СПбГЭТУ «ЛЭТИ»
ПЕРВЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ

Н.А. Жукова

Машинное обучение

Методические указания к выполнению практических заданий

СПбГЭТУ «ЛЭТИ», 2022 г.

1 ОБЩИЕ ТРЕБОВАНИЯ К ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ ЗАДАНИЙ

Студенты выполняют практические задания самостоятельно. На каждое задание отводится один академический час. В качестве решения, студент отправляет программный код на языке Python преподавателю посредством электронной почты или иным способом, определенным преподавателем.

Работы должны быть выполнены и представлены преподавателю для проверки в сроки до начала зачетной недели. При этом при оценивании используются следующие правила:

- Работа отправлена в срок и полностью выполнена – 2,5 балла.
- Работа отправлена в срок, выполнена частично – 1,5 – 2,5 балла (на усмотрение преподавателя).
- Работа не отправлена или отправлена за пределами установленного срока – 0 баллов.

2 ПРАКТИЧЕСКИЕ ЗАДАНИЯ

2.1 Задание №1. Основы Python

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_1.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Задача №1

Написать функцию на вход которой подается строка, состоящая из латинских букв. Функция должна вернуть количество гласных букв (a, e, i, o, u) в этой строке.

Задача №2

Написать функцию на вход, которой подается строка. Функция должна вернуть true, если каждый символ в строке встречается только 1 раз, иначе должна вернуть false.

Задача №3

Написать функцию, которая принимает положительное число и возвращает количество бит равных 1 в этом числе.

Задача №4

Написать функцию, которая принимает положительное. Функция должна вернуть то, сколько раз необходимо перемножать цифры числа или результат перемножения, чтобы получилось число состоящее из одной цифры.

Например, для входного числа:

39 функция должна вернуть 3, так как 3*9=27 => 2*7=14 => 1*4=4

4 функция должна вернуть 0, так как число уже состоит из одной цифры

999 функция должна вернуть 4, так как 9*9*9=729 => 7*2*9=126 => 1*2*6=12 => 1*2=2

Задача №5

Написать функция, которая принимает два целочисленных вектора одинаковой длины и возвращает среднеквадратическое отклонение двух векторов.

Задача №6

Написать функцию, которая принимает список чисел и возвращает кортеж из двух элементов. Первый элемент кортежа – мат. ожидание, второй элемент – СКО. Запрещается использовать функции для расчета соответствующих характеристик.

Задача №7

Написать функцию, принимающая целое положительное число. Функция должна вернуть строку вида “(n1**p1)(n2**p2)…(nk**pk)” представляющая разложение числа на простые множители (если pi == 1, то выводить только ni).

Например, для числа 86240 функция должна вернуть “(2**5)(5)(7**2)(11)”

Задача №8

Написать функцию, принимающая 2 строки вида “xxx.xxx.xxx.xxx” представляющие ip-адрес и маску сети. Функция должна вернуть 2 строки: адрес сети и широковещательный адрес.

Задача №9

Написать функцию, принимающая целое число n, задающее количество кубиков. Функция должна определить, можно ли из данного кол-ва кубиков построить пирамиду, то есть можно ли представить число n как

1^2+2^2+3^2+…+k^2. Если можно, то функция должна вернуть k, иначе строку “It is impossible”.

Задача №10

Написать функцию, которая принимает положительное целое число n и определяющая является ли число n сбалансированным. Число является сбалансированным, если сумма цифр до средних цифр равна сумме цифр после средней цифры. Если число нечетное, то средняя цифра одна, если четное, то средних цифр две. При расчете, средние числа не участвуют.

Например:

- Число 23441 сбалансированное, так как 2+3=4+1
- Число 7 сбалансированное, так как 0=0
- Число 1231 сбалансированное, так как 1=1
- Число 123456 несбалансированное, так как 1+2!=5+6

Задача №11

Написать функцию, которая принимает двумерный массив M в первой колонке которой стоит буква латинского алфавита (данная буква обозначает принадлежность к классу) и вещественное число r, которое удовлетворяет условию 0<r<1. Функция должна разбить данный массив на 2 так, что количество строк в новых массивах пропорционально r и (r-1). Также, в каждом новом массиве, количество строк одного класса должно быть пропорционально количеству строк этого класса в исходном массиве.

