Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Дисциплина «Технологии машинного обучения»

Отчёт

по лабораторной работе №2

«Изучение библиотек обработки данных»

Вариант 3

Студент:

Белкина Е.В.

Группа ИУ5-61Б

Преподаватель:

Гапанюк Ю.Е.

Москва, 2020 г.

Цель лабораторной работы:

Изучение библиотеки обработки данных Pandas.

Задание:

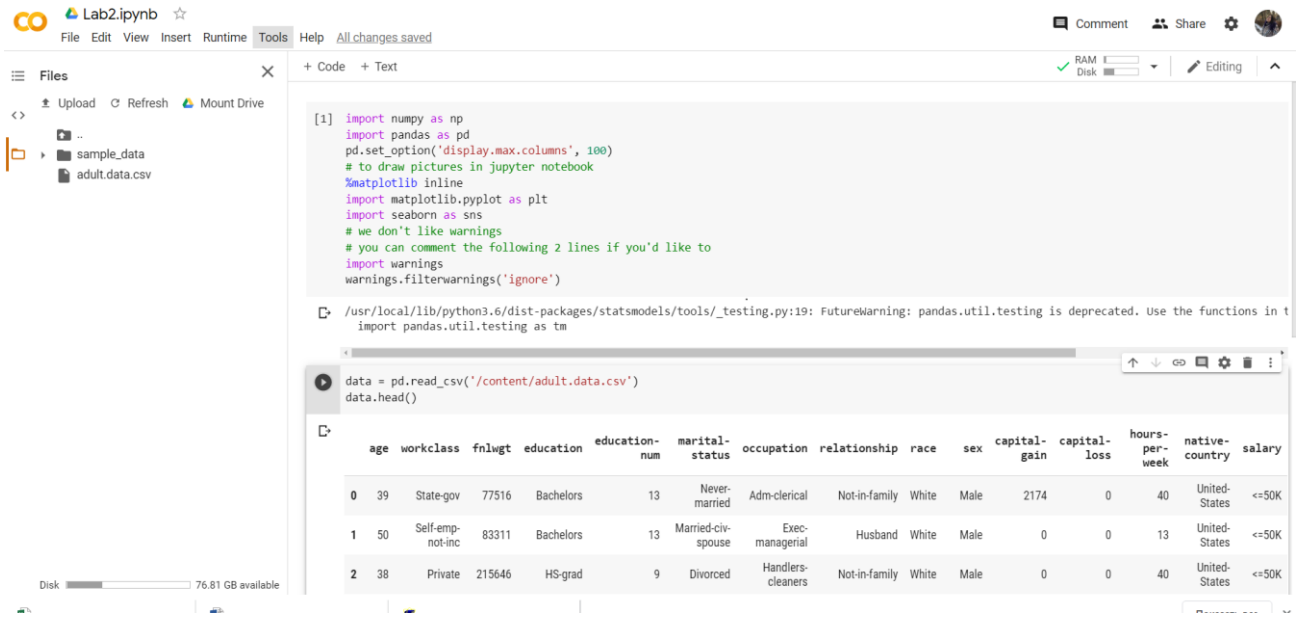
- Условие задания
  - https://nbviewer.jupyter.org/github/Yorko/mlcourse_open/blob/master/jupyter_english/assignments_demo/assignment01_pandas_uci_adult.ipynb?flush_cache=true
- Официальный датасет находится здесь, но данные и заголовки хранятся отдельно, что неудобно для анализа - https://archive.ics.uci.edu/ml/datasets/Adult
- Поэтому готовый набор данных для лабораторной работы удобнее скачать здесь - https://raw.githubusercontent.com/Yorko/mlcourse.ai/master/data/adult.data.csv (удобнее всего нажать на данной ссылке правую кнопку мыши и выбрать в контекстном меню пункт "сохранить ссылку", будет предложено сохранить файл в формате CSV)

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')


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


data['sex'].value_counts()


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


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


ages1 = data.loc[data['salary'] == '>50K', 'age']
ages2 = data.loc[data['salary'] == '<=50K', 'age']
print("Average age of those, who recieve more than 50K per year : {0} +- {
1} years, less than 50K per year : {2} +- {3} years.".format(
    round(ages1.mean()), round(ages1.std(), 1),
```

```
    round(ages2.mean()), round(ages2.std(), 1)))

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

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

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

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

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

max_load = data['hours-per-week'].max()
print("Maximum time = {0} hours./week.".format(max_load))

num_workers = data[data['hours-per-week'] == max_load].shape[0]
print("Number of workers, who work such a number of hours: {0}".format(num
_workers))

rich_share = float(data[(data['hours-per-week'] == max_load)
                 & (data['salary'] == '>50K')].shape[0]) / num_workaholics
print("Percentage of those who earn a lot (>50K) among them: {0}%".format(
int(100 * rich_share)))

