**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5.

Курс «Технологии машинного обучения»

Отчет по лабораторной работе №2

«Изучение библиотек обработки данных»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-62 |  | преподаватель каф. ИУ5 |
| Король Константин |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2019 г.

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 приведены в коде программы.

#### Часть 2.

Выполните следующие запросы с использованием двух различных библиотек - [Pandas](https://pandas.pydata.org/) и [PandaSQL](https://github.com/yhat/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.

**Код программы**

**import** **numpy** **as** **np**

**import** **pandas** **as** **pd**

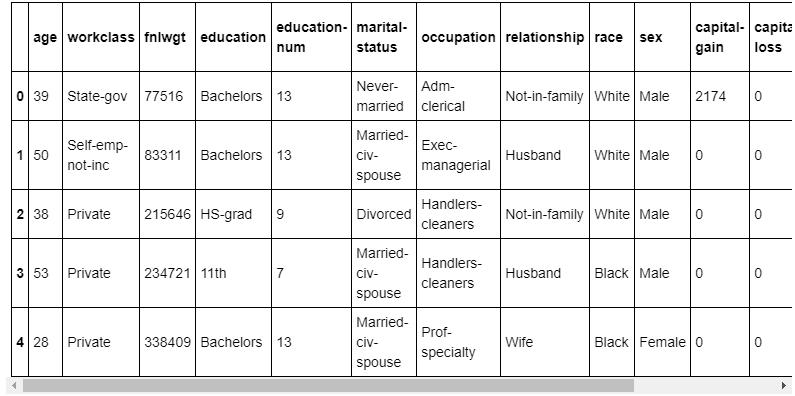
**import** **matplotlib.pyplot** **as** **plt**

%matplotlib inline

pd.set\_option("display.width", 70)

data = pd.read\_csv('data/lab2.csv')

data.head()



data.columns

OUT: Index(['age', 'workclass', 'fnlwgt', 'education', 'education-num',

'marital-status', 'occupation', 'relationship', 'race', 'sex',

'capital-gain', 'capital-loss', 'hours-per-week',

'native-country', 'salary'],

dtype='object')

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

data["sex"].value\_counts()

OUT: Male 21790

Female 10771

Name: sex, dtype: int64

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

data[data['sex'] == ' Female']['age'].mean()

OUT: 36.85823043357163

1. What is the percentage of German citizens (native-country feature)?

print("**{0:%}**".format(data[data['native-country'] == ' Germany']

.shape[0] / data.shape[0]))

OUT: 0.420749%

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("Зарплата <=50K: = **{0}** ± **{1}** лет".format(ages1.mean(), ages1.std()))

print("Зарплата >50K: = **{0}** ± **{1}** лет".format(ages2.mean(), ages2.std()))

OUT: Зарплата <=50K: = 36.78373786407767 ± 14.020088490824813 лет

Зарплата >50K: = 44.24984058155847 ± 10.51902771985177 лет

1. 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)

edu = set([" Bachelors", " Prof-school", " Assoc-acdm",

" Assoc-voc", " Masters", " Doctorate"])

**def** education(e):

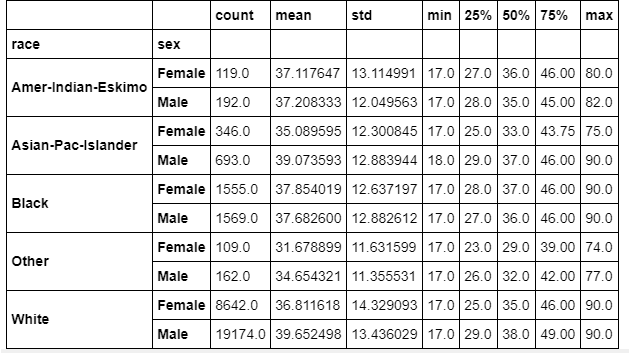
**return** e **in** edu

data[data["salary"] == " >50K"]["education"].map(education).all()

OUT: False

1. 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()



1. 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.

**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: True 5965

False 697

Name: married, dtype: int64

1. 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("Максимальное кол-во рабочих часов в неделю: **{}**".format(m))

people = data[data["hours-per-week"] == m]

c = people.shape[0]

print("**{}** человек работает максимальное кол-во часов в неделю.".format(c))

s = people[people["salary"] == " >50K"].shape[0]

print("**{0:%}** из них зарабатывают >50K.".format(s / c))

OUT: Максимальное кол-во рабочих часов в неделю: 99

85 человек работает максимальное кол-во часов в неделю.

