РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра математического моделирования и искусственного интеллекта

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

Дисциплина: Интеллектуальный анализ данных

Студент: Генер

Группа: НПИ02

Москва 2024

Вариант № 24

Ozone Level Detection Data Set

`eighthr.data`

http://archive.ics.uci.edu/ml/datasets/Ozone+Level+Detection

1. скачать датасет

In [1]:

```
from ucimlrepo import fetch_ucirepo

# fetch dataset
ozone_level_detection = fetch_ucirepo(id=172)
```

In [2]:

```
original = ozone_level_detection['data']['original']
```

In [3]:

```
o8hr = original[original['Dataset'] == '8hr']
```

In [4]:

```
o8hr
```

Out[4]:

	Dataset	Date	WSR0	WSR1	WSR2	WSR3	WSR4	WSR5	WSR6
0	8hr	1/1/1998	0.8	1.8	2.4	2.1	2.0	2.1	1.5
1	8hr	1/2/1998	2.8	3.2	3.3	2.7	3.3	3.2	2.9
2	8hr	1/3/1998	2.9	2.8	2.6	2.1	2.2	2.5	2.5
3	8hr	1/4/1998	4.7	3.8	3.7	3.8	2.9	3.1	2.8
4	8hr	1/5/1998	2.6	2.1	1.6	1.4	0.9	1.5	1.2
...	...	...	...	...	...	...	...	...	...
2529	8hr	12/27/2004	0.3	0.4	0.5	0.5	0.2	0.3	0.4
2530	8hr	12/28/2004	1.0	1.4	1.1	1.7	1.5	1.7	1.8
2531	8hr	12/29/2004	0.8	0.8	1.2	0.9	0.4	0.6	0.8
2532	8hr	12/30/2004	1.3	0.9	1.5	1.2	1.6	1.8	1.1
2533	8hr	12/31/2004	1.5	1.3	1.8	1.4	1.2	1.7	1.6

2534 rows × 75 columns

2. определить числовые признаки

Из видимых данных и названия признаков кажется, что все признаки кроме `Date` и `Class` -- числовые, а последний является категориальным. Это можно проверить, посмотрев на количество уникальных значений в колонках.

