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

Лабораторная работа №2
по дисциплине
«Методы машинного обучения»
на тему
«Изучение библиотек обработки данных»

Выполнил:
студент группы ИУ5-24М
Зубаиров В. А.

_____

Москва — 2020 г.

[6]:
```
import numpy as np
import pandas as pd
pd.set_option('display.max.columns', 100)
# to draw pictures in jupyter notebook
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
# we don't like warnings
# you can comment the following 2 lines if you'd like to
import warnings
warnings.filterwarnings('ignore')
```

[7]:
```
data = pd.read_csv('../DATA/adult.data.csv')
data.head()
```

[7]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K

[8]:
```
data['sex'].value_counts()
```

[8]:
```
Male      21790
Female    10771
Name: sex, dtype: int64
```

[9]:
```
data.loc[data['sex'] == 'Female', 'age'].mean()
```

[9]:
```
36.85823043357163
```

[10]:
```
len(data.loc[data['native-country'] == 'Germany']) / len(data)
```

[10]:
```
0.004207487485028101
```

[11]:
```
m50K = data.loc[data['salary'] == '>50K', 'age']
l50K = data.loc[data['salary'] == '<=50K', 'age']
```

```
print('Average more 50K - ', str(m50K.mean()), ' std - ', str(m50K.std()))
print('Average more 50K - ', str(l50K.mean()), ' std - ', str(l50K.std()))
```

```
Average more 50K -  44.24984058155847  std -  10.519027719851826
Average more 50K -  36.78373786407767  std -  14.02008849082488
```

[12]:
```
data.loc[data['salary'] == '>50K', 'education'].unique()
```

[12]:
```
array(['HS-grad', 'Masters', 'Bachelors', 'Some-college', 'Assoc-voc',
       'Doctorate', 'Prof-school', 'Assoc-acdm', '7th-8th', '12th',
       '10th', '11th', '9th', '5th-6th', '1st-4th'], dtype=object)
```

[13]:
```
for (race, sex), groupped in data.groupby(['race', 'sex']):
    print("Race: {0}, sex: {1}".format(race, sex))
    print(groupped['age'].describe())
```

```
Race: Amer-Indian-Eskimo, sex: Female
count   119.000000
mean     37.117647
std      13.114991
min      17.000000
25%      27.000000
50%      36.000000
75%      46.000000
max      80.000000
Name: age, dtype: float64
Race: Amer-Indian-Eskimo, sex: Male
count   192.000000
mean     37.208333
std      12.049563
min      17.000000
25%      28.000000
50%      35.000000
75%      45.000000
max      82.000000
Name: age, dtype: float64
Race: Asian-Pac-Islander, sex: Female
count   346.000000
mean     35.089595
std      12.300845
min      17.000000
25%      25.000000
50%      33.000000
75%      43.750000
max      75.000000
Name: age, dtype: float64
Race: Asian-Pac-Islander, sex: Male
count   693.000000
```

```
mean      39.073593
std       12.883944
min       18.000000
25%       29.000000
50%       37.000000
75%       46.000000
max       90.000000
Name: age, dtype: float64
Race: Black, sex: Female
count    1555.000000
mean       37.854019
std        12.637197
min        17.000000
25%        28.000000
50%        37.000000
75%        46.000000
max        90.000000
Name: age, dtype: float64
Race: Black, sex: Male
count    1569.000000
mean       37.682600
std        12.882612
min        17.000000
25%        27.000000
50%        36.000000
75%        46.000000
max        90.000000
Name: age, dtype: float64
Race: Other, sex: Female
count    109.000000
mean      31.678899
std       11.631599
min       17.000000
25%       23.000000
50%       29.000000
75%       39.000000
max       74.000000
Name: age, dtype: float64
Race: Other, sex: Male
count    162.000000
mean      34.654321
std       11.355531
min       17.000000
25%       26.000000
50%       32.000000
75%       42.000000
max       77.000000
Name: age, dtype: float64
Race: White, sex: Female
count    8642.000000
```

```
mean      36.811618
std       14.329093
min       17.000000
25%       25.000000
50%       35.000000
75%       46.000000
max       90.000000
Name: age, dtype: float64
Race: White, sex: Male
count    19174.000000
mean        39.652498
std         13.436029
min         17.000000
25%         29.000000
50%         38.000000
75%         49.000000
max         90.000000
Name: age, dtype: float64
```

