

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 7108

**WEB SUSTAV ZA DUBINSKU ANALIZU PODATAKA
TEMELJEN NA DUBOKIM NEURONSKIM MREŽAMA**

Marija Tišljar

Zagreb, lipanj 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 7108

**WEB SUSTAV ZA DUBINSKU ANALIZU PODATAKA
TEMELJEN NA DUBOKIM NEURONSKIM MREŽAMA**

Marija Tišljar

Zagreb, lipanj 2021.

ZAVRŠNI ZADATAK br. 7108

Pristupnica: **Marija Tišljar (0130309559)**

Studij: Računarstvo

Modul: Računarska znanost

Mentor: izv. prof. dr. sc. Alan Jović

Zadatak: **Web sustav za dubinsku analizu podataka temeljen na dubokim neuronskim mrežama**

Opis zadatka:

U ovom završnom radu potrebno je ostvariti web sustav za dubinsku analizu skupa podataka koji treba koristiti duboke neuronske mreže. Sustav se treba sastojati od: dijela za učitavanje datoteke sa skupom podataka, dijela za vizualizaciju osnovnih značajki, uključujući statistike za svaki atribut koji čini skup podataka, dijela za izbor između više metoda za pripremu podataka za analizu, barem sljedeće: uzorkovanje skupa podataka, uklanjanje određenog atributa, zamjenu nedostajućih vrijednosti, dijela za odabir metoda vrednovanja skupa podataka (npr. izdvajanje podskupa za testiranje u određenom postotku, unakrsnu validaciju, izgradnju modela na čitavom skupu), dijela za odabir cilja analize (klasifikacija, regresija), dijela za definiranje arhitekture duboke neuronske mreže uz mogućnost izbora između različitih algoritama za izgradnju mreže i optimizaciju, broja slojeva, aktivacijskih funkcija i funkcija gubitka, pokretanja analize sa zadanim parametrima i dijelom za prikaz rezultata analize ovisno o odabranom cilju analize u sučelju. Sustav na strani klijenta treba biti implementiran u radnom okviru ili knjižnici za tzv. single-page web aplikacije, kao što su React ili Vue.JS te treba imati mogućnost registracije i prijave korisnika. Izbor jezika na strani poslužitelja je proizvoljan.

Rok za predaju rada: 11. lipnja 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 7108

**Web sustav za dubinsku analizu
podataka temeljen na dubokim
neuronskim mrežama**

Marija Tišljar

Zagreb, lipanj 2021.

ZAVRŠNI ZADATAK br. 7108

Pristupnica: **Marija Tišljar (0130309559)**

Studij: Računarstvo

Modul: Računarska znanost

Mentor: izv. prof. dr. sc. Alan Jović

Zadatak: **Web sustav za dubinsku analizu podataka temeljen na dubokim neuronskim mrežama**

Opis zadatka:

U ovom završnom radu potrebno je ostvariti web sustav za dubinsku analizu skupa podataka koji treba koristiti duboke neuronske mreže. Sustav se treba sastojati od: dijela za učitavanje datoteke sa skupom podataka, dijela za vizualizaciju osnovnih značajki, uključujući statistike za svaki atribut koji čini skup podataka, dijela za izbor između više metoda za pripremu podataka za analizu, barem sljedeće: uzorkovanje skupa podataka, uklanjanje određenog atributa, zamjenu nedostajućih vrijednosti, dijela za odabir metoda vrednovanja skupa podataka (npr. izdvajanje podskupa za testiranje u određenom postotku, unakrsnu validaciju, izgradnju modela na čitavom skupu), dijela za odabir cilja analize (klasifikacija, regresija), dijela za definiranje arhitekture duboke neuronske mreže uz mogućnost izbora između različitih algoritama za izgradnju mreže i optimizaciju, broja slojeva, aktivacijskih funkcija i funkcija gubitka, pokretanja analize sa zadanim parametrima i dijelom za prikaz rezultata analize ovisno o odabranom cilju analize u sučelju. Sustav na strani klijenta treba biti implementiran u radnom okviru ili knjižnici za tzv. single-page web aplikacije, kao što su React ili Vue.JS te treba imati mogućnost registracije i prijave korisnika. Izbor jezika na strani poslužitelja je proizvoljan.

Rok za predaju rada: 11. lipnja 2021.

