Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Технологии машинного обучения»

Отчет по лабораторной работе №5:
«Линейные модели, SVM и деревья решений»

Выполнил:
студент группы ИУ5-63
Курганова Александра

Подпись и дата:

Проверил:

Подпись и дата:

Москва, 2019 г.

Отчет по лаборторной работе №5 "Линейные модели, SVM и деревья решений"

In [237]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import warnings
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.svm import LinearSVC
from sklearn.preprocessing import Normalizer
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score, f1_score
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV, KFold, ShuffleSplit
%matplotlib inline
sns.set(style="ticks")
%matplotlib inline
sns.set(style='ticks')
```

подготовка датасета

Medical Cost (personal datasets)

In [175]:
```
data = pd.read_csv('reg.csv')
```

In [176]:
```
data.head()
```

Out[176]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [177]:
```
data.shape
```

Out[177]:
```
(1338, 7)
```

In [178]:
```
data.isnull().sum()
```

Out[178]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [179]:
```
data.dtypes
```

Out[179]:
```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

In [180]:
```
cat_coll = []
for col in data.columns:
    if data[col].dtype == 'object':
        cat_coll.append(col)
en_cat = {}
for col in cat_coll:
    le = LabelEncoder()
    data[[col]] = le.fit_transform(data[col])
    en_cat[col] = le
data.dtypes
```

Out[180]:
```
age           int64
sex           int64
bmi         float64
children      int64
smoker        int64
region        int64
charges     float64
dtype: object
```

In [181]:
```
# разделение выборки на обучающую и тестовую
x = data.iloc[:, 0:5]
y = data.iloc[:, 6]
```

In [182]:
```
x.dtypes
```

Out[182]:
```
age           int64
sex           int64
bmi         float64
children      int64
smoker        int64
dtype: object
```

In [183]:
```
y.dtypes
```

Out[183]:
```
dtype('float64')
```

In [184]:
```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.3, random_state=42)
len(xtrain), len(xtest), len(ytrain), len(ytest)
```

Out[184]:
```
(936, 402, 936, 402)
```

обучение моделей

линейная регрессия

In [185]:
```
lr = LinearRegression()
lr.fit(xtrain, ytrain)
```

Out[185]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [186]:
```
lr.intercept_, lr.coef_
```

Out[186]:
```
(-12538.439849853163,
 array([  261.91061673,   136.65119758,   333.36099462,   432.1792927 ,
        23618.76182167]))
```

In [187]:

```
y_predtest = lr.predict(xtest)
y_predtest
```

```
Out[187]: array([ 8504.25952786,  6897.83941087, 36805.01178274,  9525.01640181,
        26834.15783554, 11236.53827843,   -56.51533813, 16996.53931939,
          558.86901551, 11292.49638115, 28504.83781808,  9398.41804442,
         5353.29915879, 38679.23753917, 40572.74207294, 37372.20787154,
        15387.38620563, 36033.31348368,  9292.40371143, 31304.88883148,
         4274.20100087, 10592.29487706,  2708.68730205,  6493.02810907,
        11227.74003336, 12498.97075764, 14877.16806499,  5963.85891638,
         9503.12017865,  2360.10282672,  9434.17913244, 12999.67895742,
         4585.98098388,  3265.96837276,  4840.89651399, 12653.62349137,
         2194.59265213,  9123.3897013 , 33175.85413453, 32812.24167496,
         4132.88277765,  4243.29138982, 14495.05758702, 11538.30865769,
         9023.27584285, 12650.54437942,  5036.97193595,  3406.53561573,
        35766.6625433 ,  9379.5276352 , 16062.93310871,  2535.59415356,
        12177.97650916,  1021.65843332, 13742.82492927, 12262.32230678,
         3942.90077261, 32097.02477108, 13686.62484229, 12427.07590785,
        14562.80863291, 10557.76332545, 16905.62875202,  7655.1890887 ,
        11373.4329289 ,  3872.95410672, 27013.19085325, 11314.83155364,
         1947.12765493,  6528.00420329, 10334.0189268 , 11141.36723127,
        11144.19170379,  9457.97855155, 12242.17723173,  6864.5033114 ,
         6828.90388721, 10679.2641989 ,  6660.32882643,  9111.45334274,
         3947.34563302, 36354.50315789,  6641.21611054, 30192.97241472,
        34939.30712732, 34921.71765958,  7276.27432785, 12929.20186644,
         9817.25737478, 14977.02431552, 17263.76406273, 35506.57070869,
        32531.35623517,  5797.56374799, 32209.85837182,  9935.12924576,
        29674.42577177,  3700.68530721, 28119.8326795 ,  5315.10605979,
         5220.17699689,  2071.99461253, 11757.66027701, 15596.07566839,
        11460.41329585,  4479.08875221, 10217.48599406, 32065.48064352,
         -530.13699665, 33021.06350798,  3608.76198753, 10351.35375509,
        13883.78463378, 31058.34518156, 10988.21107693,  4269.5157975 ,
        12845.30599255, 32107.57563287,  8284.1301888 ,  3175.09163801,
         7882.95348274, 10601.851235  , 14739.66577893,  5838.39141632,
         3608.40892066, 10085.16850355, 11036.88181043, 10808.16932962,
        14648.59354772,  7593.31481706,  5486.83345982,  9502.5701467 ,
         9447.11558813, 12010.26358196,  8610.3148261 , 15850.80992779,
         7990.56995517, 32218.04891058, 35588.79984832, 31031.86620517,
         5852.75269273, 12124.93973582,  6215.53381214, 14454.64212853,
         2726.68876747, 33448.99343673,  6258.04639246,  5281.59421218,
        14244.87927064,  7364.77256184, 38545.8087982 ,  2859.61550364,
         6027.07376757, 31241.22244649, 11369.13162531,  8062.46480496,
        14540.8911221 ,  9994.35084105, 27094.20622156, 33051.48372991,
        14125.94815955,  1452.4199672 , 13758.89086597,  1792.83351063,
```

```
 5819.59793469, 11857.1776157 , 40148.40032285, 36396.75571766,
33624.08677948,  4110.16370537,  7827.94884791,  8942.48582041,
12032.23862568,  4795.33715448,  2257.6766724 , 32429.6543216 ,
25427.37443824, 17854.94388055, 26247.97836193, 10117.77315891,
36828.12481641, -1000.67248132,  6864.06429134,  8032.96593032,
 3995.77990482,  4863.33731726,  5506.61840611,  4552.97820298,
15311.45870528, 11219.05996403,  7286.38631103,  1898.42465938,
  950.90157348, 32301.56497811, 16781.07700597, 12186.61151983,
 1040.71913884, 12096.55756348,   948.34582471,  9091.13244881,
 1681.85109549, 33957.08205205, 11061.11161712,  2527.95916931,
25698.12269471, 26402.59324033,  9497.80605204,  1614.05356179,
13554.36488469,  1375.41354916, 10970.5948697 , 10816.64124731,
16398.47011646, 27047.39771877,  7263.01863791,  4629.65123175,
 5960.9051128 , 13488.26089546, 11353.54235183,  8464.89065431,
 4869.33782931, 12296.70499117, 14096.72912449, 36031.20772936,
 4064.46645304, 29111.34674737,  -694.90164283,  2772.02589103,
11398.662762  , 15848.37798753,  5382.68695082,  6836.16755614,
 4122.00708833, 31746.56533914,  7193.08605633, 12716.18421917,
 5564.76667698,  9966.06716687, 36402.70981263,  4617.84570411,
 9288.37099015, 31511.05494896,  5505.27946786,  4677.57643684,
  966.87610515,  4763.74668056,  4826.843356  ,  6912.40170628,
18826.43518202, -1598.33674847,  2567.2579255 , 11138.81700506,
 3395.77107975,  9801.08021402,  3479.2548145 ,  5185.72971769,
12946.98109591,  6146.00480794,  7937.45436126,  7043.17067778,
 8798.64963302, 10468.69114706, 28006.03281425, 39414.97932447,
11709.79414418,  7325.43595033, 41028.14003009, 12652.12691199,
 7024.35894769,  7878.69852551,  9368.26654624, 11277.92000142,
10058.53188603, 17923.06064849,  1324.26588745, 23120.2941195 ,
12251.28911777, 32740.79129706,  4841.48440127, 13483.61354741,
10271.00813382, 17334.92753915, 10407.70984191, 11429.63301587,
32644.37117734,  2886.70204489, 13828.891223  , 39635.17339979,
 5103.47383856,  6021.96219931,  2816.83419957, 11894.56045004,
25001.76386727, 13774.62355483,  9649.09134319,  9753.71615672,
13800.56106102,  1303.96317126,  2574.36961729, 30887.27190492,
30389.45277247, 13821.67397116,  3499.73327885, 25302.84086992,
13816.11786028, 30918.60781012,  2897.28531017, 39273.52313771,
11412.02960527,  4937.01012536,  7159.06916471,  2755.61241001,
25808.7393878 , 14767.60340866,   964.46377177, 13168.78996585,
13004.20809023, 14813.38500411, 35263.84306666, 14267.18211036,
31929.35826095, 10364.00173872, 18662.88566557,  6355.24437337,
 9182.3945832 ,  9816.00823149, 15646.96876137,  9602.73594478,
 7785.46867108, 15466.30127146, 12360.29255559, 14411.65819538,
```

```
 7869.59223275, 26277.44497452,  9520.8080031 ,  1769.72047696,
 4496.81597133, 14622.26357999, 35986.61613422, 10007.93984956,
12702.65428292,  4929.07401387,  4863.98987008,  4376.58929987,
 2215.11760427,  8895.48189188,  7407.10939401,  2626.81843263,
13520.07367605,  8756.88251039,  6368.27775664,  1102.86772751,
 9851.89455716,  5078.3551588 , 32913.00815694, 28655.75343521,
37125.48288019,  5929.31463888,  8952.3486867 ,  8540.75348912,
 3800.6936056 , 31198.69011786,  6749.38813748, 28726.12482468,
36028.31859131,  7296.6826041 , 13343.70748973,  9676.91243847,
 8264.65175086, 12275.46132083, 29782.12812667, 17527.17125884,
11715.6694893 ,  3931.70994246,  -805.65629947, 11724.35091704,
31297.78266225, 13326.37825471, 11717.24474781,  7667.34775393,
 3192.20415963,  7565.86800559,  7633.90043041, 11073.50503204,
33704.45906951, 39542.38235325, 12319.84476434,  8201.39198245,
16369.15394166, 15506.4607321 ,  9982.79988213,  9543.83676437,
 8781.5371114 ,  3172.90154057, 10532.66970439,  4158.86670092,
11053.07298477, 15473.32869085,  6927.81509017,  1581.73171449,
14816.69180385,   602.54485665])
```

