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
_____ Д. В. Лужевский

Москва, 2019 г.

1 Цель лабораторной работы

Изучить библиотеки обработки данных Pandas и PandaSQL.

2 Задание

Задание состоит из двух частей.

2.1 Часть 1

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

2.2 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.

3 Ход выполнения работы

3.1 Часть 1

Ниже приведён демонстрационный Jupyter-ноутбук «Exploratory data analysis with Pandas» курса mlcourse.ai (файл assignment01_pandas_uci_adult.ipynb). Все пояснения переведены на русский.

В этой задаче вы должны использовать Pandas, чтобы ответить на несколько вопросов о dataset Adult.

Уникальные значения всех функций (для получения дополнительной информации см. ссылки выше):

- age: continuous.
- workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- fnlwgt: continuous.
- education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
- education-num: continuous.
- marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.

- occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
- relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
- race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
- sex: Female, Male.
- capital-gain: continuous.
- capital-loss: continuous.
- hours-per-week: continuous.
- native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.
- salary: >50K, <=50K.

Пояснения:

- age возраст
- workclass: класс работы: индивидуальная, самостоятельная Эми-не-Инк, собственн-Эми-Инк, федеральные правительства, местные правительства, государства-гов, без-платить, не работал.
- fnlwgt
- education: образование: бакалавры, некоторые-колледж, 11-й, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9-й, 7-8-й, 12-й, магистры, 1-4-й, 10-й, докторантура, 5-6-й, дошкольное.
- education-num кол-во
- marital-status:супружеский статус: Женат-Сив-супруга, разведенных, никогда не состоявших в браке, разведенными, вдовами, вышли замуж-супруг-отсутствуют, женат-АФ-супруга.
- occupation:профессия: техник-поддержки, ремесло-ремонт, другое-сервиса, продаж, метод exec-управленческих, Проф-специальность, обработчиков-очистителей, машина-ОП-inspct, Адм-делопроизводство, сельское хозяйство-рыболовство, транспорт-переезд, прив-дом-серв, защитно-серв Вооруженных Сил.
- relationship: отношения: жена, собственный ребенок, муж, не член семьи, другой родственник, не женат.
- race:раса: Белый, азиат-Пак-островитянин, Амер-индеец-Эскимос, другой, черный.
- sex:пол: женский, мужской.
- capital-gain: прирост капитала
- capital-loss: потери капитала
- hours-per-week:часы работы в неделю: непрерывно.
- native-country:родная страна: Соединенные Штаты, Камбоджа, Англия, Пуэрто-Рико, Канада, Германия, окраинные США (Гуам-USVI-etc), Индия, Япония, Греция, Юг, Китай, Куба, Иран, Гондурас, Филиппины, Италия, Польша, Ямайка, Вьетнам, Мексика, Португалия, Ирландия, Франция, Доминиканская Республика, Лаос, Эквадор, Тайвань, Гаити, Колумбия, Венгрия, Гватемала, Никарагуа, Шотландия, Таиланд, Югославия, Сальвадор, Тринадад&Тобаго, Перу, Хонг, Холанд-Нидерланды.
- salary зарплата: > 50к, <=50к.

Импорт всех необходимых пакетов:

```
In [4]: import pandas as pd
```

Установка максимальной ширины отображения для текстового отчета:

```
In [16]: pd.set_option("display.width", 70)
```

Загрузка данных:

```
In [20]: data = pd.read_csv('data/adult2.data.csv', sep=",")

         data.head()
```

```
Out[20]:    age         workclass  fnlwgt  education  education-num  \
         0   39         State-gov   77516  Bachelors             13
         1   50  Self-emp-not-inc   83311  Bachelors             13
         2   38           Private  215646    HS-grad              9
         3   53           Private  234721       11th              7
         4   28           Private  338409  Bachelors             13

                marital-status         occupation   relationship   race  \
         0       Never-married       Adm-clerical  Not-in-family  White
         1  Married-civ-spouse    Exec-managerial        Husband  White
         2            Divorced  Handlers-cleaners  Not-in-family  White
         3  Married-civ-spouse  Handlers-cleaners        Husband  Black
         4  Married-civ-spouse     Prof-specialty           Wife  Black

               sex  capital-gain  capital-loss  hours-per-week  \
         0    Male          2174             0              40
         1    Male             0             0              13
         2    Male             0             0              40
         3    Male             0             0              40
         4  Female             0             0              40

           native-country salary
         0  United-States  <=50K
         1  United-States  <=50K
         2  United-States  <=50K
         3  United-States  <=50K
         4           Cuba  <=50K
```

1. Сколько мужчин и женщин (половые признаки) представлены в этом наборе данных?

```
In [21]: data["sex"].value_counts()
```

```
Out[21]: Male      21790
         Female    10771
         Name: sex, dtype: int64
```

2.Каков средний возраст (возрастная характеристика) женщин? Мужчины

```
In [22]: data[data["sex"] == "Female"]["age"].mean()

Out[22]: 36.85823043357163

In [23]: data[data["sex"] == "Male"]["age"].mean()

Out[23]: 39.43354749885268
```

3. Каков процент граждан Германии (особенность родной страны)?

```
In [28]: print("{0:%}".format(data[data["native-country"] == "Germany"].shape[0] / data.shape[0]

0.420749%
```

data.shape[0] все записи можно сказать id

4-5. Каковы среднее и стандартное отклонение возраста для тех, кто зарабатывает более 50 тыс. в год (функция зарплаты), и тех, кто зарабатывает менее 50 тыс. в год?

```
In [43]: 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 ± 14.02008849082488 years
 >50K: = 44.24984058155847 ± 10.519027719851826 years
```

6. Правда ли, что люди, которые зарабатывают более 50 тысяч, имеют хотя бы среднее образование? (образование – бакалавры, Проф-училище, ДОЦ-acdm, ДОЦ-лов, мастера или особенность доктора)

```
In [44]: 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[44]: False
```

7. Отображение статистики возраста для каждой расы (функция расы) и каждого пола (функция пола). Используйте groupby () и describe (). Найти максимальный возраст мужчин Амер-Индо-эскимосской расы.

```
In [51]: data.groupby(["race", "sex"])["age"].describe()
```

```
Out[51]:                                  count       mean        std   min  \
         race                 sex
         Amer-Indian-Eskimo   Female    119.0  37.117647  13.114991  17.0
                              Male      192.0  37.208333  12.049563  17.0
         Asian-Pac-Islander   Female    346.0  35.089595  12.300845  17.0
                              Male      693.0  39.073593  12.883944  18.0
         Black                Female   1555.0  37.854019  12.637197  17.0
                              Male     1569.0  37.682600  12.882612  17.0
         Other                Female    109.0  31.678899  11.631599  17.0
                              Male      162.0  34.654321  11.355531  17.0
         White                Female   8642.0  36.811618  14.329093  17.0
                              Male    19174.0  39.652498  13.436029  17.0

                                        25%   50%    75%   max
         race                 sex
         Amer-Indian-Eskimo   Female  27.0  36.0  46.00  80.0
                              Male    28.0  35.0  45.00  82.0
         Asian-Pac-Islander   Female  25.0  33.0  43.75  75.0
                              Male    29.0  37.0  46.00  90.0
         Black                Female  28.0  37.0  46.00  90.0
                              Male    27.0  36.0  46.00  90.0
         Other                Female  23.0  29.0  39.00  74.0
                              Male    26.0  32.0  42.00  77.0
         White                Female  25.0  35.0  46.00  90.0
                              Male    29.0  38.0  49.00  90.0
```

```
In [54]: data[(data["race"] == "Amer-Indian-Eskimo")  &
              (data["sex"] == "Male")]["age"].max()
```

```
Out[54]: 82
```

8. Среди кого доля тех, кто зарабатывает много (>50 тыс.) больше: женатые или одинокие мужчины (особенность семейного положения)? Считать женатыми тех, кто имеет семейное положение, начиная с женатого (женатый-гражданский супруг, женатый-отсутствующий супруг или женатый-AF-супруг), остальные считаются холостяками.

```
In [55]: 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[55]: True     5965
         False     697
         Name: married, dtype: int64
```

9. Какое максимальное количество часов человек работает в неделю (функция часов в неделю)? Сколько человек работает такое количество часов, и каков процент тех, кто зарабатывает больше (>50K) среди них?

```
In [56]: 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.
```

10. Подсчитайте среднее время работы (часов в неделю) для тех, кто зарабатывает мало и много (зарплата) для каждой страны (родной страны). Что это будет для Японии?

```
In [58]: p = pd.crosstab(data["native-country"], data["salary"],
                         values=data['hours-per-week'], aggfunc="mean")
         p
```

Out[58]:

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
Ireland	40.947368	48.000000
Italy	39.625000	45.400000
Jamaica	38.239437	41.100000
Japan	41.000000	47.958333

```
Laos                          40.375000  40.000000
Mexico                        40.003279  46.575758
Nicaragua                     36.093750  37.500000
Outlying-US(Guam-USVI-etc)    41.857143        NaN
Peru                          35.068966  40.000000
Philippines                   38.065693  43.032787
Poland                        38.166667  39.000000
Portugal                      41.939394  41.500000
Puerto-Rico                   38.470588  39.416667
Scotland                      39.444444  46.666667
South                         40.156250  51.437500
Taiwan                        33.774194  46.800000
Thailand                      42.866667  58.333333
Trinadad&Tobago               37.058824  40.000000
United-States                 38.799127  45.505369
Vietnam                       37.193548  39.200000
Yugoslavia                    41.600000  49.500000
```

```
In [59]: p.loc["Japan"]
```

```
Out[59]: salary
         <=50K    41.000000
         >50K     47.958333
         Name: Japan, dtype: float64
```

3.2 Часть 2

Импортируем pandasql:

```
In [1]: from pandasql import sqldf
        pysqldf = lambda q: sqldf(q, globals())
```

Для выполнения данного задания возьмём два набора данных из исходных данных, представленных NASA для своего хакатона по предсказанию мощности солнечного излучения:

```
In [5]: wind = (pd.read_csv('data/wind speed.csv', header=None,
                            names=["row", "UNIX", "date",
                                   "time", "speed", "text"])
                            .drop("text", axis=1))
        temp = (pd.read_csv('data/temperature.csv', header=None,
                            names=["row", "UNIX", "date",
                                   "time", "temperature", "text"])
                            .drop("text", axis=1))
```

Посмотрим на эти наборы данных:

```
In [6]: wind.head()
```

```
Out[6]:    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 [7]: temp.head()

Out[7]:    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
```

Объединим эти наборы данных различными способами, проверяя время их выполнения:

```
In [17]: wind.merge(temp[["UNIX", "temperature"]], on="UNIX").head()

Out[17]:    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 [18]: %%timeit
         wind.merge(temp[["UNIX", "temperature"]], on="UNIX")

24 ms ± 1.17 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)


In [22]: %%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
                 """)

1.11 s ± 33.5 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

При запросе на соединения pandoSql показывает лучший результат чем pandas. Примерно в 25 раз.

Сгруппируем набор данных с использованием функций агрегирования различными способами:

```
In [24]: wind.groupby("date")["speed"].mean().head()
```

```
Out[24]: 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 [25]: %%timeit
         wind.groupby("date")["speed"].mean()

5.16 ms ± 127 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)


In [26]: pysqldf("""SELECT date, AVG(speed)
                    FROM wind
                    GROUP BY date
                 """).head()

Out[26]:          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 [27]: %%timeit
         pysqldf("""SELECT date, AVG(speed)
                    FROM wind
                    GROUP BY date
                 """).head()

439 ms ± 13.7 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

При запросе с группировкой pandos показывает лучший результат чем pandaSql. Примерно в 100 раз.

Таким образом для разных задач, нужно подбирать разные библиотеки.