POPIS SLIKA

1.1. TensorFlow Playground [1]	2
2.1. Umjetna neuronska mreža [2]	3
2.2. Podjela skupa podataka [3]	4
4.1. Isječak koda konfiguriranja registracije i prijave	7
4.2. Uspješno ukrcavanje podataka u platformu	8
4.3. Implementacija zamjene nedostajućih vrijednosti	8
4.4. Implementacija odabira značajki	9
4.5. Podjela skupa podataka	9
4.6. Izdvajanje podskupa za testiranje u određenom postotku	10
4.7. Dio za definiranje arhitekture neuronske mreže	11
4.8. Isječak koda definiranja arhitekture neuronske mreže	11
4.9. Sažetak modela	12
4.10. Vizualizacija modela	13
4.11. Rezultati vrednovanja modela	13
4.12. Rezultati testiranja modela	13

SADRŽAJ

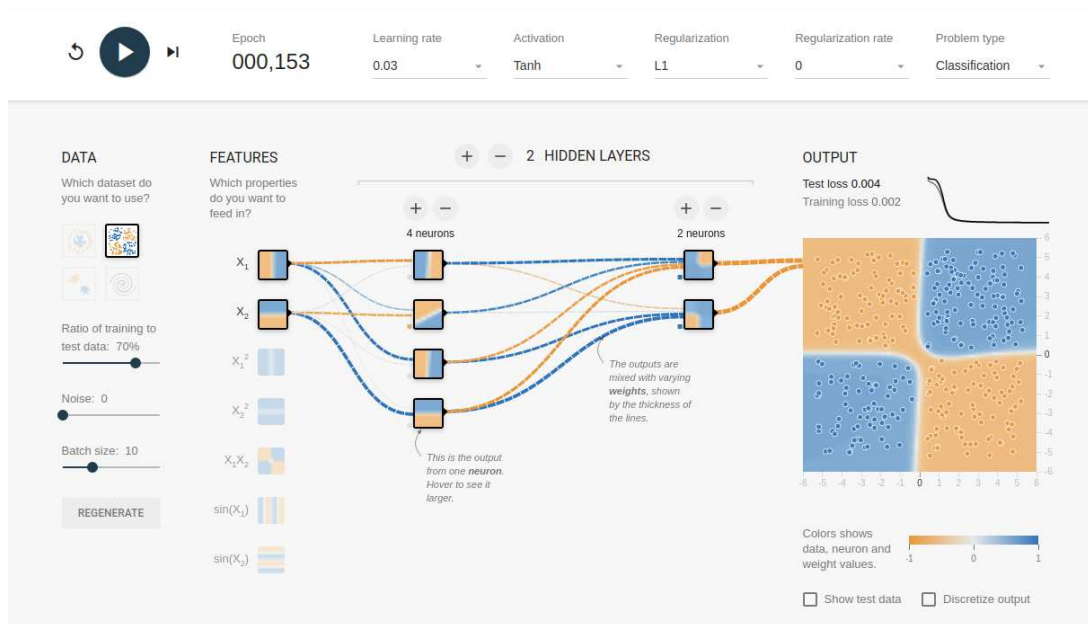
Popis slika	iv
1. Uvod	1
2. Neuronske mreže	3
3. Korištene tehnologije i alati	5
3.1. Vue.js	5
3.2. Flask	5
3.3. Scikit-learn	5
3.4. Keras	6
4. Implementacija	7
4.1. Registracija i prijava korisnika	7
4.2. Ukcavanje podataka u platformu	7
4.3. Metode za pripremu podataka	8
4.3.1. Zamjena nedostajućih vrijednosti	8
4.3.2. Uklanjanje određenog atributa	9
4.4. Odabir metode vrednovanja skupa podataka	9
4.5. Odabir cilja analize	10
4.6. Definiranje arhitekture neuronske mreže	10
4.7. Prikaz rezultata	12
5. Zaključak	14
Literatura	15

1. Uvod

Dubinska analiza podataka je relativno novo područje računarskih znanosti koje se razvija velikom brzinom i čija popularnost brzo raste. Proces dubinske analize podataka uključuje pripremu podataka, izvođenje računarskih postupaka analize te interpretaciju rezultata.