Например, для входных данных (r = 0.5) :

a	1	2
a	3	4
b	5	6
b	7	8
c	9	10
c	11	12

Должны получиться две выборки:

a	1	2

b	5	6
c	9	10

И

a	3	4
b	7	8
c	11	12

2.2 Задание №2. Создание простой нейронной сети с использованием библиотеки Keras

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_2.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Необходимо дополнить следующий фрагмент кода моделью ИНС, которая способна провести бинарную классификацию по сгенерированным данным:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mclr
from tensorflow.keras import layers
from tensorflow.keras import models
def genData(size=500):
    #Функцию выбрать в зависимости от варианта
def drawResults(data, label, prediction):
    p_label = np.array([round(x[0]) for x in prediction])
    plt.scatter(data[:, 0],
data[:, 1], s=30, c=label[:, 0], cmap=mclr.ListedColormap(['red', 'blue']))
    plt.scatter(data[:, 0],
data[:, 1], s=10, c=p_label, cmap=mclr.ListedColormap(['red', 'blue']))
    plt.grid()
    plt.show()
(train_data, train_label), (test_data, test_label) = genData()
#В данном месте необходимо создать модель и обучить ее
#Получение ошибки и точности в процессе обучения
loss = H.history['loss']
val_loss = H.history['val_loss']
acc = H.history['accuracy']
val_acc = H.history['val_accuracy']
epochs = range(1, len(loss) + 1)
#Построение графика ошибки
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
```

```
plt.ylabel('Loss')
plt.legend()
plt.show()
#Построение графика точности
plt.clf()
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
#Получение и вывод результатов на тестовом наборе
results = model.evaluate(test_data, test_label)
print(results)
#Вывод результатов бинарной классификации
all_data = np.vstack((train_data, test_data))
all_label = np.vstack((train_label, test_label))
pred = model.predict(all_data)
drawResults(all_data, all_label, pred)
```

Вариант 1

```
def genData(size=500):
    data = np.random.rand(size, 2)*2 - 1
    label = np.zeros([size, 1])
    for i, p in enumerate(data):
        if (p[0] + .5 >= p[1]) and (p[0] - 0.5 <= p[1]):
            label[i] = 1.
        else:
            label[i] = 0.
    div = round(size*0.8)
    train_data = data[:div, :]
    test_data = data[div:, :]
    train_label = label[:div, :]
    test_label = label[div:, :]
    return (train_data, train_label), (test_data, test_label)
```

Вариант 2

```
def genData(size=500):
    data = np.random.rand(size, 2)*2 - 1
    label = np.zeros([size, 1])
    for i, p in enumerate(data):
        if p[0]*p[1] >= 0:
            label[i] = 1.
        else:
            label[i] = 0.
    div = round(size*0.8)
    train_data = data[:div, :]
    test_data = data[div:, :]
    train_label = label[:div, :]
    test_label = label[div:, :]
    return (train_data, train_label), (test_data, test_label)
```

Вариант 3

```
def genData(size=500):
    data = np.random.rand(size, 2)*2 - 1
    label = np.zeros([size, 1])
    for i, p in enumerate(data):
```

```
        if (p[0]+0.2)**2 + (0.6*p[1])**2 >= 0.25:
            label[i] = 0.
        else:
            label[i] = 1.
    div = round(size*0.8)
    train_data = data[:div, :]
    test_data = data[div:, :]
    train_label = label[:div, :]
    test_label = label[div:, :]
    return (train_data, train_label), (test_data, test_label)
```

Вариант 4

```
def genData(size=500):
    size1 = size//2
    size2 = size - size1
    t1 = np.random.rand(size1)
    x1 = np.asarray([i * math.cos(i*2*math.pi) + (np.random.rand(1)-
1)/2*i for i in t1])
    y1 = np.asarray([i * math.sin(i*2*math.pi) + (np.random.rand(1)-
1)/2*i for i in t1])
    data1 = np.hstack((x1, y1))
    label1 = np.zeros([size1, 1])
    div1 = round(size1*0.8)
    t2 = np.random.rand(size2)
    x2 = np.asarray([-
i * math.cos(i*2*math.pi) + (np.random.rand(1)-1)/2*i for i in t2])
    y2 = np.asarray([-
i * math.sin(i*2*math.pi) + (np.random.rand(1)-1)/2*i for i in t2])
    data2 = np.hstack((x2, y2))
    label2 = np.ones([size2, 1])
    div2 = round(size2*0.8)
    div = div1 + div2
    order = np.random.permutation(div)
    train_data = np.vstack((data1[:div1], data2[:div2]))
    test_data = np.vstack((data1[div1:], data2[div2:]))
    train_label = np.vstack((label1[:div1], label2[:div2]))
    test_label = np.vstack((label1[div1:], label2[div2:]))
    return (train_data[order, :], train_label[order, :]),
(test_data, test_label)
```

Вариант 5

