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

Отчет по ЛР №2

по курсу «Технологии машинного обучения»

«Изучение библиотек обработки данных»

ИСПОЛНИТЕЛЬ:
Аушева Л.И.
Группа ИУ5-61Б

_____

"__"_____2020 г.

ПРЕПОДАВАТЕЛЬ:
Гапанюк Ю.Е.

_____

"__"_____2020 г.

Москва 2020

Цель лабораторной работы:

Изучение библиотеки обработки данных Pandas.

Задание:

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

Выполнение:

In [1]:
```
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 [2]:
```
data = pd.read_csv('adult-data.csv')
data
```

Out[2]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
32556	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Female	0	0	38	United-States	<=50K
32557	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0	0	40	United-States	>50K
32558	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Female	0	0	40	United-States	<=50K
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	0	0	20	United-States	<=50K
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	15024	0	40	United-States	>50K

32561 rows × 15 columns

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

In [3]:
```
data['sex'].value_counts()
```

Out[3]:
```
Male      21790
Female    10771
Name: sex, dtype: int64
```

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

In [4]:
```
data.loc[data['sex']=='Female', 'age'].mean()
```

Out[4]:
```
36.85823043357163
```

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

In [5]:
```
print("Germany: ", data['native-country'].value_counts()['Germany'])
print("All: ", data['native-country'].count())
print("Germany(perc): ", round(data['native-country'].value_counts()['Germany'] / data['native-country'].count() * 100, 2), '%')
```

```
Germany:  137
All:  32561
Germany(perc):  0.42 %
```

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?

In [6]:
```
a = [round(data.loc[data['salary']=='>50K', 'age'].mean(), 0), round(data.loc[data['salary']=='>50K', 'age'].std(), 2)]
b = [round(data.loc[data['salary']=='<=50K', 'age'].mean(), 0), round(data.loc[data['salary']=='<=50K', 'age'].std(), 1)]
df = pd.DataFrame([a, b], columns=['mean', 'std'], index=['>50K', '<=50K'])
df
```

Out[6]:

	mean	std
>50K	44.0	10.52
<=50K	37.0	14.00

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)

In [7]:
```
data.loc[data['salary']=='>50K', 'education'].value_counts()
```

Out[7]:
```
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
```

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.

In [8]:
```
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
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
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
```

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.

In [9]:
```
stat = data.loc[data['salary']=='>50K', 'marital-status'].value_counts()
stat
married_count = 0
for i in stat.items():
    if i[0].startswith('Married'):
        married_count += i[1]
all_stat = data.loc[data['salary']=='>50K', 'marital-status'].count()

married = married_count / all_stat * 100
df = pd.DataFrame([married, 100-married], columns=['%'], index=['Marry', 'Not marry'])
df
```

Out[9]:

	%
Marry	85.90741
Not marry	14.09259

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?

In [10]:
```
data['hours-per-week'].describe()['max']
```

Out[10]:
```
99.0
```

In [11]:
```
many_hours = data.loc[data['hours-per-week']==99, 'workclass'].count()
many_hours
```

Out[11]: 85

In [12]:
```
salary = data.loc[data['hours-per-week']==99, 'salary']
count_big_salary = 0
for i in salary.items():
    if i[1] == '>50K':
        count_big_salary += 1
count_big_salary
small_salary = many_hours - count_big_salary
perc_big_salary = count_big_salary / many_hours * 100
perc_small_salary = small_salary / many_hours * 100
```

In [13]:
```
df = pd.DataFrame([[count_big_salary, perc_big_salary], [small_salary, perc_small_salary]],
                  columns=['count people', '%'], index=['>50K', '<=50K'])
df
```

Out[13]:

	count people	%
>50K	25	29.411765
<=50K	60	70.588235

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?

In [14]:
```
df = pd.crosstab(data['native-country'], data['salary'],
          values=data['hours-per-week'], aggfunc=np.mean).T
df
```

Out[14]:

native-country	?	Cambodia	Canada	China	Columbia	Cuba	Dominican-Republic	Ecuador	El-Salvador	England	France	Germany	Greece	Gua
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.
>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.

In [15]:
```
df['Japan']
```

Out[15]: salary

Вывод:

Изучила библиотеки обработки данных Pandas.