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:
студент группы ИУ5-63
Курганова Александра

Подпись и дата:

Проверил:

Подпись и дата:

Москва, 2019 г.

Отчет по ЛР2: "Изучение библиотек обработки данных"

Exploratory data analysis with Pandas

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, Without-pay, Never-worked.
- fnlwgt: continuous.
- education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
- education-num: continuous.
- marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
- occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-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-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.
- salary: >50K,<=50K

In [12]:
```
# Импорт библиотек
import numpy as np
import pandas as pd
pd.set_option('display.max.columns', 100)

# Картинки
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns

# Warnings
import warnings
warnings.filterwarnings('ignore')
```

In [13]:
```
data = pd.read_csv('adult.data.txt')
data.head()
```

Out[13]:

	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 [14]:
```
data.dtypes
```

Out[14]:
```
age                int64
workclass         object
fnlwgt             int64
education         object
education-num      int64
marital-status    object
occupation        object
relationship      object
race              object
sex               object
capital-gain       int64
capital-loss       int64
hours-per-week     int64
native-country    object
salary            object
dtype: object
```

1. Сколько мужчин и женщин (половая особенность) представлено в этом наборе данных?

In [15]:
```
data['sex'].value_counts()
```

Out[15]:
```
 Male      21790
 Female    10771
Name: sex, dtype: int64
```

2. Каков средний возраст (age feature) женщины?

In [27]:
```
data.loc[data['sex'] == ' Female', 'age'].mean()
```

Out[27]:
```
36.85823043357163
```

3. Каков процент граждан Германии (native-country feature)?

In [26]:
```
(float((data['native-country'] == ' Germany').sum())/data.shape[0])*100
```

Out[26]:
```
0.42074874850281013
```

4-5. Каково среднее значение и стандартное отклонение возраста тех, кто получает >50K в год (salary feature) и тех, кто получает <=50K в год?

In [43]:
```
agemore = data.loc[data['salary'] == ' >50K', 'age']
ageless = data.loc[data['salary'] == ' <=50K', 'age']

print(f"The mean age for more than 50k: {agemore.mean()} years and {agemore.std()} years, for less than 50k:
 {ageless.mean()} years and {ageless.std()} years")
```

```
The mean age for more than 50k: {agemore.mean()} years and {agemore.std()} years, for less than 50k: {ageles
s.mean()} years and {ageless.std()} years
```

6. Правда ли, что люди, которые зарабатывают >50K, имеют хотя бы среднее образование? (education – Bachelors, Prof-school, Assoc-acdm, Assoc-voc, Masters or Doctorate feature)

In [83]:
```
all_gt_50K = data.loc[data['salary'] == ' >50K'].shape[0]
gt_50K_and_education = data.loc[(data['salary'] == ' >50K') & (data['education'].isin(['Bachelors', 'Prof-sch
ool', 'Assoc-acdm', 'Assoc-voc', 'Masters', 'Doctorate']))].shape[0]
print('yes, it\'s true' if all_gt_50K == gt_50K_and_education else 'no')
```

```
no
```

7. Показать возрастную статистику для каждой расы (race feature) и каждый пол (sex feature). Используйте groupby() и describe(). Найти максимальный возраст мужчин американо-индийско-эскимосской расы.

In [31]:
```
data.groupby(['race', 'sex'])['age'].describe()
```

Out[31]:

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

In [32]:
```
data.groupby(['race', 'sex'])['age'].describe().loc[' Amer-Indian-Eskimo'].loc[' Male', 'max']
```

Out[32]: 82.0

8. Среди кого больше доля тех, кто много зарабатывает (> 50 тыс.): замужние или одинокие мужчины (marital-status feature)? Считается, что в браке находятся те, кто имеет семейное положение, начиная с женатых (женатых гражданских супругов, женатых супругов нет или женатых супругов), остальные считаются холостяками.

In [55]:
```
notmarried = data.loc[(data['sex'] == ' Male') & (data ['salary'] == ' >50K') &
        (data['marital-status'].isin([' Never-married', ' Separated', ' Divorced', ' Widowed']))].shape[0]
married = data.loc[(data['sex'] == ' Male') & (data ['salary'] == ' >50K') &
        (data['marital-status'].isin([' Married-civ-spouse', ' Married-spouse-absent' ,' Married-AF-spouse'
]))].shape[0]

print(f"notmarried: {notmarried}, married: {married}")
```

```
notmarried: 697, married: 5965
```

9. Какое максимальное количество часов человек работает в неделю (hours-per-week feature)? Сколько человек работает такое количество часов, и каков процент тех, кто зарабатывает много (>50K) Среди них?