In [5]:

```python
for label in list(o8hr):
    print(f"{label = }, {len(set(o8hr[label])) = }")
```

```
label = 'Dataset', len(set(o8hr[label])) = 1
label = 'Date', len(set(o8hr[label])) = 2534
label = 'WSR0', len(set(o8hr[label])) = 367
label = 'WSR1', len(set(o8hr[label])) = 362
label = 'WSR2', len(set(o8hr[label])) = 359
label = 'WSR3', len(set(o8hr[label])) = 358
label = 'WSR4', len(set(o8hr[label])) = 357
label = 'WSR5', len(set(o8hr[label])) = 355
label = 'WSR6', len(set(o8hr[label])) = 357
label = 'WSR7', len(set(o8hr[label])) = 356
label = 'WSR8', len(set(o8hr[label])) = 359
label = 'WSR9', len(set(o8hr[label])) = 357
label = 'WSR10', len(set(o8hr[label])) = 364
label = 'WSR11', len(set(o8hr[label])) = 369
label = 'WSR12', len(set(o8hr[label])) = 364
label = 'WSR13', len(set(o8hr[label])) = 366
label = 'WSR14', len(set(o8hr[label])) = 365
label = 'WSR15', len(set(o8hr[label])) = 364
label = 'WSR16', len(set(o8hr[label])) = 356
label = 'WSR17', len(set(o8hr[label])) = 356
label = 'WSR18', len(set(o8hr[label])) = 356
label = 'WSR19', len(set(o8hr[label])) = 357
label = 'WSR20', len(set(o8hr[label])) = 362
label = 'WSR21', len(set(o8hr[label])) = 362
label = 'WSR22', len(set(o8hr[label])) = 368
label = 'WSR23', len(set(o8hr[label])) = 362
label = 'WSR_PK', len(set(o8hr[label])) = 347
label = 'WSR_AV', len(set(o8hr[label])) = 328
label = 'T0', len(set(o8hr[label])) = 472
label = 'T1', len(set(o8hr[label])) = 469
label = 'T2', len(set(o8hr[label])) = 474
label = 'T3', len(set(o8hr[label])) = 467
label = 'T4', len(set(o8hr[label])) = 467
label = 'T5', len(set(o8hr[label])) = 474
label = 'T6', len(set(o8hr[label])) = 478
label = 'T7', len(set(o8hr[label])) = 494
label = 'T8', len(set(o8hr[label])) = 498
label = 'T9', len(set(o8hr[label])) = 499
label = 'T10', len(set(o8hr[label])) = 515
label = 'T11', len(set(o8hr[label])) = 522
label = 'T12', len(set(o8hr[label])) = 523
label = 'T13', len(set(o8hr[label])) = 526
label = 'T14', len(set(o8hr[label])) = 527
label = 'T15', len(set(o8hr[label])) = 526
label = 'T16', len(set(o8hr[label])) = 521
label = 'T17', len(set(o8hr[label])) = 511
label = 'T18', len(set(o8hr[label])) = 505
label = 'T19', len(set(o8hr[label])) = 494
label = 'T20', len(set(o8hr[label])) = 491
label = 'T21', len(set(o8hr[label])) = 479
label = 'T22', len(set(o8hr[label])) = 479
label = 'T23', len(set(o8hr[label])) = 473
label = 'T_PK', len(set(o8hr[label])) = 505
label = 'T_AV', len(set(o8hr[label])) = 471
label = 'T85', len(set(o8hr[label])) = 350
label = 'RH85', len(set(o8hr[label])) = 205
label = 'U85', len(set(o8hr[label])) = 1468
label = 'V85', len(set(o8hr[label])) = 1641
label = 'HT85', len(set(o8hr[label])) = 463
label = 'T70', len(set(o8hr[label])) = 352
label = 'RH70', len(set(o8hr[label])) = 215
label = 'U70', len(set(o8hr[label])) = 1694
label = 'V70', len(set(o8hr[label])) = 1586
label = 'HT70', len(set(o8hr[label])) = 541
label = 'T50', len(set(o8hr[label])) = 301
label = 'RH50', len(set(o8hr[label])) = 225
label = 'U50', len(set(o8hr[label])) = 1897
label = 'V50', len(set(o8hr[label])) = 1719
label = 'HT50', len(set(o8hr[label])) = 197
label = 'KI', len(set(o8hr[label])) = 1183
label = 'TT', len(set(o8hr[label])) = 782
label = 'SLP', len(set(o8hr[label])) = 166
label = 'SLP_', len(set(o8hr[label])) = 214
label = 'Precp', len(set(o8hr[label])) = 176
label = 'Class', len(set(o8hr[label])) = 2
```

Учитывая, что в признаке `Class` существуют только два разных значения, а во всех остальных признаках -- больше сотни, то, действительно, этот один признак является категориальным, а остальные -- числовыми.

Заменим значения NaN медианными значениями по своим колонкам.

