VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

DUOMENŲ MOKSLAS. BAKALAURAS

Dirbtinis neuronas

Praktinė užduotis Nr. 1

Užduotį atliko: Ugnius Vilimas 3 kursas, 2 grupė

VU el. paštas: ugnius.vilimas@mif.stud.vu.lt

2023

Turinys

Tikslas

Užduoties tikslas – išanalizuoti dirbtinio neurono modelį bei suprasti jo veikimą.

Uždaviniai

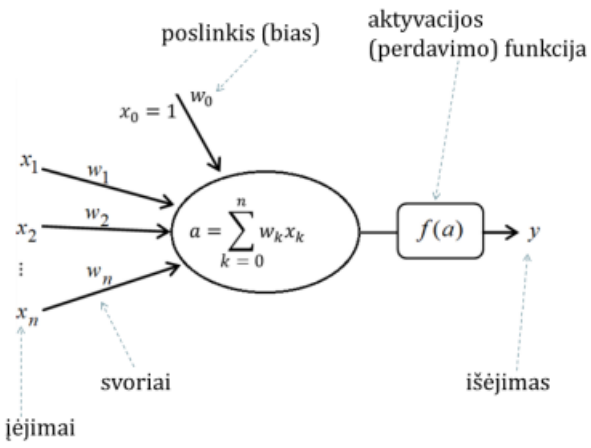
- Sukurti dirbtinio neurono modelį, kuriam būtų pateikiamos įėjimo reikšmės bei duodamas pasirinkimas tarp sigmoidinės arba slenkstinės funkcijų.
- Pagal sąlygoje pateiktus duomenis lentelėje, rasti užduoties svorius ir poslinkį.
- Išanalizuoti nelygybių sistemas.

Duomenys

1 lentelė. Duomenys klasifikavimui

Duomenys		**Klasė**
x_1	x_2	t
–0,2	0,5	0
0,2	–0,7	0
0,8	–0,8	1
0,8	1	1

Schema



1 pav. Dirbtinio neurono schema

Svarbios formulės

1) Įėjimo ir svorių reikšmių sandaugos formulė: $$a = \sum_{k=0}^{n} w_k x_k$$

2) Slenkstinė aktyvacijos funkcija: $$f(a) = \begin{cases} 1, jei\ a \geq 0 \\ 0, jei\ a < 0 \end{cases}$$

3) Sigmoidinė aktyvacijos funkcija: $$f(a) = \frac{1}{1 + e^{-a}}$$

Python programinis kodas su paaiškinimais ir komentarais

Programa rašyta „python“ programavimo kalba.

Kodas parašytas kuo labiau jį supaprastinus, kad būtų kuo lengviau paaiškinama ir suprantama. Pradžioje buvau susikūręs šiek tiek ilgesnį kodą, tačiau jį modifikavau. Pirmojo kodo varianto jau nebeturiu.

Pirmiausia importavau dvi bibliotekas, kurių reikės programos vykdymui.

```
import random
import math
```

1 pav. Bibliotekos

Tada pasidariau 2 sąrašus su duomenimis, kurie buvo pateikti lentelėje, užduotyje. Taip pat sąrašams skiriami vardai, kuriuos reikės toliau naudoti generuoti svorius.

```
x_poros = [[-0.2, 0.5], [0.2, -0.7], [0.8, -0.8], [0.8, 1]]
x_klases = [0, 0, 1, 1]
```

2 pav. Tikrinamos duotų duomenų poros ir reikšmės

Čia parašiau funkciją, atsitiktiniams skaičiams rasti. Pasirinkau naudoti antrą strategiją, tai yra atsitiktinai generuojant svorių ir poslinkių reikšmes *w* iš tam tikro intervalo ieškant tinkamos kombinacijos. Pasirinkau intervalą nuo -12 iki 12 su dešimtąja dalimi.

```
def atsitiktinis():
    return round(random.uniform(-12, 12), 2)
```

