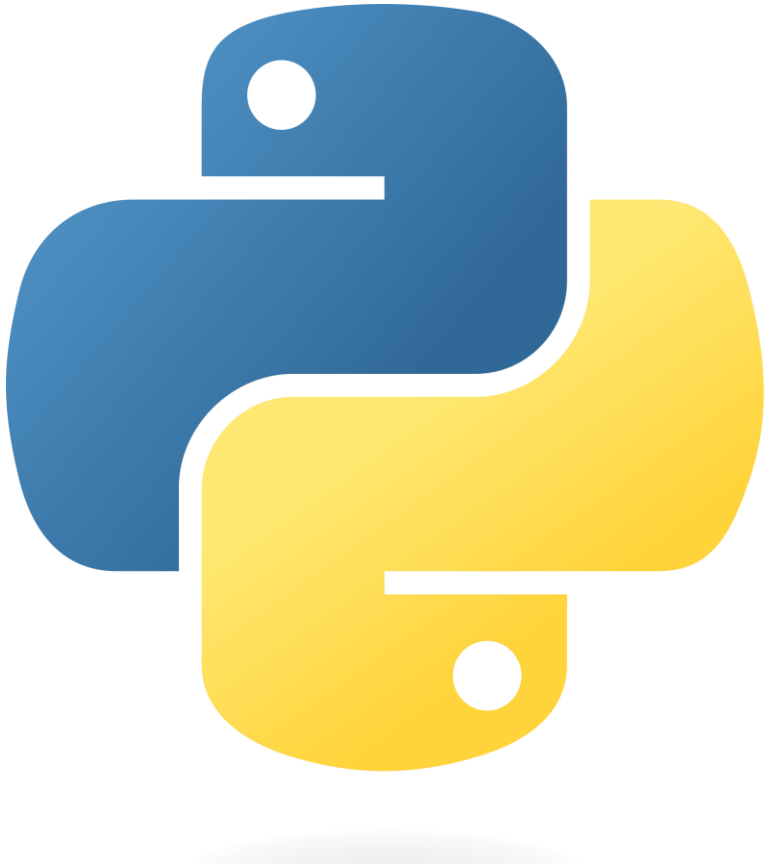
Obrada podataka i algoritmi mašinskog učenja

Nikola Vukotić

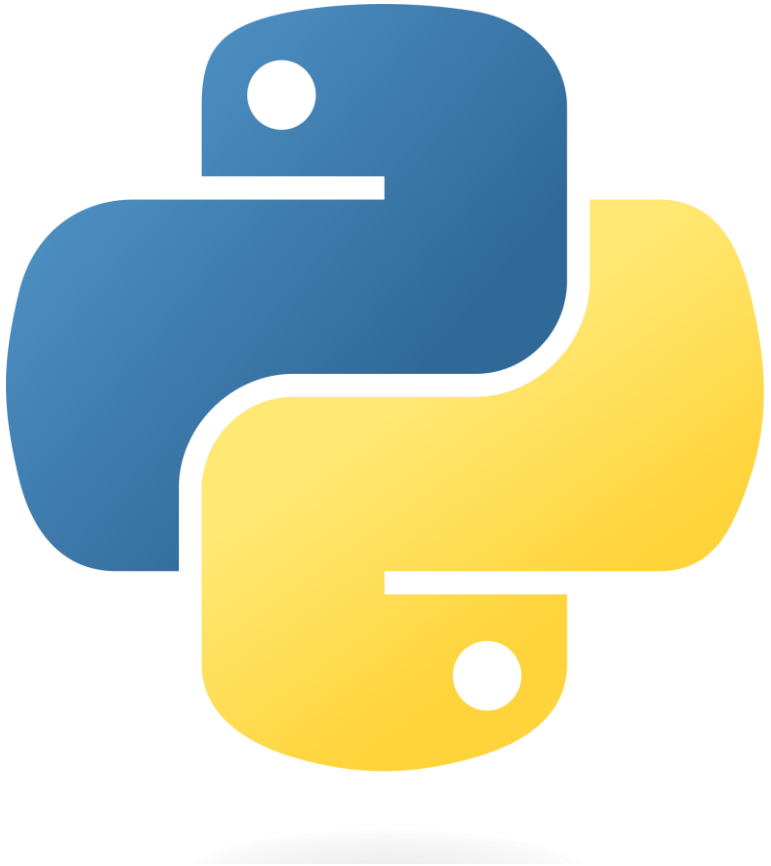
ODSEK NIŠ
AKADEMIJA TEHNIČKO-VASPITAČKIH STRUKOVNIH STUDIJA

Programski jezik
Python

- Python je programski jezik opšte namene koji se često primenjuje u pisanju raznih skripti posebne namene
- Dakle, Python je programski jezik kao i skriptni jezik
- Python se takođe naziva i interpretirani jezik
- Osmišljen sa ciljem da bude jednostavan za čitanje i pisanje



Programski jezik
Python

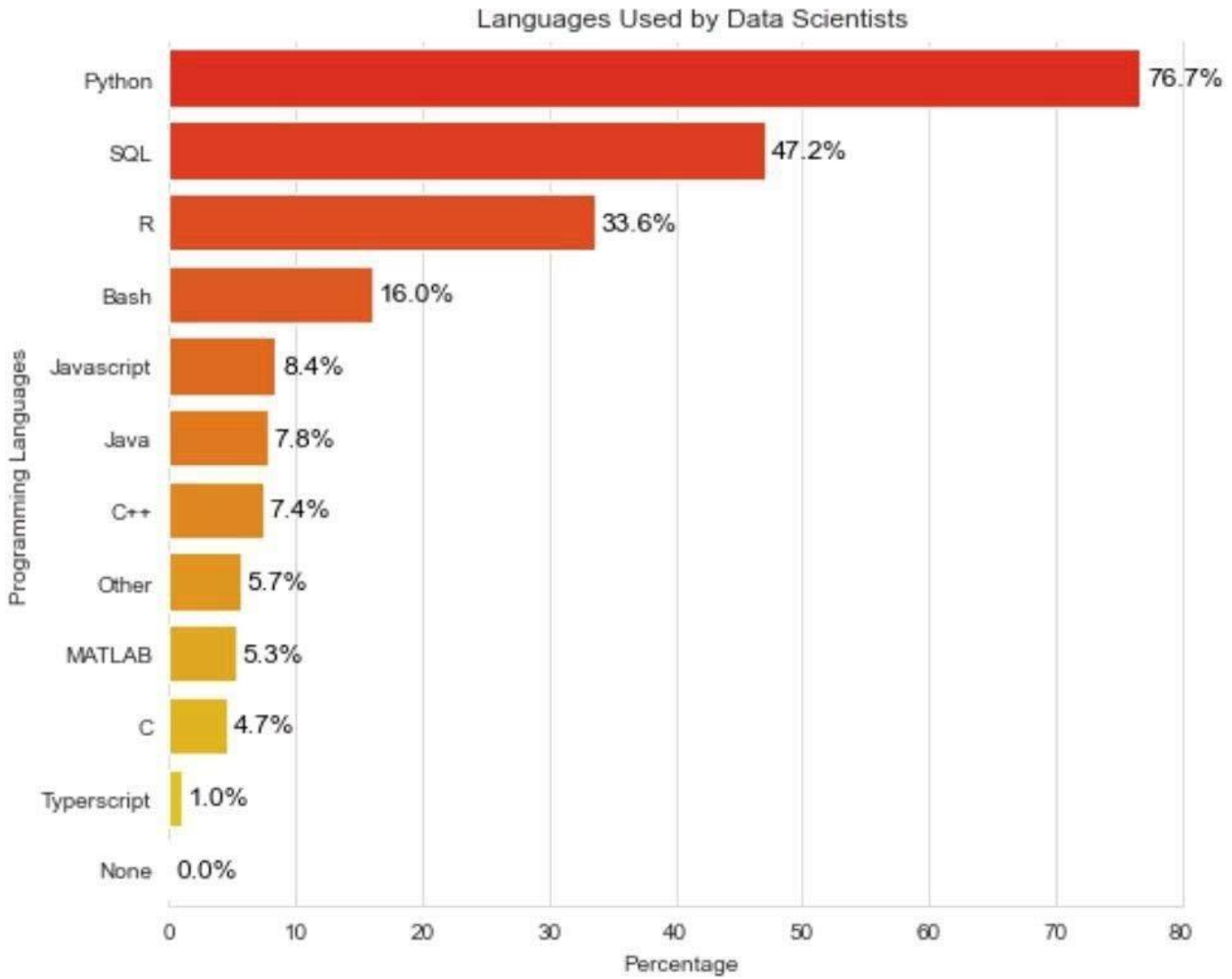


- Ima ogromnu i aktivnu zajednicu programera koji stvaraju i održavaju mnoge korisne biblioteke i alate koji olakšavaju razvoj aplikacija

