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 ИУ5-23М
Богомолов Д.Н.

Москва — 2020 г.

In this task you should use Pandas to answer a few questions about the Adult dataset. (You don't here). Choose the answers in the web-form. This is a demo version of an assignment, so by sub solution .ipynb file.

Unique values of all features (for more information, please see the links above):

- `age`: continuous.
- `workclass`: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Withou
- `fnlwgt`: continuous.
- `education`: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9t Doctorate, 5th-6th, Preschool.
- `education-num`: continuous.
- `marital-status`: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married
- `occupation`: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, 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 South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Port Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Peru, Hong, Holand-Netherlands.
- `salary`: >50K,<=50K

```
import pandas as pd
```

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

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relati
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-i
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	H
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-i
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	H
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	

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

```
data['sex'].value_counts()
```

```
Male      21790
Female    10771
Name: sex, dtype: int64
```

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

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

```
36.85823043357163
```

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

```
float((data['native-country'] == 'Germany').sum()) / data.shape[0] * 100
```

```
0.42074874850281013
```

*4-5. What are the mean and standard deviation of age for those who earn more than 50K per year (
50K per year?**

```
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.".forma
    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 earn more than 50K have at least high school education? (*education*

```
data.loc[data['salary'] == '>50K', 'education'].unique()
```

```
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 age statistics for each race (*race* feature) and each gender (*sex* feature). Use ***groupby(***
men of ***Amer-Indian-Eskimo*** race.

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

```
Race: Amer-Indian-Eskimo, sex: Female
count    119.000000
mean      37.117647
std       13.114991
min       17.000000
25%       27.000000
50%       36.000000
75%       46.000000
max       80.000000
Name: age, dtype: float64
Race: Amer-Indian-Eskimo, sex: Male
count    192.000000
mean      37.208333
std       12.049563
min       17.000000
25%       28.000000
50%       35.000000
75%       45.000000
max       82.000000
Name: age, dtype: float64
Race: Asian-Pac-Islander, sex: Female
count    346.000000
mean      35.089595
std       12.300845
min       17.000000
25%       25.000000
50%       33.000000
75%       43.750000
max       75.000000
Name: age, dtype: float64
Race: Asian-Pac-Islander, sex: Male
count    693.000000
mean      39.073593
std       12.883944
min       18.000000
25%       29.000000
50%       37.000000
75%       46.000000
max       90.000000
Name: age, dtype: float64
Race: Black, sex: Female
count    1555.000000
mean       37.854019
std        12.637197
min        17.000000
25%        28.000000
50%        37.000000
75%        46.000000
max        90.000000
Name: age, dtype: float64
Race: Black, sex: Male
count    1569.000000
mean       37.682600
std        12.882612
min        17.000000
25%        27.000000
50%        36.000000
75%        46.000000
```

8. Among whom is the proportion of those who earn a lot (>50K) greater: married or single men (those who have a *marital-status* starting with *Married* (Married-civ-spouse, Married-spouse-abs considered bachelors.

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

```
<=50K    7552
>50K      697
Name: salary, dtype: int64
```

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

```
<=50K    7576
>50K     5965
Name: salary, dtype: int64
```

9. What is the maximum number of hours a person works per week (*hours-per-week* feature)? Ho hours, and what is the percentage of those who earn a lot (>50K) among them?

```
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*) for those who earn a little and a lot (*salary*) these be for Japan?