In [69]:
```
max_value = data['hours-per-week'].max()
print(max_value)
people_with_max_hpw = data[data['hours-per-week'] == max_value]
print(people_with_max_hpw.shape[0])
print(people_with_max_hpw[data['salary'] == ' >50K'].shape[0])
```

```
99
85
25
```

10. Посчитайте среднее время работы (hours-per-week) для тех, кто зарабатывает мало и много (salary) для каждой страны (native-country). Что это будет для Японии?

In [75]:
```
(data.groupby(['native-country', 'salary'])['hours-per-week']).mean()
```

Out[75]:

```
native-country              salary
 ?                          <=50K     40.164760
                            >50K      45.547945
 Cambodia                   <=50K     41.416667
                            >50K      40.000000
 Canada                     <=50K     37.914634
                            >50K      45.641026
 China                      <=50K     37.381818
                            >50K      38.900000
 Columbia                   <=50K     38.684211
                            >50K      50.000000
 Cuba                       <=50K     37.985714
                            >50K      42.440000
 Dominican-Republic         <=50K     42.338235
                            >50K      47.000000
 Ecuador                    <=50K     38.041667
                            >50K      48.750000
 El-Salvador                <=50K     36.030928
                            >50K      45.000000
 England                    <=50K     40.483333
                            >50K      44.533333
 France                     <=50K     41.058824
                            >50K      50.750000
 Germany                    <=50K     39.139785
                            >50K      44.977273
 Greece                     <=50K     41.809524
                            >50K      50.625000
 Guatemala                  <=50K     39.360656
                            >50K      36.666667
 Haiti                      <=50K     36.325000
                            >50K      42.750000
                                         ...
 Mexico                     >50K      46.575758
 Nicaragua                  <=50K     36.093750
                            >50K      37.500000
 Outlying-US(Guam-USVI-etc) <=50K     41.857143
 Peru                       <=50K     35.068966
                            >50K      40.000000
 Philippines                <=50K     38.065693
                            >50K      43.032787
 Poland                     <=50K     38.166667
```

```
                              >50K      39.000000
 Portugal                     <=50K     41.939394
                              >50K      41.500000
 Puerto-Rico                  <=50K     38.470588
                              >50K      39.416667
 Scotland                     <=50K     39.444444
                              >50K      46.666667
 South                        <=50K     40.156250
                              >50K      51.437500
 Taiwan                       <=50K     33.774194
                              >50K      46.800000
 Thailand                     <=50K     42.866667
                              >50K      58.333333
 Trinadad&Tobago              <=50K     37.058824
                              >50K      40.000000
 United-States                <=50K     38.799127
                              >50K      45.505369
 Vietnam                      <=50K     37.193548
                              >50K      39.200000
 Yugoslavia                   <=50K     41.600000
                              >50K      49.500000
Name: hours-per-week, Length: 82, dtype: float64
```

In [77]:

```
(data.groupby(['native-country', 'salary'])['hours-per-week']).mean().loc[' Japan']
```

Out[77]:

```
salary
 <=50K    41.000000
 >50K     47.958333
Name: hours-per-week, dtype: float64
```

In [29]:
```
import numpy as np
import pandas as pd
pd.set_option('display.max.columns', 100)
import pandasql as pds
# 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 [30]:
```
user_usage = pd.read_csv("user_usage.csv")
user_device = pd.read_csv("user_device.csv")
android_devices = pd.read_csv("android_devices.csv")
```

In [31]:
```
user_usage.head()
```

Out[31]:

	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 [32]:
```
user_device.head()
```

Out[32]:

	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 [33]:
```
android_devices = android_devices.rename(index=str, columns={'Device': 'device'})
android_devices.head()
```

Out[33]:

	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

Merge using pandas

In [34]:
```
user_usage_and_user_device = pd.merge(user_usage, user_device[['use_id', 'device']], on='use_id')
user_usage_and_user_device.head()
```

Out[34]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	device
0	21.97	4.82	1557.33	22787	GT-I9505
1	1710.08	136.88	7267.55	22788	SM-G930F
2	1710.08	136.88	7267.55	22789	SM-G930F
3	94.46	35.17	519.12	22790	D2303
4	71.59	79.26	1557.33	22792	SM-G361F

In [35]:
```
user_usage_and_user_device_and_android_devices = pd.merge(user_usage_and_user_device,
                                android_devices[['Model', 'Retail Branding']],
                                left_on='device', right_on='Model').drop_duplicates()
user_usage_and_user_device_and_android_devices.head()
```

Out[35]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	device	Model	Retail Branding
0	21.97	4.82	1557.33	22787	GT-I9505	GT-I9505	Samsung
1	69.80	14.70	25955.55	22801	GT-I9505	GT-I9505	Samsung
2	249.26	253.22	1557.33	22875	GT-I9505	GT-I9505	Samsung
3	249.26	253.22	1557.33	22876	GT-I9505	GT-I9505	Samsung
4	83.46	114.06	3114.67	22880	GT-I9505	GT-I9505	Samsung

In [36]:
```
user_usage_and_user_device_and_android_devices.groupby('Retail Branding').agg({
    "outgoing_mins_per_month": "mean",
    "outgoing_sms_per_month": "mean",
    "monthly_mb": "mean",
    "use_id": "count"
})
```

Out[36]:

Retail Branding	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id
HTC	289.315789	97.678421	7080.200000	19
Huawei	81.526667	9.500000	1561.226667	3
LGE	111.530000	12.760000	1557.330000	2
Lava	60.650000	261.900000	12458.670000	2
Lenovo	215.920000	12.930000	1557.330000	1
Motorola	96.780000	68.844000	4195.424000	5
OnePlus	308.740000	51.772500	8824.890000	4
Samsung	196.975556	93.815354	3725.970707	99
Sony	143.703846	39.114615	2715.352308	13
Vodafone	42.750000	46.830000	5191.120000	1
ZTE	42.750000	46.830000	5191.120000	1

Merge using pandasql

In [37]:
```
query = """
    SELECT uu.*, ud.device FROM user_usage AS uu
    JOIN user_device AS ud ON uu.use_id = ud.use_id
"""
user_usage_and_user_device_sql = pds.sqldf(query, {'user_usage': user_usage, 'user_device': user_device})
user_usage_and_user_device_sql.head()
```

Out[37]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	device
0	21.97	4.82	1557.33	22787	GT-I9505
1	1710.08	136.88	7267.55	22788	SM-G930F
2	1710.08	136.88	7267.55	22789	SM-G930F
3	94.46	35.17	519.12	22790	D2303
4	71.59	79.26	1557.33	22792	SM-G361F

In [38]:
```
query = """
    SELECT DISTINCT ud.*, ad.`Retail Branding` FROM user_usage_and_user_device AS ud
    JOIN android_devices AS ad ON ud.device = ad.Model
"""
user_usage_and_user_device_and_android_devices_sql = pds.sqldf(
    query, {'user_usage_and_user_device': user_usage_and_user_device_sql, 'android_devices': android_devices
})
user_usage_and_user_device_and_android_devices_sql.head()
```

Out[38]:

	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id	device	Retail Branding
0	21.97	4.82	1557.33	22787	GT-I9505	Samsung
1	1710.08	136.88	7267.55	22788	SM-G930F	Samsung
2	1710.08	136.88	7267.55	22789	SM-G930F	Samsung
3	94.46	35.17	519.12	22790	D2303	Sony
4	71.59	79.26	1557.33	22792	SM-G361F	Samsung

In [39]:
```
query = """
    SELECT `Retail Branding`,
           AVG(outgoing_mins_per_month) AS outgoing_mins_per_month,
           AVG(outgoing_sms_per_month) AS outgoing_sms_per_month,
           AVG(monthly_mb) AS monthly_mb,
           COUNT(use_id) AS use_id
           FROM user_usage_and_user_device_and_android_devices
    GROUP BY `Retail Branding`
"""
pds.sqldf(
    query, {'user_usage_and_user_device_and_android_devices': user_usage_and_user_device_and_android_devices_
sql})
```

Out[39]:

	Retail Branding	outgoing_mins_per_month	outgoing_sms_per_month	monthly_mb	use_id
0	HTC	289.315789	97.678421	7080.200000	19
1	Huawei	81.526667	9.500000	1561.226667	3
2	LGE	111.530000	12.760000	1557.330000	2
3	Lava	60.650000	261.900000	12458.670000	2
4	Lenovo	215.920000	12.930000	1557.330000	1
5	Motorola	96.780000	68.844000	4195.424000	5
6	OnePlus	308.740000	51.772500	8824.890000	4
7	Samsung	196.975556	93.815354	3725.970707	99
8	Sony	143.703846	39.114615	2715.352308	13
9	Vodafone	42.750000	46.830000	5191.120000	1
10	ZTE	42.750000	46.830000	5191.120000	1