Московский Государственный Технический Университет
им. Н. Э. Баумана

Лабораторная работа №2
по курсу: «Технологии машинного обучения»

Изучение библиотек обработки данных.

Выполнила:
Студентка группы ИУ5-63
Нурлыева Д.Д.

Москва
2019

Задание:
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

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.
Текст программы:

```
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')
In [62]:
data = pd.read_csv('/Users/user/Desktop/adult.data.txt')
data.head()
Out[62]:
```

	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 [64]:
```
#1. How many men and women (sex feature) are represented
in this dataset?
data['sex'].value_counts()
```
Out[64]:
```
Male      21790
Female    10771
Name: sex, dtype: int64
```
In [65]:
```
#2. What is the average age (age feature) of women?
data.loc[data['sex'] == 'Female', ['age']].mean()
```
Out[65]:
```
age    36.85823
dtype: float64
```
In [68]:
```
#3. What is the percentage of German citizens (native-
country feature)?
float((data['native-country'] == 'Germany').sum()) /
data.shape[0]
```
Out[68]:
```
0.004207487485028101
```
In [69]:
```
#4-5. What are the mean and standard deviation of age for
those who earn more than 50K per year (salary feature)
and those who earn less than 50K per year?
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 [70]:

```
#6. Is it true that people who earn more than 50K have at
least high school education? (education – Bachelors,
Prof-school, Assoc-acdm, Assoc-voc, Masters or Doctorate
feature)
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[70]:

```
False
```

In [71]:

```
#7. Display age statistics for each race (race feature)
and each gender (sex feature). Use groupby() and
describe(). Find the maximum age of men of Amer-Indian-
Eskimo race.
data.groupby(["race", "sex"])["age"].describe()
```

Out[71]:

		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 [72]:
```
#8. Among whom is the proportion of those who earn a lot
(>50K) greater: married or single men (marital-status
feature)? Consider as 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.
data.loc[(data['sex'] == 'Male') &
     (data['marital-status'].isin(['Never-married',
                                   'Separated',
                                   'Divorced',
                                   'Widowed'])),
'salary'].value_counts()
```

Out[72]:
```
<=50K    7552
>50K      697
Name: salary, dtype: int64
```

In [73]:
```
data.loc[(data['sex'] == 'Male') &
     (data['marital-status'].str.startswith('Married')),
'salary'].value_counts()
```

Out[73]:
```
<=50K    7576
>50K     5965
Name: salary, dtype: int64
```

In [74]:
```
data['marital-status'].value_counts()
```

Out[74]:
```
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
In [75]:
#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 (>50K) among them?
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 [76]:
#10. Count the average time of work (hours-per-week) for
those who earn a little and a lot (salary) for each
country (native-country). What will these be for Japan?
p = pd.crosstab(data["native-country"], data["salary"],
                values=data['hours-per-week'],
aggfunc="mean")
p
Out[76]:
```

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 [77]:
p.loc["Japan"]
Out[77]:
salary
<=50K    41.000000
>50K     47.958333
Name: Japan, dtype: float64
In [78]:
!pip install pandasql

Requirement already satisfied: pandasql in /anaconda3/
lib/python3.6/site-packages (0.7.3)
Requirement already satisfied: numpy in /anaconda3/lib/
python3.6/site-packages (from pandasql) (1.14.0)
Requirement already satisfied: pandas in /anaconda3/lib/
python3.6/site-packages (from pandasql) (0.22.0)
Requirement already satisfied: sqlalchemy in /anaconda3/
lib/python3.6/site-packages (from pandasql) (1.2.1)
Requirement already satisfied: python-dateutil>=2 in /
anaconda3/lib/python3.6/site-packages (from pandas-
>pandasql) (2.6.1)
Requirement already satisfied: pytz>=2011k in /anaconda3/
lib/python3.6/site-packages (from pandas->pandasql)
(2017.3)
Requirement already satisfied: six>=1.5 in /anaconda3/
lib/python3.6/site-packages (from python-dateutil>=2-
>pandas->pandasql) (1.11.0)
You are using pip version 19.0.1, however version 19.0.3
is available.
You should consider upgrading via the 'pip install --
upgrade pip' command.
In [80]:
```

```
android_devices = pd.read_csv('/Users/user/Desktop/
pandas/android_devices.csv')
user_device = pd.read_csv('/Users/user/Desktop/pandas/
user_device.csv')
user_usage = pd.read_csv('/Users/user/Desktop/pandas/
user_usage.csv')
```

In [81]:

```
android_devices.head()
```

Out[81]:

	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 [82]:

```
user_device.head()
```

Out[82]:

	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 [83]:

```
user_usage.head()
```

Out[83]:

	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

In [91]:
```
result = pd.merge(user_usage,
                  user_device[['use_id', 'platform',
'device']],
                  on='use_id',
                  how='left')
result.head()
```
Out[91]:

	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

In [92]:
```
result.tail()
```
Out[92]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	platform	device
235	260.66	68.44	896.96	25008	NaN	NaN
236	97.12	36.50	2815.00	25040	NaN	NaN
237	355.93	12.37	6828.09	25046	NaN	NaN
238	632.06	120.46	1453.16	25058	NaN	NaN
239	488.70	906.92	3089.85	25220	NaN	NaN

In [94]:
```
#агрегация
result.agg({'outgoing_mins_per_month':
['sum','min','max','mean'],'outgoing_sms_per_month':
['sum','min','max','mean']})
```
Out[94]:

	outgoing_mins_per_month	outgoing_sms_per_month
sum	65894.200000	23752.390000
min	0.500000	0.250000
max	1816.630000	906.920000
mean	274.559167	98.968292

In [97]:
```
#Pandasql
import pandasql as ps
def ex_pandasql(result):
    simple_query = '''
        SELECT
            outgoing_mins_per_month,
            monthly_mb,
            platform
        FROM result
        where platform='ios'
        LIMIT 10
        '''
    return ps.sqldf(simple_query, locals())
```
In [98]:
```
ex_pandasql(result)
```
Out[98]:

	outgoing_mins_per_month	monthly_mb	platform
0	681.44	1271.39	ios
1	50.68	650.92	ios

In [99]:
```
def agg_ps(result):
    agg_ps = '''
    SELECT
            count(*),
            platform
        FROM result
```

