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 ИУ5-21М

Мьоу Зо У

Москва — 2021 г.

# 1. Цель лабораторной работы

Изучить библиотеки обработки данных Pandas и PandaSQL.

## Часть 1

Требуется выполнить первое демонстрационное задание под названием «Exploratory data analysis with Pandas» со страницы курса mlcourse.ai.

In [1]:
```
import pandas as pd
```

In [2]:
```
pd.set_option("display.width", 70)
```

In [3]:
```
data = pd.read_csv('adult.data.csv')
data.head()
```

Out[3]:

| | 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 |

In [4]:
```
data["sex"].value_counts()
```

Out[4]:
```
Male      21790
Female    10771
Name: sex, dtype: int64
```

In [5]:
```
data[data["sex"] == "Female"]["age"].mean()
```

Out[5]:
```
36.85823043357163
```

In [6]:
```
print("{0:%}".format(data[data["native-country"] == "Germany"]
.shape[0] / data.shape[0]))
```

```
0.420749%
```

In [7]:
```
ages1 = data[data["salary"] == "<=50K"]["age"]
ages2 = data[data["salary"] == ">50K"]["age"]
print("<=50K: = {0} ± {1} years".format(ages1.mean(), ages1.std()))
print(" >50K: = {0} ± {1} years".format(ages2.mean(), ages2.std()))
```

```
<=50K: = 36.78373786407767 ± 14.020088490824813 years
 >50K: = 44.24984058155847 ± 10.51902771985177 years
```

In [8]:
```
high_educations = set(["Bachelors", "Prof-school", "Assoc-acdm",
"Assoc-voc", "Masters", "Doctorate"])
def high_educated(e):
  return e in high_educations
data[data["salary"] == ">50K"]["education"].map(high_educated).all()
```

Out[8]:
```
False
```

In [9]:
```
data.groupby(["race", "sex"])["age"].describe()
```

Out[9]:

| | | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|---|
| **race** | **sex** | | | | | | | | |
| **Amer-Indian-Eskimo** | **Female** | 119.0 | 37.117647 | 13.114991 | 17.0 | 27.0 | 36.0 | 46.00 | 80.0 |
| | **Male** | 192.0 | 37.208333 | 12.049563 | 17.0 | 28.0 | 35.0 | 45.00 | 82.0 |
| **Asian-Pac-Islander** | **Female** | 346.0 | 35.089595 | 12.300845 | 17.0 | 25.0 | 33.0 | 43.75 | 75.0 |
| | **Male** | 693.0 | 39.073593 | 12.883944 | 18.0 | 29.0 | 37.0 | 46.00 | 90.0 |
| **Black** | **Female** | 1555.0 | 37.854019 | 12.637197 | 17.0 | 28.0 | 37.0 | 46.00 | 90.0 |
| | **Male** | 1569.0 | 37.682600 | 12.882612 | 17.0 | 27.0 | 36.0 | 46.00 | 90.0 |
| **Other** | **Female** | 109.0 | 31.678899 | 11.631599 | 17.0 | 23.0 | 29.0 | 39.00 | 74.0 |
| | **Male** | 162.0 | 34.654321 | 11.355531 | 17.0 | 26.0 | 32.0 | 42.00 | 77.0 |
| **White** | **Female** | 8642.0 | 36.811618 | 14.329093 | 17.0 | 25.0 | 35.0 | 46.00 | 90.0 |
| | **Male** | 19174.0 | 39.652498 | 13.436029 | 17.0 | 29.0 | 38.0 | 49.00 | 90.0 |

In [10]:
```
data[(data["race"] == "Amer-Indian-Eskimo")
& (data["sex"] == "Male")]["age"].max()
```

Out[10]:
```
82
```

In [11]:
```
def is_married(m):
  return m.startswith("Married")
data["married"] = data["marital-status"].map(is_married)
(data[(data["sex"] == "Male") & (data["salary"] == ">50K")]
["married"].value_counts())
```

Out[11]:
```
True     5965
False     697
Name: married, dtype: int64
```

In [12]:
```
m = data["hours-per-week"].max()
print("Maximum is {} hours/week.".format(m))
people = data[data["hours-per-week"] == m]
c = people.shape[0]
print("{} people work this time at week.".format(c))
s = people[people["salary"] == ">50K"].shape[0]
print("{0:%} get >50K salary.".format(s / c))
```

```
Maximum is 99 hours/week.
85 people work this time at week.
29.411765% get >50K salary.
```

In [13]:
```
p = pd.crosstab(data["native-country"], data["salary"],
values=data['hours-per-week'], aggfunc="mean")
p
```