- Može se izvoditi na različitim platformama
  - Windows, Mac, Linux

- Podrška za objektno orijentisano programiranje, funkcionalno programiranje i strukturno programiranje

Programski jezik Python



Osnove

Python-a

```
print("Hello World!")
```

```
Hello World!
```

Osnovne karakteristike

Python-a

- Radi se o jeziku sa dinamičkom i strogom tipizacijom, gde ne postoji deklaracija promenljivih i promenljive mogu da uzimaju vrednosti različitog tipa

- Stroga tipizacija znači da se tip vrednosti neće promeniti na neočekivan način
  - Primer - string koji sadrži samo cifre ne može da postane broj

- Dinamička tipizacija znači da runtime objekti imaju tip, nasuprot statičkoj tipizaciji gde promenljive imaju svoj tip

Osnovne karakteristike

Python-a

```
x = 7
print(x)
print(type(x))

x = 'Zdravo!'
print(x)
print(type(x))

x = 7.0
print(x)
print(type(x))

x = '2'
print(x + '2')
```

```
7
<class 'int'>
Zdravo!
<class 'str'>
7.0
<class 'float'>
22
```

Osnovne programske strukture

Python-a

- U Python-u blok nema separatore već se svaki blok identifikuje tabulatorima

- Osnovne strukture:
  - If-else
  - For petlja
  - While petlja

Osnovne programske strukture

Python-a

```
# If-elif-else

x = 5.0

if isinstance(x, int):
    print('Integer')
elif isinstance(x, str):
    print('String')
else:
    print('Other')
```

```
Other
```

Osnovne programske strukture

Python-a

```python
# for

ingredients = ['eggs', 'milk', 'flour', 'sugar', 'chocolate', 'cream']

for sastojak in ingredients:
    print(sastojak)
```

```
eggs
milk
flour
sugar
chocolate
cream
```

Osnovne programske strukture

Python-a

```
# for

for i in range(0, len(ingredients)):
    print(ingredients[i])
    print(i)
```

```
eggs
0
milk
1
flour
2
sugar
3
chocolate
4
cream
5
```

Osnovne programske strukture

Python-a

```
# while

i = 0
while i < len(ingredients):
    print(ingredients[i])
    i+=1
```

```
eggs
milk
flour
sugar
chocolate
cream
```

Rad sa poljima

u Python-u

```
sentence = ['Marija', 'jede', 'tortu', 'na', 'terasi']
sentence_len = len(sentence)

print(sentence[0]) # indeksiranje
print(sentence[len(sentence)-1])
print(sentence[-1])  # indeksiranje unazad - poslednji element niza
print(sentence[:3])  # uzimanje prva 3 elementa
print(sentence[3:]) # uzimanje sve osim prva tri
print(sentence[1:3]) # uzimanje od indeksa 1 do indeksa 3, ne uključujući i poslednji
print(sentence[::2]) # uzimanje svakog drugog elementa
print(sentence[::1])
print(sentence[::-1]) # uzimanje elemenata unazad
```

```
Marija
terasi
terasi
['Marija', 'jede', 'tortu']
['na', 'terasi']
['jede', 'tortu']
['Marija', 'tortu', 'terasi']
['Marija', 'jede', 'tortu', 'na', 'terasi']
['terasi', 'na', 'tortu', 'jede', 'Marija']
```

Rad sa poljima

u Python-u

```
#konkatenacija operatorom +
print(sentence + ['svakog', 'dana'])

sentence.append('pored')
sentence.append('mora')
print(sentence)

sentence.append('svakog dana')
print(sentence)

print('Mara' in sentence)
print(sentence.index('jede'))

#sentence.extend(["svakog", "dana"])
#print(sentence)

print(sorted(sentence)) # sortiranje po default komparatoru
print(sorted(sentence, key = lambda x: len(x)))

pom = sentence[0]
sentence[0] = sentence[1]
sentence[1] = pom

print(sentence)
```

```
['Marija', 'jede', 'tortu', 'na', 'terasi', 'svakog', 'dana']
['Marija', 'jede', 'tortu', 'na', 'terasi', 'pored', 'mora']
['Marija', 'jede', 'tortu', 'na', 'terasi', 'pored', 'mora', 'svakog dana']
False
1
['Marija', 'jede', 'mora', 'na', 'pored', 'svakog dana', 'terasi', 'tortu']
['na', 'jede', 'mora', 'tortu', 'pored', 'Marija', 'terasi', 'svakog dana']
['jede', 'Marija', 'tortu', 'na', 'terasi', 'pored', 'mora', 'svakog dana']
```

Rad sa poljima

u Python-u

```python
# List comprehension

numbers = [1, 2, 3, 4, 5, 6, 7]

numbers_squared = [x ** 2 for x in numbers]
print(numbers_squared)

numbers = [1, 2, 3, 4, 5, 6, 7]

numbers_squared = [x ** 2 for x in numbers]
print(numbers_squared)

numbers_even = [x for x in numbers if x % 2 == 0]
print(numbers_even)
```

```
[1, 4, 9, 16, 25, 36, 49]
[1, 4, 9, 16, 25, 36, 49]
[2, 4, 6]
```

Definicija funkcija u Python-u

```
def saberi(x, y):
    result = 0
    result = x + y
    return result

print(saberi(4, 5))
```

```
9
```

```
def saberiElemente(x):
    result = 0
    for el in x:
        result += el
    return result

x = [1, 2, 3, 4, 5, 6, 7]
print(saberiElemente(x))
```

```
28
```

Tuple
u Python-u

- Python Tuple je kolekcija objekata odvojenih zapetama
- Na neki nacin, Tuple je slican listi u smislu indeksiranja, ugnjezdjenih objekata i ponavljanja, ali je Tuple nepromenljiv, za razliku od listi koje mogu biti promenljive

- https://www.w3schools.com/python/python_tuples.asp
- https://docs.python.org/3/c-api/tuple.html
- https://www.programiz.com/python-programming/tuple
- https://www.geeksforgeeks.org/tuples-in-python/
- https://www.tutorialspoint.com/python/python_tuples.htm

Tuple

u Python-u

```
empty_tuple = ()
print(empty_tuple)
```

```
()
```

```
int_tuple = (1, 2, 3)
print(int_tuple)
```

```
(1, 2, 3)
```

```
string_tuple = ("Zdravo!", "Kako", "si", "?")
print(string_tuple)
```

```
('Zdravo!', 'Kako', 'si', '?')
```

```
mix_tuple = ("Cao", 1, 3.4)
print(mix_tuple)
```

```
('Cao', 1, 3.4)
```

```
nested_tuple = ("Cao", [1, 2, 3], (5, 6, 7))
print(nested_tuple)
```

```
('Cao', [1, 2, 3], (5, 6, 7))
```

Tuple sa jednim elementom u Python-u

```
one_el_tuple = ("Hello")
print(one_el_tuple)
```

```
Hello
```

```
print(type(one_el_tuple)) ##!!!!
```

```
<class 'str'>
```

```
one_el_tuple_correct = ("Hello",)
print(one_el_tuple_correct)
```

```
('Hello',)
```

```
print(type(one_el_tuple_correct))
```

```
<class 'tuple'>
```

Tuple

u Python-u

```
languages = ("Python", "Swift", "C++", "C#")
# X code
```

```
for language in languages:
    print(language)
```

```
Python
Swift
C++
C#
```

Tuple

u Python-u

```
change_tuple[0] = 3
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_10380/1754832812.py in <module>
----> 1 change_tuple[0] = 3

TypeError: 'tuple' object does not support item assignment
```

Prednosti Tuple u odnosu na liste u Python-u

- Obično koristimo Tuple za heterogene tipove podataka i liste za homogene
- Pošto su Tuple nepromenljive, iteriranje kroz Tuple je brže, te postoji blago povecanje u performansama
- Elementi Tupla mogu se koristiti kao ključ za neki recnik, obzirom da su nepromenljive. Sa listama to nije bezbedno
- Ukoliko imate podatke koji su nepromenljivi, Tuple garantuje da su podaci zasticeni od ažuriranja

Okruženja za razvoj aplikacija

u Python-u

Jupyter notebook

i Python

- Koraci za instalaciju
  - Instalirati Python: https://www.python.org/downloads/windows/
  - U cmd-u ili PowerShell-u ukucati naredbu: pip install jupyter
  - Pokrenuti jupyter notebook naredbom: jupyter notebook

Lab 3 Osnove Pythona I

10.11.2022.

In [2]:
```
print("Hello World!")
print("Undo - proba")
```

```
Hello World!
Undo - proba
```

1. Osnovne karakteristike

Radi se o jeziku sa dinamičkom i strogom tipizacijom, gde ne postoji deklaracija promenljivih i promenljive mogu da uzimaju vrednosti različitog tipa.

