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

Лабораторная работа № 3

По курсу «методы машинного обучения в АСОИУ»

«Обработка признаков. Часть 2»

Выполнил:

студент ИУ5-23М
Семенов И.А.

Проверил:

Балашов А.М.

Подпись:

29.02.2024

Москва, 2024

Описание задания

- 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.
- Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:
  - масштабирование признаков (не менее чем тремя способами);
  - обработку выбросов для числовых признаков (по одному способу для удаления выбросов и для замены выбросов);
  - обработку по крайней мере одного нестандартного признака (который не является числовым или категориальным);
  - 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).
    - ✓

Текст программы и экранные формы с примерами выполнения программы

Загрузка и первичный анализ данных

Используем набор данных web log dataset.

```
import numpy as np
import pandas as pd
import seaborn as sns
import datetime
import ipaddress
import tqdm
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import cross_val_score, cross_validate
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

```
[ ] data = pd.read_csv('beer_production.csv', sep=',')
```

```
[ ] data.shape
```

```
(476, 2)
```

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

```
[ ]
```

	Month	Monthly beer production
0	1956-01	93.2
1	1956-02	96.0
2	1956-03	95.2
3	1956-04	77.1
4	1956-05	70.9

```
[ ] data.dtypes
```

```
Month                       object
Monthly beer production    float64
dtype: object
```

```
[ ] data['Month'] = pd.to_datetime(data['Month'],infer_datetime_format=True)
df = data.set_index(['Month'])
df.head()
```

Month	Monthly beer production
1956-01-01	93.2
1956-02-01	96.0
1956-03-01	95.2
1956-04-01	77.1
1956-05-01	70.9


```
[ ] df.index.freq = 'MS'
df.index
```

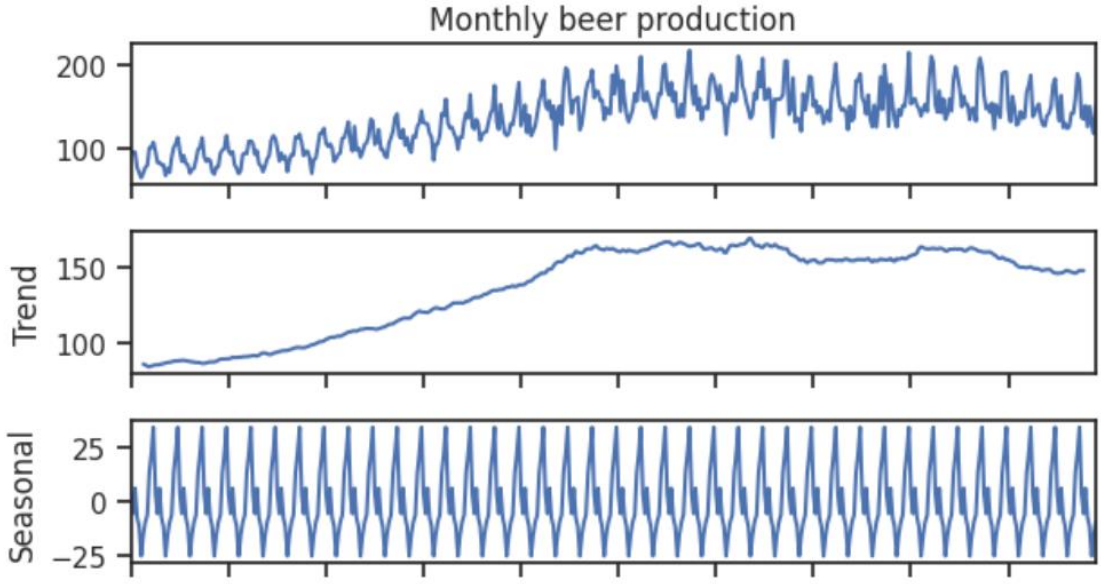
```
DatetimeIndex(['1956-01-01', '1956-02-01', '1956-03-01', '1956-04-01',
               '1956-05-01', '1956-06-01', '1956-07-01', '1956-08-01',
               '1956-09-01', '1956-10-01',
               ...
               '1994-11-01', '1994-12-01', '1995-01-01', '1995-02-01',
               '1995-03-01', '1995-04-01', '1995-05-01', '1995-06-01',
               '1995-07-01', '1995-08-01'],
              dtype='datetime64[ns]', name='Month', length=476, freq='MS')
```

```
[ ] train_df = df[:440]
    test_df = df[440:]
    len(test_df)
```

```
36
```

```
from statsmodels.tsa.filters.hp_filter import hpfilter
from statsmodels.tsa.seasonal import seasonal_decompose

decompose_df = seasonal_decompose(df['Monthly beer production'], model='cumulative')
decompose_df.plot();
```



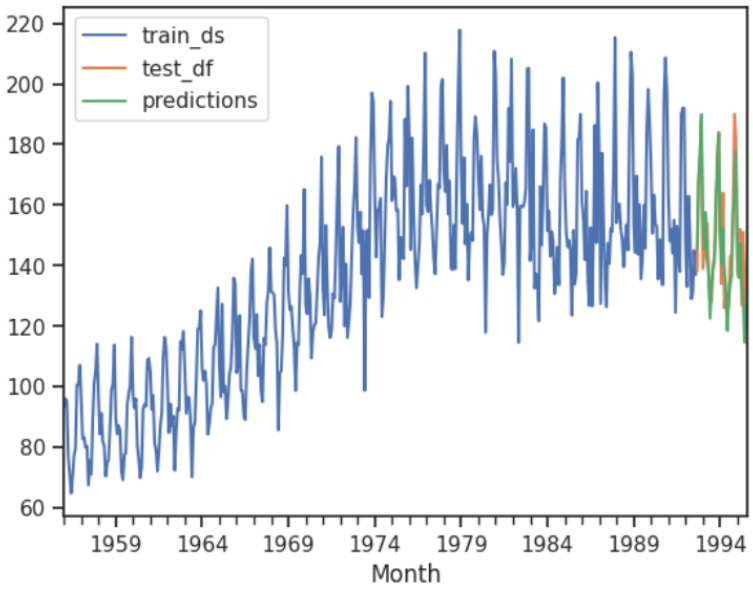
```
[ ] from statsmodels.tsa.holtwinters import SimpleExpSmoothing
    from statsmodels.tsa.holtwinters import ExponentialSmoothing

    model = ExponentialSmoothing(train_df['Monthly beer production'], trend='mul', seasonal='mul', seasonal_periods=12).fit()
    predictions = model.forecast(36).rename('HW')
    predictions[:12]
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/holtwinters/model.py:83: RuntimeWarning: overflow encountered in matmul
  return err.T @ err
1992-09-01    145.711438
1992-10-01    167.907772
1992-11-01    176.634185
1992-12-01    189.934477
1993-01-01    156.085951
1993-02-01    145.572899
1993-03-01    157.500476
1993-04-01    142.400903
1993-05-01    137.741258
1993-06-01    122.551526
1993-07-01    132.968263
1993-08-01    138.522124
Freq: MS, Name: HW, dtype: float64
```

```
[ ] train_df['Monthly beer production'].plot(legend=True, label='train_ds')
    test_df['Monthly beer production'].plot(legend=True, label='test_df')
    predictions.plot(legend=True, label='predictions')
```

```
<Axes: xlabel='Month'>
```



Масштабирование признаков

Загрузка и предобработка данных

```
[ ] import pandas as pd
    import matplotlib.pyplot as plt
    import numpy as np
    import seaborn as sns
    #from sklearn.datasets import load_boston
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler
    from sklearn.preprocessing import MinMaxScaler
    from sklearn.preprocessing import RobustScaler
    from sklearn.preprocessing import MaxAbsScaler
```

```
data = pd.read_csv('Customers.csv', encoding='windows-1251')
data = data.drop('Gender', axis=1)
data = data.drop('Profession', axis=1)
data.shape
```

```
(2000, 8)
```

```
data.head()
```

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
0	1	19	15000	39	1	4
1	2	21	35000	81	3	3
2	3	20	86000	6	1	1
3	4	23	59000	77	0	2
4	5	31	38000	40	2	6

```
[ ] # Нужно ли масштабирование
    data.describe()
```

Масштабирование данных на основе Z-оценки

$$x' = \frac{x - \mu(x)}{\sigma(x)}$$

где x - признак, $\mu(x) = mean(x)$ - среднее значение, $\sigma(x) = std(x)$ - среднеквадратичное отклонение.

Особенности метода:

- Среднее значение приводится к 0.
- Среднеквадратичное отклонение приводится к 1.
- Форма исходного распределения сохраняется.
- Максимальные и минимальные значения могут варьироваться.
- Выбросы сохраняются.

Метод реализован с использованием класса StandardScaler.

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

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
0	-1.731185	-1.054089	-2.093501	-0.428339	-0.791207	0.117497
1	-1.729453	-0.983723	-1.656133	1.075546	-0.281162	-0.390051
2	-1.727721	-1.018906	-0.540845	-1.609962	-0.791207	-1.405148
3	-1.725989	-0.913356	-1.131292	0.932319	-1.046230	-0.897599
4	-1.724257	-0.631891	-1.590528	-0.392532	-0.536185	1.132594
...	...	...	...	...	...	...
1995	1.724257	0.775438	1.610720	-0.392532	0.993950	1.640142
1996	1.725989	1.479103	-0.821679	-0.678986	0.738928	1.640142
1997	1.727721	1.338370	-0.432356	-1.323508	1.248972	-0.897599
1998	1.729453	0.986538	1.560904	-1.681576	0.738928	-0.897599
1999	1.731185	1.443919	-0.002664	0.037150	0.228883	-0.897599

2000 rows × 6 columns

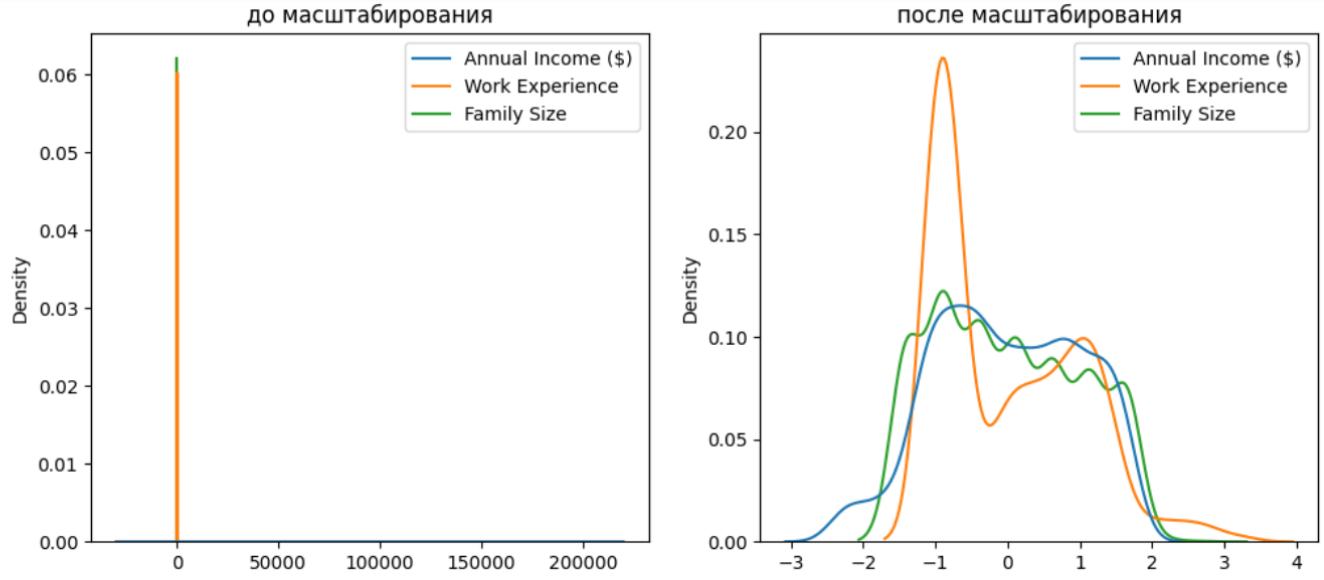
```
[ ] data_cs11_scaled.describe()
```

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
count	2000.000000	2.000000e+03	2.000000e+03	2.000000e+03	2.000000e+03	2.000000e+03
mean	0.000000	-3.552714e-17	-1.136868e-16	4.085621e-17	-5.329071e-18	2.842171e-17
std	1.000250	1.000250e+00	1.000250e+00	1.000250e+00	1.000250e+00	1.000250e+00
min	-1.731185	-1.722571e+00	-2.421527e+00	-1.824803e+00	-1.046230e+00	-1.405148e+00
25%	-0.865592	-8.429900e-01	-7.907571e-01	-8.222131e-01	-7.912071e-01	-8.975992e-01
50%	0.000000	-3.377589e-02	-1.501968e-02	-3.446402e-02	-2.811622e-01	1.174974e-01
75%	0.865592	8.458046e-01	8.388918e-01	8.607053e-01	7.389275e-01	6.250458e-01
max	1.731185	1.760568e+00	1.732899e+00	1.755875e+00	3.289152e+00	2.655239e+00

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

```
[ ] draw_kde(['Annual Income ($)', 'Work Experience', 'Family Size'], data, data_cs11_scaled, 'до масштабирования', 'после масштабирования')
```

[ ]




```
# Обучаем StandardScaler на обучающей выборке
# и масштабируем обучающую и тестовую выборки
cs12 = StandardScaler()
cs12.fit(X_train)
data_cs12_scaled_train_temp = cs12.transform(X_train)
data_cs12_scaled_test_temp = cs12.transform(X_test)
# формируем DataFrame на основе массива
data_cs12_scaled_train = arr_to_df(data_cs12_scaled_train_temp)
data_cs12_scaled_test = arr_to_df(data_cs12_scaled_test_temp)
```

[ ]

```
data_cs12_scaled_train.describe()
```

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
count	1.600000e+03	1.600000e+03	1.600000e+03	1.600000e+03	1.600000e+03	1.600000e+03
mean	4.662937e-17	2.109424e-17	1.376677e-16	7.771561e-17	1.676437e-16	1.143530e-16
std	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00
min	-1.725200e+00	-1.718127e+00	-2.394178e+00	-1.821801e+00	-1.039760e+00	-1.374476e+00
25%	-8.690342e-01	-8.393009e-01	-7.917217e-01	-8.257591e-01	-7.860440e-01	-8.707744e-01
50%	-5.946437e-03	-3.078089e-02	-1.235402e-02	-7.581483e-03	-2.786119e-01	1.366291e-01
75%	8.627654e-01	8.480453e-01	8.380605e-01	8.550623e-01	9.899683e-01	6.403308e-01

```
[ ] data_cs12_scaled_test.describe()
```

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	0.022183	0.014830	0.036344	-0.044577	0.005550	0.100111
std	0.996568	0.996863	0.958469	0.967294	0.975324	0.958530
min	-1.720008	-1.718127	-2.394178	-1.821801	-1.039760	-1.374476
25%	-0.804140	-0.804148	-0.703119	-0.825759	-0.786044	-0.870774
50%	0.032990	0.004372	0.005538	-0.078727	-0.024896	0.136629
75%	0.895212	0.883198	0.850959	0.775023	0.736252	0.640331
max	1.718499	1.762024	1.720051	1.735493	3.019697	2.151436

```
# распределения для обучающей и тестовой выборки немного отличаются
draw_kde(['Annual Income ($)', 'Work Experience', 'Family Size'], data_cs12_scaled_train, data_cs12_scaled_test, 'обучающая', 'тестовая')
```