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

pd.crosstab(data['native-country'], data['salary'],
           values=data['hours-per-week'], aggfunc=np.mean).T
```

Выполнение работы:




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

	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

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]
```

```
0.004207487485028101
```

4. 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.loc[data['salary'] == '>50K', 'age']
ages2 = data.loc[data['salary'] == '<=50K', 'age']
print("Average age of those, who recieve more than 50K per year : {0} +- {1} years, less than 50K per year : {2} +- {3} years.".format(
    round(ages1.mean()), round(ages1.std(), 1),
    round(ages2.mean()), round(ages2.std(), 1)))
```

```
Average age of those, who recieve more than 50K per year : 44 +- 10.5 years, less than 50K per year : 37 +- 14.0 years.
```

5. 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)

```
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)
```

Answer: No, it’s not true

6. 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.

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

```
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
max        90.000000
Name: age, dtype: float64
Race: Other, sex: Female
count    109.000000
mean      31.678899
std       11.631599
min       17.000000
25%       23.000000
50%       29.000000
75%       39.000000
max       74.000000
Name: age, dtype: float64
Race: Other, sex: Male
count    162.000000
mean      34.654321
std       11.355531
min       17.000000
25%       26.000000
50%       32.000000
75%       42.000000
max       77.000000
Name: age, dtype: float64
Race: White, sex: Female
count    8642.000000
mean       36.811618
std        14.329093
```

```
Race: White, sex: Female
count    8642.000000
mean       36.811618
std        14.329093
min        17.000000
25%        25.000000
50%        35.000000
75%        46.000000
max        90.000000
Name: age, dtype: float64
Race: White, sex: Male
count    19174.000000
mean        39.652498
std         13.436029
min         17.000000
25%         29.000000
50%         38.000000
75%         49.000000
max         90.000000
Name: age, dtype: float64
```

Answer: maximum age of men of Amer-Indian-Eskimo race = 80

7. 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'].str.startswith('Married')), 'salary'].value_counts()
```

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

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

```
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
```

8. 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?

```
max_load = data['hours-per-week'].max()
print("Maximum time = {0} hours./week.".format(max_load))

num_workers = data[data['hours-per-week'] == max_load].shape[0]
print("Number of workers, who work such a number of hours: {0}".format(num_workers))

rich_share = float(data[(data['hours-per-week'] == max_load)
                   & (data['salary'] == '>50K')].shape[0]) / num_workaholics
print("Percentage of those who earn a lot (>50K) among them: {0}%".format(int(100 * rich_share)))
```

```
Maximum time = 99 hours./week.
Number of workers, who work such a number of hours: 85
Percentage of those who earn a lot (>50K) among them: 29%
```

9. 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?

```
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
```

```
pd.crosstab(data['native-country'], data['salary'],
            values=data['hours-per-week'], aggfunc=np.mean).T
```

native-country	?	Cambodia	Canada	China	Columbia	Cuba	Dominican-Republic	Ecuador	El-Salvador	England	France	Germany	Greece	Guatema
salary														
<=50K	40.164760	41.416667	37.914634	37.381818	38.684211	37.985714	42.338235	38.041667	36.030928	40.483333	41.058824	39.139785	41.809524	39.3606
>50K	45.547945	40.000000	45.641026	38.900000	50.000000	42.440000	47.000000	48.750000	45.000000	44.533333	50.750000	44.977273	50.625000	36.6666

```
pd.crosstab(data['native-country'], data['salary'],
            values=data['hours-per-week'], aggfunc=np.mean).T
```

Guatemala	Haiti	Holand-Netherlands	Honduras	Hong	Hungary	India	Iran	Ireland	Italy	Jamaica	Japan	Laos	Mexico	Nicaragua	Outlying US(Guam USVI-etc
39.360656	36.325	40.0	34.333333	39.142857	31.3	38.233333	41.44	40.947368	39.625	38.239437	41.000000	40.375	40.003279	36.09375	41.85714
36.666667	42.750	NaN	60.000000	45.000000	50.0	46.475000	47.50	48.000000	45.400	41.100000	47.958333	40.000	46.575758	37.50000	Na

```
pd.crosstab(data['native-country'], data['salary'],
            values=data['hours-per-week'], aggfunc=np.mean).T
```

tlying-S(Guam-VI-etc)	Peru	Philippines	Poland	Portugal	Puerto-Rico	Scotland	South	Taiwan	Thailand	Trinadad&Tobago	United-States	Vietnam	Yugoslavia
41.857143	35.068966	38.065693	38.166667	41.939394	38.470588	39.444444	40.15625	33.774194	42.866667	37.058824	38.799127	37.193548	41.6
NaN	40.000000	43.032787	39.000000	41.500000	39.416667	46.666667	51.43750	46.800000	58.333333	40.000000	45.505369	39.200000	49.5