In [6]:

```python
import warnings
with warnings.catch_warnings(action="ignore"):
    for label in list(o8hr):
        try:
            med = o8hr[label].median()
        except:
            print(f"{label=} no median")
            continue
        nans = sum(o8hr[label].isna())
        o8hr[label] = o8hr[label].fillna(med)
        print(f"{label=} filled {nans} cells with {med}")
```

```
label='Dataset' no median
label='Date' no median
label='WSR0' filled 299 cells with 1.3
label='WSR1' filled 292 cells with 1.3
label='WSR2' filled 294 cells with 1.2
label='WSR3' filled 292 cells with 1.3
label='WSR4' filled 293 cells with 1.3
label='WSR5' filled 292 cells with 1.3
label='WSR6' filled 291 cells with 1.4
label='WSR7' filled 289 cells with 1.9
label='WSR8' filled 290 cells with 2.5
label='WSR9' filled 287 cells with 2.8
label='WSR10' filled 288 cells with 2.9
label='WSR11' filled 292 cells with 2.9
label='WSR12' filled 287 cells with 3.0
label='WSR13' filled 288 cells with 3.0
label='WSR14' filled 288 cells with 3.1
label='WSR15' filled 286 cells with 3.2
label='WSR16' filled 284 cells with 3.2
label='WSR17' filled 283 cells with 2.9
label='WSR18' filled 286 cells with 2.5
label='WSR19' filled 292 cells with 2.2
label='WSR20' filled 294 cells with 2.0
label='WSR21' filled 293 cells with 1.7
label='WSR22' filled 300 cells with 1.6
label='WSR23' filled 297 cells with 1.4
label='WSR_PK' filled 273 cells with 4.1
label='WSR_AV' filled 273 cells with 2.2
label='T0' filled 190 cells with 20.4
label='T1' filled 185 cells with 20.2
label='T2' filled 187 cells with 19.9
label='T3' filled 184 cells with 19.85
label='T4' filled 184 cells with 19.7
label='T5' filled 183 cells with 19.6
label='T6' filled 183 cells with 19.7
label='T7' filled 183 cells with 20.4
label='T8' filled 185 cells with 21.4
label='T9' filled 185 cells with 22.9
label='T10' filled 188 cells with 24.0
label='T11' filled 192 cells with 24.8
label='T12' filled 189 cells with 25.3
label='T13' filled 191 cells with 25.5
label='T14' filled 192 cells with 25.7
label='T15' filled 187 cells with 25.6
label='T16' filled 184 cells with 25.25
label='T17' filled 182 cells with 24.4
label='T18' filled 184 cells with 23.4
label='T19' filled 188 cells with 22.5
label='T20' filled 189 cells with 21.8
label='T21' filled 185 cells with 21.3
label='T22' filled 192 cells with 20.9
label='T23' filled 189 cells with 20.7
label='T_PK' filled 175 cells with 26.6
label='T_AV' filled 175 cells with 22.2
label='T85' filled 99 cells with 14.3
label='RH85' filled 105 cells with 0.64
```

```
label='U85' filled 180 cells with 1.875
label='V85' filled 180 cells with 1.545
label='HT85' filled 95 cells with 1535.0
label='T70' filled 107 cells with 6.8
label='RH70' filled 115 cells with 0.38
label='U70' filled 157 cells with 5.09
label='V70' filled 157 cells with 0.86
label='HT70' filled 100 cells with 3153.5
label='T50' filled 115 cells with -10.1
label='RH50' filled 125 cells with 0.23
label='U50' filled 210 cells with 9.25
label='V50' filled 210 cells with 0.36
label='HT50' filled 112 cells with 5835.0
label='KI' filled 136 cells with 14.925
label='TT' filled 125 cells with 41.1
label='SLP' filled 95 cells with 10160.0
label='SLP_' filled 158 cells with 0.0
label='Precp' filled 2 cells with 0.0
label='Class' filled 0 cells with 0.0
```

3. метка класса

Уже определили, что метка класса -- это признак `Class`, который принимает 2 значения, поэтому дискретизация не требуется.

In [7]:

```
del o8hr['Dataset']
```

In [8]:

```
numerics = set(o8hr) - set(["Date", "Class"])
numerics = list(numerics)
numerics
```

```
Out[8]: ['WSR2',
          'T21',
          'SLP',
          'T22',
          'T_PK',
          'RH70',
          'HT85',
          'WSR5',
          'RH85',
          'WSR13',
          'T20',
          'T17',
          'WSR17',
          'T3',
          'U50',
          'WSR18',
          'TT',
          'WSR9',
          'T14',
          'T6',
          'T23',
          'U70',
          'WSR7',
          'WSR11',
          'WSR19',
          'WSR6',
          'HT50',
          'WSR10',
          'SLP_',
          'WSR16',
          'T8',
          'V85',
          'T2',
          'Precp',
          'KI',
          'T7',
          'V50',
          'T18',
          'T4',
          'T5',
          'T50',
          'RH50',
          'T16',
          'WSR23',
          'WSR_PK',
          'T_AV',
          'WSR12',
          'WSR15',
          'T10',
          'WSR4',
          'T15',
          'WSR0',
          'T0',
          'WSR1',
          'T12',
          'U85',
          'T1',
          'HT70',
          'WSR21',
          'T11',
          'T70',
          'T13',
          'V70',
          'WSR8',
          'WSR22',
          'T85',
          'WSR3',
          'WSR20',
          'T9',
          'WSR_AV',
          'T19',
          'WSR14']
```

4. признаки с макс. взаимосвязью

In [9]:
```
import sklearn.feature_selection
klass = sklearn.feature_selection.SelectKBest(sklearn.feature_selection.f_cl
```

In [10]:
```
klass.fit(o8hr[numerics], o8hr['Class'])
```

Out[10]:
```
▼  SelectKBest  ⓘ ?
SelectKBest(k=2)
```

In [11]:
```
high_corr = klass.get_feature_names_out()
high_corr
```

Out[11]:
```
array(['T16', 'T15'], dtype=object)
```

5. матрица корреляции

In [12]:
```
corr_in = o8hr[list(high_corr) + ["Class"]]
```

In [13]:
```
corr_mat = corr_in.corr()
```

In [14]:
```
corr_mat.style.background_gradient(cmap='Blues')
```

Out[14]:

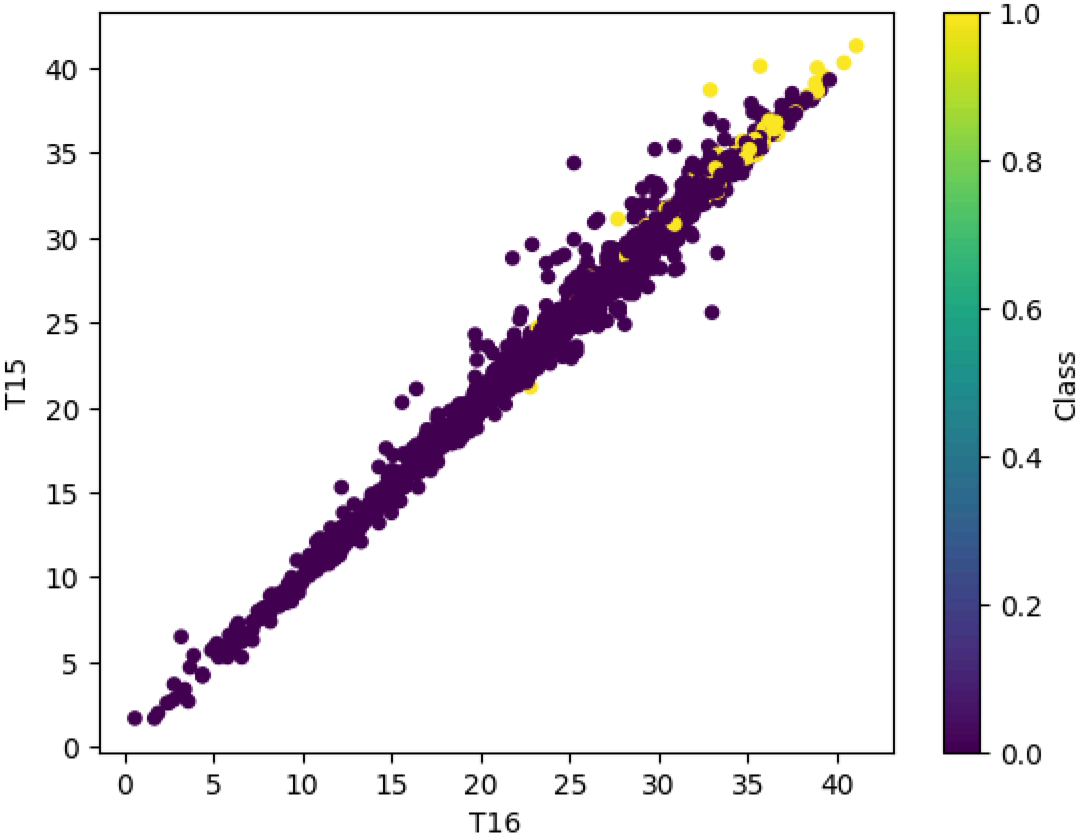
	T16	T15	Class
T16	1.000000	0.993745	0.249164
T15	0.993745	1.000000	0.251189
Class	0.249164	0.251189	1.000000

6. визуализация данных

In [15]:
```
visdata = o8hr[list(high_corr) + ["Class"]]
```

In [16]:
```
visdata.plot.scatter(high_corr[0], high_corr[1], c='Class', cmap='viridis')
```

Out[16]:
```
<Axes: xlabel='T16', ylabel='T15'>
```



7. метод главных компонент: размерность

In [17]:
```
from sklearn.decomposition import PCA
```

In [18]:
```
explained_ratio = 0
n = -1
while explained_ratio < 0.975:
    n += 1
    p = PCA(n_components=n)
    p.fit(o8hr[numerics])
    explained_ratio = sum(p.explained_variance_ratio_)
    print(f"{n=} {explained_ratio=}")
print("необходимая размерность:", n)
```

```
n=0 explained_ratio=0
n=1 explained_ratio=np.float64(0.6168094803096223)
n=2 explained_ratio=np.float64(0.8671047959110314)
n=3 explained_ratio=np.float64(0.9331933895443932)
n=4 explained_ratio=np.float64(0.9562651425610001)
n=5 explained_ratio=np.float64(0.9765877080299148)
необходимая размерность: 5
```

