**Московский Государственный Технический Университет имени Н. Э. Баумана**



**Машинное обучение**

**Лабораторная работа №2**

**«Python »**

ИСПОЛНИТЕЛЬ:

ФИО Гладова Анастасия

ИУ5-41 (ИУ5Ц- 61Б)

ПРЕПОДАВАТЕЛЬ:

ФИО Гапанюк Ю.Е.

**Москва 2018**

**Задание и порядок выполнения:**

Необходимо реализовать скрипт, выполняющий следующие действия:

**1. Скачивание 1000 последних объявлений с hh.ru**

Для выполнения этого пункта вам понадобится библиотека [requests](http://docs.python-requests.org/en/master/). С помощью этой библиотеки можно делать HTTP-запросы в API hh.ru. [Что такое api](https://ru.wikipedia.org/wiki/API). С API, предоставляемым hh.ru, можно ознакомиться [здесь](https://dev.hh.ru/?utm_source=hh_ru&utm_medium=main&utm_campaign=niz). Общая информация [здесь](https://github.com/hhru/api/blob/master/docs/general.md)

Вам понадобится [этот метод](https://github.com/hhru/api/blob/master/docs/vacancies.md#search). Чтобы тестировать запросы в api и смотреть, что они возвращают, можно использовать [Postman](https://www.getpostman.com/). 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**

Построить гистограммы распределений с легендой и подписями осей с использованием библиотек [matplotlib](https://matplotlib.org/) или [seaborn](https://seaborn.pydata.org/).

**Примечание:**

Для выполнения заданий необходимо использовать функции [map, list, reduce, filter] (<http://book.pythontips.com/en/latest/map_filter.html>). Чтобы не писать все в одном файле можете установить [jupyter](http://jupyter.org/).

**Листинг:**

# coding: utf-8

# In[1]:

#библиотека requests, с помощью которой можно делать HTTP-запросы в API hh.ru.

#API - набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом)

#или операционной системой для использования во внешних программных продуктах.

#Используется программистами при написании всевозможных приложений.

import requests

# matplotlib — это библиотека Python для построения качественных двухмерных графиков

import matplotlib.pyplot as plt

# In[2]:

#скачивание 1000 последних объявлений с hh.ru

arr1 = []

#range - универсальная функция для создания списков (list), содержащая арифметическую прогрессию

#принимает от 1 до 3 аргументов

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+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[3]:

#получение значений зарплат + перевод валюты

#в случае четного числа зарплат берется полусумма двух средних

#в случае нечетного нужно отсортировать и взять зарплату со средним номером

vac\_sal = {} #словарь - структура данных, отображающая одни объекты (ключи) в другие (значения)

for i in arr1:

if ((i['salary'] != None) and (i['salary']['currency'] == 'RUR')): #валюта - рубли

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')): #валюта - доллары

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')): #валюта - евро

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[4]:

#сортировка и поиск медианы для data\_science

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[5]:

#сортировка и поиск медианы для machine\_learning

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[6]:

#сортировка и поиск медианы для programmer

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[7]:

#сортировка и поиск медианы для analyst

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[8]:

#сортировка и поиск медианы для developer

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[9]:

#для графика - присваивание номеров

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[10]:

#распределение зарплат по диапазонам

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[11]:

#присваивание номеров зарплатам

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)

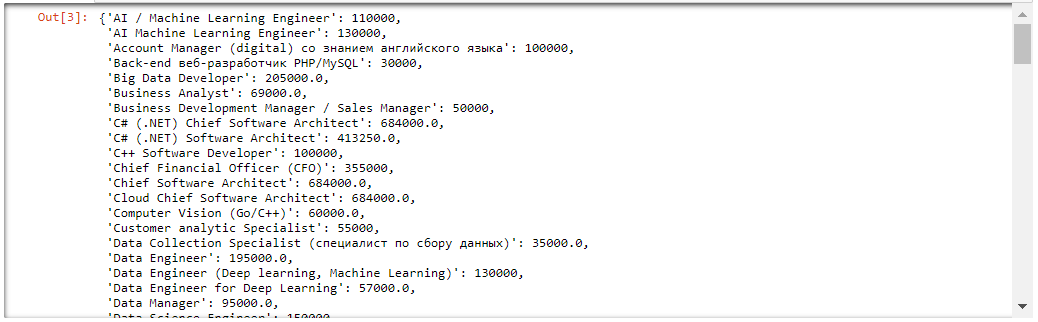
plt.show()

**Результаты выполнения:**

# In[2]:



# In[3]:



# In[4]:



# In[5]:



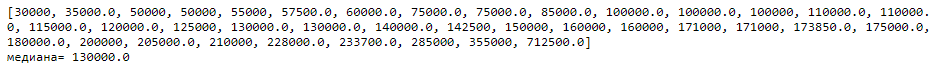
# In[6]:



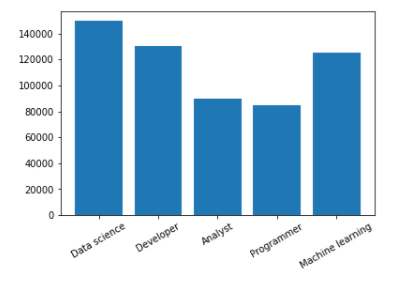
# In[7]:



# In[8]:



# In[9]:



# In[10]:



# In[11]:

