Московский государственный технический университет им. Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Отчет по

лабораторной работе № 2 по курсу

«Введение в машинное обучение»

Исполнитель: ИУ5-41, Черепанов Е.

Преподаватель: Гапанюк Ю.

Москва, 2018г.

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 и результаты.

In [1]:
```
# 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+data+science+OR+data+a
    #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)
```


```
[{'salary': None, 'snippet': {'requirement': '...<highlighttext>big</highlighttext> network. You are experienced in solving p
roblems using <highlighttext>machine</highlighttext> <highlighttext>learning</highlighttext> techniques (clustering, classifi
cation, outlier analysis, etc) and deep <highlighttext>learning</highlighttext>...', 'responsibility': '...<highlighttext>dat
a</highlighttext> models, to identify inconsistencies in <highlighttext>data</highlighttext>. You will come up with scalable
algorithms (for <highlighttext>data</highlighttext> processing and <highlighttext>machine</highlighttext> <highlighttext>lear
ning</highlighttext>...'}, 'archived': False, 'premium': False, 'name': 'Senior Data Science Engineer', 'area': {'url': 'http
s://api.hh.ru/areas/1', 'id': '1', 'name': 'Москва'}, 'url': 'https://api.hh.ru/vacancies/25604231?host=hh.ru', 'created_at':
'2018-06-03T23:38:35+0300', 'alternate_url': 'https://hh.ru/vacancy/25604231', 'apply_alternate_url': 'https://hh.ru/applican
t/vacancy_response?vacancyId=25604231', 'relations': [], 'employer': {'logo_urls': {'90': 'https://hhcdn.ru/employer-logo/310
471.jpeg', '240': 'https://hhcdn.ru/employer-logo/383328.jpeg', 'original': 'https://hhcdn.ru/employer-logo-original/231202.j
pg'}, 'vacancies_url': 'https://api.hh.ru/vacancies?employer_id=5135', 'name': 'NVIDIA', 'url': 'https://api.hh.ru/employers/
5135', 'alternate_url': 'https://hh.ru/employer/5135', 'id': '5135', 'trusted': True}, 'response_letter_required': False, 'pu
blished_at': '2018-06-03T23:38:35+0300', 'address': {'building': '12', 'city': 'Москва', 'description': None, 'metro': {'line
_name': 'Люблинско-Дмитровская', 'station_id': '10.185', 'line_id': '10', 'lat': 55.793723, 'station_name': 'Марьина Роща',
'lng': 37.61618}, 'metro_stations': [{'line_name': 'Люблинско-Дмитровская', 'station_id': '10.185', 'line_id': '10', 'lat': 5
5.793723, 'station_name': 'Марьина Роща', 'lng': 37.61618}, {'line_name': 'Серпуховско-Тимирязевская', 'station_id': '9.128',
'line_id': '9', 'lat': 55.794054, 'station_name': 'Савеловская', 'lng': 37.587163}], 'raw': 'Двинцев, 12', 'street': 'Двинце
в', 'lat': 55.796273, 'lng': 37.599687, 'id': '99751'}, 'department': None, 'sort_point_distance': None, 'type': {'id': 'ope
n', 'name': 'Открытая'}, 'id': '25604231'}, {'salary': None, 'snippet': {'requirement': '...компьютерными <highlighttext>наук
```


In [2]:
```
# 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]:
```
# 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)
```

```
[35000.0, 50000.0, 60000, 75000.0, 85000.0, 90000.0, 100000.0, 100000.0, 100000, 120000, 128250.0, 130000, 142000.0, 150000, 16
0000, 160000.0, 170000, 200000.0, 200000, 200000, 200000, 228000.0, 276900.0, 285000.0, 342000.0, 712500.0]
медиана= 150000
```

In [4]:
```
# 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)
```

```
[35000.0, 85000.0, 110000, 130000, 145000.0, 200000.0, 213750.0, 250000]
медиана= 145000.0
```

In [5]:
```
# 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)
```

```
[25000, 35000.0, 65000.0, 130000, 150000.0, 175000.0, 199500.0, 200000.0]
медиана= 150000.0
```

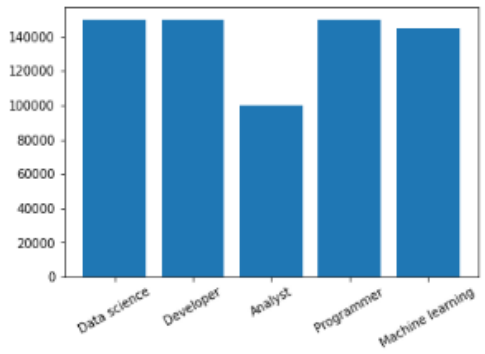
In [6]:
```
# In[6]:
analyst = list()
for i  in vac_sal:
     if (('нали' in i) or ('naly' in i)) :
         analyst.append(vac_sal[i])
analyst.sort()
print(analyst)
med_anl = (analyst[len(analyst)//2])
print('медиана=', med_anl)
```

```
[20000, 32000, 40000.0, 60000, 60000, 60000.0, 70000, 70000, 75000.0, 80000, 80000, 90000.0, 100000.0, 100000.0, 100000.0, 1100
00.0, 115000.0, 120000, 120000.0, 120000, 130000, 142000.0, 228000.0, 256500.0, 299250.0]
медиана= 100000.0
```

In [7]:
```
# 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)
```

```
[50000.0, 50000, 60000.0, 71250.0, 80000, 85000.0, 92500.0, 100000, 106500.0, 120000.0, 120000, 120000, 125000.0, 130000.0, 140
000, 142000.0, 150000, 150000, 150000.0, 171000, 171000, 171000, 173850.0, 180000.0, 200000.0, 205000.0, 210000, 240000.0, 2500
00, 260000, 568000.0, 712500.0]
медиана= 150000
```

In [8]:
```
# 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]:
```
# 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)
```

```
140
38 26 18 22 25 11
```

In [10]:
```
# 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()
```