8. метод главных компонент: сжатие и визуализация

In [19]:

```
p = PCA(n_components=2)
coords = p.fit_transform(o8hr[numerics])
```

In [20]:

```
coords
```

Out[20]:

```
array([[  5.34385419, 175.30488734],
       [ 14.62103152, 108.23753092],
       [ -8.33647694,  66.94707866],
       ...,
       [ 47.66417271, 113.22842784],
       [ 42.55531334,  83.36474789],
       [ 13.16492786,  53.88670953]])
```

In [21]:

```
xs = coords[:, 0]
ys = coords[:, 1]
```

In [22]:

```
import matplotlib
from matplotlib import pyplot as pyplt
```

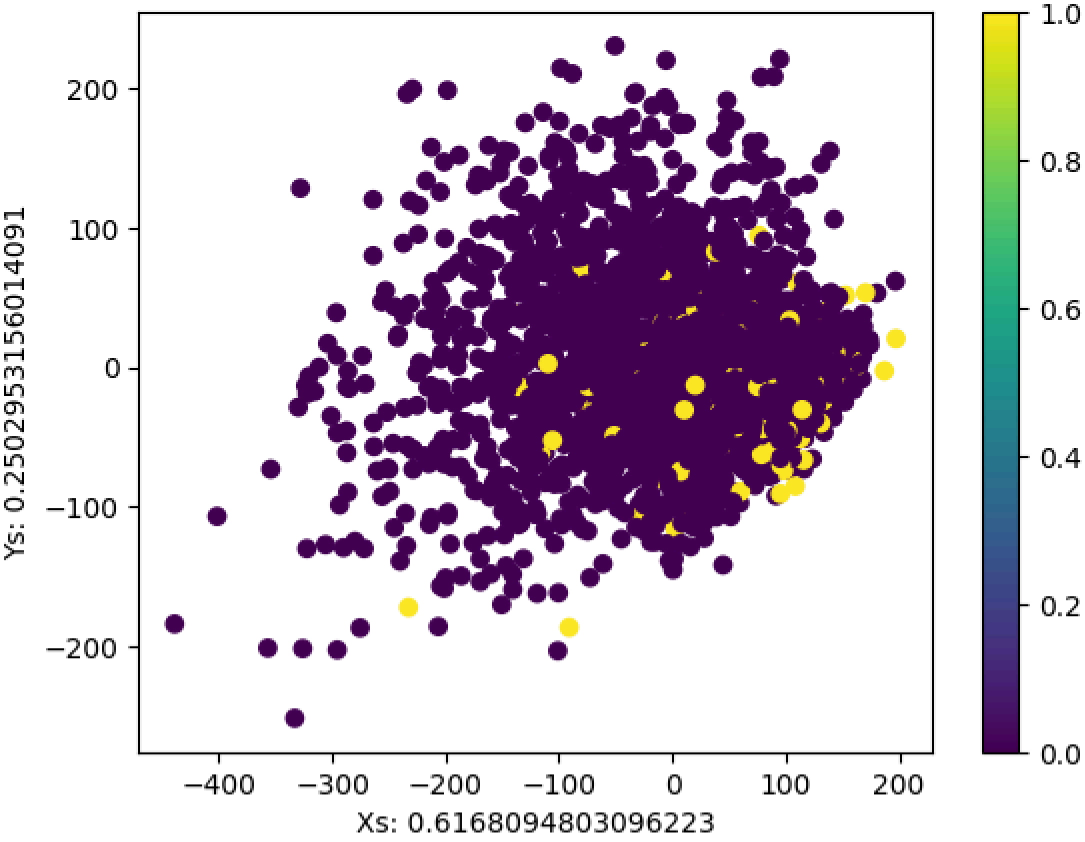
In [23]:

```
%matplotlib
pyplt.scatter(xs, ys, c=o8hr['Class'])
pyplt.colorbar()
pyplt.xlabel(f"Xs: {p.explained_variance_ratio_[0]}")
pyplt.ylabel(f"Ys: {p.explained_variance_ratio_[1]}")
```

```
Using matplotlib backend: module://matplotlib_inline.backend_inline
```

Out[23]:

```
Text(0, 0.5, 'Ys: 0.2502953156014091')
```


200
100
0
−100
−200
Ys: 0.2502953156014091
−400
−300
−200
−100
0
100
200
Xs: 0.6168094803096223
1.0
0.8
0.6
0.4
0.2
0.0

```
In [ ]:
```