In [188]:

```
y_predtrain = lr.predict(xtrain)
y_predtrain
```

```
Out[188]: array([ 1.38256364e+04,  8.84686317e+03,  1.35354871e+04,  3.58512392e+04,
                  3.38103569e+04,  3.55911066e+04,  6.13903101e+03,  5.67096539e+03,
                  1.67676441e+04,  7.95099695e+03,  3.77437564e+04,  5.22458952e+03,
                  8.16931607e+03,  1.05685742e+04,  3.04312340e+04,  5.03648650e+03,
                  3.33007217e+03,  1.61765824e+04,  3.33935980e+03,  6.42236104e+03,
                  9.87609203e+03, -5.24914346e+02,  2.98210581e+04,  8.10253272e+03,
                  1.03688911e+04,  5.83263995e+03,  7.93654581e+03,  1.18355359e+04,
                  2.93262858e+04,  9.78747712e+03,  1.10151810e+04,  5.93664857e+03,
                  3.99691789e+03,  1.28137343e+03,  8.07719180e+03,  1.14957032e+04,
                  1.05625879e+04,  9.04557861e+03,  5.88077324e+03,  4.47638952e+03,
                  7.08441041e+03,  3.40634889e+04,  3.34643225e+04,  3.40543859e+03,
                  4.02327139e+04,  2.88466581e+04,  1.12328980e+04,  4.37549228e+03,
                  1.71477741e+04,  3.94212298e+03,  9.56114464e+03,  2.97400191e+04,
                  1.66160492e+04,  1.23144701e+04,  8.29262731e+03,  1.26467340e+04,
                  2.42553301e+03,  1.39699085e+04,  3.81022308e+03,  1.26318242e+04,
                  1.10284366e+04,  5.56582585e+03,  1.08490617e+04,  5.75518906e+03,
                  1.01960721e+04,  8.31718491e+03,  3.08887291e+04,  1.39102636e+04,
                  2.74891286e+03,  3.35126977e+04,  2.86194439e+04,  4.22536867e+03,
                  6.65045883e+03,  2.95562387e+04,  8.85756311e+03,  1.41702260e+04,
                  1.11036186e+04,  4.16150127e+03,  8.73410287e+03,  1.11017816e+04,
                  5.38602056e+03,  3.65154329e+03,  6.43634258e+03,  4.63054025e+03,
                  4.34348959e+03,  3.42290905e+04,  1.11466364e+04,  3.40039931e+04,
                  2.69765298e+04,  1.80426261e+04,  1.66235352e+04,  6.63552218e+03,
                  1.49760975e+04,  1.13971394e+04,  1.25228152e+04,  1.73368812e+04,
                  1.16453176e+04,  7.27920988e+03,  1.43734596e+04,  1.00958739e+04,
                  5.08977375e+03,  1.39459852e+04,  5.00368225e+03,  1.00929146e+04,
                  1.64198839e+04,  7.99973372e+02,  1.13069562e+04,  2.61853389e+04,
                  1.14201877e+04,  4.08163502e+04,  3.35687434e+04,  1.32210362e+04,
                  3.15108959e+03,  2.88961981e+04, -8.24939241e+02,  2.98758815e+04,
                  1.02783476e+04,  1.83164131e+03,  2.32905573e+04,  7.22915932e+03,
                  1.10892195e+04,  6.71878435e+02,  6.82371357e+03,  7.36983764e+03,
                  1.14319469e+04,  3.54178321e+03,  1.36569093e+04,  9.84508392e+03,
                  3.14460440e+04,  1.54091011e+04,  6.88581158e+03,  5.30796756e+03,
                  8.88684681e+03,  3.70948980e+04,  1.34515912e+04,  1.18935604e+04,
                  6.38601920e+03,  2.33749341e+03,  9.84815433e+03,  3.95915031e+04,
                  1.36416211e+04,  5.70943980e+03,  1.23543300e+04,  3.43051433e+04,
                  3.42352440e+03,  1.20068047e+04,  1.54567449e+04,  1.38768488e+04,
                  1.44376604e+04,  2.59596676e+04,  9.99187391e+03,  1.90435187e+03,
                  1.04277721e+04,  3.02355511e+04,  9.35323856e+03,  2.88306048e+04,
                  3.69564224e+04,  1.23256609e+04,  8.95370898e+03,  9.27766256e+03,
                  3.79103959e+04,  3.78658508e+04,  3.79022149e+03,  3.13630371e+04,
```

```
-8.93473671e+02,  6.00996118e+03,  3.72197129e+04,  6.73738711e+03,
 1.30733464e+03,  1.33844942e+04,  1.01621749e+04,  3.06666497e+03,
 7.08004990e+03,  1.20189309e+04,  5.55020990e+03,  3.35290001e+04,
 8.60687566e+03,  1.39077079e+04,  9.26876739e+03,  2.76621422e+04,
 7.78969115e+03,  1.75260796e+03,  1.13696604e+04,  5.11203010e+03,
 1.75643966e+04,  8.01852024e+03,  4.16167709e+03,  9.15364676e+03,
 1.91712612e+04,  1.10425080e+04, -2.60628542e+01,  6.86197990e+03,
 9.01765514e+03,  8.04480931e+03,  1.14957300e+04,  1.47643541e+04,
 3.36729009e+04,  5.59706489e+03,  2.84037325e+04,  4.99111340e+01,
 3.49725265e+04,  2.60641207e+04,  1.00526890e+04,  1.30461919e+04,
 1.54972842e+04,  5.77244507e+03,  1.80852827e+03,  2.98283865e+04,
 1.22259481e+04,  1.03402220e+04,  7.17390018e+03,  3.45656094e+04,
 1.30413223e+04,  5.28493334e+03,  5.57206282e+03,  6.59434294e+03,
 7.37912528e+03,  3.91411592e+04,  3.86790152e+04,  1.35360300e+04,
 3.97267042e+04,  7.80547026e+03,  1.02817868e+04,  3.18394943e+04,
 1.00150192e+04,  3.59467184e+04,  4.94265044e+03,  3.32363360e+04,
 3.81279836e+04,  2.99522266e+04,  1.25563088e+04,  7.31282736e+03,
 1.28388878e+04,  3.93778928e+03,  2.63601776e+04,  3.36975374e+04,
 2.51800476e+03,  8.57801668e+03,  3.04539673e+04,  2.81257094e+04,
 5.96612776e+03,  3.25396959e+04,  1.04314714e+04,  1.21553922e+04,
 1.30266867e+04,  4.01794731e+04,  1.82308505e+03,  2.58840790e+04,
 2.77942375e+04,  9.29791333e+03,  2.74469864e+04,  2.98585845e+04,
 9.35259859e+03,  3.00540451e+04,  2.25595345e+04,  1.09954071e+04,
 1.17902453e+04,  1.00649966e+04,  1.16188836e+04,  2.42965619e+04,
 5.11478848e+03,  1.11079200e+04,  1.59848999e+04,  7.90745194e+03,
 1.38027315e+04,  9.38933297e+03,  2.99349565e+04,  7.53775456e+03,
 1.20581564e+04,  1.19303412e+04,  1.19077050e+04,  2.78275736e+04,
 6.57324276e+03,  9.42302206e+03,  2.89620980e+04,  1.63043708e+04,
 4.61976156e+03,  7.69622484e+03,  3.58050131e+04,  3.33470118e+04,