```
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
Cambodia <=50K 41.42
Cambodia >50K 40.0
Canada <=50K 37.91
Canada >50K 45.64
China <=50K 37.38
China >50K 38.9
Columbia <=50K 38.68
Columbia >50K 50.0
Cuba <=50K 37.99
Cuba >50K 42.44
Dominican-Republic <=50K 42.34
Dominican-Republic >50K 47.0
Ecuador <=50K 38.04
Ecuador >50K 48.75
El-Salvador <=50K 36.03
El-Salvador >50K 45.0
England <=50K 40.48
England >50K 44.53
France <=50K 41.06
France >50K 50.75
Germany <=50K 39.14
Germany >50K 44.98
Greece <=50K 41.81
Greece >50K 50.62
Guatemala <=50K 39.36
Guatemala >50K 36.67
Haiti <=50K 36.33
Haiti >50K 42.75
Holand-Netherlands <=50K 40.0
Honduras <=50K 34.33
Honduras >50K 60.0
Hong <=50K 39.14
Hong >50K 45.0
Hungary <=50K 31.3
Hungary >50K 50.0
India <=50K 38.23
India >50K 46.48
Iran <=50K 41.44
Iran >50K 47.5
Ireland <=50K 40.95
Ireland >50K 48.0
Italy <=50K 39.62
Italy >50K 45.4
Jamaica <=50K 38.24
Jamaica >50K 41.1
Japan <=50K 41.0
Japan >50K 47.96
Laos <=50K 40.38
Laos >50K 40.0
Mexico <=50K 40.0
Mexico >50K 46.58
Nicaragua <=50K 36.09
Nicaragua >50K 37.5
Outlying-US(Guam-USVI-etc) <=50K 41.86
Peru <=50K 35.07
Peru >50K 40.0

mlcourse.ai. Assignment #1 (demo)

Всего 10/10

Exploratory data analysis with Pandas

Часть 2.

- `age`: continuous.
- `workclass`: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Withou
- `fnlwgt`: continuous.
- `education`: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9t Doctorate, 5th-6th, Preschool.
- `education-num`: continuous.
- `marital-status`: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married

- `occupation` : Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, 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 South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Port Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Peru, Hong, Holand-Netherlands.
- `salary`: >50K,<=50K

```
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
```

```
df = pd.read_csv('/content/adult.data.csv', sep=',')
df.head()
```

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife

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

```
df.sex.value_counts()
```

```
Male      21790
Female    10771
Name: sex, dtype: int64
```

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

```
df[df.sex == 'Female'].age.mean()
```

```
36.85823043357163
```

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

```
df['native-country'].value_counts(normalize=True)['Germany']*100
```

```
0.42074874850281013
```

*4-5. What are the mean and standard deviation of age for those who earn more than 50K per year (
50K per year?**

```
df.salary.value_counts()
```

```
<=50K    24720
>50K      7841
Name: salary, dtype: int64
```

```
df.groupby(by='salary').agg({'age':['mean','std']})
```

	age	
	mean	std
salary		
<=50K	36.783738	14.020088
>50K	44.249841	10.519028

6. Is it true that people who earn more than 50K have at least high school education? (*education* *Assoc-voc, Masters* or *Doctorate* feature)

```
df[df.salary=='>50K'].education.value_counts()
```

```
Bachelors       2221
HS-grad         1675
Some-college    1387
Masters          959
Prof-school      423
Assoc-voc        361
Doctorate        306
Assoc-acdm       265
10th              62
11th              60
7th-8th           40
12th              33
9th               27
5th-6th           16
1st-4th            6
Name: education, dtype: int64
```

No

7. Display age statistics for each race (*race* feature) and each gender (*sex* feature). Use *groupby(* men of *Amer-Indian-Eskimo* race.

```
df.groupby(by=['race', 'sex']).age.describe()
```

race	sex	count	mean	std	min	25%	50%	75%	max
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

```
df1 = df.groupby(by=['race', 'sex']).age.describe()
df1.loc['Amer-Indian-Eskimo', 'Male']['max']
```

```
82.0
```

8. Among whom is the proportion of those who earn a lot (>50K) greater: married or single men (those who have a *marital-status* starting with *Married* (Married-civ-spouse, Married-spouse-abs

considered bachelors.

```
df[df.salary=='>50K'].groupby(by='marital-status').age.count()
```

```
marital-status
Divorced                  463
Married-AF-spouse          10
Married-civ-spouse       6692
Married-spouse-absent      34
Never-married             491
Separated                  66
Widowed                    85
Name: age, dtype: int64
```

answer = among married

9. What is the maximum number of hours a person works per week (*hours-per-week* feature)? Ho hours, and what is the percentage of those who earn a lot (>50K) among them?

```
#df.sort_values(by='hours-per-week', ascending=False)
mx = df['hours-per-week'].max()
mx
```

```
99
```

```
df[df['hours-per-week'] == mx].count()
```

```
age               85
workclass         85
fnlwgt            85
education         85
education-num     85
marital-status    85
occupation        85
relationship      85
race              85
sex               85
capital-gain      85
capital-loss      85
hours-per-week    85
native-country    85
salary            85
dtype: int64
```

```
df[df['hours-per-week'] == mx].salary.value_counts(normalize=True)
```

```
<=50K    0.705882
>50K     0.294118
Name: salary, dtype: float64
```

```

