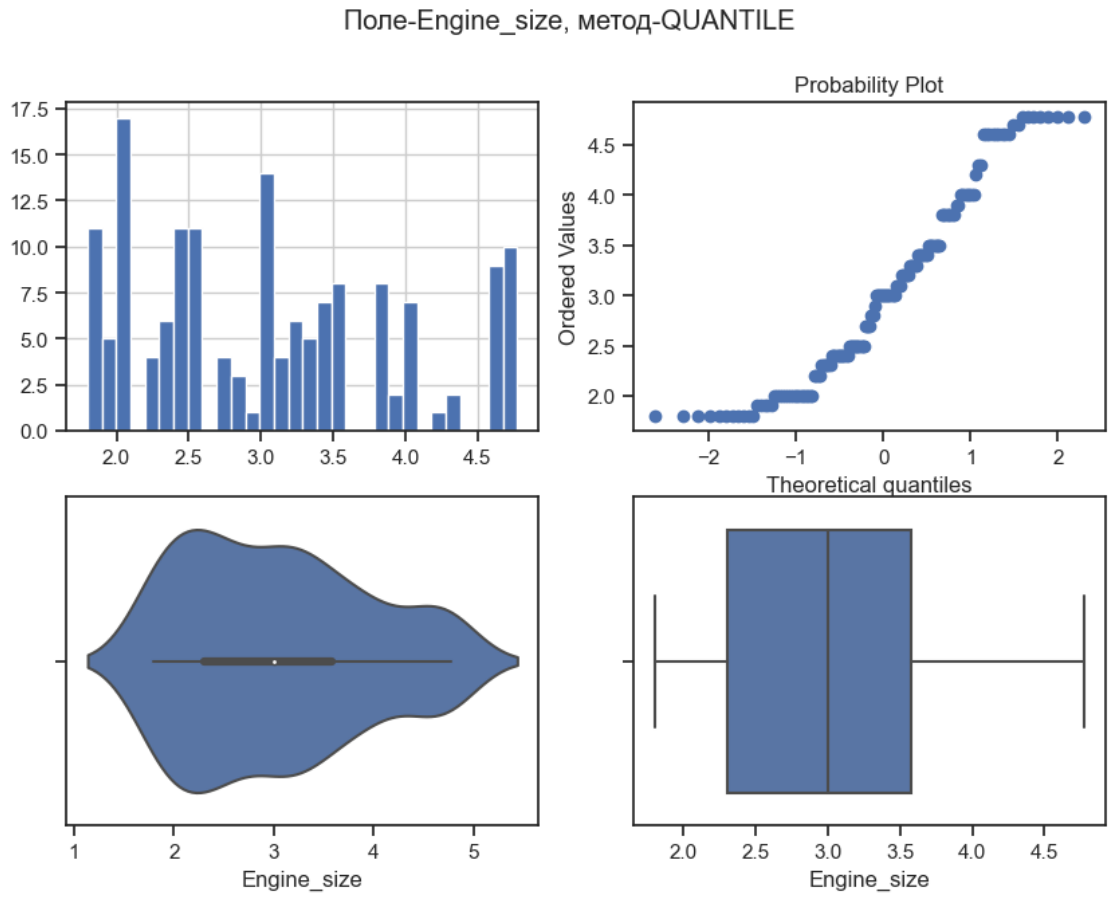
```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_64500/4201870494.py:4
: MatplotlibDeprecationWarning: Auto-removal of overlapping axes is deprecated
since 3.6 and will be removed two minor releases later; explicitly call
ax.remove() as needed.
  plt.subplot(2, 2, 1)
```



0.6

[ ]:
```
#
lower_boundary, upper_boundary = get_outlier_boundaries(data2, "Engine_size")
#
data2["Engine_size"] = np.where(data2["Engine_size"] > upper_boundary,␣
 ↪upper_boundary,
                      np.where(data2["Engine_size"] < lower_boundary,␣
 ↪lower_boundary, data2["Engine_size"]))
title = '  -{},    -{}'.format("Engine_size", "QUANTILE")
diagnostic_plots(data2, "Engine_size", title)
```

```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_64500/4201870494.py: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)
```