Stroga tipizacija znači da se tip vrednosti neće promeniti na neočekivan način. Primer - string koji sadrži samo cifre ne može da postane broj. Dinamička tipizacija znači da runtime objekti imaju tip, nasuprot statičkoj tipizaciji gde promenljive imaju svoj tip.

Algoritmi
Mašinskog učenja

- Priprema podataka (Data Preprocessing)
- Deskriptivna analiza podataka
- Redukcija dimenzionalnosti
- Nadgledano učenje (klasifikacija i regresija)
- Nenadgledano učenje (klasterizacija)
- Veštačke neuronske mreže

Priprema podataka

Za algoritme Mašinskog učenja

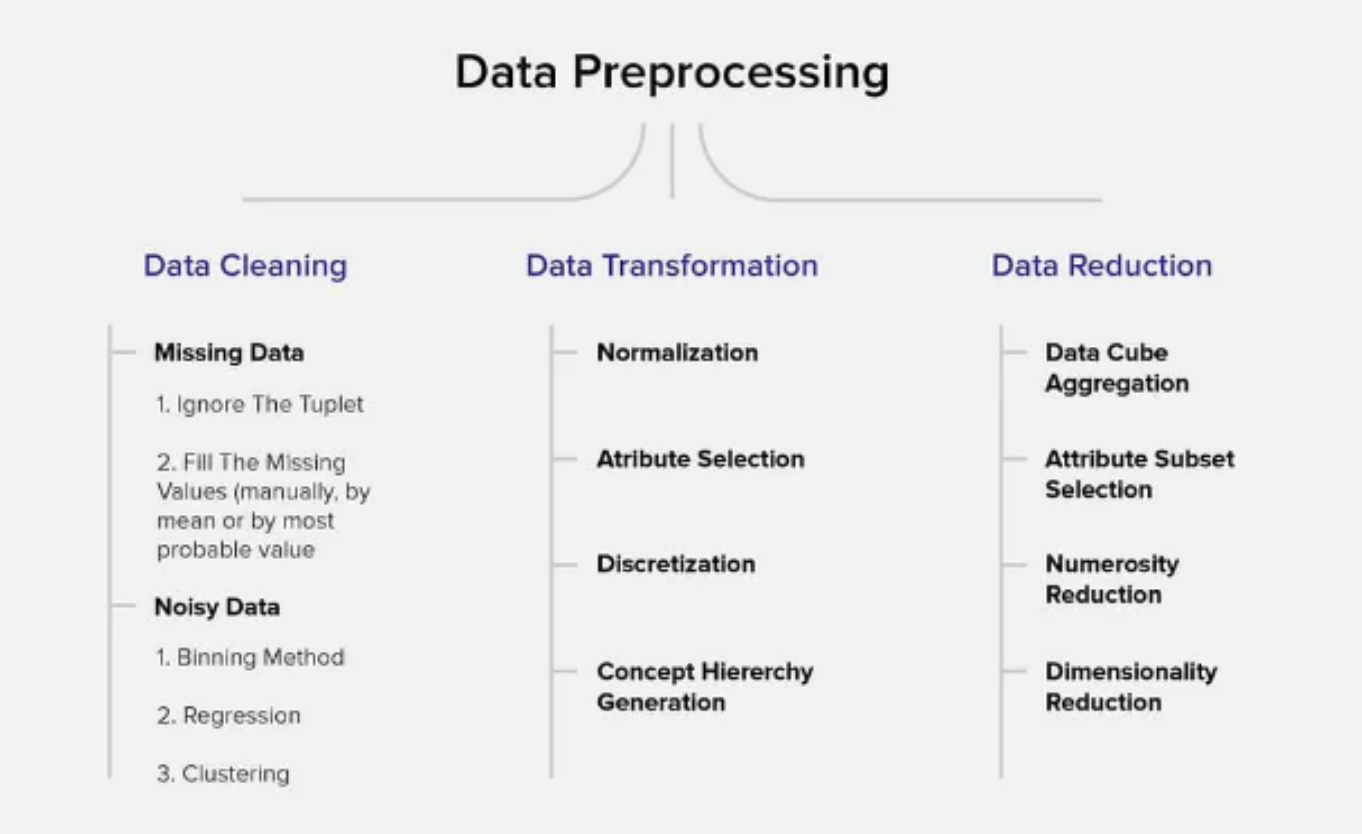
- Priprema podataka je prvi korak koji se primenjuje kada je potrebno primenjivati algoritme Mašinskog učenja

- Ova faza je naizostavna!
- Postoji čitav tim ljudi koji se u okviru projekta bavi samo ovom tematikom

- Podaci koji se prikupljaju su često vrlo složeni i nestruktuisani te je potrebno pažljivo ih primeniti kako bi bili pogodni za korišćenje

Priprema podataka

Za algoritme Mašinskog učenja



Deskriptivna analiza

Faza nakon preprocesiranja

- Deskriptivna analiza podataka (deskriptivna statistika) je naredni korak u pripremi podataka za algoritme mašinskog učenja
- Podrazumeva se upoznavanje sa prirodom podataka koji će se koristiti u algoritmima mašinskog učenja
- Od prirode podataka zavisi koji algoritmi se mogu primeniti i kakav će biti kvalitet dobijenih rezultata

Deskriptivna analiza

Faza nakon preprocesiranja

Statistics is the science of data

Deskriptivna analiza

Faza nakon preprocesiranja

Data is any collection of observations on a population of interest

Deskriptivna analiza

Faza nakon preprocesiranja

Statistics give us a concrete way to compare populations using numbers rather than ambiguous description

Deskriptivna analiza

Faza nakon preprocesiranja

- Kada se bavimo deskriptivnom analizom, treba da postavimo sebi sledeća pitanja:
  - *What does the middle of our data look like?*
  - *How much does my data vary?*

Deskriptivna analiza

Faza nakon preprocesiranja

- Deskriptivnu statistiku delimo u dve grupe mera:
  - Mere centralne tencencije (engl. Measures of central tendency)
  - Mere širenja (engl. Measures of spread)

Deskriptivna analiza

Measures of central tendency

- Sredina (engl. Mean)
- Medijana (engl. Median)
- Mode (Value that appears the most frequently in our data!)

- Primer: 15, 14, 13, 10
  - Mean: (15+14+13+10)/4
  - Median: 15, 14, 13, 10 -> 10, 13, 14, 15
    - (13+14)/2
- Primer: 15, 13, 15, 11, 12, 15, 15
  - Mode: 15

Deskriptivna analiza

Measures of central tendency

- Sredina (engl. Mean)
- Medijana (engl. Median)
- Mode

- ***OUTLIERS:***
  - Mean is not robust to outliers!
  - Median is robust to outliers!

Deskriptivna analiza

Measures of spread

- Range
  - price_range = max_price – min_price
- Standard deviation
  - *Standard deviation summaries how much your data differs from the mean*
  - *Što je veća standardna devijacija, podaci su više rasprostranjeni oko srednje vrednosti i obrnuto*
- Variance
  - *Standardna devijacija na kvadrat*
- ***Variance and Standard deviation are affected by the outliers***

Algoritmi Mašinskog učenja

Nadgledano učenje

- Cilj učenja je da se dobije takav model koji će dati dobre rezultate ne samo nad podacima koji su korišćeni pri obuci modela (trening skup) već i na kasnijim do tada neviđenim podacima koji nisu korišćeni prilikom obuke. Ovakva sposobnost modela naziva se sposobnost generalizacije

- Ispituje se dovođenjem skupa parova podataka (x, y) koji nisu korišćeni tokom obuke modela. Ovaj skup podataka naziva se skup podataka za testiranje modela ili kraće test skup

- Sposobnost generalizacije modela ispituje se dovođenjem skupa parova podataka (x, y) koji nisu korišćeni tokom obuke modela. Ovaj skup podataka naziva se skup podataka za testiranje modela ili kraće test skup

Algoritmi Mašinskog učenja

Nadgledano učenje

- Dataset = $\{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})\}$, $t = 1,2,...T$
  - gde je T ukupan broj uzoraka u test skupu

- U toku procesa testiranja modela ispituje se koliko model greši na svojim izlazima u odnosu na željene vrednosti za sve uzorke is test skupa koji se dovode na ulaz modela i na osnovu toga se ocenjuje stepen postignute generalizacije modela, odnosno kvalitet obuke modela