Out[13]:

| salary | <=50K | >50K |
|---|---|---|
| native-country | | |
| ? | 40.164760 | 45.547945 |
| Cambodia | 41.416667 | 40.000000 |
| Canada | 37.914634 | 45.641026 |
| China | 37.381818 | 38.900000 |
| Columbia | 38.684211 | 50.000000 |
| Cuba | 37.985714 | 42.440000 |
| Dominican-Republic | 42.338235 | 47.000000 |
| Ecuador | 38.041667 | 48.750000 |
| El-Salvador | 36.030928 | 45.000000 |
| England | 40.483333 | 44.533333 |
| France | 41.058824 | 50.750000 |
| Germany | 39.139785 | 44.977273 |
| Greece | 41.809524 | 50.625000 |
| Guatemala | 39.360656 | 36.666667 |
| Haiti | 36.325000 | 42.750000 |
| Holand-Netherlands | 40.000000 | NaN |
| Honduras | 34.333333 | 60.000000 |
| Hong | 39.142857 | 45.000000 |
| Hungary | 31.300000 | 50.000000 |
| India | 38.233333 | 46.475000 |
| Iran | 41.440000 | 47.500000 |

```
In [14]: p.loc["Japan"]

Out[14]: salary
         <=50K    41.000000
         >50K     47.958333
         Name: Japan, dtype: float64
```

## Часть 2

Требуется выполнить следующие запросы с использованием двух различных библиотек — Pandas и PandaSQL:

• один произвольный запрос на соединение двух наборов данных,

• один произвольный запрос на группировку набора данных с использованием функций агрегирования.

Также требуется сравнить время выполнения каждого запроса в Pandas и PandaSQL.

```
In [15]: !pip install pandasql

         Collecting pandasql
           Downloading https://files.pythonhosted.org/packages/6b/c4/ee4096ffa2eeeca0c749b26f0371bd26aa5c8b611c43de99a4f86d3de0a7/pandasql-0.7.3.t
         ar.gz
         Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from pandasql) (1.19.5)
         Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (from pandasql) (1.1.5)
         Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.7/dist-packages (from pandasql) (1.3.23)
         Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (from pandas->pandasql) (2.8.1)
         Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-packages (from pandas->pandasql) (2018.9)
         Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.7.3->pandas->pandasql) (1.15.
         0)
         Building wheels for collected packages: pandasql
           Building wheel for pandasql (setup.py) ... done
           Created wheel for pandasql: filename=pandasql-0.7.3-cp37-none-any.whl size=26820 sha256=3b1156e24c0b8ac95aed7a14d3eee140c5802273d78c32f
         056995cbd4a1866da
           Stored in directory: /root/.cache/pip/wheels/53/6c/18/b87a2e5fa8a82e9c026311de56210b8d1c01846e18a9607fc9
         Successfully built pandasql
         Installing collected packages: pandasql
         Successfully installed pandasql-0.7.3

In [16]: from pandasql import sqldf
         pysqldf = lambda q: sqldf(q, globals())

In [17]: wind = (pd.read_csv('wind speed.csv', header=None,
         names=["row", "UNIX", "date",
         "time", "speed", "text"])
         .drop("text", axis=1))
         temp = (pd.read_csv('temperature.csv', header=None,
         names=["row", "UNIX", "date",
         "time", "temperature", "text"])
         .drop("text", axis=1))
```

In [18]:
```
wind.head()
```

Out[18]:

| | row | UNIX | date | time | speed |
|---|---|---|---|---|---|
| **0** | 1 | 1475315718 | 2016-09-30 | 23:55:18 | 7.87 |
| **1** | 2 | 1475315423 | 2016-09-30 | 23:50:23 | 7.87 |
| **2** | 3 | 1475315124 | 2016-09-30 | 23:45:24 | 9.00 |
| **3** | 4 | 1475314821 | 2016-09-30 | 23:40:21 | 13.50 |
| **4** | 5 | 1475314522 | 2016-09-30 | 23:35:22 | 15.75 |

In [19]:
```
wind.dtypes
```

Out[19]:
```
row        int64
UNIX       int64
date      object
time      object
speed    float64
dtype: object
```

In [20]:
```
temp.head()
```

Out[20]:

| | row | UNIX | date | time | temperature |
|---|---|---|---|---|---|
| **0** | 1 | 1475315718 | 2016-09-30 | 23:55:18 | 48 |
| **1** | 2 | 1475315423 | 2016-09-30 | 23:50:23 | 48 |
| **2** | 3 | 1475315124 | 2016-09-30 | 23:45:24 | 48 |
| **3** | 4 | 1475314821 | 2016-09-30 | 23:40:21 | 48 |
| **4** | 5 | 1475314522 | 2016-09-30 | 23:35:22 | 48 |