Web sustav za dubinsku analizu podatka temeljen na dubokim neuronskim mrežama omogućuje analiziranje podataka i jednostavan postupak izgradnje modela neuronske mreže bez potrebe za pisanjem koda i instaliranjem visokosloženih alata na lokalno računalo. Pristupačan je i omogućuje eksperimentiranje s različitim metodama analize i brz prikaz rezultata. Sustavu je moguće pristupiti s bilo koje lokacije i zahtjeva samo registraciju i prijavu. Potencijalni problem sustava bila bi pohrana osjetljivih podataka.

Aplikacija koja na sličan način omogućava eksperimentiranje s neuronskim mrežama je TensorFlow Playground. TensorFlow Playground je interaktivna web aplikacija otvorenog koda za vizualizaciju neuronskih mreža, slika 1.1. Omogućava kreiranje i učenje neuronske mreže bez potrebe za pisanjem koda. Neuronsku mrežu je moguće prilagoditi odabirom različitih opcija koje nam stoje na raspolaganju. Neke od opcija su odabir između četiri skupa podataka, izdvajanje skupa za testiranje u određenom postotku, izbor funkcije aktivacije, dodavanje novih skrivenih slojeva, odabir cilja analize i odabir stope učenja. Za vizualizaciju koriste se narančasta i plava boja. Općenito narančastom bojom su prikazane negativne vrijednosti a plavom pozitivne. U skrivenim slojevima linije su obojene i zadebljanje prema težinama veza između neurona. U izlaznom sloju točke su obojene narančasto ili plavo ovisno o izvornim vrijednostima. Boja pozadine pokazuje što mreža predviđa za to područje. Intenzitet boje pokazuje koliko je to predviđanje sigurno.[1] Aplikacija je odličan prvi korak kada želimo vizualizirati kako promjena neke postavke utječe na neuronsku mrežu.

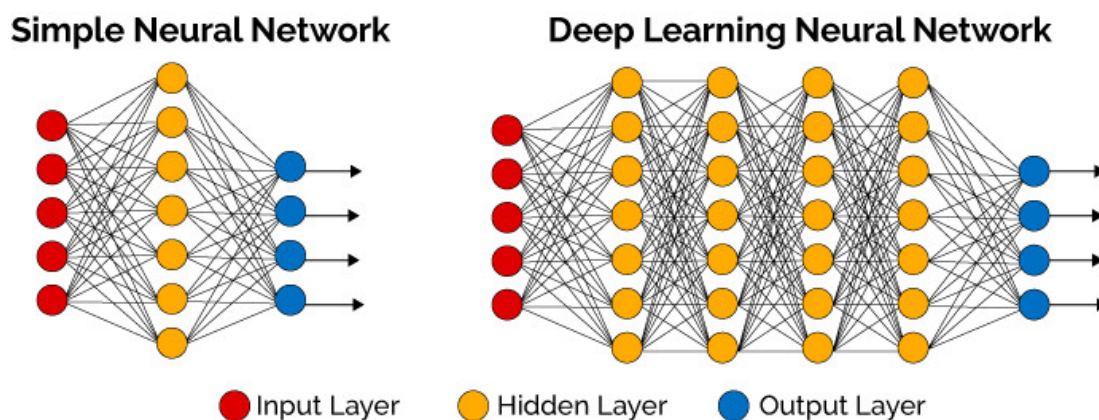


Slika 1.1: TensorFlow Playground [1]

2. Neuronske mreže

Umjetna neuronska mreža (engl. *Artificial neural network*, kraće: ANN) je skup međusobno povezanih neurona modeliran prema biološkim neuronskim mrežama koje čine naš mozak. Svaka veza između neurona može prenositi signal na druge neurone. Signal koji izlazi iz neurona računa se pomoću neke nelinearne funkcije zbroja njegovih ulaza.[4] Svakom neuronu i vezi dodjeljuje se težina koja povećava ili smanjuje jačinu signala koji njima prolazi.

Arhitektura neuronske mreže govori nam kako su neuroni međusobno povezani i koliko ih ima. Neuronske mreže koje imaju više skrivenih slojeva nazivaju se duboke neuronske mreže, slika 2.1.



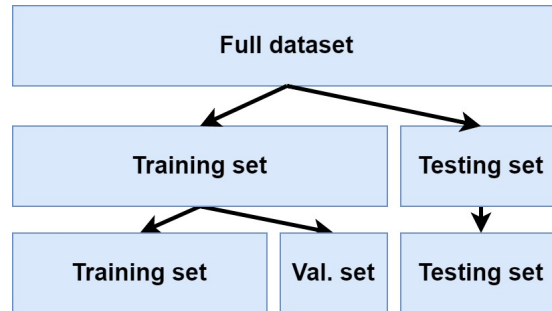
Slika 2.1: Umjetna neuronska mreža [2]

Učenje umjetne neuronske mreže je iterativan postupak predočavanja ulaznih primjera i eventualno očekivanog izlaza pri čemu dolazi do postupnog prilagođavanja težina između veza neurona.[5]

Učenje neuronskih mreža može biti:

- nadzirano (učenje na temelju označenih primjera)
- nenadzirano (učenje gdje nije poznata ciljna varijabla)
- podržano

Skup podataka za učenje može se podijeliti na skup za učenje, skup za provjeru i skup za testiranje, slika 2.2. Skup za učenje služi za iterativno podešavanje težina u mreži. Skup za provjeru služi da vidimo koliko dobro naša mreža generalizira nove primjere. Skup za testiranje je konačna provjera mreže i pomoću njega možemo raditi usporedbu s drugim modelima.



Slika 2.2: Podjela skupa podataka [3]

U nadzirano učenje spadaju klasifikacija i regresija. Klasifikacija je postupak razvrstavanja uzoraka u klase. Razvrstavanje uzoraka u samo dvije klase zovemo binarna klasifikacija. Primjer klasifikacije je kategorizacija biljne vrste. Regresija je postupak predviđanja ciljane numeričke vrijednosti. Primjer regresije je predviđanje kretanja cijena dionica.

3. Korištene tehnologije i alati

3.1. Vue.js

Vue je JavaScript radni okvir za izgradnju korisničkih sučelja i jednostranih web aplikacija. Jednostavan je za upotrebu i početnicima pristupačniji od Reacta i Angulara. To ga ne čini ništa manje moćnim jer nam pruža sve potrebne alate za razvoj modernih korisničkih sučelja. Lako ga je integrirati s postojećim projektima i knjižnicama.

3.2. Flask

Flask je internetski razvojni okvir napisan u Pythonu. Spada u "*mikrookvire*" zbog svoje jednostavne jezgre koja podržava brojna proširenja kao što su integracija baze podataka, testiranje komponenti, izrada obrazaca, prijenos podataka i isporuka REST-ful API web usluga. Flask korisniku ne nameće određene metode i vrlo je fleksibilan. Izvršna dokumentacija olakšava učenje te ga čini pristupačnim početnicima.

3.3. Scikit-learn

Scikit-learn je besplatna knjižnica za strojno učenje u Pythonu. Dizajnirana je za korištenje s numeričkim i znanstvenim knjižnicama NumPy i SciPy [7]. Pruža nam puno različitih algoritama za nadzirano i nenadzirano učenje. Neke od funkcionalnosti koje podržava su regresija, klasifikacija, grupiranje, priprema podataka i odabir modela. Sučelje je vrlo jednostavno i omogućuje laku usporedbu različitih metoda.

Scikit-learn dolazi s nekoliko standardnih skupova podataka kao što su *iris* i *digits* nad kojima možemo raditi klasifikaciju i *diabetes* nad koji možemo raditi regresiju.

3.4. Keras

Keras je knjižnica otvorenog koda za neuronske mreže napisana u Pythonu. Temelji se na TensorFlowu za kojeg nam pruža API visoke razine. Razvijen je za omogućavanje brzog eksperimentiranja i što bržeg prijelaza od ideje do rezultata. Neka od glavnih načela Kerasa su pristupačnost korisnicima, razumljivost koda i fleksibilnost.

Uz standardne neuronske mreže, Keras ima podršku za konvolucijske i rekurentne neuronske mreže.[8] Omogućava učinkovito izvršavanje operacija niske razine na CPU-u, GPU-u ili TPU-u.

Model je osnovna struktura podataka u Kerasu. Za stvaranje novih modela koristimo module koji nam omogućuju definiranje broja i vrste slojeva, aktivacijskih funkcija, funkcija gubitka i optimizaciju.