3 pav. atsitiktinių skaičių generavimas

Čia sukūriau funkciją, kuri leidžia vartotojui pasirinkus atitinkamai 0 arba 1, pasirinkti aktyvacijos funkciją slenkstinę ar sigmoidinę. Poslinkio ir svorių reikšmės būtent ir bus skaičiuojamos pagal pasirinktą.

```
def pasirinkimas():
    while True:
        pasirink = input("Spausti 0, jei norite, kad butu skaiciuojama\npagal slenkstine funkcija , Spauskite 1, jei pagal sigmoidine funkcija: ")
        if pasirink in ('0', '1'):
            return pasirink
```

4 pav. aktyvacijos funkcijos pasirinkimo funkcija (slenkstinė ar sigmoidinė)

Čia parašiau funkciją „gauti_a_reikšmes“, kurios būtent tai ir daro. Tai yra *for loop*, kuriame yra sąlyga, kaip apskaičiuojamos *a* reikšmės. *a* formulė buvo nurodyta anksčiau, formulių skiltyje. Čia funkcija dar nėra priskirta nei vienai iš dviejų naudojamų funkcijų, tai bus padaryta vėliau.

```
def gauti_a_reiksme(w0, w1, w2):
    return [w0 + w1 * p[0] + w2 * p[1] for p in x_poros]
```

5 pav. funkcija a reikšmėms gauti

Čia aprašoma slenkstinė ir sigmoidinė funkcijos.

Slenkstinė aktyvacijos funkcija apskaičiuojama pagal šią formulę: $$f(a) = \begin{cases} 1, jei\ a \geq 0 \\ 0, jei\ a < 0 \end{cases}$$

Sigmoidinė aktyvacijos funkcija apskaičiuojama pagal šią formulę: $$f(a) = \frac{1}{1 + e^{-a}}$$

```
def slenkstine(a):
    return 1 if a >= 0 else 0


def sigmoidine(a):
    return 1 / (1 + math.exp(-a))
```

6 pav. slenkstinė ir sigmoidinė funkcijos

Čia aprašoma sekanti funkcija, kuri rastu ir grąžintų ieškomus svorius, kuriuos čia pavadinau vienodai, tačiau jie grįš iš eilės. Atitinkamai iš eilės čia yra w_0, w_1, w_2, kur w_0 yra poslinkis.

```
def svoriai():
    return atsitiktinis(), atsitiktinis(), atsitiktinis()
```

7 pav. svorių generavimo funkcija

Tada sukūriau kitą funkciją „gautos_klases“, kuri randa *a* reikšmes, pagal tai, kurią funkciją, ar slenkstinę ar sigmoidinę, pasirinko vartotojas. Tam panaudojau paprasčiausią *if* funkciją. Jei bus įvedama netinkama reikšmė, tai pavyzdžiui kitas skaičius ar raidė, programa bus sustabdoma su *exit()* ir iškarto paleidžiama atgal, tikintis tinkamos įvesties.

```
def gautos_klases(w0, w1, w2, pasirink):
    a_reiksmes = gauti_a_reiksme(w0, w1, w2)
    return [slenkstine(a) if pasirink == '0' else round(sigmoidine(a)) for a in a_reiksmes]
```

8 pav. funkcija gautos klasės

Paskutinė funkcija yra „atitinkantys_svoriai“. Čia yra paprasčiausia paduodama, ką pasirinko vartotojas, tai yra ar sigmoidinę ar slenkstinę aktyvacijos funkciją. Į sukurtą *For* ciklą paduodami sugeneruoti svoriai bei gautos klasės. Rezultatas atspausdinamas tada, kai klasės atitinka tikrinamas klases. Rezultate matome atitinkamus svorius.

```
def atitinkantys_svoriai():
    pasirink = pasirinkimas()
    for _ in range(501):
        w0, w1, w2 = svoriai()
        klases = gautos_klases(w0, w1, w2, pasirink)
        if klases == x_klases:
            print(f"Gaunamos klases {klases}, atitinka pradine salyga su svoriais\natitinkančiais: w0 = {w0}, w1 = {w1}, w2 = {w2}")
            exit()
```

9 pav. atitinkančių svorių funkcija

Generavimo strategija

Užduoties sąlygoje buvo nurodyta pasirinkti pirmą arba antrą strategija. Tiek ir slenkstinės, tiek ir sigmoidinės aktyvacijos funkcijos naudojimui aš rinkausi antrą strategiją. Joje atsitiktinai generavau svorių ir poslinkių reikšmes iš pasirinkto intervalo. Mano intervalas buvo nuo -12 iki 12 su dešimtąja dalimi, tačiau buvo galima rinktis pagal norą.

Reikšmių rinkiniai

Slenkstinė aktyvacijos funkcija

Čia bus pateikti 5 sugeneruoti pavyzdžiai, tinkantys pateiktiems duomenims užduotyje, naudojant slenkstinę aktyvacijos funkciją.

Formulė: $$f(a) = \begin{cases} 1, jei\ a \geq 0 \\ 0, jei\ a < 0 \end{cases}$$

Paleidus programą dabar matysime 5 pavyzdžius su svoriais ir poslinkiu.

Rinkinys 1:
```
Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais
atitinkančiais: w0 = -5.62, w1 = 10.14, w2 = -0.9
```

Rinkinys 2: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -4.92, w1 = 11.33, w2 = 3.66

Rinkinys 3: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -1.83, w1 = 10.6, w2 = 2.31

Rinkinys 4: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -3.96, w1 = 8.37, w2 = -1.56

Rinkinys 5: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -1.91, w1 = 10.63, w2 = 8.01

Sigmoidinė aktyvacijos funkcija

Čia bus pateikti 5 sugeneruoti pavyzdžiai, tinkantys pateiktiems duomenims užduotyje, naudojant sigmoidinę aktyvacijos funkciją.

Formulė: $$f(a) = \frac{1}{1 + e^{-a}}$$

Paleidus programą dabar matysime 5 pavyzdžius su svoriais ir poslinkiu.

Rinkinys 1: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -1.43, w1 = 5.38, w2 = 1.53

Rinkinys 2: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -3.5, w1 = 10.94, w2 = 5.24

Rinkinys 3: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -4.0, w1 = 11.12, w2 = 5.19

Rinkinys 4: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -4.1, w1 = 10.43, w2 = 3.18

Rinkinys 5: Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais atitinkančiais: w0 = -0.76, w1 = 6.81, w2 = 1.27

Nelygybių sistema

Dabar patikrinsime ar mūsų sugeneruoti poslinkis ir svoriai yra mūsų duotų duomenų sprendiniai. Tai darysime susidarę tokia nelygybę:

$$\begin{cases} -0.2w_1 + 0.5w_2 + w_0 < 0 \\ 0.2w_1 - 0.7w_2 + w_0 < 0 \\ 0.8w_1 - 0.8w_2 + w_0 \geq 0 \\ 0.8w_1 + w_2 + w_0 \geq 0 \end{cases}$$

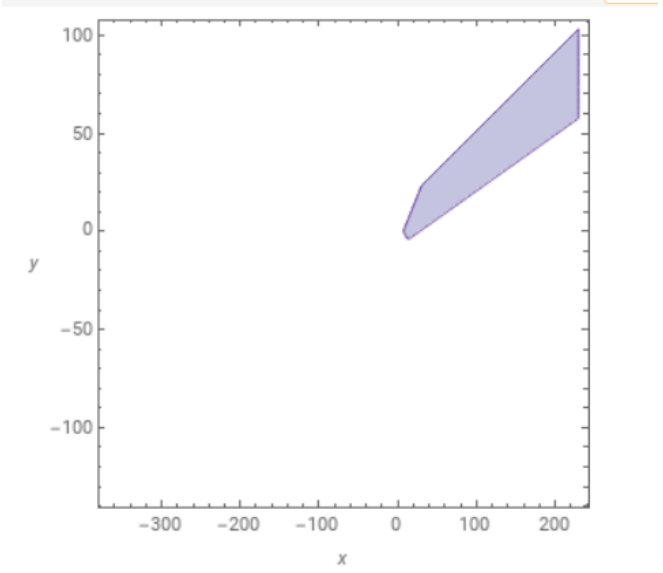
Kai norime patikrinti, ar mūsų sukurtos programos sugeneruoti svoriai ir poslinkis yra tinkami, poslinkio konstanta yra imama kaip konstanta ir žiūrima, ar svoriai, kurie bus pažymėti x ir y patenka į nuspalvintą plotą. Naudosiu programą WolframAlfa, kuri nubrėš dvimatį grafiką iš duotos lygčių sistemos ir tada galėsime matyti, ar taškas patenka į reikiamą plotą ar ne.

Pavyzdžių ir gautų svorių testavimas

1. Pirmas pavyzdys. Duomenys: Poslinkis: -5.62, svoriai: 10.14 ir -0.9

```
Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais
atitinkančiais: w0 = -5.62, w1 = 10.14, w2 = -0.9
```

Lygtis: $\begin{cases} -0.2w_1 + 0.5w_2 - 5.62 < 0 \\ 0.2w_1 - 0.7w_2 - 5.62 < 0 \\ 0.8w_1 - 0.8w_2 - 5.62 \geq 0 \\ 0.8w_1 + w_2 - 5.62 \geq 0 \end{cases}$

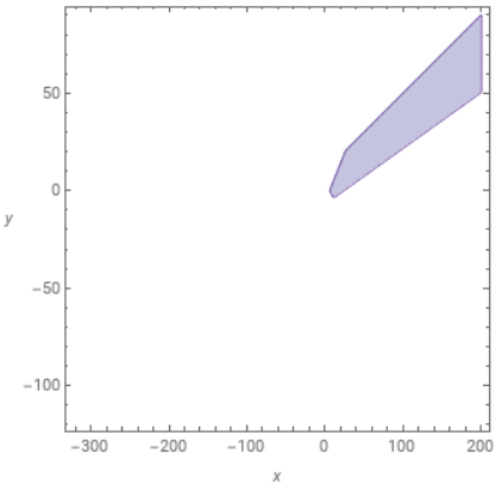


1 pavyzdys

2. Antras pavyzdys. Duomenys: poslinkis -4.92, svoriai 11.33 ir 3.66

```
Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais
atitinkančiais: w0 = -4.92, w1 = 11.33, w2 = 3.66
```

Lygtis: $\begin{cases} -0.2w_1 + 0.5w_2 - 4.92 < 0 \\ 0.2w_1 - 0.7w_2 - 4.92 < 0 \\ 0.8w_1 - 0.8w_2 - 4.92 \geq 0 \\ 0.8w_1 + w_2 - 4.92 \geq 0 \end{cases}$

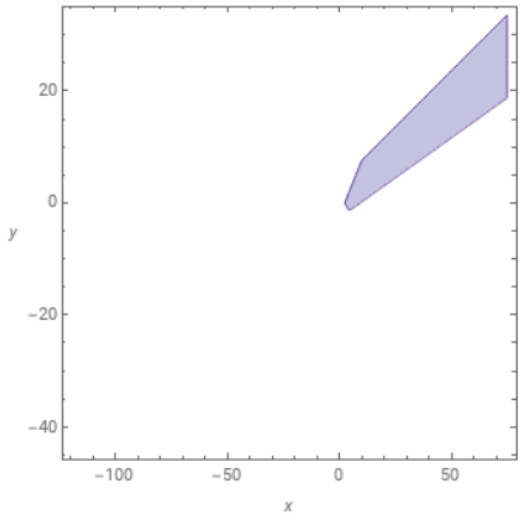


2 pavyzdys

3. Trečia pavyzdys. Duomenys: poslinkis -1.83, svoriai 10.6 ir 2.31

```
Gaunamos klases [0, 0, 1, 1], atitinka pradine salyga su svoriais
atitinkančiais: w0 = -1.83, w1 = 10.6, w2 = 2.31
```

