Московский государственный технический университет им. Н. Э. Баумана

***Факультет** «Информатика и системы управления»*

***Кафедра** «Системы обработки информации и управления»*

Отчёт по лабораторной работе №2

*по курсу «**Введение в машинное обучение**»*

Исполнила: Алиева Д.Г., ИУ5-41
Проверил: Гапанюк Ю.Е.

Москва 2018 г.

Задание:

Необходимо реализовать скрипт, выполняющий следующие действия:

1. Скачивание 1000 последних объявлений с hh.ru

Для выполнения этого пункта вам понадобится библиотека requests. С помощью этой библиотеки можно делать HTTP-запросы в API hh.ru. Что такое api. С API, предоставляемым hh.ru, можно ознакомиться здесь. Общая информация здесь Вам понадобится этот метод. Чтобы тестировать запросы в api и смотреть, что они возвращают, можно использовать Postman. В качестве поискового запроса можно вводить ключевые слова, связанные с тематикой анализа данных: machine learning, data science, машинное обучение, big data, data analytics и тд. В ответе API будут интересующие нас поля: salary, area, name, employer

2. Получить медианное значение зарплат

Необходимо сделать обработку полученных на первом шаге данных и получить следующую структуру: Словарь, где ключом является название вакансии (как оно задано на hh.ru), а значением - медианное значение зарплаты по этой вакансии. То есть необходимо сгруппировать данные по имени вакансии. Также можно использовать другие варианты, например, сгруппировать по городу или любому другому интересному параметру из выдачи. В поле salary hh.ru отдает значения диапазона. Значением зарплаты считать среднее значение из диапазона, например, если зп от 100 до 150, то фиксировать значение 125.

3. Получить распределение зарплат по диапазонам

Необходимо выделить диапазоны зарплат, например:
до 80к, 80-120к, 120-150к, 150-200к, 200-300к, 300к+ Для каждого диапазона подсчитать количество предлагаемых вакансий.

***. Построить графики по пунктам 2 и 3.**

Скрипт:

```
# In[1]:
import requests
import matplotlib.pyplot as plt
arr1 = []
for i in range(10):
   #print(i)
   str0 =
'https://api.hh.ru/vacancies/?per_page=100&page='+str(i)+'&text=machine+learning+OR+big+data+OR+da
ta+science+OR+data+analytics'
   #print(str0)
   req0 = requests.get(str0)
   if req0.status_code != requests.codes.ok:
      print("Error: server return status code: " + str(req.status_code))
   arr1 += (req0.json()['items'])
print(arr1)
#len(arr2)


# In[2]:
vac_sal = {}
for i in arr1:
   if ((i['salary'] != None) and (i['salary']['currency'] == 'RUR')):
      if i['salary']['to'] == None:
         vac_sal[i['name']] = (i['salary']['from'])
      elif i['salary']['from'] == None:
         vac_sal[i['name']] = (i['salary']['to']/2)
      elif ((i['salary']['from'] != None) and (i['salary']['to'] != None)):
         vac_sal[i['name']] = ((i['salary']['to'] + i['salary']['from']) / 2)
   elif ((i['salary'] != None) and (i['salary']['currency'] == 'USD')):
      if i['salary']['to'] == None:
         vac_sal[i['name']] = (i['salary']['from'] * 57)
      elif i['salary']['from'] == None:
         vac_sal[i['name']] = ((i['salary']['to'] / 2) * 57)
      elif ((i['salary']['from'] != None) and (i['salary']['to'] != None)):
         vac_sal[i['name']] = (i['salary']['to'] + i['salary']['from'] / 2) * 57
   elif   ((i['salary'] != None) and (i['salary']['currency'] == 'EUR')):
      if i['salary']['to'] == None:
         vac_sal[i['name']] = (i['salary']['from'] * 71)
      elif i['salary']['from'] == None:
         vac_sal[i['name']] = ((i['salary']['to'] / 2) * 71)
      elif ((i['salary']['from'] != None) and (i['salary']['to'] != None)):
         vac_sal[i['name']] = (i['salary']['to'] + i['salary']['from'] / 2) * 71
vac_sal


# In[3]:
```

