Програмиране на невронна мрежа с Python

В темата ще бъдат разгледани следните основни въпроси:

- Същност на невронните мрежи
- Съставни части на невронните мрежи
- Невронни мрежи
- Примери за неврон и невронна мрежа
- Код за невронна мрежа на Python

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Програмиране на невронна мрежа с Python

Изкуствена невронна мрежа или често наричана само *невронна мрежа*, е модел за обработка на данни, наподобяваща действието на биоелектричните мрежи в мозъка на човека и животните, образувани от неврони и техните синапси.

Невронната мрежа е математически аналог на биологичната невронна мрежа и представлява множество от взаимносвързани прости изчислителни елементи, наречени *неврони*. Всеки неврон приема сигнали от другите неврони или от входа под формата на числа, сумира ги, като сумата преминава през функция за активация. Така се определя своята степента на възбуда (активация), която се предава по изходящите връзки към другите неврони. Всяка връзка има тегло, което, умножавайки сигнала, определя неговата значимост (сила). Теглата на връзките са аналогични на силата на синаптичните импулси, предавани между биологичните неврони. Отрицателна стойност на теглото съответства на потискащ биологичен импулс, а положителна – на възбуждащ импулс.

Всяка невронна мрежа винаги има входен и изходен слой от неврони, като във входния се въвеждат данните към мрежата. След това сигналите от входните неврони преминават през един или няколко скрити слоя от междинни неврони, като сигналите накрая достигат до изходния слой, откъдето се извежда изходната информация.

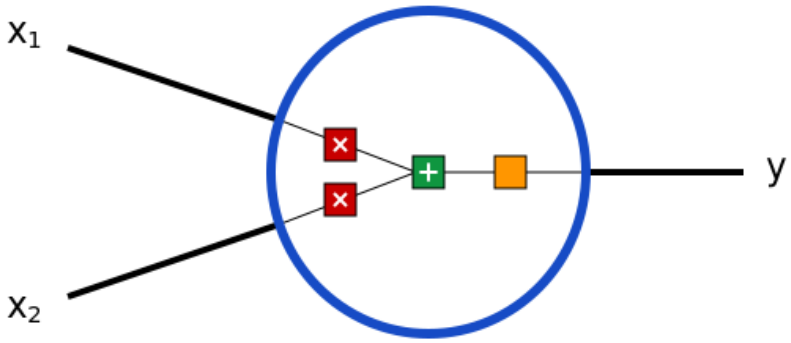
Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Доказано е, че всяка невронна мрежа с поне един скрит слой от достатъчен на брой неврони между входния и изходния слой, може да моделира поведението на всяка реално съществуваща функция.

За да се разбере действието на невронните мрежи последователно ще бъдат въведени техните основни елементи и ще бъде реализирана проста невронна мрежа с код на Python.

1. Съставни части на невронните мрежи

Невронът, като основен елемент на невронната мрежа, приема множество входни сигнали, извършва някои математически операции върху тях и след това извежда един изход. На фиг. 5.1 е представен неврон с два входа x_1 и x_2 и един изход y.



Фиг. 5.1. Основни операции в неврон с два входа и един изход

Три операции се извършват вътре в неврона. Първо, входните стойности се умножават по *техните тегла*:

$$x_1 \rightarrow x_1 * w_1, x_2 \rightarrow x_2 * w_2$$

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

След това претеглените входове се събират и към тях се добавя *праговата стойност* b:

$$x_1 * w_1 + x_2 * w_2 + b$$

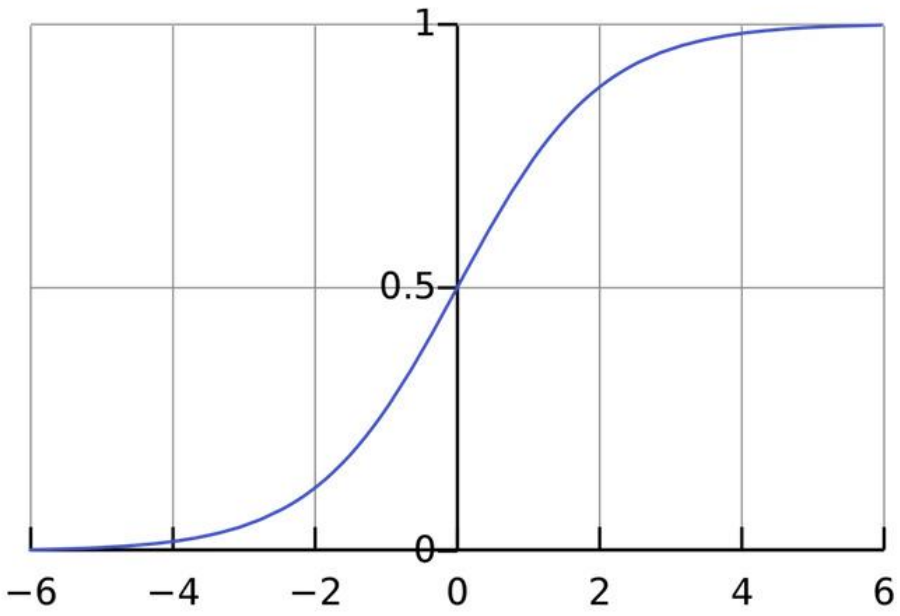
Накрая получената сума се предава на *функция за активиране* f:

$$y = f(x_1 * w_1 + x_2 * w_2 + b)$$

Функцията за активиране трансформира неограничени входни стойности в изход, който има ясна и предвидима форма. Една често използвана функция за активиране е *сигмоидната функция*

$$f(x) = \frac{1}{1 + e^{-x}}.$$

Графиката на сигмоидната функция е показана на фиг. 5.2.



Фиг. 5.2. Сигмоидна функция

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Сигмоидът е гладка, монотонна, нарастваща нелинейна „S-образна“ функция, която често се използва за изглаждане на стойностите на някаква величина. Сигмоидът често се използва и с наименованието като логистична функция. Сигмоидната функция дава резултати в интервала (0, 1). Реално той преобразува интервала от (-∞, +∞) в (0, 1), като големите отрицателни числа се превръщат в числа, близки до 0, а големите положителни числа се превръщат в числа, близки до 1.

Производната на сигмоидната функция е камбанообразна крива с връх при нула, клоняща асимптотично към нула при ∞.

Невронните мрежи често използват сигмоиди като функции за активация, защото техните производни могат да бъдат изразени по отношение на самата функция. Това позволява значително да се намали изчислителната сложност на алгоритъма за обратно разпространение на грешката, което го прави приложим на практика.

Производната на логистичната функция е

$$f'(x) = f(x).(1 - f(x)).$$

Освен логистичната функция, често се прилага и функцията за хиперболичен тангенс.

Ще бъде даден прост пример 5.1 за изчисленията, които се извършват в един неврон.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Пример 5.1. Нека невронът с два входа използва функция за активиране на сигмоид и има следните параметри:

$$w = [0, 1] \text{ и } b = 4$$

Интервалът $w = [0, 1]$ се представя като $w_1 = 0$ и $w_2 = 1$ във векторна форма. Нека входните данни на неврона са $x = [2, 3]$. Като се използва скаларно произведение на вектори, формули () и () могат да се запишат в съкратена форма и да се пресметне:

$$w.x + b = x_1 * w_1 + x_2 * w_2 + b = 0 * 2 + 1 * 3 + 4 = 7$$

$$y = f(w.x + b) = f(7) = 0{,}999$$

Следователно, невронът извежда 0,999 за входове $x = [2, 3]$.

Процесът на подаване на входни стойности, за да се получи изход, се нарича *предаване напред (feed forward)*.

За написване на кода за неврона се използва популярната и мощна библиотека за изчисления на Python NumPy. Класът `Neuron` има метод за инициализация `__init__(self, weights, bias)` с входни параметри теглата `weights` и прага `bias`. Методът `feedforward(self, inputs)` умножава входовете с теглата, прибавя прага и използва функцията за активация за завършване на процеса на предаване на входните сигнали напред към изхода.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

```
import numpy as np

def sigmoid(x):
  # Функция за активация: f(x) = 1 / (1 + e^(-x))
  return 1 / (1 + np.exp(-x))

class Neuron:
  def __init__(self, weights, bias):
    self.weights = weights
    self.bias = bias

  def feedforward(self, inputs):
    # Умножаване на входа с теглото, прибавяне на прага,
използване на функцията за активация
    total = np.dot(self.weights, inputs) + self.bias
    return sigmoid(total)

weights = np.array([0, 1]) # w1 = 0, w2 = 1
bias = 4                   # b = 4
n = Neuron(weights, bias)

x = np.array([2, 3])       # x1 = 2, x2 = 3
print("Изходът от неврона е", n.feedforward(x)
```

а)

```
Изходът от неврона е 0.9990889488055994
```

б)