Lygtis: $\begin{cases} -0.2w_1 + 0.5w_2 - 1.83 < 0 \\ 0.2w_1 - 0.7w_2 - 1.83 < 0 \\ 0.8w_1 - 0.8w_2 - 1.83 \geq 0 \\ 0.8w_1 + w_2 - 1.83 \geq 0 \end{cases}$



3 pavyzdys

Visi trys grafikai parodė, kad mūsų sugeneruoti taškai patenka į nuspalvintą plotą, tai reiškia, kad programa tikrai generuoja tinkamus svorius ir poslinkį pagal duotus duomenis.

Išvados

Šios užduoties metu buvo išsiaiškintas ir pilnai suprastas neurono modelio veikimo principas. Taip pat buvo sukurtas programinis kodas, naudojant python programavimo kalbą. Kodas skaičiavo funkcijos išėjimo reikšmes su slenkstine arba sigmoidine aktyvacijos funkcijomis, kurias buvo galima pasirinkti pačiam vartotojui.

Phyton programinis kodas

```
#biblioteku importavimas, kuriu reikes programai veikti
import random
import math

#lenteleje duotos reiksmes susidedamos i sarasus [x1, x2]
#viskas tiesiog sudedama i du sarasus su pavadinimais
x_poros = [[-0.2, 0.5], [0.2, -0.7], [0.8, -0.8], [0.8, 1]]
x_klases = [0, 0, 1, 1]

#sukuriama funkcija - atsitiktiniai skaiciai nuo -12 iki 12 su desimtaja dalimi
#buvo galima rinktis ir kitoki intervala bei skaiciu po kablelio kieki
#naudojama paprasciausia random funkcija kuri duos skaicius nuo -12 iki 12, su dviem skaiciais po kablelio
def atsitiktinis():
    return round(random.uniform(-12, 12), 2)

#slenkstines arba sigmoidines funkcijos pasirinkimas
#neatmeciau galimybes parasyti ne toki skaiciu, todel programa prasys tol kol irasys tinkama
def pasirinkimas():
    while True:
        pasirink = input("Spausti 0, jei norite, kad butu skaiciuojama\npagal slenkstine funkcija , Spauskite 1, jei pagal sigmoidine funkcija: ")
```

```
        if pasirink in ('0', '1'):
            return pasirink

#funkcija a reiksmei gauti, nepriklausomai ar tai bus sigmoidine ar slekstine fukcija
#tiesiog uzrasome funkcija pagal duota sumos formule
def gauti_a_reiksme(w0, w1, w2):
    return [w0 + w1 * p[0] + w2 * p[1] for p in x_poros]

#slenkstine aktyvacijos f-ja bei jai priskiriama a reiksme
def slenkstine(a):
    return 1 if a >= 0 else 0

#sigmoidine aktyvacijos f-ja bei jai priskirama a reiksme
def sigmoidine(a):
    return 1 / (1 + math.exp(-a))

#svoriu generavimas
def svoriai():
    return atsitiktinis(), atsitiktinis(), atsitiktinis()

#gaunamos klases
def gautos_klases(w0, w1, w2, pasirink):
    a_reiksmes = gauti_a_reiksme(w0, w1, w2)
    return [slenkstine(a) if pasirink == '0' else round(sigmoidine(a)) for a in a_reiksmes]

#atsitiktinai generuosime skaicius 500 kartu ir suzinosime ar yra tinkamu
def atitinkantys_svoriai():
    pasirink = pasirinkimas()
    for _ in range(501):
        w0, w1, w2 = svoriai()
```

```
        klases = gautos_klases(w0, w1, w2, pasirink)
        if klases == x_klases:
            print(f"Gaunamos klases {klases}, atitinka pradine salyga su svoriais\natitinkančiais: w0 = {w0}, w1 = {w1}, w2 = {w2}")
            exit()


#kai failas yra leidziamas kaip scriptas, koda vykdyti vistiek leidzia
if __name__ == "__main__":
    atitinkantys_svoriai()
```