[14]:
```
data.loc[(data['sex'] == 'Male') &
    (data['marital-status'].isin(['Never-married',
                                  'Separated',
                                  'Divorced',
                                  'Widowed'])), 'salary'].value_counts()
```

[14]:
```
<=50K    7552
>50K      697
Name: salary, dtype: int64
```

[15]:
```
data.loc[(data['sex'] == 'Male') &
    (data['marital-status'].str.startswith('Married')), 'salary'].value_counts()
```

[15]:
```
<=50K    7576
>50K     5965
Name: salary, dtype: int64
```

[16]:
```
data['marital-status'].value_counts()
```

[16]:
```
Married-civ-spouse       14976
Never-married            10683
Divorced                  4443
Separated                 1025
Widowed                    993
Married-spouse-absent      418
Married-AF-spouse           23
Name: marital-status, dtype: int64
```

[17]:
```
max_load = data['hours-per-week'].max()
print("Max time - {0} hours./week.".format(max_load))

num_workaholics = data[data['hours-per-week'] == max_load].shape[0]
```

```
print("Total number of such hard workers {0}".format(num_workaholics))

rich_share = float(data[(data['hours-per-week'] == max_load)
                & (data['salary'] == '>50K')].shape[0]) / num_workaholics
print("Percentage of rich among them {0}%".format(int(100 * rich_share)))
```

Max time - 99 hours./week.
Total number of such hard workers 85
Percentage of rich among them 29%

[18]:
```
pd.crosstab(data['native-country'], data['salary'],
            values=data['hours-per-week'], aggfunc=np.mean).T
```

[18]:
native-country ? Cambodia Canada China Columbia \
salary
<=50K 40.164760 41.416667 37.914634 37.381818 38.684211
>50K 45.547945 40.000000 45.641026 38.900000 50.000000

native-country Cuba Dominican-Republic Ecuador El-Salvador \
salary
<=50K 37.985714 42.338235 38.041667 36.030928
>50K 42.440000 47.000000 48.750000 45.000000

native-country England France Germany Greece Guatemala Haiti \
salary
<=50K 40.483333 41.058824 39.139785 41.809524 39.360656 36.325
>50K 44.533333 50.750000 44.977273 50.625000 36.666667 42.750

native-country Holand-Netherlands Honduras Hong Hungary India \
salary
<=50K 40.0 34.333333 39.142857 31.3 38.233333
>50K NaN 60.000000 45.000000 50.0 46.475000

native-country Iran Ireland Italy Jamaica Japan Laos \
salary
<=50K 41.44 40.947368 39.625 38.239437 41.000000 40.375
>50K 47.50 48.000000 45.400 41.100000 47.958333 40.000

native-country Mexico Nicaragua Outlying-US(Guam-USVI-etc) Peru \
salary
<=50K 40.003279 36.09375 41.857143 35.068966
>50K 46.575758 37.50000 NaN 40.000000

native-country Philippines Poland Portugal Puerto-Rico Scotland \
salary
<=50K 38.065693 38.166667 41.939394 38.470588 39.444444
>50K 43.032787 39.000000 41.500000 39.416667 46.666667

native-country South Taiwan Thailand Trinadad&Tobago \
salary

```
<=50K          40.15625  33.774194  42.866667       37.058824
>50K           51.43750  46.800000  58.333333       40.000000

native-country  United-States    Vietnam  Yugoslavia
salary
<=50K            38.799127  37.193548       41.6
>50K             45.505369  39.200000       49.5
```

[19]:
```
user_usage = pd.read_csv("../DATA/merging/user_usage.csv")
user_device = pd.read_csv("../DATA/merging/user_device.csv")
devices = pd.read_csv("../DATA/merging/android_devices.csv")
devices.rename(columns={"Retail Branding": "manufacturer"}, inplace=True)
```