```

answer = 0.705882

10. Count the average time of work (*hours-per-week*) for those who earn a little and a lot (*salary*) these be for Japan?

```
# You code here
```

```
df.columns
```

```
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')
```

```
df_hpw = df.groupby(by=['native-country', 'salary']).agg({'hours-per-week':'mean'})
```

```
df_hpw.loc['Japan']
```

	hours-per-week
salary	
<=50K	41.000000
>50K	47.958333

```
df1 = df.iloc[0:4]
df2 = df.iloc[50:53]
```

Получим из таблицы с исходными данными топ3 людей, чей возраст ме

```
!pip install pandasql
!pip install pandas

```

```
Requirement already satisfied: pandasql in /usr/local/lib/python3.6/dist-packages (0.
Requirement already satisfied: sqlalchemy in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dis
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (1.0.
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dis
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (fr
```

```
import pandasql as ps
import pandas as pd
```

```
simple_query = '''
    SELECT
        age,
        workclass,
        fnlwgt,
        education
    FROM df
    WHERE age < 40
    ORDER BY age desc
    LIMIT 3

    '''
%time df_ps = ps.sqldf(simple_query, locals())
df_ps
```

```
CPU times: user 563 ms, sys: 41.9 ms, total: 605 ms
Wall time: 610 ms
```

	age	workclass	fnlwgt	education
0	39	State-gov	77516	Bachelors
1	39	Private	367260	HS-grad
2	39	Private	365739	Some-college

```
columns = ['age', 'workclass', 'fnlwgt', 'education']
%time df_pd = df.loc[df.age < 40, columns].sort_values(by='age', ascending=False).hea
df_pd
```

```
CPU times: user 9.93 ms, sys: 986 µs, total: 10.9 ms
Wall time: 13.3 ms
```

	age	workclass	fnlwgt	education
0	39	State-gov	77516	Bachelors
12603	39	Private	185053	HS-grad
1608	39	Private	379350	10th

```
def example2_pandasql(data):
    aggr_query = '''
        SELECT
            count(age) as count,
            avg(age)  as mean,
            min(age)   as mean
        FROM data
        GROUP BY race
        '''
    return ps.sqldf(aggr_query, locals()).set_index('age')
```

```
df.groupby(by=['race', 'sex']).age.describe()
```

race	sex	count	mean	std	min	25%	50%	75%	max
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

```
%time pd.concat([df1, df2])
```

```
CPU times: user 5.93 ms, sys: 255 µs, total: 6.19 ms
Wall time: 6.84 ms
```

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
50	25	Private	32275	Some-college	10	Married-civ-spouse	Exec-managerial	Wife
51	18	Private	226956	HS-grad	9	Never-married	Other-service	Own-child
52	47	Private	51835	Prof-school	15	Married-civ-spouse	Prof-specialty	Wife

Список литературы

[1] Гапанюк Ю. Е. Лабораторная работа «Разведочный анализ данных. Исследование и визуализация данных» [Электронный ресурс] // GitHub. – 2019. – Режим досту- па: https://github.com/ugapanyuk/ml_course/wiki/LAB_EDA_VISUALIZATION (дата обращения: 15.02.2020).

[2] Scikit-learn. Boston house-prices dataset [Electronic resource] // Scikit-learn. — 2018. — Access mode: https://scikit- learn.org/0.20/modules/generated/sklearn.datasets.load_boston.html (online;accessed: 18.02.2020).

[3] 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: 20.02.2020).

[4] Waskom M. seaborn 0.9.0 documentation [Electronic resource] // PyData. — 2018. — Access mode: https://seaborn.pydata.org/(online; accessed: 20.02.2020).

[5] pandas 0.24.1 documentation [Electronic resource] // PyData. — 2019. — Access mode: http://pandas.pydata.org/pandas-docs/stable/(online; accessed: 20.02.2020).