VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMOS

„Dirbtinis neuronas“

Praktinė užduotis Nr. 1

Darbą atliko: Jaroslav Kochanovskis
Vadovė: Prof., Dr., Olga Kurasova

Vilnius 2022

Darbo tikslas – išanalizuoti dirbtinio neurono modelį ir jo veikimo principus.

Duomenys – pateikta lentelė sudaryta iš keturių eilučių, kuriuose yra po dvi įvestis ir norimas klasifikatoriaus rezultatas.

Programos kodas – parašyta naudojant „Python“ programavimo kalbą. Neurono modelį galima pamatyti „neuron.py“ faile. Failas „main.py“ naudoja neurono modelį ir ieško poslinkio ir svorių pagal duotus duomenis iš lentelės. Programa išsaugo faile visus galimus svorius iš nurodyto intervalo.

```
32      # Sums all inputs with weights and bias
33      def __activation(self):
34          output = 0
35          for (i, v) in zip(self._inputs, self._weights):
36              output += i * v
37          return output
38
39      # Output, depends on chosen activation type
40      def activation_func(self):
41          if self._activation_type == 0:
42              if self.__activation() >= 0:
43                  return 1
44              else:
45                  return 0
46
47          if self._activation_type == 1:
48              return 1 / (1 + math.e ** (-1 * self.__activation()))
```

1 pav. Neurono modelio kodo dalis

„neuron.py“ faile neurono modelis susideda iš konstruktoriaus, geterių, seterių ir keleto savų funkcijų. Viena iš jų yra „__activation“ (žr. 1 pav.), kuri sudaugina įvestis su svoriais ir sudeda juos visus kartu su poslinkiu ir gražina tą sumą. Šią funkciją naudoja kita funkcija „activation_func“ (žr. 1 pav.), kuri pagal pasirinkimą naudoja slenkstinę arba sigmoidinę aktyvacijos funkciją.

```
6   def read_input():
7       data = []
8       input_file = open("input.txt", "r")
9       for line in input_file:
10          data += [list(map(float, line.split()))]
11      return data
12
```

2 pav. Failo skaitymo funkcija

Įvesties duomenys programa nuskaito iš „input.txt“ (žr. 2 pav.) failo su „read_input“ funkcija ir juos konvertuoja, kad gražintu masyvą susidedanti iš įvesčių ir su tikėtina klase.

```
14  def check_set(neuron, input_file, weights):
15      result = True
16      for val in input_file:
17          neuron.set_inputs([1] + val[:-1])
18          neuron.set_weights(weights)
19          result = result and (float(val[-1]) == round(neuron.activation_func()))
20      if result:
21          f = open("output.txt", "a")
22          f.write(str(weights) + "\n")
23
```

pav. 3 Duomenų tikrinimas

Funkcija „check_set“ (žr. 3 pav.) patikrina ar pasirinkti svoriai tinka visoms įvestims. Jeigu tinka, tada išsaugo tą svorį „output.txt“ faile. Prieš tai funkcija taip pat į įvesčių masyvo priekį įdeda vienetą, nes svorių pirmasis elementas atitinka poslinkį (w_0).

```
25  def main(argv):
26      argv = list(map(int, argv))
27      if not len(argv) == 3:
28          print("python main.py <bias> <weight_from> <weight_to>")
29          sys.exit(0)
30      f = open("output.txt", "w")
31      f.write("")
32      neuron = Neuron(0, [], [])
33      neuron.set_activation(0)
34      input_file = read_input()
35      for i in list(range(argv[1], argv[2])):
36          for j in list(range(argv[1], argv[2])):
37              check_set(neuron, input_file, [argv[0], i, j])
38
39
40  # Press the green button in the gutter to run the script.
41  if __name__ == '__main__':
42      main(sys.argv[1:])
```