Фиг. 5.3. Пример за дефиниране и използване на клас Neuron

а. Код на Python **б.** Изход от програмата

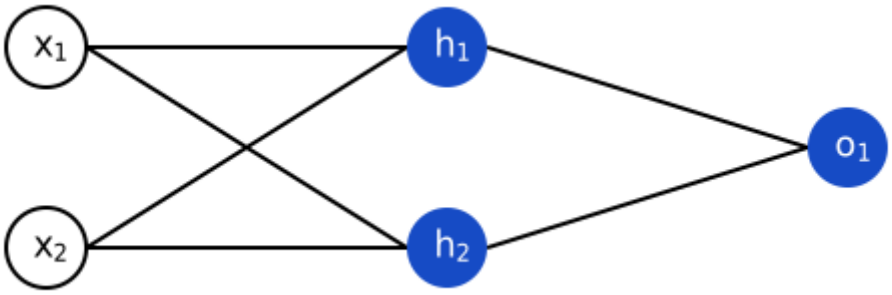
Изведеният изход от неврона е същия като получения резултат от разгледания по-горе пример 5.1.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

2. Невронна мрежа

Невронната мрежа е съставена от свързани един с друг неврони. На фиг. 5.4 е представена една проста невронна мрежа. Тази невронна мрежа има два входа x_1 и x_2, *скрит слой* с два неврона h_1 и h_2 и изходен слой с един неврон o_1, като входовете за x_1 и x_2 са изходите от h_1 и h_2.

Скрит слой е всеки слой между входния (първи) слой на мрежата и изходния (последен) слой на мрежата. Невронната мрежа може да има много скрити слоеве.



Фиг. 5.4. Проста невронна мрежа с два входа, един изход и един скрит слой

Пример 5.2. Нека в невронната мрежа от фиг. 5.4 всички неврони имат едни и същи тегла w = [0, 1], еднакви прагови стойности b = 0 и една и съща активираща функция сигмоид. Нека h_1, h_2 и o_1 са изходните стойности на невроните от фиг. 5.4.

При подаване на вход x = [2, 3] на изходите на невроните се получава

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

$$h_1 = h_2 = f(w \cdot x + b) = f((0 * 2) + (1 * 3) + 0) = f(3) = 0.9526$$

$$o_1 = f(w \cdot [h_1, h_2] + b) = f\big((0 * h_1) + (1 * h_2) + 0\big) =$$
$$= f(0{,}9526) = 0{,}7216$$

Следователно, ако се подаде x = [2, 3] на входа на невронната мрежа от фиг. 5.4, изходът ще бъде 0,7216.

Една невронна мрежа може да има произволен брой слоеве и тези слоеве могат да имат произволен брой неврони. Основната идея е да се предават входните данни по невроните на мрежата, докато се получат изходни стойности.

3. Код за невронна мрежа на Python

Ще бъде реализирана предавателната връзка за невронната мрежа от фиг. 5.4 с код на Python. В кода от фиг. 5.5 е дефиниран класа `NeuralNetwork` методът за инициализация на обекта при неговото създаване `__init__(self)` и метода за разпространение на входните данни през невроните към изхода `feedforward(self, x)`.

Създава се обекта network и с него се извиква метода `feedforward` с входните данни `x = [2, 3]`.

Както се вижда от фиг. 5.5.б на изхода на невронната мрежа се получава 0.7216, който е резултатът от пример 5.2.

В дисциплината „Машинно обучение и самообучение“ невронната мрежа ще бъде обучена и използвана за предсказване на нови данни.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

```
import numpy as np

# вмъкнете кода на функцията sigmoid и класа Neuron

class NeuralNetwork:
  '''
  Невронна мрежа с:
    - 2 входа
    - скрит слой с 2 неврона (h1, h2)
    - изходен слой с 1 неврон (o1)
  Всички неврони имат еднакви тегла и прагове:
    - w = [0, 1]
    - b = 0
  '''
  def __init__(self):
    weights = np.array([0, 1])
    bias = 0

    # Използване на класс Neuron
    self.h1 = Neuron(weights, bias)
    self.h2 = Neuron(weights, bias)
    self.o1 = Neuron(weights, bias)

  def feedforward(self, x):
    out_h1 = self.h1.feedforward(x)
    out_h2 = self.h2.feedforward(x)

    # Входове за o1 са изходите h1 и h2
    out_o1 = self.o1.feedforward(np.array([out_h1, out_h2]))

    return out_o1

network = NeuralNetwork()
x = np.array([2, 3])
print(„Изходът от невронната мрежа е“,
network.feedforward(x))
```

а)

```
Изходът от невронната мрежа е 0.7216325609518421
```

б)

Фиг. 5.5. Пример за дефиниране и използване на клас `NeuralNetwork`

а. Код на Python **б.** Изход от програмата

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.