Algoritmi Mašinskog učenja

Klasifikacija

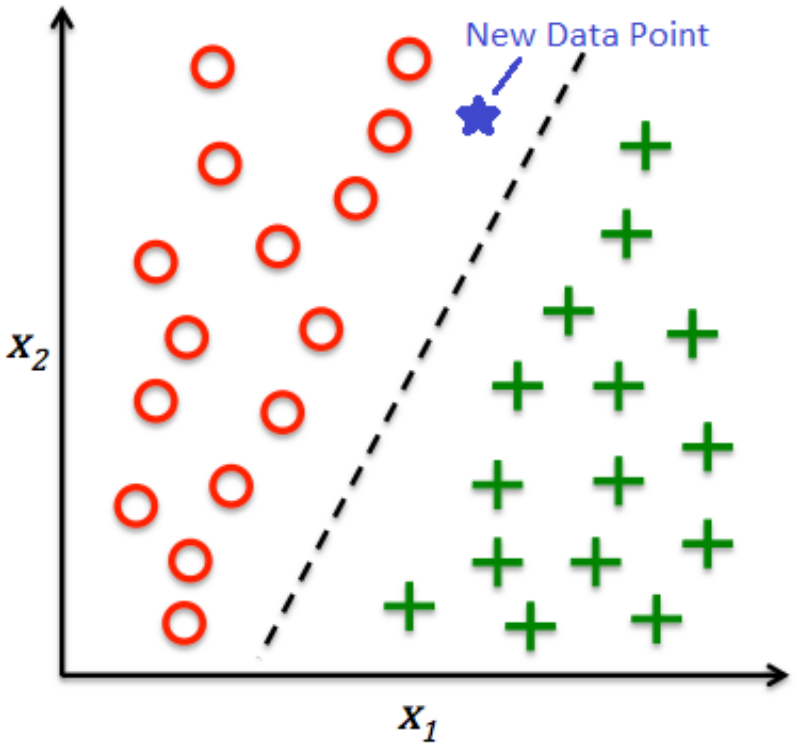
- Dataset
  - X skup atributa
  - Y klasni atribut

- Delimo na trening i test podatke
  - 50/50
  - 80/20
  - 70/30

Algoritmi Mašinskog učenja

Klasifikacija

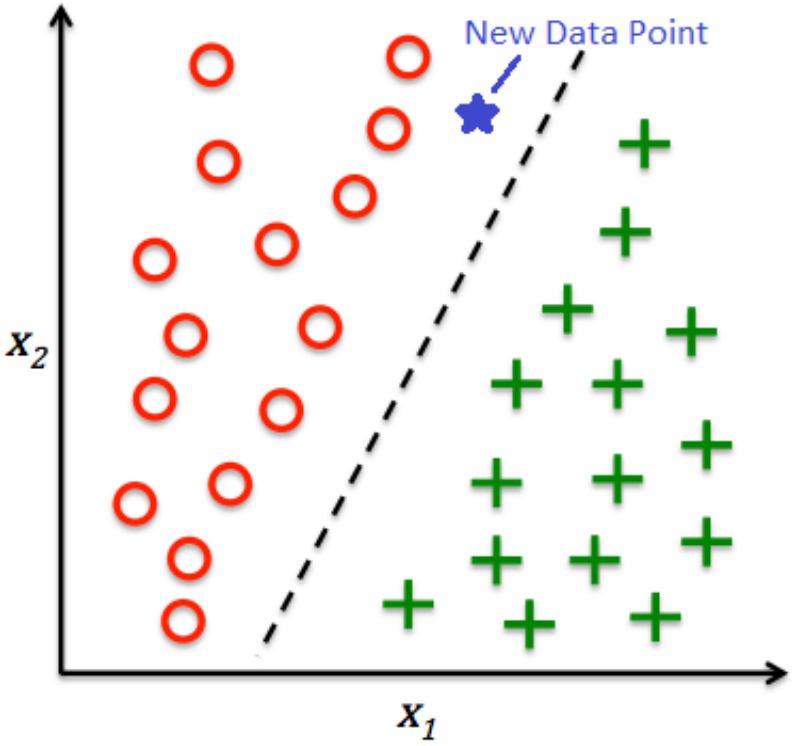
- Služi za predviđanje diskretnih vrednosti (kategorije).
- Da li je email spam ili ne
- Da li je na slici pas ili mačka
- Da li će vreme biti sunčano ili kišovito



Algoritmi Mašinskog učenja

Klasifikacija

- Navedite neki primer gde je moguće upotrebiti klasifikaciju



Algoritmi Mašinskog učenja

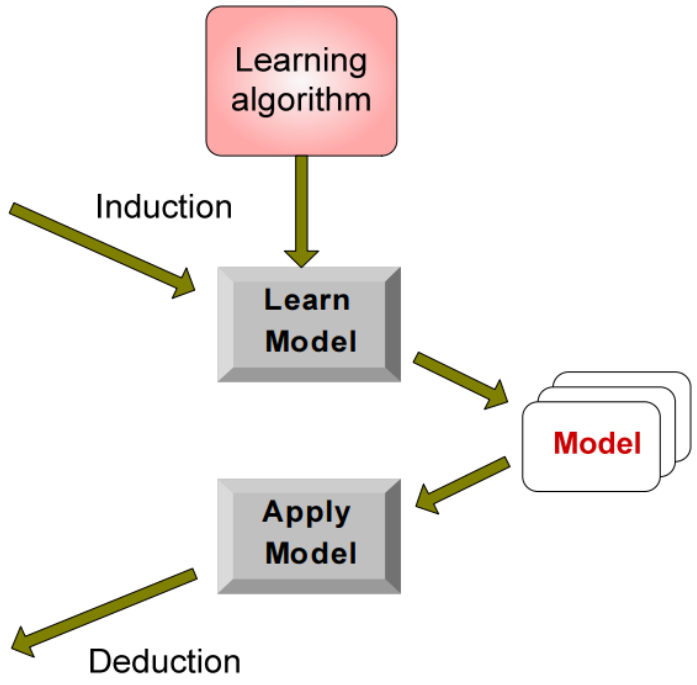
Klasifikacija

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

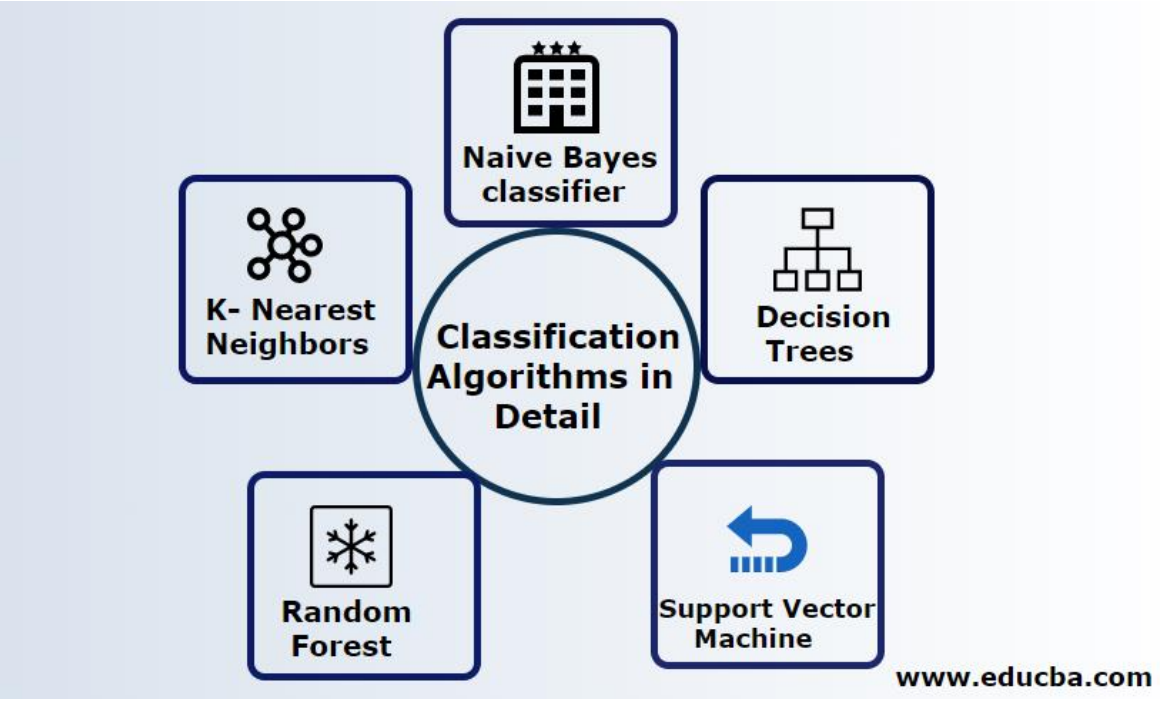
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Algoritmi Mašinskog učenja

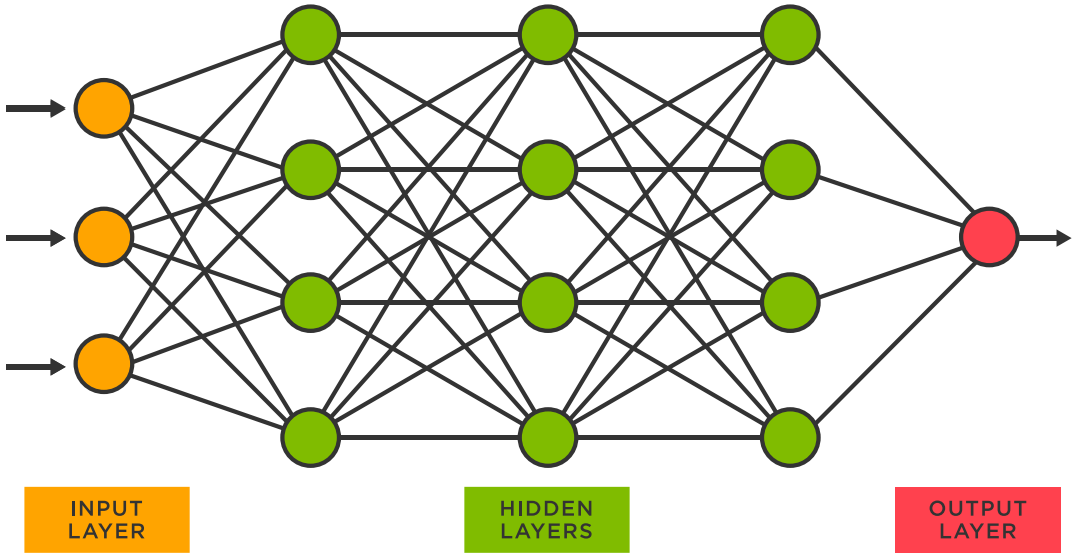
Klasifikacija



Algoritmi Mašinskog učenja

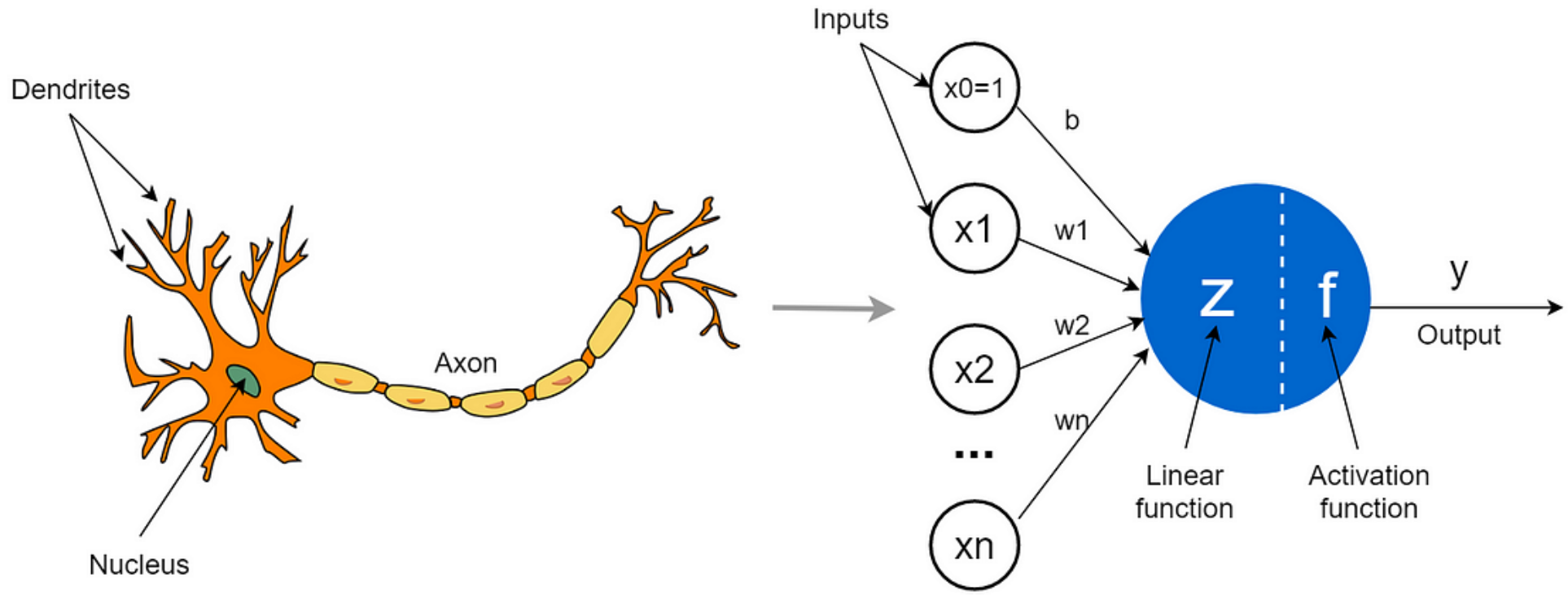
Veštačke neuronske mreže

- Veštačke neuronske mreže su računarski modeli inspirisani strukturom i funkcijom ljudskog mozga
- One se sastoje od velikog broja međusobno povezanih jedinica, nazvanih neuronske ćelije ili neuroni
- Svaki neuron prima ulazne signale od drugih neurona, obrađuje ih i zatim šalje izlazne signale drugim neuronima



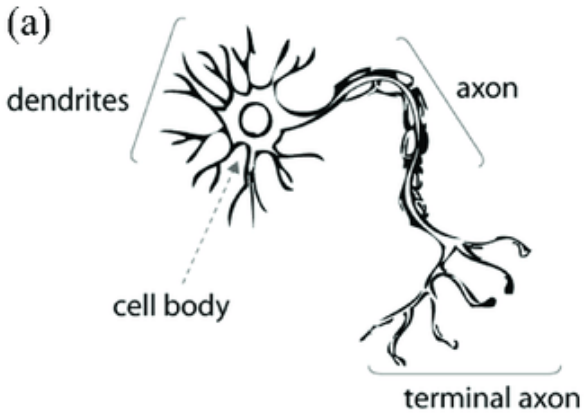
Algoritmi Mašinskog učenja

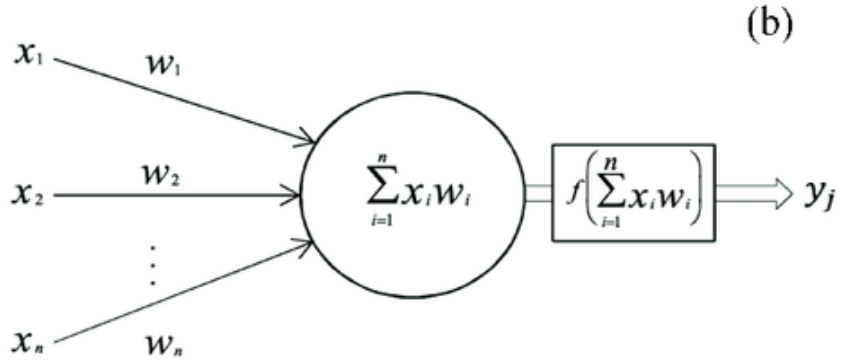
Veštačke neuronske mreže

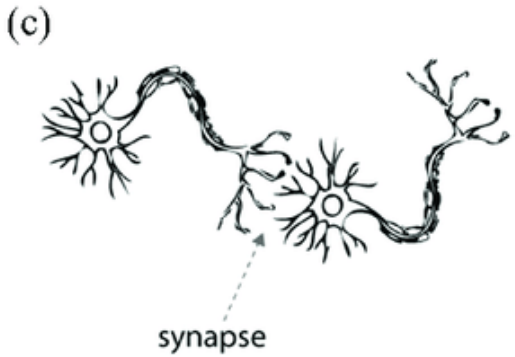


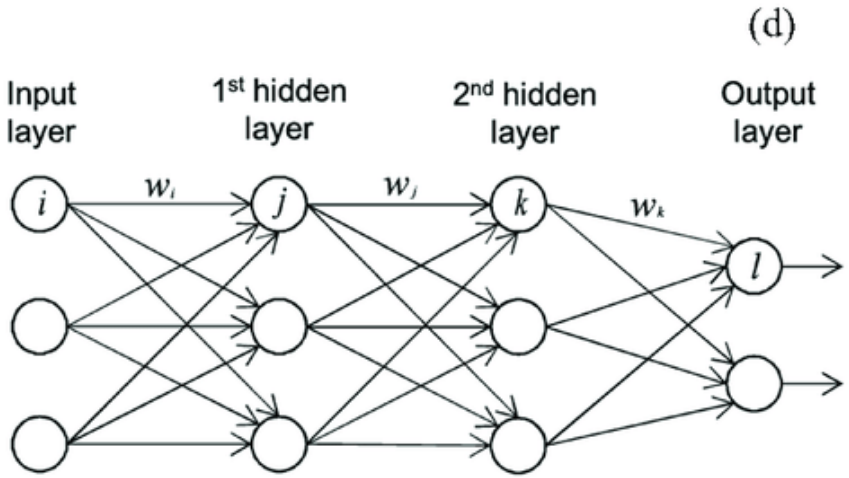
Algoritmi Mašinskog učenja

Veštačke neuronske mreže









Algoritmi Mašinskog učenja

Veštačke neuronske mreže

- Veštačke neuronske mreže koriste se u oblasti mašinskog učenja za rešavanje različitih problema, kao što su prepoznavanje slika, prevođenje jezika, preporučivanje proizvoda itd.

- One se obučavaju na velikim skupovima podataka, u procesu koji se naziva "učenje„

- Tokom ovog procesa neuronske mreže prilagođavaju svoje parametre tako da mogu da prepoznaju obrasce u podacima

Algoritmi Mašinskog učenja

Veštačke neuronske mreže

- Postoji mnogo vrsta veštačkih neuronskih mreža
  - potpuno povezane neuronske mreže,
  - konvolutivne neuronske mreže,
  - rekurentne neuronske mreže
  - neuronske mreže sa samoorganizujućim mapama

- Svaka od ovih vrsta neuronskih mreža ima svoje prednosti i nedostatke, i najbolje se koristi za različite vrste problema

Algoritmi Mašinskog učenja

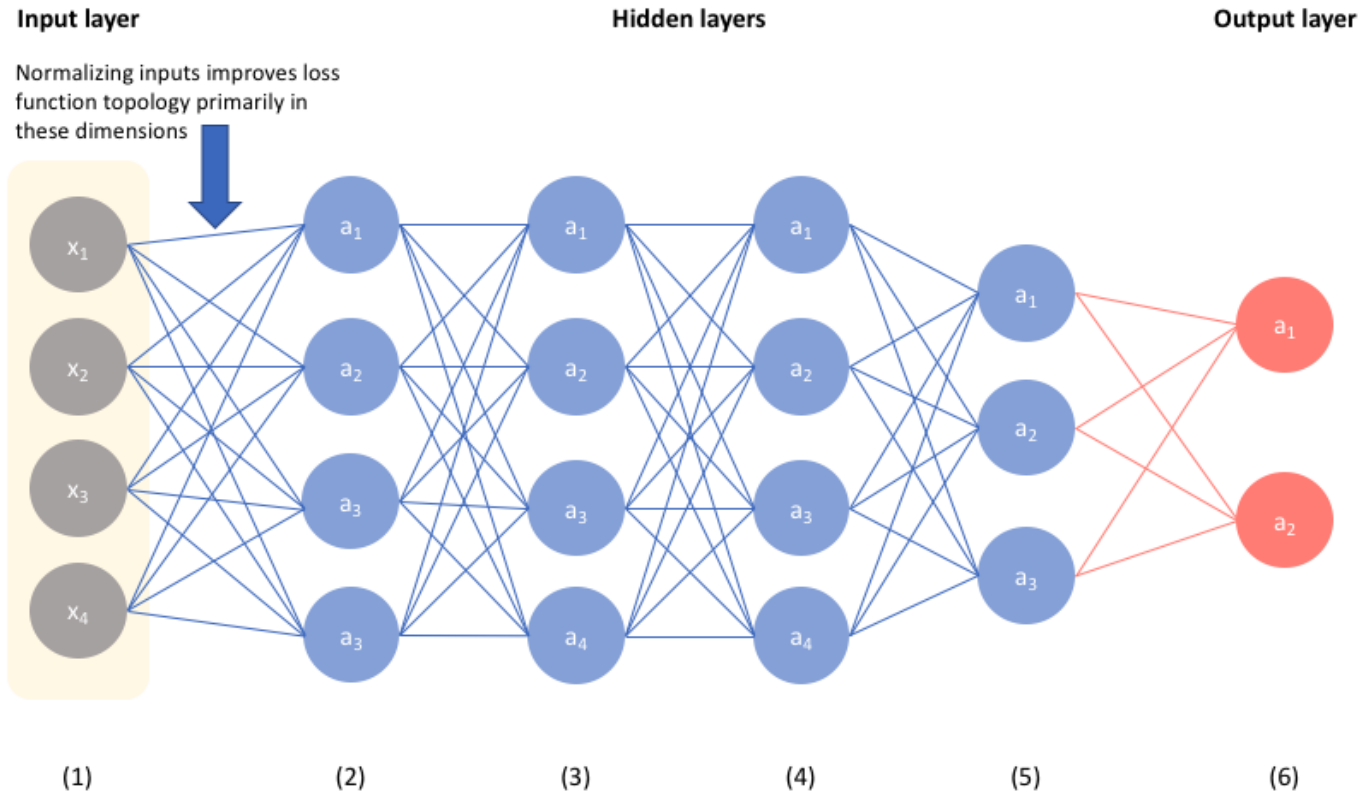
Ulaz u Veštačke neuronske mreže

- Ulaz u veštačku neuronsku mrežu (ANN) su podaci koje mreža prima na svom ulazu, a to mogu biti bilo kakvi podaci, kao što su slike, zvukovi, tekstovi ili bilo koja druga vrsta podataka
- Ovi ulazni podaci se obrađuju u neuronskoj mreži kako bi se proizveo odgovarajući izlaz
- Ulazni podaci se najčešće predstavljaju kao vektori, tj. kao nizovi brojeva koji opisuju karakteristike ulaznih podataka
- Na primer, ako želimo da koristimo neuronsku mrežu za prepoznavanje slika, ulazni podaci mogu biti vektori piksela slike, gde svaki piksel ima određenu vrednost koja opisuje boju tog piksela

Algoritmi Mašinskog učenja

Ulaz u Veštačke neuronske mreže

- Ulazni podaci se obično normalizuju pre nego što se proslede neuronskoj mreži
- Normalizacija podataka podrazumeva skaliranje vrednosti podataka na određenom opsegu, kako bi se postigla bolja performansa neuronske mreže
- Na primer, ako su ulazni podaci pikseli slike, oni se obično normalizuju tako da vrednosti piksela budu između 0 i 1



Algoritmi Mašinskog učenja

Ulaz u Veštačke neuronske mreže

- Nakon što se ulazni podaci normalizuju, oni se prosleđuju neuronskoj mreži
- Ulazni podaci se obrađuju u mreži pomoću niza matematičkih operacija koje se izvode na svakom neuronu u mreži
- Ove operacije omogućavaju mreži da "uči" iz ulaznih podataka i da prilagođava svoje parametre kako bi što bolje odgovarala ulaznim podacima

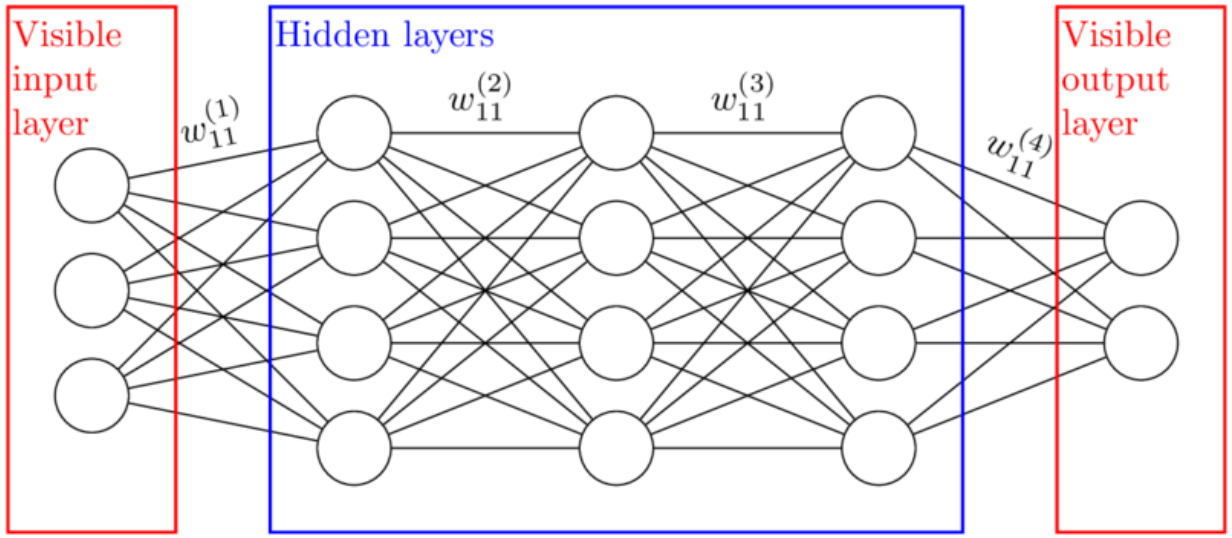
Algoritmi Mašinskog učenja

Skriveni slojevi Veštačke neuronske mreže

- Hidden layer-i (skriveni slojevi) su slojevi veštačke neuronske mreže (ANN) koji se nalaze između ulaznog i izlaznog sloja, a u kojima se obrađuju ulazni podaci kako bi se proizveo odgovarajući izlazni signal