pav. 4 Main funkcija

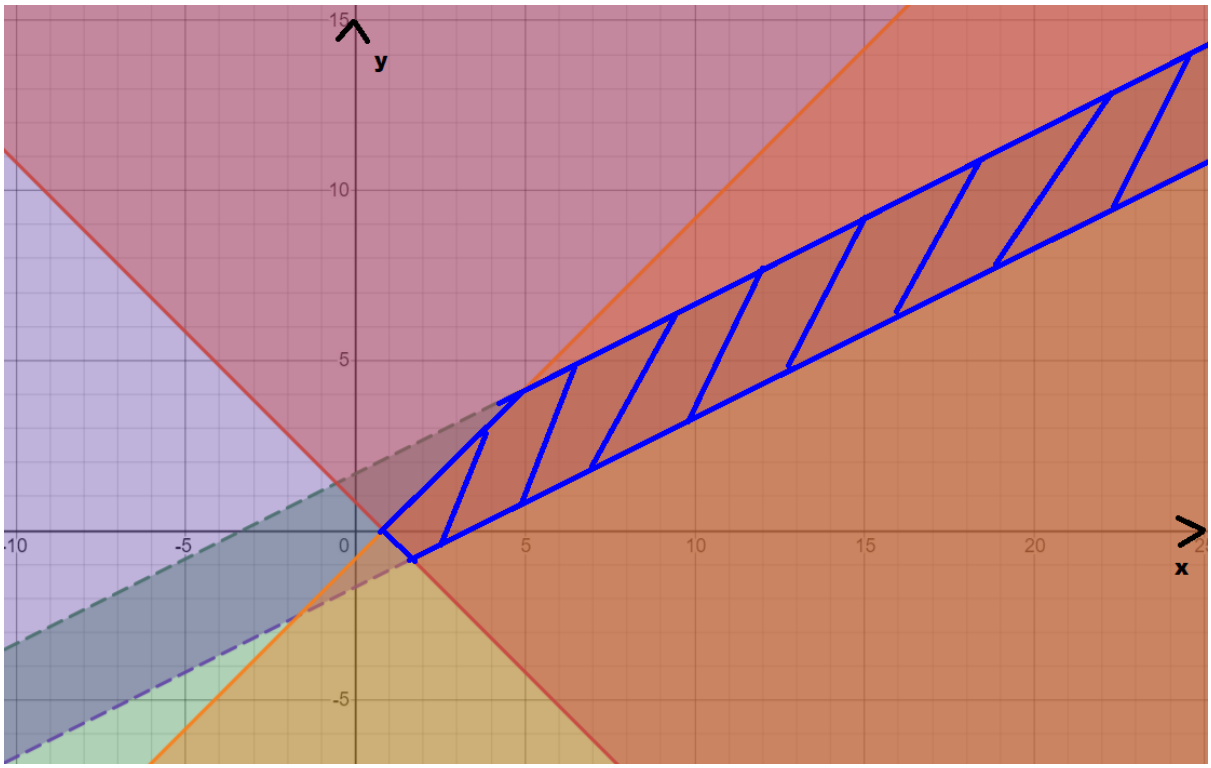
Paleista programa nuskaito trys parametrus: poslinkį ir svorių intervalo ribas nuo iki. „main“ funkcijoje (žr. 4 pav.) yra sukuriamas tuščias neuronas, kad po to būtų galima įdėti tikrinamus svorius. 33 eilutėje galima nusistatyti kuria aktyvacijos funkciją norime naudoti. Nulis atitinka slenkstinę, o vienetas – sigmoidinę. Tada paleidžiame ciklą iš nurodyto intervalo ir tikriname, ar tinka pasirinktas svorių rinkinys.

Kadangi svorių intervalas yra perrenkamas iš sveikų skaičių, todėl rezultatas ar renkantis slenkstinę, ar sigmoidinę lieka toks pat. (žr. lentelė 1)

lentelė 1 Svorių sprendimai

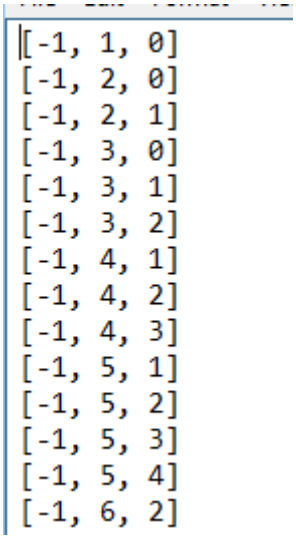
w_0	w_1	w_2
-1	1	0
-1	2	0
-1	2	1
-1	3	0

Norint teisingai parinkti svorių ir poslinkio reikšmes, kai aktyvacijos funkcija būtų slenkstinė reikia išspręsti tokią nelygybių sistemą: $\begin{cases} -0.3x + 0.6y + w_0 < 0 \\ 0.3x - 0.6y + w_0 < 0 \\ 1.2x - 1.2y + w_0 \geq 0 \\ 1.2x + 1.2y + w_0 \geq 0 \end{cases}$, kur (x, y) atitinka (w_1, w_2). Pasinaudojus „desmos“ (žr. 5 pav.) įrankiu nupiešiame nelygybių sistemas ir nuspalvintas plotas ir bus atsakymų rinkinys. Pastaba! Poslinkis būtinai turi būti neigiamas. Kitu atveju nelygybių sistema neturės sprendinių.



pav. 5 Nelygybių grafikas

Matome, kad programos rezultatas atitinka grafiko ploto sprendimus $[w_0, w_1, w_2]$ (žr. 6 pav.)



pav. 6 Programos išvestis

Apibendrinimas. Šio laboratorinio tyrimo metu pavyko išnagrinėti dirbtinio neurono modelį, bei realizuoti jį kode. Keičiant svorių ir poslinkio reikšmes pavyko gauti jų reikšmes pagal nurodytos lentelės įeitis ir klases. Pagal tai pavyko sudaryti nelygybių sistemą ir rasti sprendinius grafiniu būdu, įsitikinti, naudojantis programa, kad grafike gauti sprendiniai atitinka programos duodamus rezultatus.