 8.24100966e+00,  1.11487067e+04,  1.34519119e+04,  9.32671310e+03,
 3.78042315e+03,  1.63538785e+04,  1.08233787e+04,  1.69040736e+03,
 9.94656912e+03,  9.20832346e+03,  4.45485895e+03,  2.94203259e+04,
 3.47022175e+03,  1.49496564e+04,  9.60965773e+03,  9.23807145e+03,
 6.40135929e+03,  6.90720587e+03,  1.42775627e+04,  8.01015940e+03,
 7.61942807e+03,  3.76125233e+04,  9.17794979e+03,  5.19327725e+03,
 4.24921308e+03,  9.18914062e+03,  1.42527042e+04,  2.66843676e+04,
 5.44449663e+03,  3.02010574e+04,  9.49881888e+03,  5.33748772e+03,
 1.09355328e+03,  2.96277088e+04,  5.88793687e+03,  6.91184063e+03,
 1.88001651e+03,  1.17513586e+04,  7.49047645e+03,  3.40628717e+02,
 1.11942014e+04,  1.56701228e+04,  2.53573954e+04,  1.16651885e+04,
 1.20341544e+04,  3.70631498e+04,  2.27845619e+03,  6.23134518e+03,
 5.69063366e+03,  9.89912624e+03,  8.56145419e+03, -2.36695608e+03,
```

```
 9.40892801e+03,  3.86703242e+03,  1.12410745e+04,  4.06225563e+04,
 6.32754313e+03,  6.12610878e+03,  4.46030390e+03,  4.93701013e+03,
 3.30169000e+03,  4.15550083e+03,  3.91792734e+04,  2.78410050e+04,
 1.07853841e+04,  7.50762683e+03,  3.73458723e+04,  2.70263478e+03,
 2.76955894e+04,  2.92667899e+04,  1.67677694e+04,  1.10506339e+04,
 1.12331386e+03,  2.14719626e+03,  7.77079211e+03,  3.55122434e+04,
 3.91478461e+04,  7.16972405e+03, -1.50184283e+02,  5.03957970e+03,
 1.14281364e+04,  3.38087956e+04,  2.54507169e+03,  2.26852548e+03,
 7.85076090e+03,  1.04107975e+04,  8.66157364e+03,  7.39480186e+03,
 9.63248387e+03,  6.43532976e+03,  1.17988015e+04,  1.38920918e+04,
 1.32348475e+04,  3.04882979e+04,  1.61284153e+04,  6.58674036e+03,
 6.84776196e+03,  5.07811877e+03,  1.04127512e+04,  1.53746593e+04,
 1.41748411e+04,  1.43379390e+04,  3.22987397e+03,  1.23101364e+04,
 3.59084521e+04,  8.65616109e+03,  4.06783240e+03,  4.50772542e+03,
 3.30403717e+04,  1.30902926e+04,  1.31098962e+04,  6.20597745e+03,
 5.95657141e+03,  1.44189402e+04,  3.33457610e+03,  7.49570463e+03,
 9.36897118e+03,  1.33651537e+04,  1.62462817e+04,  1.59548974e+04,
 1.41926527e+04,  8.87796275e+03,  5.10813541e+03,  1.19406558e+04,
 1.49117123e+04,  1.18523670e+04,  5.88434317e+03,  1.23549967e+04,
 1.53270802e+04,  1.11805154e+02, -1.62831243e+03,  3.52152188e+04,
 6.26498227e+03,  8.44249634e+03,  7.69427665e+03,  1.56996540e+04,
 6.76219936e+03,  7.70275409e+03,  1.48405954e+04,  8.90648834e+03,
 1.11842274e+04,  7.09562502e+03,  1.15519301e+04,  1.47302205e+04,
 1.58725431e+04,  1.45128423e+04,  2.95896354e+04,  6.56628871e+03,
 1.06591893e+04,  1.05563383e+04,  1.59858015e+04,  1.23183214e+04,
 5.18760136e+02,  6.78908495e+03,  1.00446939e+04,  2.44592098e+04,
 1.80370505e+04,  3.76288170e+04,  8.23753840e+03,  2.91149996e+04,
 5.62440041e+03,  1.27785030e+04,  1.13148780e+04,  6.50978036e+03,
 5.61557550e+03,  1.00240721e+04,  5.60373211e+03,  3.65747705e+04,
 2.65417159e+04,  2.44864989e+04,  9.12072832e+03,  5.60728789e+03,
 7.37105445e+03,  3.02799993e+04,  1.56974119e+04,  2.76026195e+04,
 1.16610123e+04,  1.32499921e+04,  2.89074212e+04,  1.15015279e+04,
 2.54940734e+04,  2.30153122e+04,  1.08607237e+04,  2.56714677e+03,
-1.38993453e+02,  3.34702442e+04,  1.02430058e+04,  7.84398252e+03,
 6.26060214e+03,  7.32801851e+03,  7.85598907e+03,  3.60828408e+04,
 1.14878728e+04,  7.91264076e+03,  3.73005887e+03,  6.02932839e+03,
 1.12849543e+04,  8.19182154e+03,  1.25851733e+04,  2.30765401e+03,
 8.73344168e+03,  1.50100538e+04,  1.06352661e+04,  6.93543040e+03,
 2.21909111e+03,  1.18122203e+04,  1.10961877e+04,  3.80999617e+04,
 2.90982669e+04,  3.72324272e+03,  1.52066777e+04,  7.80303112e+03,
 1.11310653e+04,  1.81059915e+04,  1.34653110e+04,  1.07986075e+04,
 1.72786866e+04,  1.58614451e+04,  8.80822008e+03,  1.20737283e+03,
```

```
1.46900146e+04,  4.26103829e+03,  1.74582711e+04,  3.44187027e+04,
2.92736935e+04,  5.16283892e+03,  3.06352509e+04,  4.92140677e+03,
7.54340760e+03,  1.05490171e+04,  3.43514481e+04,  2.76034438e+04,
7.52786488e+03,  3.26675574e+04,  1.05839231e+04,  5.78864184e+03,
3.68021237e+03,  1.55738516e+04,  9.55453656e+03,  3.35659512e+04,
4.11646336e+04,  9.87948478e+03,  2.68875224e+04,  2.94817840e+03,
2.66688635e+03,  5.93898215e+03,  2.37043698e+03,  2.65089086e+04,
3.01795733e+04,  4.37563497e+02,  1.18108025e+04,  9.22040330e+03,
2.61306676e+04,  1.00023192e+04,  2.54843004e+04,  1.14817753e+04,
1.29747415e+04,  3.14996095e+04,  5.68207740e+03,  2.42067444e+04,
2.98712143e+04,  1.38647239e+04,  8.06952449e+03,  1.56239676e+03,
1.36741793e+04,  1.19704880e+04,  1.31936682e+04,  5.60276428e+03,
9.36629876e+03,  3.65778086e+04,  3.67476737e+04,  4.77376569e+03,
1.08669064e+03,  2.14197361e+03,  8.35642984e+02,  4.41882920e+03,
2.84220153e+04,  1.01921774e+04,  1.53796077e+04,  1.62148346e+04,
2.95717450e+04,  8.42960630e+03,  6.50795599e+03,  4.55753967e+03,
7.50425536e+03,  1.01987011e+04,  5.38452398e+03,  2.54907398e+04,
1.16105638e+04,  5.52884243e+03,  1.91437175e+04,  1.08872815e+04,
8.26087919e+03,  1.08354262e+04,  8.63929034e+03,  2.73565992e+04,
1.85381243e+04,  1.56322039e+04,  1.51328291e+04,  5.80725800e+03,
8.83792165e+03,  1.77572099e+03,  8.30765536e+03,  2.62298067e+04,
1.70915472e+04,  3.74331456e+03,  9.88387043e+03,  3.39980658e+04,
3.15676885e+04,  1.20817139e+04,  2.55347901e+03,  4.74363394e+03,
5.92971269e+03,  1.85361635e+03,  3.11545163e+04,  7.55762546e+03,
9.76144262e+03,  2.56840892e+04,  2.77202589e+03,  2.21511760e+03,
5.60681661e+03,  1.42343370e+04,  1.00374064e+04,  5.03916203e+03,
3.35549699e+04,  2.85126304e+04,  1.22734020e+04,  9.12777535e+03,
1.55208094e+04,  2.48477981e+03,  1.03782630e+04,  1.33788271e+04,
1.15660044e+04,  3.04517233e+03,  1.24078576e+04,  1.59225417e+04,
1.38441666e+04,  3.47574677e+04,  1.23861568e+04,  2.47414461e+03,
1.23738351e+04,  2.81961218e+03,  1.46874392e+04,  4.05193668e+03,
1.17371621e+04,  8.23784498e+03,  1.42533117e+04,  5.56070169e+03,
4.22050615e+03,  1.22129006e+04,  4.90034042e+03,  5.89072908e+03,
5.35938394e+03,  3.08419348e+04,  2.70396820e+03, -6.16588619e+02,
4.79351278e+03,  3.04203765e+04,  5.04586152e+03,  8.07085790e+03,
1.20340433e+04,  1.04413091e+04,  9.22976976e+03,  1.17488817e+04,
5.76711126e+03,  1.22991819e+04,  6.85753503e+03,  3.69101780e+03,
1.01245050e+04,  3.73435473e+03,  1.11478965e+04,  3.81758301e+04,
3.35776795e+03,  1.57114845e+03,  1.35486134e+04,  9.52320059e+03,
1.06609349e+04,  2.84886088e+04,  1.40989656e+04,  1.19556570e+04,
3.07009412e+04,  2.92550984e+03,  1.23214848e+04,  3.37954146e+04,
4.31246733e+03,  2.07717078e+03,  9.41938195e+03,  1.28044729e+04,
```

```
 1.04143912e+04,  1.26711285e+04,  2.70036163e+04,  2.43237273e+04,
 1.17739429e+04,  3.51753219e+03,  6.74435546e+03,  1.24364424e+04,
 2.69532591e+04,  4.72027899e+03,  3.17876280e+04,  1.28264479e+04,
 7.73158081e+03,  1.30980528e+04,  3.38335792e+03,  1.45941036e+04,
 1.16575676e+04,  7.69575356e+03,  3.90444900e+04,  9.25973984e+03,
 1.34179607e+03,  5.22422933e+03,  3.14408536e+04,  6.38796904e+02,
 1.34400079e+04,  2.18223150e+03,  1.19724559e+04,  2.35858040e+04,
 1.52148541e+04,  1.06835192e+04,  2.75718014e+04,  8.48900931e+03,
 7.62137618e+03,  3.09479703e+04,  3.60598444e+04,  2.98376158e+02,
 6.32543179e+03,  9.54974982e+03,  8.58523393e+03,  9.52193191e+03,
 4.02063849e+03,  1.19579258e+04,  1.23226227e+04,  3.83275684e+04,
 1.14043190e+03,  1.45942557e+04,  3.21233744e+04,  1.00707425e+04,
 1.24959296e+04,  3.04956373e+04,  3.31237907e+04,  8.09742122e+03,
 6.61492704e+03,  4.55495159e+03,  1.68471342e+03,  6.06717401e+03,
 2.98890582e+04,  1.60997140e+04,  6.25774541e+03,  3.58999099e+04,
 2.71106252e+04,  7.41720170e+03,  5.45038606e+03,  3.42402025e+04,
 5.24126303e+03,  2.37201951e+03,  1.25366643e+04,  1.47540199e+04,
 7.98285063e+03,  1.43660988e+04,  1.67544349e+04,  3.52921086e+04,
 7.75169355e+03,  3.47272429e+04,  1.52130848e+04,  2.98332167e+04,
 1.12526774e+04,  1.37149787e+04,  1.09109233e+04,  5.10975573e+03,
 3.34065543e+03,  4.79014684e+03,  5.97046699e+03,  2.75746912e+03,
 3.62876358e+03,  9.64602093e+03,  3.80853262e+04,  8.43844394e+03,
 2.83105758e+03,  3.50946800e+04,  7.68623809e+03,  2.35890730e+03,
 1.11885287e+04,  1.14538108e+04,  1.58639166e+04,  9.52863691e+03,
 2.68326750e+03,  8.99993505e+03,  7.36558269e+03,  3.31594730e+04,
 1.48472364e+03,  3.29512067e+04,  4.00238253e+03,  7.55642280e+03,
 1.35078054e+04,  3.80442520e+03,  3.97399599e+04,  2.55559407e+04,
 2.97222343e+04,  3.08909656e+04,  7.75183144e+03,  3.53353353e+03,
 9.81122460e+03,  1.26672912e+04,  1.81339080e+03,  6.13777634e+03,
-3.08465557e+01,  1.08279488e+04,  6.90128417e+03,  2.98818172e+03,
 1.11825281e+04,  2.77893482e+04,  1.17273102e+04,  1.62777469e+04,
 1.12258707e+04, -6.58258744e+02,  3.23311175e+04,  7.46462322e+03,
 9.58620586e+03,  3.13401195e+04,  7.68579921e+03,  1.03307768e+04,
 2.83305771e+03,  1.22045342e+04,  8.48600356e+03,  1.29007409e+04,
 7.76827733e+03,  1.32900096e+04,  1.60504932e+04,  3.64675126e+04,
 3.50922490e+03,  8.29986281e+03,  2.98771044e+03,  1.18516680e+04,
 1.10846367e+04,  9.18131574e+03,  1.27614102e+04,  3.43919235e+03,
 1.11077498e+04,  1.28915321e+04,  1.22755921e+04,  1.48806971e+04,
 3.22853503e+03,  6.07613696e+03,  3.32519252e+04,  9.58681342e+03,
 3.25345195e+04,  2.45955550e+03,  4.20861628e+03,  3.27439026e+04,
 1.45811167e+04,  9.66740250e+03,  1.01730056e+04,  1.11069396e+04,
 3.14208341e+03,  2.29556871e+03,  1.31645547e+04,  2.61481741e+03,
```

```
6.53654079e+03,  1.09599865e+04,  1.55595439e+04,  3.92212446e+04,
1.78470867e+04,  4.91556390e+03,  4.07831003e+03,  8.83211656e+03,
1.16624301e+04,  7.02516914e+03,  3.44932504e+04,  9.67784780e+03,
2.37924878e+04,  3.23330585e+04,  1.26701284e+04,  5.31972667e+03,
5.36644996e+02,  1.47563858e+04,  6.91627126e+03,  3.34718469e+03,
1.02102253e+04,  1.77527652e+03,  1.08710903e+04,  8.36952257e+03,
1.34529443e+04,  1.37999676e+04, -2.34282848e+01,  1.07238843e+03,
1.55186389e+04,  3.18730938e+03,  9.47805896e+03,  6.73158923e+03,
1.12096739e+04,  2.80790657e+04,  1.06597394e+04,  1.61815041e+04,
1.53039545e+04, -1.36162363e+03,  2.60370540e+03,  2.84797515e+04,
4.49597352e+03,  7.22306181e+03,  4.85470224e+03,  3.11169899e+04,
1.33631126e+04,  2.17144729e+03,  1.17315652e+04,  1.00406880e+04,
1.52573258e+03,  3.74786498e+03,  9.70655473e+03,  3.91573541e+03,
1.54029895e+04,  1.23018165e+04,  5.23036780e+03,  1.00138024e+04,
7.19502575e+03,  3.46586123e+03,  1.51788598e+04,  6.98562997e+03,
1.10727595e+04,  5.50439045e+03,  2.78586873e+04,  4.83730834e+03,
3.12839249e+04,  8.17187182e+03,  3.24698649e+03,  6.62554816e+03,
2.64446810e+04,  7.44094918e+03,  1.34016067e+04,  4.63354592e+03,
3.50240453e+04,  3.46896141e+04, -1.73218496e+02,  3.76287847e+04,
6.55195417e+03,  1.02354355e+04,  2.70339002e+04,  3.11108555e+02,
9.31404544e+03,  5.36035178e+03,  4.04213128e+03,  1.11868619e+04,
1.63545972e+04,  4.77072754e+03,  1.17500197e+04,  4.53089760e+03,
5.76455552e+03,  1.11959737e+04,  3.96183887e+04,  6.15440343e+03,
1.31758370e+04,  2.29926071e+02,  1.41847828e+04,  4.35553560e+03,
7.79429761e+03,  1.11813902e+04,  3.75033567e+04,  1.19707890e+04])
```