```
        GROUP BY platform
        LIMIT 10
    '''
    return ps.sqldf(agg_ps, locals())
```

In [100]:
```
agg_ps(result)
```

Out[100]:

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

In [104]:
```
def join_ps(user_usage, user_device):
    join_ps = '''
    SELECT*
    FROM user_usage t0
    JOIN user_device t1
    ON t0.use_id = t1.use_id'''
    return ps.sqldf(join_ps)
```

In [105]:
```
join_ps(user_usage, user_device)
```

Out[105]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	use_id	user_id	platform	platform_version	device	use_type_id
0	21.97	4.82	1557.33	2278	2278	1292	android	4.3	GT-I9505	1
1	1710.08	136.88	7267.55	2278	2278	2871	android	6.0	SM-G930F	1
2	1710.08	136.88	7267.55	2278	2278	2871	android	6.0	SM-G930F	1
3	94.46	35.17	519.12	2279	2279	2959	android	5.1	D2303	1

4	71.59	79.26	1557.33	2279	2279	2821	android	5.1	SM-G361F	1
5	71.59	79.26	1557.33	2279	2279	2821	android	5.1	SM-G361F	1
6	71.59	79.26	519.12	2279	2279	2821	android	5.1	SM-G361F	1
7	71.59	79.26	519.12	2279	2279	2821	android	5.1	SM-G361F	1
8	30.92	22.77	3114.67	2279	2279	2964	android	6.0	ONEPLUS A3003	1
9	69.80	14.70	25955.55	2280	2280	1097	android	4.4	GT-I9505	1
10	554.41	150.06	3114.67	2280	2280	2964	android	6.0	SM-G935F	1
11	189.10	24.08	519.12	2280	2280	2964	android	4.2	GT-I9195	1
12	283.30	107.47	15573.33	2280	2280	2161	android	6.0	A0001	1
13	324.34	92.52	519.12	2280	2280	2906	android	6.0	SM-G900F	1
14	797.06	7.67	519.12	2281	2281	2341	android	4.4	HTC Desire 510	1
15	797.06	7.67	15573.33	2281	2281	2341	android	4.4	HTC Desire 510	1

16	797.06	7.67	1557 3.33	2281	2281	2341	and roid	4.4	HTC Desir e 510	1
17	797.06	7.67	1557 3.33	2281	2281	2341	and roid	4.4	HTC Desir e 510	1
18	797.06	7.67	1557 3.33	2281	2281	2341	and roid	4.4	HTC Desir e 510	1
19	78.80	327.33	1038 2.21	2281	2281	2965	and roid	4.4	HTC One mini 2	1
20	78.80	327.33	1557 3.33	2282	2282	2965	and roid	4.4	HTC One mini 2	1
21	78.80	327.33	1557 3.33	2282	2282	2965	and roid	4.4	HTC One mini 2	1
22	164.10	192.64	3114 .67	2282	2282	2965	and roid	6.0	SM-G90 0F	1
23	208.26	91.76	5191 .12	2282	2282	2895	and roid	6.0	SM-G90 0F	1
24	681.44	47.35	1271 .39	2282	2282	2965	ios	10.1	iPho ne7, 2	2
25	324.27	91.50	519. 12	2283	2283	2906	and roid	6.0	SM-G90 0F	1
26	85.97	26.94	407. 01	2283	2283	6541	and roid	4.1	GT-I819 0N	1

27	244.88	105.95	1557.33	2283	2283	2929	android	6.0	D5803	1
28	135.09	42.02	5191.12	2283	2283	2484	android	6.0	E6653	1
29	57.49	16.73	15573.33	2283	2283	2965	android	6.0	A0001	1
...	...	...	...	...	...	...	...	...	...	...
129	322.33	86.39	3114.67	2300	2300	2889	android	6.0	SM-G920F	1
130	124.70	4.64	11.68	2300	2300	2970	android	5.1	HUAWEI CUN-L01	1
131	37.27	136.10	1557.33	2300	2300	2669	android	6.0	SM-G900F	1
132	50.68	540.60	650.92	2301	2301	2971	ios	9.3	iPhone6,2	2
133	28.74	29.52	3114.67	2301	2301	1426	android	6.0	SM-G900F	1
134	87.76	140.61	1557.33	2301	2301	2894	android	6.0	SM-A300FU	1
135	99.81	403.78	3114.67	2301	2301	2971	android	6.0	SM-G900F	1
136	55.96	0.25	2076.45	2301	2301	2966	android	6.0	F3111	1
137	101.59	84.41	5191.12	2301	2301	2945	android	6.0	Moto G (4)	1
138	126.30	135.35	519.12	2301	2301	2971	android	5.1	SM-J320FN	1

139	42.93	124.33	519.12	2302	2302	2971	and roid	5.1	SM-G361F	1
140	63.56	26.87	9344.00	2302	2302	2822	and roid	6.0	SM-G930F	1
141	157.33	8.87	1557.33	2302	2302	2964	and roid	7.0	ONE PLUS A3003	1
142	70.34	18.00	212.64	2302	2302	2863	and roid	6.0	Moto E2(4G-LTE)	1
143	532.98	44.36	2076.45	2302	2302	2276	and roid	6.0	ONE A2003	1
144	60.08	261.33	12458.67	2302	2302	1810	and roid	4.4	X11	1
145	92.52	162.39	1557.33	2302	2302	2971	and roid	5.1	C6603	1
146	22.85	34.54	6577.12	2302	2302	2971	and roid	6.0	HTC One_M8	1
147	227.13	76.94	0.00	2303	2303	2797	and roid	5.1	SM-J320FN	1
148	227.13	76.94	1038.21	2303	2303	2797	and roid	5.1	SM-J320FN	1
149	227.13	76.94	1038.21	2303	2303	2797	and roid	5.1	SM-J320FN	1
150	57.66	62.85	1557.33	2303	2303	2971	and roid	5.1	VF-795	1

151	180.18	17.49	2076.45	2303	2303	2972	android	5.1	SM-G531F	1
152	12.85	58.32	74.40	2304	2304	2972	android	4.4	HTC Desire 620	1
153	198.59	90.49	5191.12	2304	2304	2895	android	6.0	SM-G900F	1
154	198.59	90.49	5191.12	2304	2304	2895	android	6.0	SM-G900F	1
155	198.59	90.49	3114.67	2304	2304	2895	android	6.0	SM-G900F	1
156	106.65	82.13	5191.12	2304	2304	2945	android	6.0	Moto G (4)	1
157	344.53	20.53	519.12	2304	2304	2972	android	6.0	SM-G900F	1
158	42.75	46.83	5191.12	2305	2305	2025	android	5.1	Vodafone Smart ultra 6	1

159 rows × 10 columns