29.411765% из них зарабатывают >50K.

1. 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



p.loc[" Japan"]

OUT: salary

<=50K 41.000000

>50K 47.958333

Name: Japan, dtype: float64

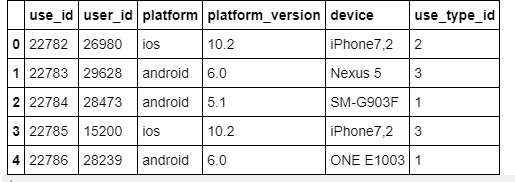
!pip install pandasql

**from** **pandasql** **import** sqldf

pysqldf = **lambda** q: sqldf(q, globals())

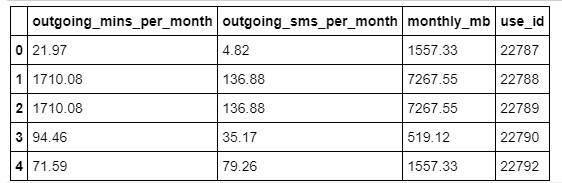
ud = pd.read\_csv('data/user\_device.csv')

ud.head()



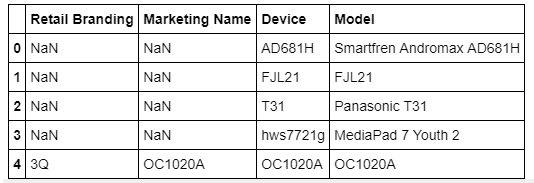
uu = pd.read\_csv('data/user\_usage.csv')

uu.head()



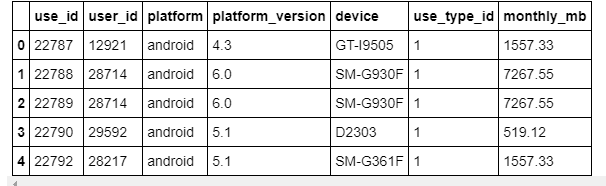
ad = pd.read\_csv('data/android\_devices.csv')

ad.head()



res1 = pd.merge(ud, uu[['monthly\_mb', 'use\_id']], on='use\_id')

res1.head()



%%timeit

res1 = pd.merge(ud, uu[['monthly\_mb', 'use\_id']], on='use\_id')

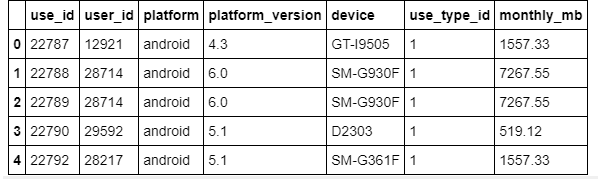
OUT: 3.05 ms ± 108 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)

pysqldf("""SELECT a.use\_id, a.user\_id, a.platform, a.platform\_version, a.device,

a.use\_type\_id, b.monthly\_mb

FROM ud AS a JOIN uu AS b

ON a.use\_id = b.use\_id""").head()



%%timeit

pysqldf("""SELECT a.use\_id, a.user\_id, a.platform, a.platform\_version, a.device,

a.use\_type\_id, b.monthly\_mb

FROM ud AS a JOIN uu AS b

ON a.use\_id = b.use\_id""")

OUT: 9.46 ms ± 215 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)

res1.groupby('device')['monthly\_mb'].mean().head()

OUT: device

A0001 15573.33

C6603 1557.33

D2303 519.12

D5503 1557.33

D5803 1557.33

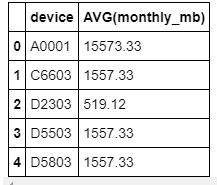
Name: monthly\_mb, dtype: float64

%%timeit

res1.groupby('device')['monthly\_mb'].mean()

OUT: 528 µs ± 8.5 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)

pysqldf("""SELECT device, AVG(monthly\_mb) FROM res1 GROUP BY device""").head()



%%timeit

pysqldf("""SELECT device, AVG(monthly\_mb) FROM res1 GROUP BY device""")

5.21 ms ± 95.9 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)