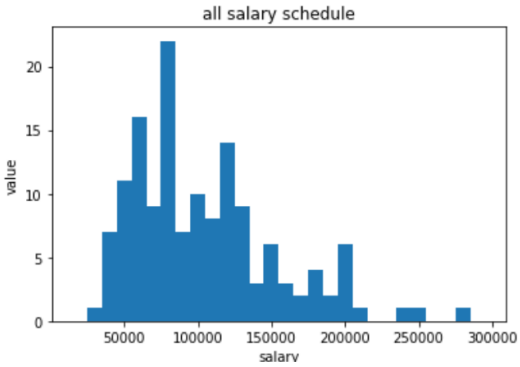
```
data_science = []
for i  in vac_sal:
   if (('ata' in i) and ('cien' in i)):
      data_science.append(vac_sal[i])
data_science.sort()
print(data_science)
med_ds = (data_science[len(data_science)//2])
print('медиана=', med_ds)


# In[4]:
machine_learning = []
for i  in vac_sal:
   if (('achine' in i) or ('earning' in i)):
      machine_learning.append(vac_sal[i])
machine_learning.sort()
print(machine_learning)
med_ml = (machine_learning[len(machine_learning)//2])
print('медиана=', med_ml)


# In[5]:
programmer = list()
for i  in vac_sal:
   if ('рограммист' in i):
      programmer.append(vac_sal[i])
programmer.sort()
print(programmer)
med_prg = (programmer[len(programmer)//2])
print('медиана=', med_prg)


# In[6]:
analyst = list()
for i  in vac_sal:
    if (('нали' in i) or ('naly' in i)) :
      analyst.append(vac_sal[i])
analyst.sort()
print(analyst)
med_anl = (analyst[len(analyst)//2])
print('медиана=', med_anl)


# In[7]:
developer = list()
for i  in vac_sal:
    if (('азработ' in i)or ('evelop' in i)) :
      developer.append(vac_sal[i])
```

```
developer.sort()
print(developer)
med_dvp = (developer[len(developer)//2])
print('медиана=', med_dvp)

# In[8]:
names = ['Data science', 'Developer', 'Analyst', 'Programmer', 'Machine learning']
x = [0, 1, 2, 3, 4]
med = [med_ds, med_dvp, med_anl, med_prg, med_ml]
plt.bar(x, med)
plt.xticks(x, names, rotation = 30)
plt.show()

# In[9]:
a = 0
b = 0
c = 0
d = 0
e = 0
f = 0
print(len(vac_sal))
for i  in vac_sal:
   if (vac_sal[i] < 80000):
      a += 1
   elif ((vac_sal[i] >= 80000) and (vac_sal[i] < 120000)):
      b += 1
   elif ((vac_sal[i] >= 120000) and (vac_sal[i] < 150000)):
      c += 1
   elif ((vac_sal[i] >= 150000) and (vac_sal[i] < 200000)):
      d += 1
   elif ((vac_sal[i] >= 200000) and (vac_sal[i] < 300000)):
      e += 1
   elif (vac_sal[i] >= 300000):
      f += 1
print(a , b, c, d, e, f)

# In[10]:



names2 = ['<80k', '80k-120k', '120k-150k', '150k-200k', '200k-300k', '>300k']
x2 = [0, 1, 2, 3, 4, 5]
count = [a, b, c, d, e, f]
plt.bar(x2, count)
plt.xticks(x2, names2, rotation = 30)
plt.show()
```

Результаты:


```
{'Data Scientist (ML/AI/Big Data/Python)': 100000, 'Big Data Developer': 100000.0, 'Руководитель разработки / Архитектор (CTO, Machine Learning)': 150000.0, 'Программист Python (big data, machine learning)': 60000.0, 'Руководитель отдела Data Science / Machine learning': 300000, 'AI Machine Learning Engineer': 130000, 'Data Scientist (Аналитик)': 170000.0, 'Senior Data Analyst (Marketing)': 2750.0, 'Junior Data Analyst': 120000.0, 'Product manager Big Data': 75000.0, 'Специалист по внедрению аналитических решений (прогностическая аналитика, big data)': 105000.0, 'Бизнес-аналитик (Big Data решения)': 115.0, 'Руководитель проектов (Big Data решения)': 240.0, 'Руководитель проектов Data Science': 130000, 'Разработчик систем машинного обучения (математик, Data scientist)': 50000.0, 'Senior Data Scientist (Machine Learning) / Senior Python Developer': 200000, 'Специалист по анализу данных / Data Scientist': 100000.0, 'Инженер BIG DATA': 75000.0, 'Программист-стажер Java / machine learning': 45000.0, 'Big data developer - Java/Scala + Python': 2000.0, 'Разработчик (machine learning)': 125000.0, 'Data Scientist (Tech Lead)': 3000, 'Руководитель отдела Data Science': 300000, 'Scala/Java (big data)': 150000, 'Data scientist developer / Senior Python Developer': 2600.0, 'Data Engineer': 195000.0, 'Аналитик/ Data scientist': 75000.0, 'Data Scientist (BigData Аналитик)': 175000.0, 'Аналитик-математик (Data scientist)': 120000, 'Аналитик-математик (Junior data scientist)': 90000.0, 'Python Developer (Data Scientist)': 150000, 'Программист-разработчик нейронных сетей (Deep Learning Engineer)': 125000, 'Графический дизайнер в отдел маркетинга': 50000, 'UI/UX дизайнер (веб\\мобильное приложение)': 100000, 'JS (Angular) разработчик': 90000.0, '.net developer': 1100.0, 'Javascript Developer (react + redux/vue.js)': 120000, 'Инженер по тестированию ПО': 75000.0, 'Junior Full Stack Developer': 550.0, 'Руководитель группы машинного обучения': 80000, 'Senior Data Scientist': 150000, 'Фулстек разработчик PHP / JS': 65000.0, 'Fullstack разработчик игр': 82500.0, 'Программист-математик/ алгоритмист (США)': 3500.0, 'Full Stack Java Developer': 150000.0, 'Senior Software Engineer (C/C++, Python)': 5900.0, 'Data Scientist / Специалист по машинному обучению': 100000, 'PHP Developer': 60000.0, 'Product Manager': 200000, 'Senior Backend Engineer': 2500.0, 'Программист Scala / Java ( Scala / Java Developer )': 57500.0, 'Ведущий программист Scala / Java ( Senior Scala / Java Developer )': 75000.0, 'Руководитель проекта BigData': 120000, 'Principal Full Stack Developer / Technical Product Manager': 200000, 'Technical Product Manager': 200000, 'Junior-Middle Data Scientist (специалист по анализу данных)': 115000.0, 'Старший Java программист': 1850.0, 'Senior Scala Developer': 1850.0, 'Java Backend Developer (Data Science Platform) ,Tokyo, Japan': 3975.0, 'Системный аналитик': 80000, 'Разработчик алгоритмов анализа данных': 90000, 'Младший специалист технической поддержки': 35000, 'DevOps Engineer': 150000, 'HTML-верстальщик': 70000, 'Junior Software Development Engineer (Инженер-программист на неполный рабочий день)': 35000.0, 'Mobile App Developer': 55000.0, 'Senior data scientist / специалист по машинному обучению': 160000, 'Account manager/ Менеджер по работе с клиентами': 75000.0, 'Senior QA automation engineer': 1750.0, 'Программист C++': 3800, 'Дизайнер': 40000.0, 'Специалист по машинному обучению / Data Scientist': 105000.0, 'Инженер-программист C++ (алгоритмы компьютерного зрения)': 110000.0, 'Digital- аналитик': 75000.0, 'Ведущий менеджер по продажам IT решений': 130000, 'Старший программист C++ (в проект по компьютернему зрению)': 120000, 'Программист зрения для робота': 175000.0, 'Senior Legal Counsel': 5500.0, 'Senior Software Engineer (Sensors)': 2700, 'Senior
```


```
[ 0  1  7 11 16  9 22  7 10  8 14  9  3  6  3  2  4  2  6  1  0  0  1  1
  0  0  1  0] [ 15000  25000  35000  45000  55000  65000  75000  85000  95000 105000
 115000 125000 135000 145000 155000 165000 175000 185000 195000 205000
 215000 225000 235000 245000 255000 265000 275000 285000 295000]
```




```
115000.0     70000     80000.0     75000.0
[3 1 0] [ 50000 100000 150000 200000]
```