[20]:
```
user_usage.head()
```

[20]:
```
   outgoing_mins_per_month  outgoing_sms_per_month  monthly_mb  use_id
0                    21.97                    4.82     1557.33   22787
1                  1710.08                  136.88      7267.55   22788
2                  1710.08                  136.88      7267.55   22789
3                    94.46                   35.17       519.12   22790
4                    71.59                   79.26      1557.33   22792
```

[22]:
```
user_device.head()
```

[22]:
```
   use_id  user_id platform  platform_version     device  use_type_id
0   22782    26980      ios             10.2  iPhone7,2            2
1   22783    29628  android              6.0    Nexus 5             3
2   22784    28473  android              5.1   SM-G903F             1
3   22785    15200      ios             10.2  iPhone7,2            3
4   22786    28239  android              6.0  ONE E1003             1
```

[23]:
```
devices.head()
```

[23]:
```
  manufacturer Marketing Name    Device                   Model
0          NaN            NaN    AD681H  Smartfren Andromax AD681H
1          NaN            NaN     FJL21                   FJL21
2          NaN            NaN       T31            Panasonic T31
3          NaN            NaN  hws7721g        MediaPad 7 Youth 2
4           3Q       OC1020A   OC1020A                 OC1020A
```

[27]:
```
result = pd.merge(user_usage,
                  user_device[['use_id', 'platform', 'device']],
                  on='use_id')
result.head()
```

[27]:
```
   outgoing_mins_per_month  outgoing_sms_per_month  monthly_mb  use_id  \
0                    21.97                    4.82     1557.33   22787
1                  1710.08                  136.88      7267.55   22788
2                  1710.08                  136.88      7267.55   22789
3                    94.46                   35.17       519.12   22790
4                    71.59                   79.26      1557.33   22792
```

```
  platform    device
0  android  GT-I9505
1  android  SM-G930F
2  android  SM-G930F
3  android     D2303
4  android  SM-G361F
```

[32]:
```
import pandasql as ps
from pandasql import sqldf
from datetime import datetime
import time
```

[33]:
```
tic = time.perf_counter()
tutorial = pd.merge(user_usage,
                    user_device[['use_id', 'platform', 'device']],
                    on='use_id')
toc = time.perf_counter()
print(f"Смержено за: {toc - tic:0.4f} seconds")
```

Смержено за: 0.0065 seconds

[35]:
```
pysqldf = lambda q: sqldf(q, globals())
q = """
SELECT * FROM user_usage, user_device WHERE user_usage.use_id = user_device.use_id;
"""
tic = time.perf_counter()
joined = pysqldf(q)
toc = time.perf_counter()
print(f"Смержено за: {toc - tic:0.4f} seconds")
```

Смержено за: 0.0286 seconds

[36]:
```
joined.head()
```

[36]:
```
   outgoing_mins_per_month  outgoing_sms_per_month  monthly_mb  use_id  \
0                    21.97                    4.82     1557.33   22787
1                  1710.08                  136.88     7267.55   22788
2                  1710.08                  136.88     7267.55   22789
3                    94.46                   35.17      519.12   22790
4                    71.59                   79.26     1557.33   22792

   use_id  user_id platform  platform_version    device  use_type_id
0   22787    12921  android               4.3  GT-I9505            1
1   22788    28714  android               6.0  SM-G930F            1
2   22789    28714  android               6.0  SM-G930F            1
3   22790    29592  android               5.1     D2303            1
4   22792    28217  android               5.1  SM-G361F            1
```

[41]:
```
result.groupby("platform")["outgoing_sms_per_month"].describe()
```

[41]:
```
          count        mean         std    min      25%      50%      75%  \
platform
android   157.0   85.354586   85.521483   0.25   22.7700   62.850  115.0200
ios         2.0  293.975000  348.780420  47.35  170.6625  293.975  417.2875

            max
platform
android   435.29
ios       540.60
```

[ ]: