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

Отчет по лабораторной работе № 2

«Изучение библиотек обработки данных.»

По курсу «Технологии машинного обучения»

ИСПОЛНИТЕЛЬ:

Коционова А. А.
Группа ИУ5-63

_____

"__"_____2019 г.

ПРЕПОДАВАТЕЛЬ:

Гапанюк Ю. Е.

_____

"__"_____2019 г.

Москва 2019

Цель лабораторной работы: изучение библиотек обработки данных Pandas и PandaSQL.

Часть 1

Выполните первое демонстрационное задание "demo assignment" под названием "Exploratory data analysis with Pandas" со страницы курса https://mlcourse.ai/assignments

Условие задания - https://nbviewer.jupyter.org/github/Yorko/mlcourse_open/blob/master/jupyter_english/assignments_demo/assignment01_pandas_uci_adult.ipynb?flush_cache=true

Набор данных можно скачать здесь - https://archive.ics.uci.edu/ml/datasets/Adult

In [ ]:

```
import numpy as np
import pandas as pd
pd.set_option('display.max.columns', 100)
import warnings
warnings.filterwarnings('ignore')
import io
import requests
```

In [6]:

```
data = pd.read_csv('../Downloads/adult.data.csv')
data.head()
```

Out[6]:

	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

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
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

1. How many men and women (sex feature) are represented in this dataset?

In [7]:
```
data['sex'].value_counts()
```

Out[7]:
```
Male      21790
Female    10771
Name: sex, dtype: int64
```

2. What is the average age (age feature) of women?

In [8]:
```
data.loc[data['sex'] == 'Female', 'age'].mean()#как бы по пересечению строки и столбца смотрим
```

Out[8]:
```
36.85823043357163
```

3. What is the proportion of German citizens (native-country feature)?

In [9]:
```
float((data['native-country'] == 'Germany').sum()) / data.shape[0]
```

Out[9]:
```
0.004207487485028101
```

4-5. What are mean value and standard deviation of the age of those who recieve more than 50K per year (salary feature) and those who receive less than 50K per year?

In [11]:
```
ages1 = data.loc[data['salary'] == '>50K', 'age']
ages2 = data.loc[data['salary'] == '<=50K', 'age']
print("The average age of the rich: {0} +- {1} years, poor - {2} +- {3} years
.".format(
    round(ages1.mean()), round(ages1.std(), 1),
    round(ages2.mean()), round(ages2.std(), 1)))
```

```
The average age of the rich: 44 +- 10.5 years, poor - 37 +- 14.0 years.
```

6. Is it true that people who receive more than 50k have at least high school education? (education - Bachelors, Prof-school, Assoc-acdm, Assoc-voc, Masters or Doctorate feature)

In [12]:
```
data.loc[data['salary'] == '>50K', 'education'].unique() # No
```

Out[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)
```

7. Display statistics of age for each race (race feature) and each gender. Use groupby() and describe(). Find the maximum age of men of Amer-Indian-Eskimo race.

In [14]:
```
for (race, sex) in data.groupby(['race', 'sex']):
    print("Race: {0}, sex: {1}".format(race, sex))
    #print(sub_df['age'].describe())
```

8. Among whom the proportion of those who earn a lot(>50K) is more: among married or single men (marital-status feature)? Consider married those who have a marital-status starting with Married (Married-civ-spouse, Married-spouse-absent or Married-AF-spouse), the rest are considered bachelors.

In [18]:
```
data.loc[(data['sex'] == 'Male') &
     (data['marital-status'].isin(['Never-married',
                                   'Separated',
                                   'Divorced',
                                   'Widowed'])), 'salary'].value_counts()
```

Out[18]:
```
<=50K    7552
>50K      697
Name: salary, dtype: int64
```

In [19]:
```
data.loc[(data['sex'] == 'Male') &
     (data['marital-status'].str.startswith('Married')), 'salary'].value_coun
ts()
```

Out[19]:
```
<=50K    7576
>50K     5965
Name: salary, dtype: int64
```

In [20]:

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

```
Out[20]:
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
```

9. What is the maximum number of hours a person works per week (hours-per-week feature)? How many people work such a number of hours and what is the percentage of those who earn a lot among them?

```
In [21]:
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%
```

10. Count the average time of work (hours-per-week) those who earning a little and a lot (salary) for each country (native-country). What will these be for Japan?

```
In [22]:
for (country, salary), sub_df in data.groupby(['native-country', 'salary']):
    print(country, salary, round(sub_df['hours-per-week'].mean(), 2))
```

```
? <=50K 40.16
? >50K 45.55
```

```
In [37]:
for salary, sub in data.loc[data['native-country']=='Japan'].groupby('salary'
):
    print("Hours-per-week: {}".format(sub['hours-per-week'].mean()))
```

```
Hours-per-week: 41.0
Hours-per-week: 47.958333333333336
```

Часть 2¶

Выполните 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

Сравните время выполнения каждого запроса в Pandas и PandaSQL.

In [0]:
```
import pandas as pd
import pandasql as ps
```

In [1]:
```
pd.__version__
```

Out[3]:
```
'0.23.4'
```

In [5]:
```
android_devices = pd.read_csv('../AndroidML/android_devices.csv')
user_device = pd.read_csv('../AndroidML/user_device.csv')
user_usage = pd.read_csv('../AndroidML/user_usage.csv')
```

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

Out[6]:

	Retail Branding	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

In [7]:
```
user_device.head()
```

Out[7]:

	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

In [8]:
```
user_usage.head()
```

Out[8]:

	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

Сравнение JOIN

In [30]:
```
%%timeit
result1 = pd.merge(user_usage,
                   user_device[['use_id', 'platform', 'device']],
                   on='use_id',
                   how='left')
```

```
4.42 ms ± 163 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

In [29]:
```
result1.head()
```

Out[29]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	platform	device
0	21.97	4.82	1557.33	22787	android	GT-I9505
1	1710.08	136.88	7267.55	22788	android	SM-G930F
2	1710.08	136.88	7267.55	22789	android	SM-G930F
3	94.46	35.17	519.12	22790	android	D2303
4	71.59	79.26	1557.33	22792	android	SM-G361F

То же самое на pandasql

In [33]:
```
%%timeit
result2 = ps.sqldf("""SELECT A.*, B.user_id, B.platform, B.device
FROM user_usage AS A
LEFT JOIN user_device AS B
ON A.USE_ID=B.USE_ID""",globals())
```

```
26.4 ms ± 1.84 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

In [32]:
```
result2.head()
```

Out[32]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	user_id	platform	device
0	21.97	4.82	1557.33	22787	12921.0	android	GT-I9505
1	1710.08	136.88	7267.55	22788	28714.0	android	SM-G930F
2	1710.08	136.88	7267.55	22789	28714.0	android	SM-G930F
3	94.46	35.17	519.12	22790	29592.0	android	D2303
4	71.59	79.26	1557.33	22792	28217.0	android	SM-G361F

Таким образом PANDASQL сработал в 6 раз медленее чем PANDAS на джойнах

Сравнение GROUP BY

```
In [59]:
%%timeit
result11 = result1.astype(str).groupby("platform")['platform'].count()
```

```
2.75 ms ± 141 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

```
In [56]:
result11
```

```
Out[56]:
platform
android    157
ios          2
nan         81
Name: platform, dtype: int64
```

```
In [60]:
%%timeit
result21 = ps.sqldf('''SELECT count(*), platform
        FROM result2
        GROUP BY platform
    ''',globals())
```

```
18.4 ms ± 1.03 ms per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

```
In [58]:
result21
```

```
Out[58]:
```

	count(*)	platform
0	81	None
1	157	android
2	2	ios

Таким образом PANDASQL сработал вновь в 6 раз медленее чем PANDAS на группировке