0.7

[ ]:
```
data2.dtypes
```

[ ]:
```
Manufacturer            object
Model                   object
Sales_in_thousands     float64
__year_resale_value    float64
Vehicle_type            object
Price_in_thousands     float64
Engine_size            float64
Horsepower             float64
Wheelbase              float64
Width                  float64
Length                 float64
Curb_weight            float64
Fuel_capacity          float64
Fuel_efficiency        float64
Latest_Launch           object
Power_perf_factor      float64
dtype: object
```

[ ]:
```
#
data2["Date"] = data2.apply(lambda x: pd.to_datetime(x["Latest_Launch"],␣
 ↪format='%m/%d/%Y'), axis=1)
```

[ ]:
```
data2.head(5)
```

[ ]:

	Manufacturer	Model	Sales_in_thousands	__year_resale_value	Vehicle_type	\
0	Acura	Integra	16.919	16.360	Passenger	
1	Acura	TL	39.384	19.875	Passenger	
2	Acura	CL	14.114	18.225	Passenger	
3	Acura	RL	8.588	29.725	Passenger	
4	Audi	A4	20.397	22.255	Passenger	

	Price_in_thousands	Engine_size	Horsepower	Wheelbase	Width	Length	\
0	21.50	1.8	140.0	101.2	67.3	172.4	
1	28.40	3.2	225.0	108.1	70.3	192.9	
2	NaN	3.2	225.0	106.9	70.6	192.0	
3	42.00	3.5	210.0	114.6	71.4	196.6	
4	23.99	1.8	150.0	102.6	68.2	178.0	

	Curb_weight	Fuel_capacity	Fuel_efficiency	Latest_Launch	\
0	2.639	13.2	28.0	2/2/2012	
1	3.517	17.2	25.0	6/3/2011	
2	3.470	17.2	26.0	1/4/2012	
3	3.850	18.0	22.0	3/10/2011	
4	2.998	16.4	27.0	10/8/2011	

```
   Power_perf_factor       Date
0          58.280150 2012-02-02
1          91.370778 2011-06-03
2                NaN 2012-01-04
3          91.389779 2011-03-10
4          62.777639 2011-10-08
```

0.8

0.9 ( )