4. Implementacija

Ovo poglavlje opisuje implementaciju i glavne dijelove web sustava za dubinsku analizu podataka. Sustav je na strani klijenta implementiran u radnom okviru Vue.js kao jednostrana web aplikacija. Jezik na strani poslužitelja je Python.

Sustav ima mogućnost registracije i prijave korisnika.

4.1. Registracija i prijava korisnika

Login i registracija implementirani su pomoću paketa Flask-Login. Flask-Login omogućuje upravljanje korisničkim sjednicama za Flask. Podržava osnovne zadatke kao što su prijava, odjava i praćenje korisničkih sjednica tokom vremena. Za preslikavanje između objekata u kodu i relacijske baze (engl. *ORM*) koristimo Flask-SQLAlchemy. Kao bazu podataka koristimo SQLite, slika 4.1.

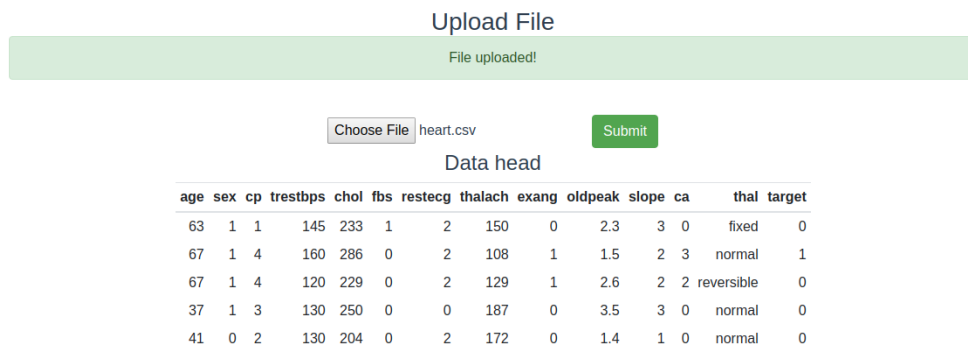
```
# instantiate the app
app = Flask(__name__)
app.config.from_object(__name__)
app.config['SECRET_KEY'] = 'r9EaRTD1vy72nPI'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///home/marija/PycharmProjects/Zavrsni/deep.db'
app.config['SECURITY_REGISTERABLE'] = True
app.config['SECURITY_RESET_URL'] = '/login'

# Create database connection object
db = SQLAlchemy(app)
```

Slika 4.1: Isječak koda konfiguriranja registracije i prijave

4.2. Ukcavanje podataka u platformu

Našu datoteku možemo odabrati pritiskom na gumb **Choose File** i učitati u sustav pritiskom na gumb **Submit**. Ukrcaj podataka radi se pomoću `<input>` elementa sa atributom `type="file"`. Ako je datoteka uspješno učitana dobijemo obavijest *File uploaded!*. Nakon ukrcaja podataka prikazuje se tablica s podacima kako bi mogli vidjeti listu značajki za naš podatkovni skup kao i prvih pet vrijednosti za svaku značajku, slika 4.2.



Slika 4.2: Uspješno ukrcavanje podataka u platformu

4.3. Metode za pripremu podataka

Biramo između nekoliko metoda za pripremu podataka.

4.3.1. Zamjena nedostajućih vrijednosti

Podatci s kojima radimo vrlo često sadrže nedostajuće vrijednosti koje mogu biti kodirane kao prazno polje ili NaN. Nad takvim podacima ne možemo koristiti Scikit-learnove transformacije jer one rade samo s numeričkim podacima. Jedna od strategija za rješavanje ovog problema bila bi uklanjanje cijelih redaka ili stupaca koji sadrže nedostajuće vrijednosti. Ova strategija je jednostavna ali nepoželjna jer riskiramo gubitak podataka koji nam mogu biti vrlo dragocjeni bez obzira na nedostajuće vrijednosti. Bolja strategija bi bila zamjena nedostajućih vrijednosti na temelju poznatog dijela podataka. Za taj zadatak koristimo razred **SimpleImputer**, slika 4.3. Ovaj razred koristi algoritam koji zamjenjuje nedostajuće vrijednosti koristeći samo vrijednosti te značajke. Kao strategiju zamjene odabiremo najčešću vrijednost.

