Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Машинное обучение»

Отчет по лабораторной работе №2

Выполнил: студент группы ИУ5-41И Кареникс Артёмс

Москва, 2018 г

Задание

Необходимо реализовать скрипт, выполняющий следующие действия:

1. Скачивание 1000 последних объявлений с hh.ru

2. Получить медианное значение зарплат

3. Получить распределение зарплат по диапазонам

*. Построить графики по пунктам 2 и 3

Код программы:

diagram.py

```
import func
import seaborn as sns
import matplotlib.pyplot as plt
#Диаграмма для диапазонов зарплат
def range_diagram(city_salary):
    sns.set(style="whitegrid", context="talk")
    f, ax = plt.subplots(figsize=(8, 6))
    sal_range = func.get_salary_range(city_salary.get('Москва'))
    sns.barplot(list(sal_range.keys()), list(sal_range.values()),
palette="viridis", label='Ads')
    ax.set_ylabel("Ads quantity")
    ax.set_xlabel("Salary ranges(RUR)")
    ax.set_title("ML vacancies in Moscow")
    ax.legend()
    plt.show()
#Диаграмма для медианы зарплат по городам
def median_diagram(city_median_salary):
    sns.set(style="whitegrid", context="talk")
    f, ax = plt.subplots(figsize=(8, 6))
    i = 0
    keys = list(city_median_salary.keys())
    values = list(city_median_salary.values())
    keys_range = []
    values_range = []
    while i < 20:
        keys_range.append(keys[i])
        values_range.append(values[i])
        i = i+1
    sns.barplot(values_range, keys_range, palette="Blues_d",
label='Salary')
    ax.set_ylabel("Cities")
```

```
    ax.set_xlabel("Salary (RUR)")
    ax.set_title("Median salary")
    ax.legend()
    plt.show()
```

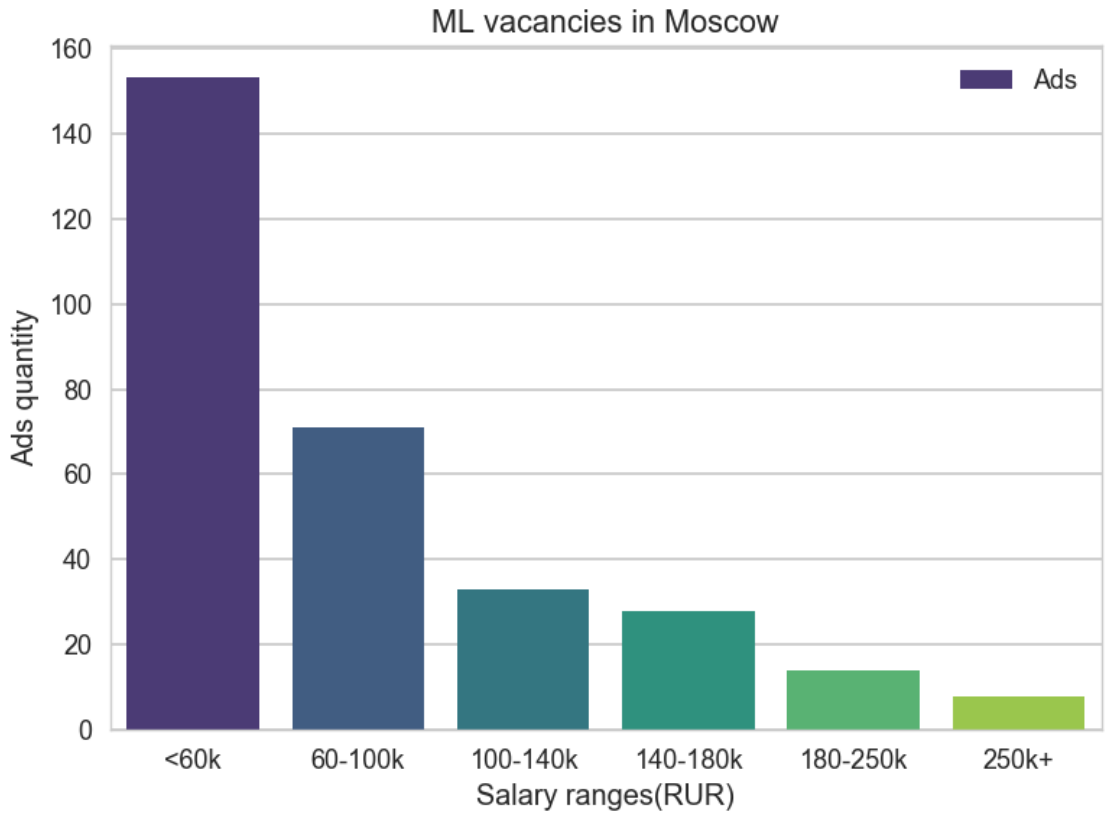
func.py

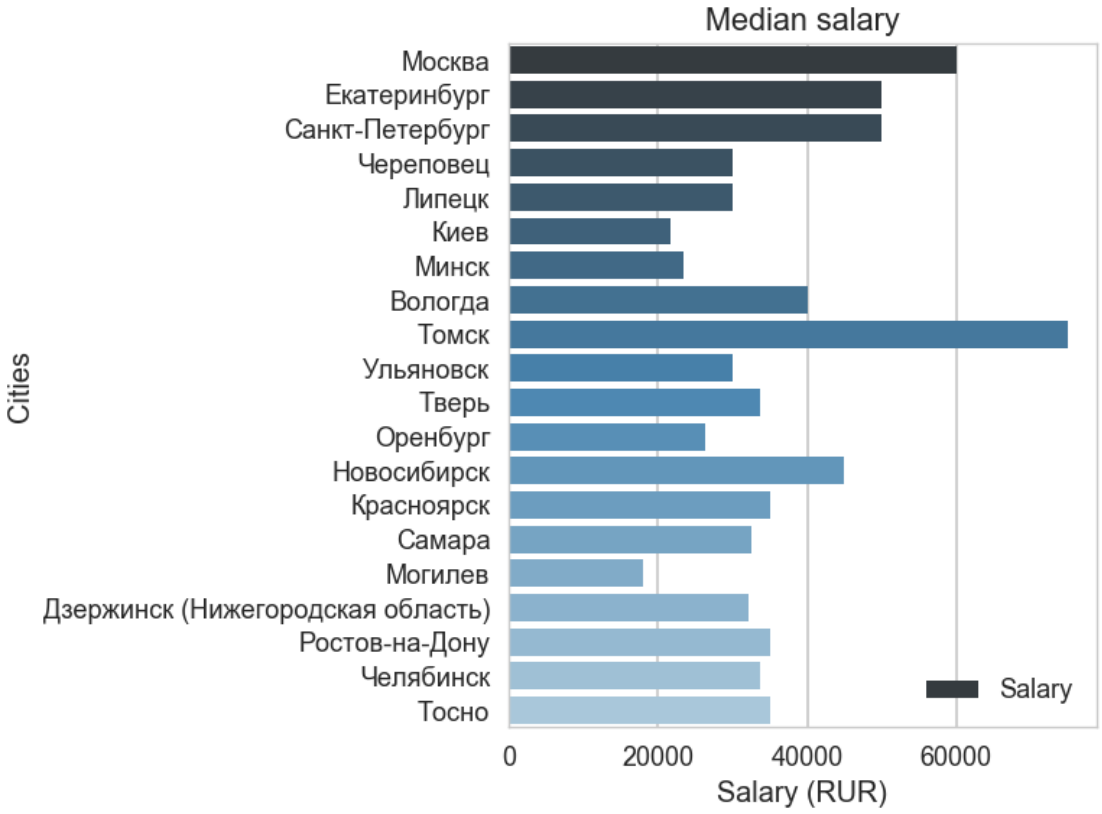
```
import requests
curr_req = requests.get('https://api.hh.ru/dictionaries')
curr_list = curr_req.json().get('currency') # Список словарей валюты
#Получение курса валюты
def get_currency_rate(currency):
    for curr in curr_list:
        if curr.get('code') == currency:
            return curr.get('rate')
#Получение зарплаты из вилки
def get_avg_salary(low, high, currency):
    if low is None:
        avg_slr = high
    elif high is None:
        avg_slr = low
    else:
        avg_slr = (low + high) / 2
    if 'RUR' in currency:
        return avg_slr*1
    else:
        return avg_slr / get_currency_rate(currency)
#Разбиение списка запрлат на диапазоны
def get_salary_range(salaries):
    count = [0, 0, 0, 0, 0, 0]
    for salary in salaries:
        if salary < 60000:
            count[0] = count[0] + 1
        elif salary < 100000:
            count[1] = count[1] + 1
        elif salary < 140000:
            count[2] = count[2] + 1
        elif salary < 180000:
            count[3] = count[3] + 1
        elif salary < 250000:
            count[4] = count[4] + 1
        else:
            count[5] = count[5] + 1
    salary_range = {'<60k': count[0], '60-100k': count[1], '100-140k':
count[2], '140-180k': count[3],
                    '180-250k': count[4],
                    '250k+': count[5]}
    return salary_range
#Получение кол-ва страниц найденых объявлений
def get_number_of_pages(request):
    return(request.json()).get('pages')
```

main.py

```
import statistics
import requests
import func
import diagram
param = {'text': 'Машинное обучение', 'vacancy_search_fields': 'name',
'per_page': '100', 'only_with_salary': 'true'}
general_request = requests.get('https://api.hh.ru/vacancies/', param)
i = 0
counter = 0
avg_salary = [] #список для хранения средней зарплаты из вилки
city_salary = {} #словарь для хранения городов и списка их зарплат
city_median_salary = {} #словарь для хранения городов и медианы зарплат
#Заполнение словаря городов и зарплат значениями
while i < func.get_number_of_pages(general_request) - 2:
    param = {'text': 'Машинное обучение', 'vacancy_search_fields': 'name',
'per_page': '100',
             'only_with_salary': 'true', 'page': i}
    req = requests.get('https://api.hh.ru/vacancies/', param)
    items = (req.json()).get('items')
    if items is not None:
        for item in items:
            salary = item.get('salary')
            city_id = (item.get('area')).get('name')
            avg_salary.append(func.get_avg_salary(salary.get('from'),
salary.get('to'), salary.get('currency')))
            #Создание списка, если в словаре еще нет такого города
            if city_salary.get(city_id) is None:
                temp = [avg_salary[counter]]
                city_salary[city_id] = temp
            #Добавление в список зарплаты конкретного города
            else:
                temp = (city_salary.get(city_id))
                temp.append(avg_salary[counter])
                city_salary[city_id] = temp
            counter = counter + 1
    else:
        break
    i = i + 1
#Вычисление медианы для каждого города
for x in city_salary.keys():
    city_median_salary[x] = statistics.median(city_salary.get(x))
#Построение диаграмм
diagram.range_diagram(city_salary)
diagram.median_diagram(city_median_salary)
```

Результаты:


ML vacancies in Moscow
Ads
Ads quantity
160
140
120
100
80
60
40
20
0
<60k
60-100k
100-140k
140-180k
180-250k
250k+
Salary ranges(RUR)


Median salary
Cities
Москва
Екатеринбург
Санкт-Петербург
Череповец
Липецк
Киев
Минск
Вологда
Томск
Ульяновск
Тверь
Оренбург
Новосибирск
Красноярск
Самара
Могилев
Дзержинск (Нижегородская область)
Ростов-на-Дону
Челябинск
Тосно
Salary
0
20000
40000
60000
Salary (RUR)