In [21]:
```
temp.dtypes
```

Out[21]:
```
row             int64
UNIX            int64
date           object
time           object
temperature     int64
dtype: object
```

In [22]:
```
wind.merge(temp[["UNIX", "temperature"]], on="UNIX").head()
```

Out[22]:

| | row | UNIX | date | time | speed | temperature |
|---|---|---|---|---|---|---|
| **0** | 1 | 1475315718 | 2016-09-30 | 23:55:18 | 7.87 | 48 |
| **1** | 2 | 1475315423 | 2016-09-30 | 23:50:23 | 7.87 | 48 |
| **2** | 3 | 1475315124 | 2016-09-30 | 23:45:24 | 9.00 | 48 |
| **3** | 4 | 1475314821 | 2016-09-30 | 23:40:21 | 13.50 | 48 |
| **4** | 5 | 1475314522 | 2016-09-30 | 23:35:22 | 15.75 | 48 |

In [23]:
```
%%timeit
wind.merge(temp[["UNIX", "temperature"]], on="UNIX")
```

```
100 loops, best of 5: 10.9 ms per loop
```

In [24]:
```
pysqldf("""SELECT w.row, w.UNIX, w.date, w.time,
w.speed, t.temperature
FROM wind AS w JOIN temp AS t
ON w.UNIX = t.UNIX
""").head()
```

Out[24]:

| | row | UNIX | date | time | speed | temperature |
|---|---|---|---|---|---|---|
| **0** | 1 | 1475315718 | 2016-09-30 | 23:55:18 | 7.87 | 48 |
| **1** | 2 | 1475315423 | 2016-09-30 | 23:50:23 | 7.87 | 48 |
| **2** | 3 | 1475315124 | 2016-09-30 | 23:45:24 | 9.00 | 48 |
| **3** | 4 | 1475314821 | 2016-09-30 | 23:40:21 | 13.50 | 48 |
| **4** | 5 | 1475314522 | 2016-09-30 | 23:35:22 | 15.75 | 48 |

In [25]:
```
%%timeit
pysqldf("""SELECT w.row, w.UNIX, w.date, w.time,
w.speed, t.temperature
FROM wind AS w JOIN temp AS t
ON w.UNIX = t.UNIX
""")
```

```
1 loop, best of 5: 776 ms per loop
```

In [26]:
```
wind.groupby("date")["speed"].mean().head()
```

Out[26]:
```
date
2016-09-01    6.396560
2016-09-02    5.804086
2016-09-03    4.960248
2016-09-04    5.184571
2016-09-05    5.830676
Name: speed, dtype: float64
```

In [27]:
```
%%timeit
wind.groupby("date")["speed"].mean()
```

```
100 loops, best of 5: 2.81 ms per loop
```

In [28]:
```
pysqldf("""SELECT date, AVG(speed)
FROM wind
GROUP BY date
""").head()
```

Out[28]:

| | date | AVG(speed) |
|---|---|---|
| 0 | 2016-09-01 | 6.396560 |
| 1 | 2016-09-02 | 5.804086 |
| 2 | 2016-09-03 | 4.960248 |
| 3 | 2016-09-04 | 5.184571 |
| 4 | 2016-09-05 | 5.830676 |

In [29]:
```
%%timeit
pysqldf("""SELECT date, AVG(speed)
FROM wind
GROUP BY date
""")
```

```
1 loop, best of 5: 327 ms per loop
```

**Список литературы**


[1] Гапанюк Ю. Е. Лабораторная работа «Изучение библиотек обработки данных»
[Электронный ресурс] // GitHub. — 2019. — Режим доступа: https://github.com/
ugapanyuk/ml_course/wiki/LAB_PANDAS (дата обращения: 20.02.2019).
[2] pandas 0.24.1 documentation [Electronic resource] // PyData. — 2019. — Access mode:
http://pandas.pydata.org/pandas-docs/stable/ (online; accessed: 20.02.2019).
[3] You are my Sunshine [Electronic resource] // Space Apps Challenge. — 2017. —
Access mode: https://2017.spaceappschallenge.org/challenges/earth-and-us/
you-are-my-sunshine/details (online; accessed: 22.02.2019).

[4] yhat/pandasql: sqldf for pandas [Electronic resource] // GitHub. — 2017. — Access mode:
https://github.com/yhat/pandasql (online; accessed: 22.02.2019).
[5] Team The IPython Development. IPython 7.3.0 Documentation [Electronic resource] //
Read the Docs. — 2019. — Access mode: https://ipython.readthedocs.io/en/stable/ (online; accessed: 20.02.2019).