In [189]:

```
ytest.values
```

```
Out[189]: array([ 9095.06825 ,  5272.1758  , 29330.98315 ,  9301.89355 ,
                33750.2918  ,  4536.259   ,  2117.33885 , 14210.53595 ,
                 3732.6251  , 10264.4421  , 18259.216   ,  7256.7231  ,
                 3947.4131  , 46151.1245  , 48673.5588  , 44202.6536  ,
                 9800.8882  , 42969.8527  ,  8233.0975  , 21774.32215 ,
                 5080.096   ,  7441.501   ,  1256.299   ,  2755.02095 ,
                11085.5868  , 10923.9332  , 12644.589   , 18804.7524  ,
                 9715.841   ,  1131.5066  , 15828.82173 , 11842.62375 ,
                 2020.5523  ,  5693.4305  ,  2904.088   ,  7448.40395 ,
                 2597.779   ,  7337.748   , 23887.6627  , 38709.176   ,
                 4687.797   ,  2643.2685  , 11674.13    , 12124.9924  ,
                 4889.9995  , 12333.828   ,  3579.8287  ,  4391.652   ,
                42124.5153  ,  4463.2051  , 13887.204   ,  1719.4363  ,
                28476.73499 ,  1708.92575 , 10594.2257  , 25333.33284 ,
                 3645.0894  , 38746.3551  , 11848.141   , 10564.8845  ,
                13880.949   ,  4753.6368  , 27941.28758 ,  8017.06115 ,
                23045.56616 ,  4133.64165 , 17942.106   , 25992.82104 ,
                 3594.17085 ,  1682.597   ,  6079.6715  ,  9411.005   ,
                 8283.6807  ,  6338.0756  ,  7152.6714  ,  4889.0368  ,
                 4846.92015 , 11454.0215  ,  4349.462   ,  9101.798   ,
                 1391.5287  , 28101.33305 ,  5152.134   , 38511.6283  ,
                44501.3982  , 41097.16175 ,  4837.5823  , 10601.63225 ,
                 8310.83915 , 11264.541   , 15230.32405 , 27037.9141  ,
                23401.30575 ,  5031.26955 , 38282.7495  ,  6875.961   ,
                19719.6947  ,  1880.07    , 18765.87545 ,  6402.29135 ,
                 4527.18295 ,  1743.214   ,  5709.1644  , 12363.547   ,
                13129.60345 ,  1727.54    ,  7731.4271  , 21195.818   ,
                 1702.4553  , 23244.7902  , 23082.95533 ,  2927.0647  ,
                13019.16105 , 37701.8768  ,  9778.3472  ,  1980.07    ,
                30259.99556 , 22478.6     ,  6313.759   ,  2789.0574  ,
                 5594.8455  ,  7261.741   , 11396.9002  ,  1986.9334  ,
                 4719.73655 ,  7749.1564  ,  7345.7266  ,  9288.0267  ,
                12244.531   ,  1837.2819  ,  3972.9247  ,  5934.3798  ,
                 5836.5204  ,  7935.29115 ,  5649.715   , 12347.172   ,
                12404.8791  , 22144.032   , 42983.4585  , 37270.1512  ,
                 5267.81815 ,  9866.30485 ,  2322.6218  , 33471.97189 ,
                 2137.6536  , 23306.547   ,  5261.46945 ,  3761.292   ,
                11436.73815 ,  4751.07    , 46661.4424  ,  2690.1138  ,
                 1146.7966  , 37607.5277  ,  6373.55735 ,  4518.82625 ,
                13555.0049  ,  8547.6913  , 34439.8559  , 39125.33225 ,
                13607.36875 ,  2710.82855 , 14988.432   ,  2396.0959  ,
```

3591.48	7162.0122	48824.45	43578.9394
39556.4945	2632.992	9182.17	6238.298
5757.41345	4239.89265	2154.361	21978.6769
16297.846	13831.1152	32734.1863	11830.6072
29523.1656	3167.45585	8428.0693	5012.471
5209.57885	2855.43755	20277.80751	3554.203
8569.8618	10594.50155	3597.596	7323.734819
2731.9122	38711.	12981.3457	9283.562
2709.1119	12096.6512	2198.18985	8932.084
3176.2877	24393.6224	4266.1658	22493.65964
17085.2676	16577.7795	8827.2099	4296.2712
7804.1605	3208.787	12957.118	22192.43711
9432.9253	17043.3414	6593.5083	4137.5227
4779.6023	14001.2867	12629.8967	5245.2269
2404.7338	6948.7008	6435.6237	42560.4304
2055.3249	34672.1472	1731.677	1639.5631
9377.9047	10977.2063	1534.3045	9644.2525
4529.477	37829.7242	9991.03765	8125.7845
3877.30425	5979.731	43896.3763	1674.6323
13204.28565	44585.45587	3021.80915	3392.9768
1632.03625	2699.56835	20177.67113	4076.497
12592.5345	1621.3402	1875.344	7196.867
3161.454	12029.2867	2719.27975	18218.16139
12146.971	3292.52985	8688.85885	6113.23105
8059.6791	13415.0381	18246.4955	47055.5321
12222.8983	6067.12675	63770.42801	9872.701
9193.8385	8534.6718	27117.99378	8596.8278
12475.3513	13405.3903	2150.469	13747.87235
6610.1097	39047.285	27375.90478	9048.0273
8988.15875	14901.5167	10096.97	8835.26495
38415.474	2721.3208	9877.6077	47269.854
4237.12655	2534.39375	2205.9808	10965.446
15006.57945	10736.87075	9788.8659	10422.91665
9304.7019	3378.91	2155.6815	38126.2465
35491.64	6356.2707	24059.68019	16450.8947
12925.886	36950.2567	2459.7201	46889.2612
5124.1887	14133.03775	6414.178	1720.3537
15817.9857	13462.52	2103.08	12105.32
20781.48892	12235.8392	41949.2441	12643.3778
21223.6758	7954.517	15170.069	3659.346
8232.6388	8027.968	13919.8229	10791.96
17878.90068	10601.412	13217.0945	11944.59435

```
       14358.36437 , 32548.3405  ,  5699.8375  ,  2352.96845 ,
        4340.4409  ,  9391.346   , 42211.1382  ,  8823.279   ,
       14256.1928  ,  7133.9025  ,  5312.16985 ,  3906.127   ,
        2203.47185 , 28340.18885 ,  5484.4673  ,  1622.1885  ,
       11299.343   ,  8026.6666  , 11737.84884 ,  2913.569   ,
        9861.025   ,  2473.3341  , 39983.42595 , 36307.7983  ,
       44400.4064  ,  3172.018   ,  7742.1098  ,  6185.3208  ,
        1880.487   , 21771.3423  ,  6858.4796  , 17179.522   ,
       42760.5022  ,  5478.0368  , 12638.195   ,  5989.52365 ,
        9566.9909  , 10370.91255 , 19594.80965 , 11576.13    ,
        6360.9936  ,  4032.2407  ,  2585.269   , 11658.37915 ,
       21082.16    , 10807.4863  ,  9724.53    ,  3201.24515 ,
        3309.7926  ,  5969.723   ,  8269.044   ,  9414.92    ,
       23967.38305 , 47928.03    , 11842.442   ,  7421.19455 ,
       12950.0712  , 11033.6617  ,  6082.405   ,  3989.841   ,
        7537.1639  ,  3484.331   ,  8116.68    , 17626.23951 ,
       11837.16    ,  9541.69555 ,  4399.731   ,  2200.83085 ,
       11363.2832  ,  1964.78    ])
```

In [190]:

```
# средняя абсолютная ошибка
print('ma (train): {}'.format(mean_absolute_error(ytrain, y_predtrain)))
print('ma (test): {}'.format(mean_absolute_error(ytest, y_predtest)))
```

```
ma (train): 4253.809194427617
ma (test): 4171.01308409371
```

In [191]:

```
# средняя квадратичная ошибка
print('ms (train): {}'.format(mean_squared_error(ytrain, y_predtrain)))
print('ms (test): {}'.format(mean_squared_error(ytest, y_predtest)))
```

```
ms (train): 37878481.85624297
ms (test): 34003912.39316076
```

In [192]:

```
# среднее квадратичное отклонение
print('sq (train): {}'.format(r2_score(ytrain, y_predtrain)))
print('sq (test): {}'.format(r2_score(ytest, y_predtest)))
```

```
sq (train): 0.7413880155089706
sq (test): 0.7680881643600721
```

svm

In [240]:
```
datac = pd.read_csv('clas.csv')[:1000]
```

In [241]:
```
datac.head()
```

Out[241]:

	Agency	Agency Type	Distribution Channel	Product Name	Claim	Duration	Destination	Net Sales	Commision (in value)	Gender	Age
0	CBH	Travel Agency	Offline	Comprehensive Plan	No	186	MALAYSIA	-29.0	9.57	F	81
1	CBH	Travel Agency	Offline	Comprehensive Plan	No	186	MALAYSIA	-29.0	9.57	F	71
2	CWT	Travel Agency	Online	Rental Vehicle Excess Insurance	No	65	AUSTRALIA	-49.5	29.70	NaN	32
3	CWT	Travel Agency	Online	Rental Vehicle Excess Insurance	No	60	AUSTRALIA	-39.6	23.76	NaN	32
4	CWT	Travel Agency	Online	Rental Vehicle Excess Insurance	No	79	ITALY	-19.8	11.88	NaN	41

In [242]:
```
datac.shape
```

Out[242]:
```
(1000, 11)
```

In [243]:
```
datac.isnull().sum()
```
Out[243]:
```
Agency                     0
Agency Type                0
Distribution Channel       0
Product Name               0
Claim                      0
Duration                   0
Destination                0
Net Sales                  0
Commision (in value)       0
Gender                   767
Age                        0
dtype: int64
```
In [244]:
```
datac.dtypes
```
Out[244]:
```
Agency                    object
Agency Type               object
Distribution Channel      object
Product Name              object
Claim                     object
Duration                   int64
Destination               object
Net Sales                float64
Commision (in value)     float64
Gender                    object
Age                        int64
dtype: object
```
In [245]:
```
datac = datac.drop(columns='Gender')
```

In [246]:
```
cat_coll = []
for col in datac.columns:
    if datac[col].dtype == 'object':
        cat_coll.append(col)
en_cat = {}
for col in cat_coll:
    le = LabelEncoder()
    datac[[col]] = le.fit_transform(datac[col])
    en_cat[col] = le
datac.dtypes
```

Out[246]:
```
Agency                    int64
Agency Type               int64
Distribution Channel      int64
Product Name              int64
Claim                     int64
Duration                  int64
Destination               int64
Net Sales               float64
Commision (in value)    float64
Age                       int64
dtype: object
```

In [247]:
```
datac.isnull().sum()
```

Out[247]:
```
Agency                  0
Agency Type             0
Distribution Channel    0
Product Name            0
Claim                   0
Duration                0
Destination             0
Net Sales               0
Commision (in value)    0
Age                     0
dtype: int64
```

In [248]:
```
xc, yc = datac[datac.columns[range(9)]], datac[datac.columns[[9]]]
print('x:\n', xc.columns)
print('y:\n', yc.columns)
```

```
x:
 Index(['Agency', 'Agency Type', 'Distribution Channel', 'Product Name',
       'Claim', 'Duration', 'Destination', 'Net Sales',
       'Commision (in value)'],
      dtype='object')
y:
 Index(['Age'], dtype='object')
```

In [249]:
```
normx = Normalizer().fit(xc)
xc_n = normx.transform(xc)
xc_n
```

Out[249]:
```
array([[ 0.01049344,  0.00524672,  0.        , ...,  0.14166138,
        -0.15215482,  0.05021109],
       [ 0.01049344,  0.00524672,  0.        , ...,  0.14166138,
        -0.15215482,  0.05021109],
       [ 0.05696054,  0.01139211,  0.01139211, ...,  0.        ,
        -0.56390935,  0.33834561],
       ...,
       [ 0.07063224,  0.01177204,  0.01177204, ...,  0.28252897,
         0.43556549,  0.        ],
       [ 0.17459279,  0.0290988 ,  0.0290988 , ...,  0.64017356,
         0.61107476,  0.        ],
       [ 0.05049332,  0.00841555,  0.00841555, ...,  0.40394654,
         0.67324423,  0.        ]])
```

In [250]:
```
xtrainc, xtestc, ytrainc, ytestc = train_test_split(xc_n, yc, test_size=0.3, random_state=42)
len(xtrainc), len(xtestc), len(ytrainc), len(ytestc)
```

Out[250]:
```
(700, 300, 700, 300)
```

In [251]:
```
lsvc = LinearSVC(C=1.0, max_iter=1000, verbose=10)
lsvc
```

Out[251]:
```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
          verbose=10)
```

In [252]:
```
lsvc.fit(xtrainc, ytrainc.values.ravel())
```

```
[LibLinear]
```

Out[252]:
```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
          verbose=10)
```

In [253]:
```
ypred_ctrain = lsvc.predict(xtrainc)
y_pred_ctest = lsvc.predict(xtestc)

print('train accuracy (%): {} %'.format(accuracy_score(ytrainc, ypred_ctrain) * 100))
print('test accuracy (%): {} %'.format(accuracy_score(ytestc, y_pred_ctest) * 100))
```

```
train accuracy (%): 47.714285714285715 %
test accuracy (%): 48.333333333333336 %
```

```
In [254]: print('матрица ошибок: строки — истинное значение, столбцы — предсказанное значение')
          print('train\n', confusion_matrix(ytrainc, ypred_ctrain))
          print('test\n', confusion_matrix(ytestc, y_pred_ctest))
```

```
матрица ошибок: строки — истинное значение, столбцы — предсказанное значение
train
 [[ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 ...
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0 13]]
test
 [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 2]]
```

```
In [255]: print('train:', precision_score(ytrainc, ypred_ctrain, average='weighted'))
          print('test:', precision_score(ytestc, y_pred_ctest, average='weighted'))
```

```
train: 0.2759646923534461
test: 0.2881639900770335
```

```
In [256]: print('train:', f1_score(ytrainc, ypred_ctrain, average='weighted'))
          print('test:', f1_score(ytestc, y_pred_ctest, average='weighted'))
```

```
train: 0.3424345726881972
test: 0.3568002352372158
```

дерево

```
In [286]: tree = DecisionTreeClassifier(random_state=42)
```

In [287]:
```
tree.fit(xtrainc, ytrainc)
```

Out[287]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=42, splitter='best')
```

In [288]:
```
list(zip(xc.columns.values, tree.feature_importances_))
```

Out[288]:
```
[('Agency', 0.08722336840906048),
 ('Agency Type', 0.07539056269469359),
 ('Distribution Channel', 0.08379155489597753),
 ('Product Name', 0.1080609600647266),
 ('Claim', 0.002129160988890403),
 ('Duration', 0.15053149373080293),
 ('Destination', 0.14379668828474473),
 ('Net Sales', 0.11815727830594033),
 ('Commision (in value)', 0.2309189326251498)]
```

In [289]:
```
yt_predtrain = tree.predict(xtrainc)
yt_predtest = tree.predict(xtestc)
```

In [290]:
```
print('train accuracy (%): {} %'.format(accuracy_score(ytrainc, yt_predtrain) * 100))
print('test accuracy (%): {} %'.format(accuracy_score(ytestc, yt_predtest) * 100))
```

```
train accuracy (%): 97.42857142857143 %
test accuracy (%): 36.0 %
```

In [291]:
```
print('матрица ошибок:')
print('train\n', confusion_matrix(ytrainc, yt_predtrain))
print('test\n', confusion_matrix(ytestc, yt_predtest))
```

```
матрица ошибок:
train
 [[ 3  0  0 ...  0  0  0]
 [ 0  3  0 ...  0  0  0]
 [ 0  0  3 ...  0  0  0]
 ...
 [ 0  0  0 ...  1  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0 18]]
test
 [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 3]]
```

In [292]:
```
print('train:', precision_score(ytrainc, yt_predtrain, average='weighted'))
print('test:', precision_score(ytestc, yt_predtest, average='weighted'))
```

```
train: 0.9741021561623425
test: 0.3756529581529582
```

In [293]:
```
print('train:', f1_score(ytrainc, yt_predtrain, average='weighted'))
print('test:', f1_score(ytestc, yt_predtest, average='weighted'))
```

```
train: 0.9730199683483137
test: 0.36515105795545705
```

подбор одного гиперпараметра с использованием GridSearchCV и кросс-валидации

svm

In [328]:
```
param = [{'C': np.array(np.arange(0.8, 3.1, 0.1)),
                   'max_iter': np.array([1000, 5000, 10000, 25000, 50000])}]
lsvc_grid = GridSearchCV(LinearSVC(), param, cv=ShuffleSplit(n_splits=20, test_size=0.3), scoring='accuracy',
                         n_jobs=-1,
                        )
lsvc_grid.fit(xc_n, yc.values.ravel())
# считает ~2 мин
```

Out[328]:
```
GridSearchCV(cv=ShuffleSplit(n_splits=20, random_state=None, test_size=0.3, train_size=None),
             error_score='raise-deprecating',
             estimator=LinearSVC(C=1.0, class_weight=None, dual=True,
                                 fit_intercept=True, intercept_scaling=1,
                                 loss='squared_hinge', max_iter=1000,
                                 multi_class='ovr', penalty='l2',
                                 random_state=None, tol=0.0001, verbose=0),
             iid='warn', n_jobs=-1,
             param_grid=[{'C': array([0.8, 0.9, 1. , 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2. ,
       2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3. ]),
                          'max_iter': array([ 1000,  5000, 10000, 25000, 50000])}],
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring='accuracy', verbose=0)
```

In [329]:
```
lsvc_grid.best_params_
```

Out[329]:
```
{'C': 2.5, 'max_iter': 1000}
```

In [330]:
```
lsvc_grid.best_estimator_
```

Out[330]:
```
LinearSVC(C=2.5, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
          verbose=0)
```

In [331]:
```
lsvc_grid.best_score_
```

Out[331]:
```
0.4776666666666667
```

дерево

In [332]:
```
param = [{'random_state': np.array([42]),
               'max_depth': np.array([None, 10, 50, 100]),
               'min_samples_split': np.array(range(2, 11)),
               'min_samples_leaf': np.array(range(1, 11))
              }]
tree_grid = GridSearchCV(DecisionTreeClassifier(), param, cv=ShuffleSplit(n_splits=20, test_size=0.3), scorin
g='accuracy',
                         n_jobs=-1,
                        )
tree_grid.fit(xc_n, yc.values.ravel())
```

Out[332]:
```
GridSearchCV(cv=ShuffleSplit(n_splits=20, random_state=None, test_size=0.3, train_size=None),
             error_score='raise-deprecating',
             estimator=DecisionTreeClassifier(class_weight=None,
                                              criterion='gini', max_depth=None,
                                              max_features=None,
                                              max_leaf_nodes=None,
                                              min_impurity_decrease=0.0,
                                              min_impurity_split=None,
                                              min_samples_leaf=1,
                                              min_samples_split=2,
                                              min_weight_fractio....0,
                                              presort=False, random_state=None,
                                              splitter='best'),
             iid='warn', n_jobs=-1,
             param_grid=[{'max_depth': array([None, 10, 50, 100], dtype=object),
                          'min_samples_leaf': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10]),
                          'min_samples_split': array([ 2,  3,  4,  5,  6,  7,  8,  9, 10]),
                          'random_state': array([42])}],
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring='accuracy', verbose=0)
```

In [333]:
```
tree_grid.best_params_
```

Out[333]:
```
{'max_depth': 10,
 'min_samples_leaf': 9,
 'min_samples_split': 2,
 'random_state': 42}
```

In [334]:
```
tree_grid.best_estimator_
```

Out[334]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=10,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=9, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=42, splitter='best')
```

In [335]:
```
tree_grid.best_score_
```

Out[335]:
```
0.4735
```

обучим снова с найденными гиперпараметрами

In [336]:
```
lsvc_grid.best_estimator_.fit(xtrainc, ytrainc.values.ravel())

y_predctrainnew = lsvc_grid.best_estimator_.predict(xtrainc)
y_predctestnew = lsvc_grid.best_estimator_.predict(xtestc)

print('train accuracy (%): {} %'.format(accuracy_score(ytrainc, ypred_ctrain) * 100))
print('test accuracy (%): {} %'.format(accuracy_score(ytestc, y_pred_ctest) * 100))

print('new train accuracy (%): {} %'.format(accuracy_score(ytrainc, y_predctrainnew) * 100))
print('new test accuracy (%): {} %'.format(accuracy_score(ytestc, y_predctestnew) * 100))
```

```
train accuracy (%): 47.714285714285715 %
test accuracy (%): 48.333333333333336 %
new train accuracy (%): 48.0 %
new test accuracy (%): 48.333333333333336 %
```

In [337]:
```
tree_grid.best_estimator_.fit(xtrainc, ytrainc.values.ravel())

yt_predtrainnew = tree_grid.best_estimator_.predict(xtrainc)
yt_predtestnew = tree_grid.best_estimator_.predict(xtestc)

print('train accuracy (%): {} %'.format(accuracy_score(ytrainc, yt_predtrain) * 100))
print('test accuracy (%): {} %'.format(accuracy_score(ytestc, yt_predtest) * 100))

print('new train accuracy (%): {} %'.format(accuracy_score(ytrainc, yt_predtrainnew) * 100))
print('new test accuracy (%): {} %'.format(accuracy_score(ytestc, yt_predtestnew) * 100))
```

```
train accuracy (%): 97.42857142857143 %
test accuracy (%): 36.0 %
new train accuracy (%): 55.285714285714285 %
new test accuracy (%): 47.333333333333336 %
```

с новыми гиперпараметрами модель обучилась лучше