Biblioteku u svoj programski kod uključujem slijedećom naredbom:

```
from sklearn.impute import SimpleImputer
```

```
def imputation_of_missing_values():
    imp = SimpleImputer(missing_values=np.nan, strategy='mean')
    ar = imp.fit_transform(df.values)
    return ar
```

Slika 4.3: Implementacija zamjene nedostajućih vrijednosti

4.3.2. Uklanjanje određenog atributa

Ponekad se može dogoditi da imamo previše značajki ili značajke koje ne koriste našem modelu. Postupci odabira značajki pomažu nam odabrati korisne značajke a izbaci štetne. Razredi u `sklearn.feature_selection` modulu nude nam brojne algoritme za odabir značajki.

VarianceThreshold uklanja sve značajke čija varijanca ne zadovoljava neki prag, slika 4.4. Prema zadanim postavkama također uklanja sve značajke koje imaju varijancu nula, tj. značajke čija je vrijednost ista u svim uzorcima.

Biblioteku u svoj programski kod uključujem slijedećom naredbom:

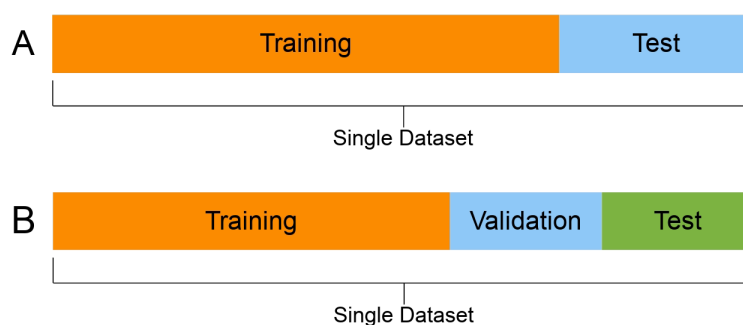
```
from sklearn.feature_selection import VarianceThreshold
```

```
def feature_selection(threshold):  
    sel = VarianceThreshold(threshold=(threshold))  
    ar = sel.fit_transform(df.values)  
    return ar
```

Slika 4.4: Implementacija odabira značajki

4.4. Odabir metode vrednovanja skupa podataka

Sustav nam nudi izbor između izgradnje modela na čitavom skupu podataka i izdvajanja podskupa za testiranje u određenom postotku, slika 4.6. Izdvajanje skupa za učenje u određenom postotku je dijeljenje našeg skupa podataka u skup za učenje i skup za vrednovanje, slika 4.5.



Slika 4.5: Podjela skupa podataka

Predloženi omjer je korištenje 80% skupa podataka za učenje i 20% za validaciju no korisnik sam može odabrati omjer.

Choose validation method

Use whole dataset

Ratio of training to test data: 80%



Slika 4.6: Izdvajanje podskupa za testiranje u određenom postotku

Implementiranje podjele skupa podataka napravljeno je koristeći argument **validation-split** metode `fit()` koju nam nudi Keras. Argument **validation-split** prima postotak skupa koji želimo ostaviti za testiranje modela.

4.5. Odabir cilja analize

Kao cilj analize možemo odabrati regresiju ili klasifikaciju.

Klasifikaciju odabiremo kada želimo svrstati primjere u određenu klasu. Za binarnu klasifikaciju sustav nam nudi izbor između tri funkcije gubitka:

- *Binary Cross-Entropy Loss*
- *Hinge Loss*
- *Squared Hinge Loss*

Kod klasifikacije u više razreda preporučena funkcija gubitka je *Cross-entropy*.

Regresiju odabiremo kada želimo predvidjeti ciljnu numeričku vrijednost. Oda-
birom regresije prikazuju nam funkcije gubitka koje možemo odabrati za naš model.

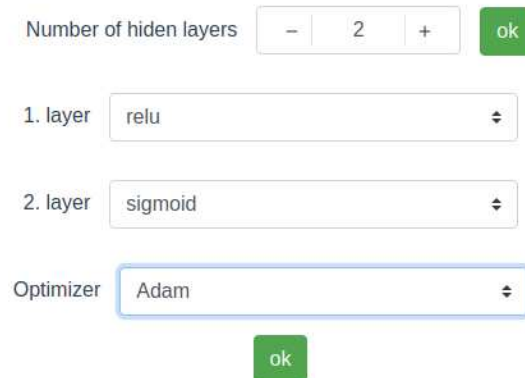
Funkcije gubitka koje sustav nudi za regresiju su:

- *MSE (Mean Squared Error Loss)*
- *MSLE (Mean Squared Logarithmic Error Loss)*
- *MAE (Mean Absolute Error Loss)*

4.6. Definiranje arhitekture neuronske mreže

Prilikom definiranja arhitekture neuronske mreže biramo kako su neuroni međusobno povezani i koliko ih ima.