- Skriveni slojevi se nazivaju "skriveni" jer se njihovi izlazi obično ne vide direktno - oni su skriveni od korisnika ili od spoljnih posmatrača

- Umesto toga, izlazi iz skrivenih slojeva se kombinuju na složen način kako bi se generisao izlazni signal

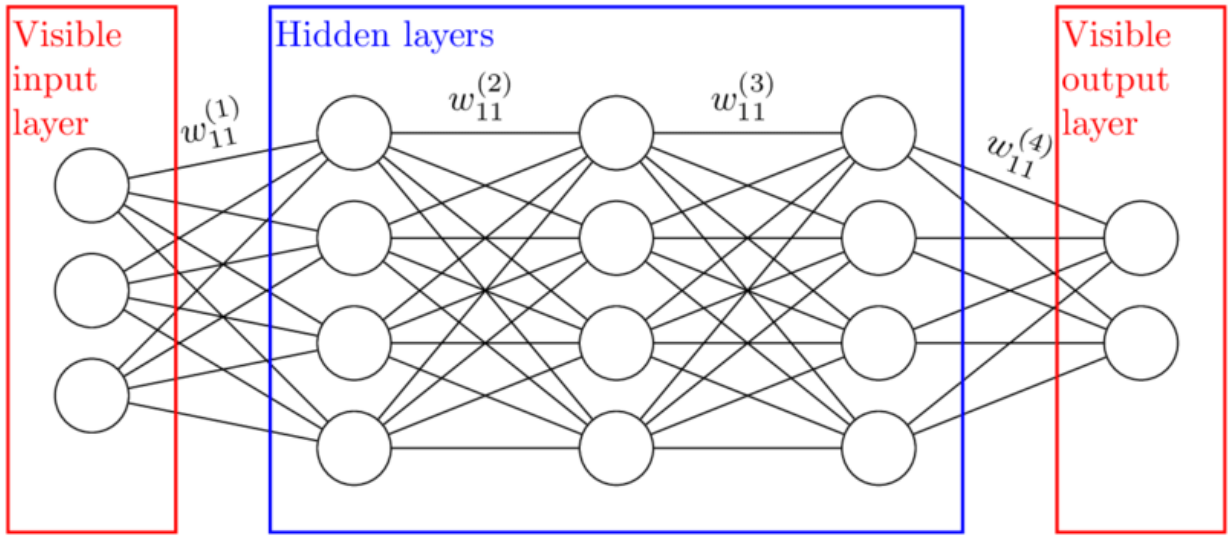


Algoritmi Mašinskog učenja

Skriveni slojevi Veštačke neuronske mreže

- Broj skrivenih slojeva koji se koriste u ANN-u može varirati od jednog do nekoliko desetina, u zavisnosti od složenosti problema koji se rešava

- Više skrivenih slojeva može pomoći ANN-u da nauči složenije funkcije i obrasce, ali takođe može dovesti do preteranog prilagođavanja (**overfitting**) trening podacima, što smanjuje sposobnost ANN-a da generalizuje i radi dobro na novim, neviđenim podacima

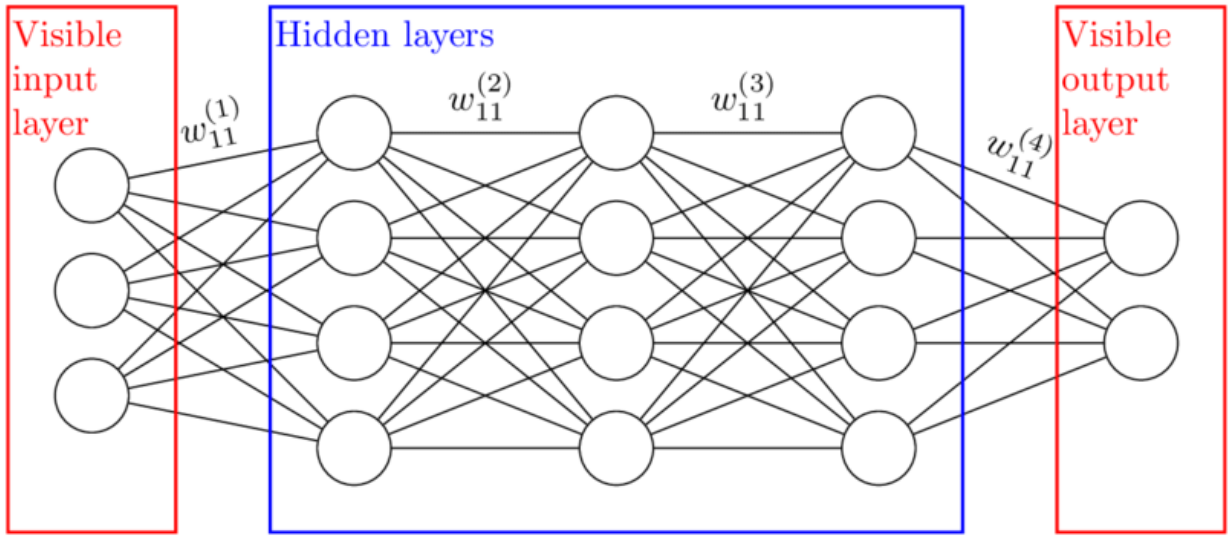


Algoritmi Mašinskog učenja

Skriveni slojevi Veštačke neuronske mreže

- Svaki skriveni sloj u ANN-u sastoji se od niza neuronskih jedinica koje obrađuju ulazne signale, kombinuju ih i proizvode izlazni signal

- Svaki neuron u sloju prima ulazne signale iz prethodnog sloja ili iz ulaznog sloja, obrađuje ih pomoću neke funkcije (npr. linearna kombinacija sa težinama, aktivaciona funkcija, itd.), i šalje izlazne signale na sledeći sloj ili izlazni sloj

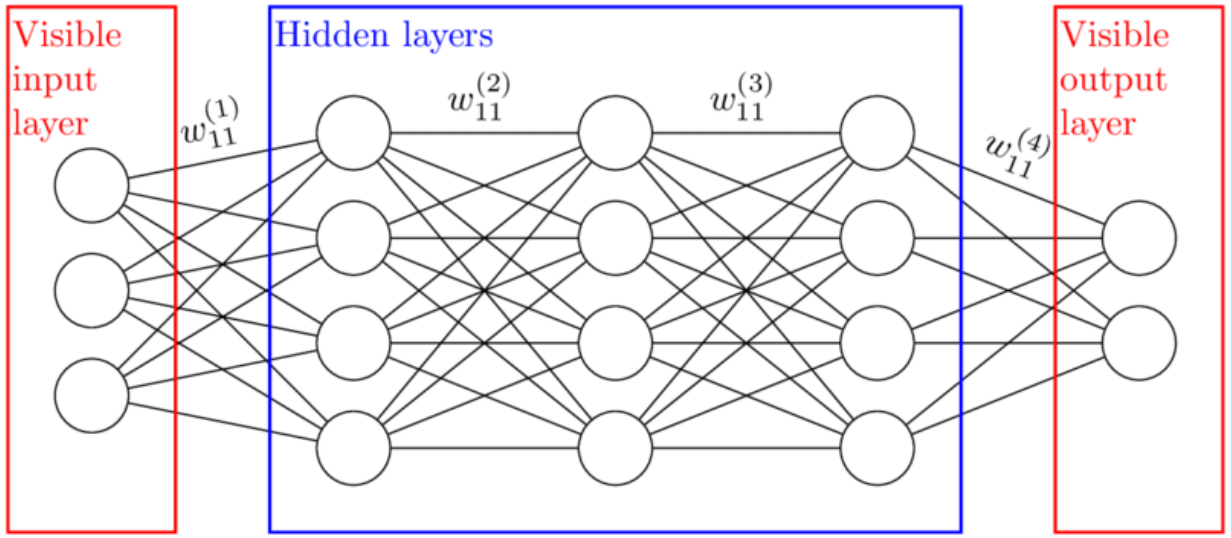


Algoritmi Mašinskog učenja

Skriveni slojevi Veštačke neuronske mreže

- Obrada ulaznih podataka u skrivenim slojevima ANN-a se obično izvodi u više koraka (iteracija), pri čemu se u svakom koraku koriste različite matematičke funkcije za obradu ulaza, kako bi se postigao optimalan rezultat

- Ovo se naziva "propagacija unapred" (forward propagation) i predstavlja jedan od osnovnih mehanizama koji se koriste za treniranje ANN-a



Algoritmi Mašinskog učenja

Izlazni sloj Veštačke neuronske mreže

- Izlazni sloj u veštačkim neuronskim mrežama (ANN) predstavlja poslednji sloj u mreži koji proizvodi izlazne signale na osnovu obrađenih ulaznih podataka

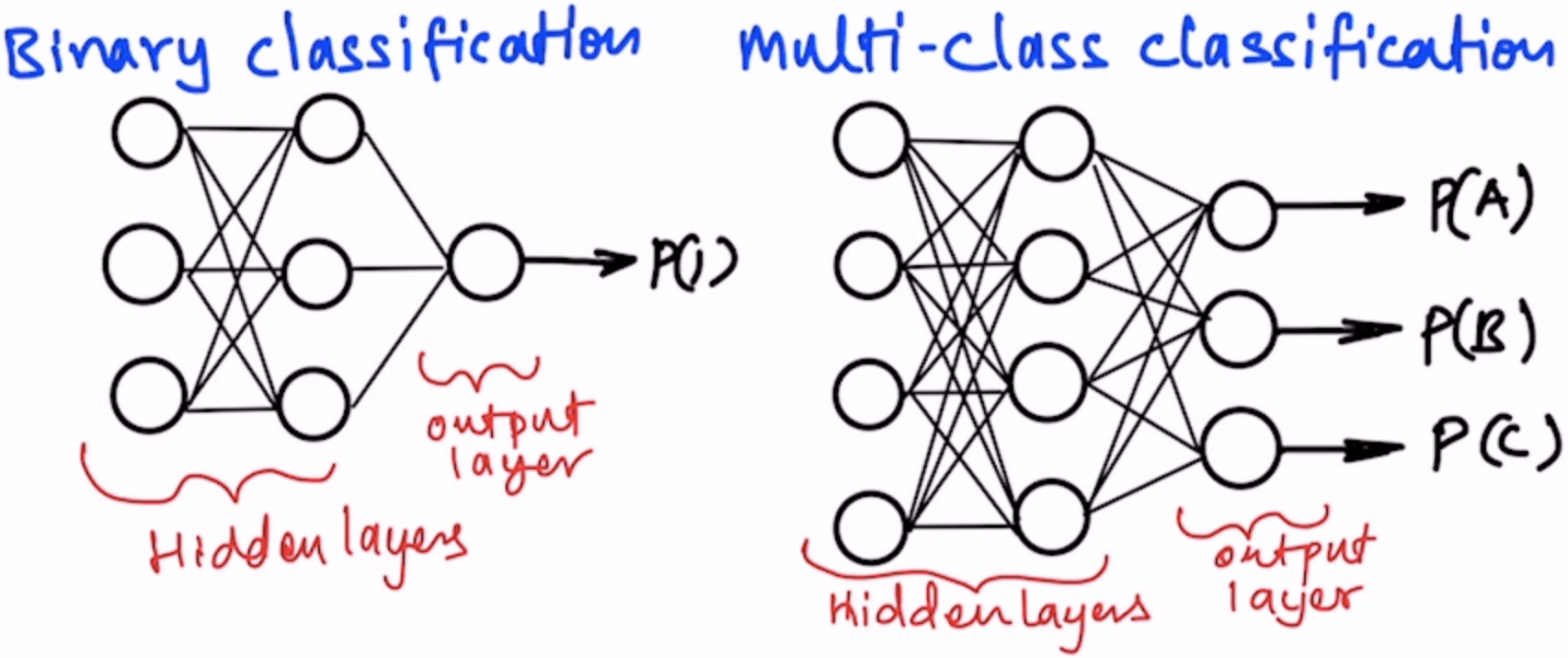
- Izlazni sloj se obično sastoji od jedne ili više neuronskih jedinica (neurona), a broj neurona u ovom sloju zavisi od prirode problema koji se rešava

- Na primer, za **binarnu klasifikaciju** (kada treba da se odluči da li je nešto prisutno ili odsutno), izlazni sloj bi se sastojao od jedne neuronske jedinice koja proizvodi izlazni signal od 0 ili 1

- Za **višeklasnu klasifikaciju**, izlazni sloj bi se sastojao od više neurona, gde svaki neuron predstavlja jednu klasu i proizvodi izlazni signal koji ukazuje na verovatnoću pripadnosti datoj klasi.

Algoritmi Mašinskog učenja

Izlazni sloj Veštačke neuronske mreže



Algoritmi Mašinskog učenja

Izlazni sloj Veštačke neuronske mreže

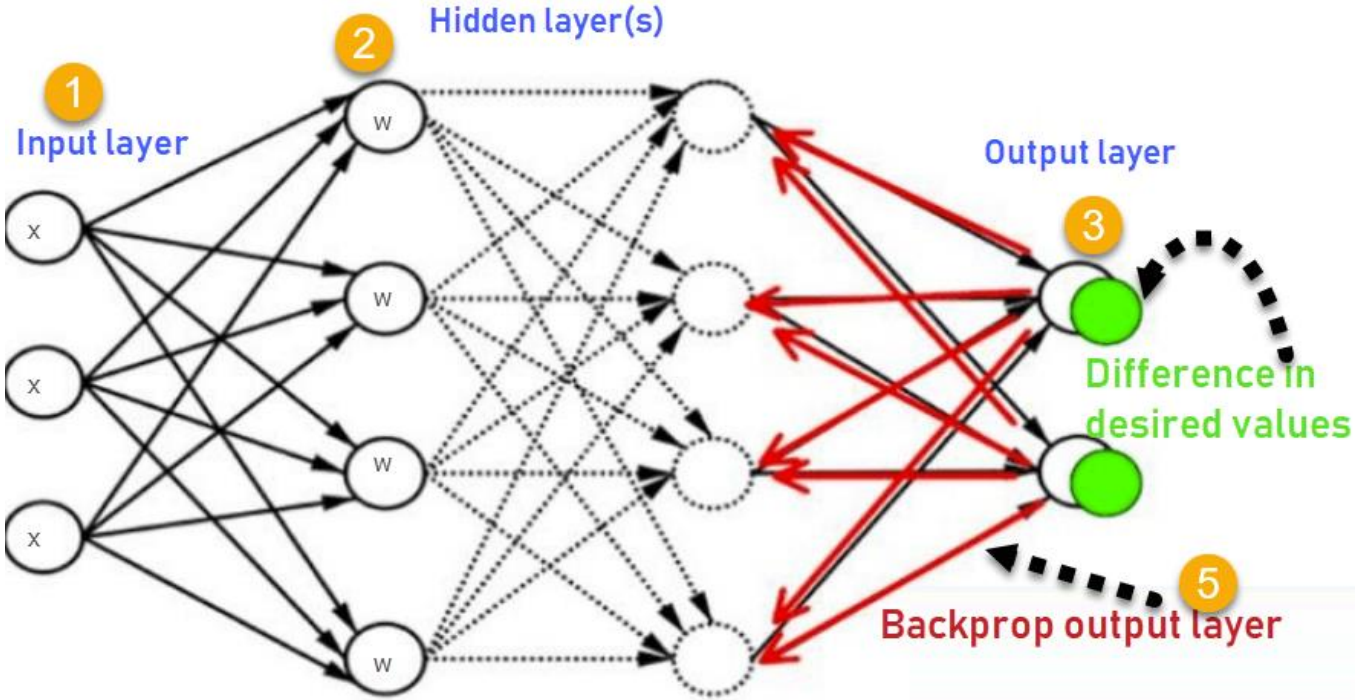
- U slučaju regresije, izlazni sloj bi se sastojao od jedne neuronske jedinice koja proizvodi kontinuiran izlazni signal koji predstavlja predikciju ciljne promenljive
- Izlazna vrednost ovog neurona obično je linearna kombinacija izlaza prethodnog sloja i odgovarajuće težine, bez aktivacione funkcije

Algoritmi Mašinskog učenja

Izlazni sloj Veštačke neuronske mreže

- Kada se ANN trenira, izlazni sloj se poredi sa ciljnim izlaznim podacima (koji su obično unapred poznati), a greška se računa kao razlika između ove dve vrednosti

- Ova greška se zatim propagira unazad (**backpropagation**) kroz mrežu kako bi se izmenile težine i poboljšala tačnost predikcije



Algoritmi Mašinskog učenja

Izlazni sloj Veštačke neuronske mreže

- U nekim slučajevima, izlazni sloj se može kombinovati sa aktivacionom funkcijom kako bi se postigao određeni oblik izlaznog signala
- Na primer, za binarnu klasifikaciju, izlazni sloj bi mogao da koristi sigmoidalnu aktivacionu funkciju kako bi se proizveo izlazni signal u opsegu 0-1, što bi se zatim moglo koristiti za odlučivanje o pripadnosti određenoj klasi

Hvala na pažnji!

nikola.vukotic@akademijanis.edu.rs