[ ]:
```
data3 = pd.read_csv("datasets/Marketing.csv")
```

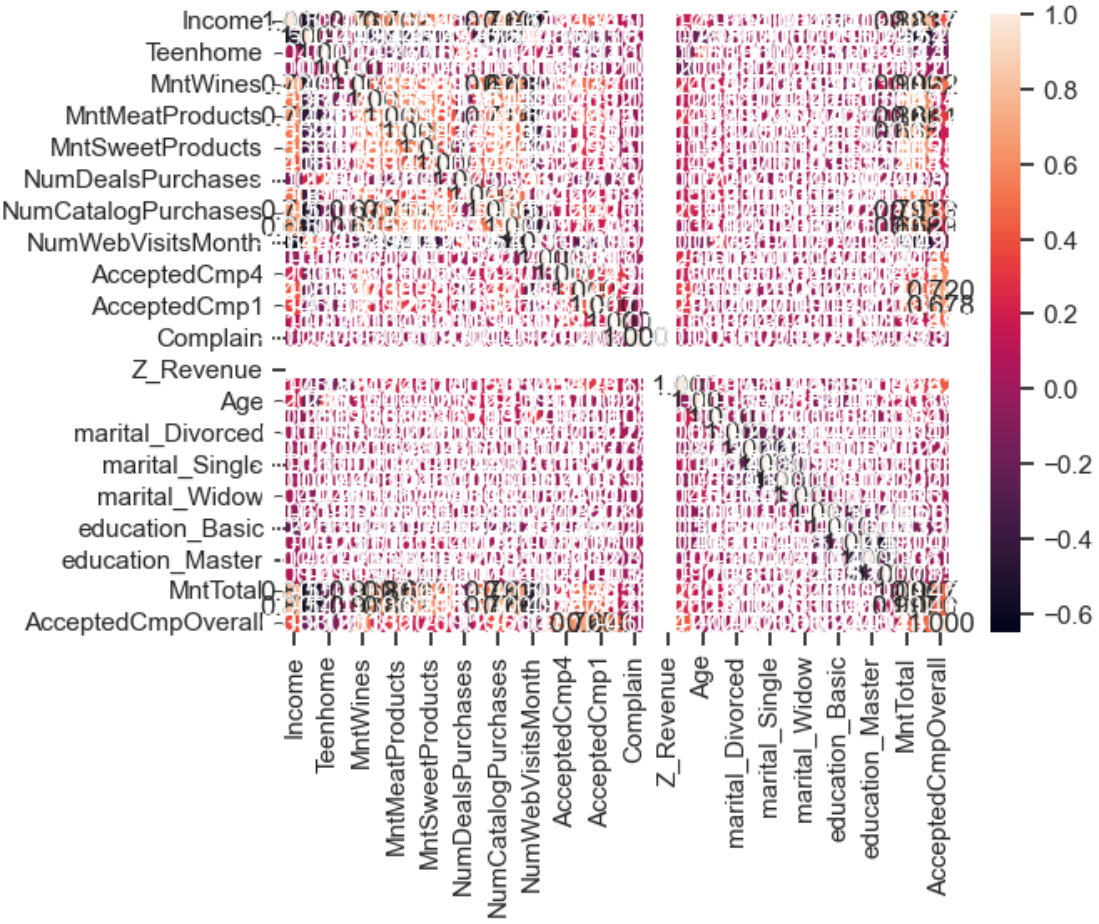
[ ]:
```
data3.head()
```

[ ]:
```
     Income  Kidhome  Teenhome  Recency  MntWines  MntFruits  MntMeatProducts  \
0  58138.0        0         0       58       635         88              546
1  46344.0        1         1       38        11          1                6
2  71613.0        0         0       26       426         49              127
3  26646.0        1         0       26        11          4               20
4  58293.0        1         0       94       173         43              118

   MntFishProducts  MntSweetProducts  MntGoldProds  …  marital_Together  \
0              172                88            88  …                 0
1                2                 1             6  …                 0
2              111                21            42  …                 1
3               10                 3             5  …                 1
4               46                27            15  …                 0

   marital_Widow  education_2n Cycle  education_Basic  education_Graduation  \
0              0                   0                0                     1
1              0                   0                0                     1
2              0                   0                0                     1
3              0                   0                0                     1
4              0                   0                0                     0

   education_Master  education_PhD  MntTotal  MntRegularProds  \
0                 0              0      1529             1441
1                 0              0        21               15
2                 0              0       734              692
3                 0              0        48               43
4                 0              1       407              392

   AcceptedCmpOverall
0                   0
1                   0
```

```
2                    0
3                    0
4                    0

[5 rows x 39 columns]
```

[ ]:
```
sns.heatmap(data3.corr(), annot=True, fmt='.3f')
```

[ ]:
```
<Axes: >
```



[ ]:
```
#         DataFrame
def make_corr_df(df):
    cr = data3.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
```

[ ]:
```
#
def corr_groups(cr):
    grouped_feature_list = []
    correlated_groups = []

    for feature in cr['f1'].unique():
        if feature not in grouped_feature_list:
            #
            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
```

[ ]:
```
#
corr_groups(make_corr_df(data3))
```

[ ]:
```
[['MntTotal',
  'MntWines',
  'MntMeatProducts',
  'Income',
  'NumCatalogPurchases',
  'NumStorePurchases',
  'MntFishProducts',
  'MntSweetProducts',
  'MntFruits',
  'Kidhome',
  'NumWebPurchases',
  'NumWebVisitsMonth',
  'AcceptedCmp5',
  'AcceptedCmpOverall',
  'MntGoldProds',
  'AcceptedCmp1',
  'MntRegularProds'],
 ['AcceptedCmpOverall', 'MntWines', 'AcceptedCmp5', 'AcceptedCmp4'],
 ['education_Graduation', 'education_PhD'],
 ['marital_Married', 'marital_Single', 'marital_Together'],
 ['AcceptedCmpOverall', 'AcceptedCmp2'],
 ['education_Graduation', 'education_Master'],
 ['AcceptedCmpOverall', 'AcceptedCmp3'],
 ['AcceptedCmpOverall', 'AcceptedCmp5', 'Response'],
 ['Teenhome', 'NumWebVisitsMonth', 'NumDealsPurchases'],
 ['Teenhome', 'Age'],
 ['education_Graduation', 'education_2n Cycle']]
```

0.10

[ ]:
```
X3_ALL = data3.drop(['Recency'], axis=1)
```

[ ]:
```
#
X3_train, X3_test, y3_train, y3_test = train_test_split(X3_ALL,␣
 ↪data3['Recency'],
                                                        test_size=0.2,
                                                        random_state=1)
```

[ ]:
```
#       L1-
e_lr1 = LogisticRegression(C=1000, solver='liblinear', penalty='l1',␣
 ↪max_iter=500, random_state=1)
e_lr1.fit(X3_train, y3_train)
#
e_lr1.coef_
```

[ ]:
```
array([[ 2.36611132e-05, -4.37617142e-01, -1.06800247e-02, …,
        -1.63170820e-03, -8.91019230e-05, -1.06604264e+00],
       [-4.08684338e-06,  2.11252986e-01,  3.26968614e-01, …,
         2.06092631e-04, -5.67253475e-04, -3.79701869e-02],
       [ 2.97309082e-05,  2.78608379e-01,  7.84637108e-01, …,
         7.60268573e-04,  9.22122307e-04, -2.72190745e-01],
       …,
       [-3.33796349e-06,  5.87316491e-01,  8.31406329e-01, …,
         1.86837335e-04,  9.46010783e-05,  2.65448813e-02],
       [-3.64459359e-05, -1.11298763e+00,  2.33999344e-01, …,
        -2.42398207e-04,  1.69350949e-04, -6.53642523e-02],
       [ 4.67890738e-05, -1.67844004e+00,  7.82124718e-01, …,
        -4.24294712e-05, -6.97529794e-04,  1.87736876e-02]])
```

[ ]:
```
#              "     "
from sklearn.feature_selection import SelectFromModel
sel_e_lr1 = SelectFromModel(e_lr1)
sel_e_lr1.fit(X3_train, y3_train)
sel_e_lr1.get_support()
```

[ ]:
```
array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True])
```

[ ]:
```
e_lr2 = LinearSVC(C=0.01, penalty="l1", max_iter=2000, dual=False)
e_lr2.fit(X3_train, y3_train)
#
e_lr2.coef_
```

/Users/seralekhin/BMSTU_Labs/.env/lib/python3.11/site-

```
packages/sklearn/svm/_base.py:1244: ConvergenceWarning: Liblinear failed to
converge, increase the number of iterations.
  warnings.warn(
```

[ ]:
```
array([[-8.74797155e-07,  0.00000000e+00,  0.00000000e+00, …,
        -1.25324679e-05,  0.00000000e+00,  0.00000000e+00],
       [-1.70092629e-06,  0.00000000e+00,  0.00000000e+00, …,
         0.00000000e+00,  5.95950619e-07,  0.00000000e+00],
       [-3.51928043e-06,  0.00000000e+00,  0.00000000e+00, …,
        -1.71853431e-05,  6.01462622e-05,  0.00000000e+00],
       …,
       [-2.93342572e-06,  0.00000000e+00,  0.00000000e+00, …,
         0.00000000e+00,  1.59097083e-05,  0.00000000e+00],
       [-2.97155805e-06,  0.00000000e+00,  0.00000000e+00, …,
         1.93890000e-05,  0.00000000e+00,  0.00000000e+00],
       [ 4.63639333e-07,  0.00000000e+00,  0.00000000e+00, …,
         6.02217848e-04, -1.19241087e-04,  0.00000000e+00]])
```

[ ]:
```
#           False  . .
sel_e_lr2 = SelectFromModel(e_lr2)
sel_e_lr2.fit(X3_train, y3_train)
sel_e_lr2.get_support()
```

```
/Users/seralekhin/BMSTU_Labs/.env/lib/python3.11/site-
packages/sklearn/svm/_base.py:1244: ConvergenceWarning: Liblinear failed to
converge, increase the number of iterations.
  warnings.warn(
```

[ ]:
```
array([ True, False, False,  True,  True,  True,  True,  True,  True,
        True,  True, False,  True, False, False, False, False, False,
       False, False, False,  True, False,  True,  True, False, False,
       False, False, False, False, False, False, False, False,  True,
        True, False])
```