```
def genData(size=500):
    size1 = size//2
    size2 = size - size1
    x1 = np.random.rand(size1, 1)*1.3 - 0.95
    y1 = np.asarray([3.5*(i+0.2)**2 - 0.8 + (np.random.rand(1)-
0.5)/3 for i in x1])
    data1 = np.hstack((x1, y1))
    label1 = np.zeros([size1, 1])
    div1 = round(size1*0.8)
    x2 = np.random.rand(size2, 1)*1.3 - 0.35
    y2 = np.asarray([-3.5*(i-0.2)**2 + 0.8 + (np.random.rand(1)-
0.5)/3 for i in x2])
    data2 = np.hstack((x2, y2))
    label2 = np.ones([size2, 1])
```

```
    div2 = round(size2*0.8)
    div = div1 + div2
    order = np.random.permutation(div)
    train_data = np.vstack((data1[:div1], data2[:div2]))
    test_data = np.vstack((data1[div1:], data2[div2:]))
    train_label = np.vstack((label1[:div1], label2[:div2]))
    test_label = np.vstack((label1[div1:], label2[div2:]))
    return (train_data[order, :], train_label[order, :]),
(test_data, test_label)
```

2.3 Задание №3. Представление данных и библиотека NumPy

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_3.pdf

Примечания к задачам:

- Необходимо использовать модуль numpy
- Все данные должны считываться из файла в виде массива numpy
- Результаты необходимо сохранять в файл

Задача 1

Дано множество из p матриц (n,n) и множество из p векторов (n,1). Написать функцию для рассчета суммы p произведений матриц (результат имеет размерность (n,1))

Пример: ссылка

Задача 2

Написать функцию преобразовывающую вектор чисел в матрицу бинарных представлений.

Задача 3

Написать функцию, которая возвращает все уникальные строки матрицы

Задача 4

Написать функцию, которая заполняет матрицу с размерами (M,N) случайными числами распределенными по нормальному закону. Затем считает мат. ожидание и дисперсию для каждого из столбцов, а также строит для каждой строки стоит гистограмму значений (использовать функцию hist из модуля matplotlib.plot)

Задача 5

Написать функцию, которая заполняет матрицу (M,N) в шахматном порядке заданными числами a и b.

Задача 6

Написать функцию, которая возвращает тензор представляющий изображение круга с заданным цветом и радиусом в схеме rgd на черном фоне.

Задача 7

Написать функцию, которая стандартизирует все значения тензор (отнять мат. ожидание и поделить на СКО)

Задача 8

Написать функцию, выделяющую часть матрицы фиксированного размера с центром в данном элементе (дополненное значением fill если необходимо)

Задача 9

Написать функцию, которая находит самое часто встречающееся число в каждой строке матрицы и возвращает массив этих значений

Задача 10

Дан трёхмерный массив, содержащий изображение, размера (height, width, numChannels), а также вектор длины numChannels. Написать функцию, которая складывает каналы изображения с указанными весами, и возвращает результат в виде матрицы размера (height, width)

2.4 Задание №4. Операции с тензорами в библиотеке Keras

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_4.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Необходимо реализовать нейронную сеть вычисляющую результат заданной логической операции. Затем реализовать функции, которые будут симулировать работу построенной модели. Функции должны принимать тензор входных данных и список весов. Должно быть реализовано 2 функции:

1. Функция, в которой все операции реализованы как поэлементные операции над тензорами

2. Функция, в которой все операции реализованы с использованием операций над тензорами из NumPy

Для проверки корректности работы функций необходимо:

1. Инициализировать модель и получить из нее веса (Как получить веса слоя, Как получить список слоев модели)
2. Прогнать датасет через не обученную модель и реализованные 2 функции. Сравнить результат.
3. Обучить модель и получить веса после обучения
4. Прогнать датасет через обученную модель и реализованные 2 функции. Сравнить результат.

Примечание: так как множество всех наблюдений ограничен, то обучение проводить можно на всем датасете без контроля.

Вариант 1

(a and b) or (a and c)

Вариант 2

(a or b) xor not(b and c)

Вариант 3

(a and b) or c

Вариант 4

(a or b) and (b or c)

Вариант 5

(a xor b) and (b xor c)

Вариант 6

(a and not b) or (c xor b)

Вариант 7

(a or b) and (a xor not b)

Вариант 8

(a and c and b) xor (a or not b)

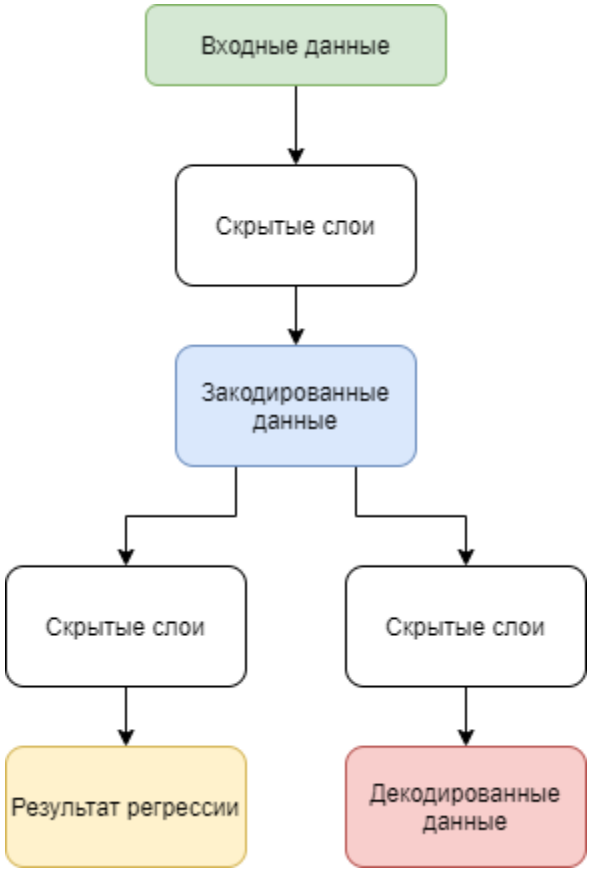
2.5 Задание №5. Оптимизация нейронных сетей в библиотеке Keras

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_5.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Необходимо в зависимости от варианта сгенерировать датасет и сохранить его в формате csv.

Построить модель, которая будет содержать в себе автокодировщик и регрессионную модель. Схематично это должно выглядеть следующим образом:



Обучить модель и разбить обученную модель на 3: Модель кодирования данных (Входные данные -> Закодированные данные), модель декодирования данных (Закодированные данные -> Декодированные данные), и регрессионную модель (Входные данные -> Результат регрессии). В качестве результата представить исходный код, сгенерированные данные в формате csv, кодированные и декодированные данные в формате csv, результат регрессии в формате csv (что должно быть и что выдает модель), и сами 3 модели в формате h5.
Сохранение и загрузка модели:

```
from keras.models import load_model
model.save('my_model.h5')  # creates a HDF5 file 'my_model.h5'
del model  # deletes the existing model
# returns a compiled model
# identical to the previous one
model = load_model('my_model.h5')
```

Вариант выбирается по списку, цель регрессии выбирается по остатку деления номера зачетки (последние 2 цифры) на 7 плюс 1. Предсказать надо одно значение, на основании шести остальных.
В названии pr необходимо будет указать номер варианта и номер цели.

Вариант 1
X ∈ N(3,10)
e ∈ N(0,0.3)

Признак	1	2	3	4	5	6	7
Формула	X^2+e	sin(X/2)+e	cos(2x)+e	X-3+e	-X+e	|X|+e	(X^3)/4+e

Вариант 2
X ∈ N(-5,10)
e ∈ N(0,0.3)

Признак	1	2	3	4	5	6	7
Формула	-X^3+e	ln(|X|)+e	sin(3X)+e	exp(X)+e	X+4+e	-X+sqrt(|X|)+e	X+e

Вариант 3
X ∈ N(0,10)
e ∈ N(0,0.3)

Признак	1	2	3	4	5	6	7
Формула	X^2+X+e	|X|+e	sin(X-PI\4)+e	lg(|X|)+e	-X^3+e	-X/4+e	-X+e

Вариант 4
X ∈ N(0,10)
e ∈ N(0,0.3)

Признак	1	2	3	4	5	6	7
Формула	cos(X)+e	-X+e	sin(X)*X+e	sqrt(|X|)+e	X^2+e	-|X|+4	X-(X^2)/5+e

2.6 Задание №6. Процесс решения задач с применением нейронных сетей в библиотеке Keras

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_6.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Необходимо построить сверточную нейронную сеть, которая будет классифицировать черно-белые изображения с простыми геометрическими фигурами на них.

К каждому варианту прилагается код, который генерирует изображения.

Для генерации данных необходимо вызвать функцию gen_data, которая возвращает два тензора:

1. Тензор с изображениями ранга 3
2. Тензор с метками классов

Обратите внимание:

- Выборки не перемешаны, то есть наблюдения классов идут по порядку
- Классы характеризуются строковой меткой
- Выборка изначально не разбита на обучающую, контрольную и тестовую
- Скачивать необходимо оба файла. Подключать файл, который начинается с var (в нем и находится функция gen_data)

Ознакомиться с построением сверточных нейронных сетей на примере MNIST можно по ссылке

Вариант 1

Классификация изображений с прямоугольником или кругом.

Файл 1

Файл 2

Вариант 2

Классификация изображений с закрашенным или не закрашенным кругом

Файл 1

Файл 2

Вариант 3

Классификация изображений с горизонтальной или вертикальной линией

Файл 1
Файл 2

Вариант 4

Классификация изображений с крестом или с линией (может быть горизонтальной или вертикальной)

Файл 1
Файл 2

Вариант 5

Классификация изображений с прямоугольником или не закрашенным кругом

Файл 1
Файл 2

Вариант 6

Классификация изображений по количеству крестов на них. Может быть 1, 2 или 3

Файл 1
Файл 2

Вариант 7

Классификация изображений по количеству линий на них. Может быть 1, 2 или 3

Файл 1
Файл 2

2.7 Задание №7. Мониторинг моделей глубокого обучения средствами библиотеки Keras

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_7.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Необходимо построить рекуррентную нейронную сеть, которая будет прогнозировать значение некоторого периодического сигнала.
Здесь находится пример простой рекуррентной сети, которая предсказывает значение зашумленной синусоиды. Модель из примера не является наилучшей, а лишь демонстрирует пример построения сети со слоями GRU и LSTM.
Информация по GRU и LSTM есть в презентациях лекций. Про эти слои в Keras можно прочитать здесь.
К каждому варианту предоставляется код, который генерирует последовательность. Для выполнения задания необходимо:

1. Преобразовать последовательность в датасет, который можно подавать на вход нейронной сети (можно использовать функцию gen_data_from_sequence из примера)
2. Разбить датасет на обучающую, контрольную и тестовую выборку
3. Построить и обучить модель
4. Построить график последовательности, предсказанной на тестовой выборке (пример построения также есть в примере). Данный график необходимо также добавить в pr

Также, в файлах с кодом вариантов есть функция draw_sequence, которая позволяет нарисовать часть последовательности

Вариант 1
Код с генерацией последовательности

Вариант 2
Код с генерацией последовательности

Вариант 3
Код с генерацией последовательности

Вариант 4
Код с генерацией последовательности

Вариант 5
Код с генерацией последовательности

Вариант 6
Код с генерацией последовательности

Вариант 7
Код с генерацией последовательности

Вариант 8

Код с генерацией последовательности

2.8 Задание №8. Ансамблирование моделей нейронных сетей с использованием библиотеки Keras

Необходимые материалы для выполнения задания: http://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_8.pdf

Студент выполняет один вариант задания, назначенный преподавателем.

Необходимо реализовать собственной CallBack, и провести обучение вашей модели из практического занятия №6 с написанным CallBack'ом. То, какой CallBack необходимо реализовать определяется вариантом.

Вариант 1
Сохранение трех наилучших моделей. Название файлов с моделями должны иметь следующий вид <текущая дата>_<префикс, задаваемый пользователем>_<номер модели>

Вариант 2
Построение и сохранение карты признаков на заданных пользователем эпохах. Карта признаков - ядро свертки представленное в виде изображения. Название карты признака должно иметь вид <номер слоя>_<номер ядра в слое>_<номер эпохи>

Вариант 3
Построение и сохранение нормированных гистограмм точности классификации наблюдений на заданных пользователем эпохах. Названия файлов с моделями должны иметь название соответствующее эпохе на которой была построена гистограмма

Вариант 4
Сохранение моделей на заданных пользователем эпохах. Название файлов с моделями должна иметь следующий вид <текущая дата>_<префикс, задаваемый пользователем>_<номер эпохи>

Вариант 5
На каждой эпохе расчет количества наблюдений для которых точность классификации меньше 90%. В конце обучения построения и сохранение графика изменения рассчитываемой величины

Вариант 6

Построение и сохранение таблицы со следующими данными: номер эпохи, номер наблюдения с наименьшей точностью классификации на заданной эпохе, к какому классу принадлежит наблюдение, точность классификации, значение ошибки.
Каждая строчка должна рассчитываться с заданным пользователем интервалом начиная с 0 эпохи, а также на самой последней