Prvo određujemo broj unutarnjih slojeva te aktivacijsku funkciju za svaki od njih, kao



Number of hidden layers:

1. layer:

2. layer:

Optimizer:

Slika 4.7: Dio za definiranje arhitekture neuronske mreže

i optimizator, slika 4.7. Aktivacijske funkcije koje sustav nudi za skrivene slojeve su

- *ReLU (Rectified Linear Activation)*
- *Sigmoid (Logistic)*
- *Tanh (Hyperbolic Tangent)*

Možemo odabrati broj epocha i vrstu optimizatora. Optimizatori koje sustav nudi su:

- *Adam*
- *SGD (Stochastic gradient descent)*

Adam je preporučeni optimizator zbog malog zauzeća memorije i jer obično dobro funkcionira čak i uz malo podešavanja hiperparametara.

Nakon što smo odabrali opciju za svaki dio možemo napraviti model neuronske mreže, slika 4.8.

```
def build_model(nLayers, actFunctions, optimizer, loss):  
    model = keras.Sequential(name="my_sequential")  
    n = nLayers-1  
    for i in range(n):  
        model.add(layers.Dense(32, activation=actFunctions[i]))  
  
    model.add(layers.Dense(1))  
    model.compile(optimizer=optimizer, loss=loss, metrics=['accuracy'])  
    return model
```

Slika 4.8: Isječak koda definiranja arhitekture neuronske mreže

4.7. Prikaz rezultata

Nakon pokretanja analize prikazuje nam se sažetak našeg modela i rezultati.

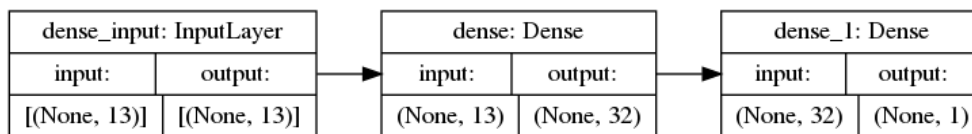
Sažetak modela, slika 4.9 sadrži podatke o slojevima i njihovom redosljedu u modelu, izlazu svakog sloja, broju parametara u svakom sloju i ukupnom broju parametara. Sažetak modela nam može biti koristan kod jednostavnijih modela, no kod kompleksnijih modela može biti zbunjujući pa tu možemo koristiti vizualizaciju modela, slika 4.10.

Rezultati pokazuju kolika je točnost za skup za učenje i skup za validaciju nakon svakog prolaska kroz podatke. Također prikazuju se točnost testiranja modela na skupu za testiranje, slika 4.11.

Model summary

```
Model: "my_sequential"
-----
Layer (type)                Output Shape                Param #
-----
dense (Dense)                (None, 32)                  448
-----
dense_1 (Dense)              (None, 1)                   33
-----
Total params: 481
Trainable params: 481
Non-trainable params: 0
-----
```

Slika 4.9: Sažetak modela



Slika 4.10: Vizualizacija modela

Results			
Epoch 1	Loss: 80.2614059,	Accuracy: 0.2644628,	Val_loss: 66.3686295, Val_accuracy: 0.3114754
Epoch 2	Loss: 66.7211990,	Accuracy: 0.2644628,	Val_loss: 54.1502266, Val_accuracy: 0.3114754
Epoch 3	Loss: 53.7243500,	Accuracy: 0.2644628,	Val_loss: 42.5201149, Val_accuracy: 0.3114754
Epoch 4	Loss: 40.8774338,	Accuracy: 0.2644628,	Val_loss: 31.1180725, Val_accuracy: 0.3114754
Epoch 5	Loss: 28.4373665,	Accuracy: 0.2644628,	Val_loss: 19.4838943, Val_accuracy: 0.3114754
Epoch 6	Loss: 15.4899893,	Accuracy: 0.2644628,	Val_loss: 7.0369983, Val_accuracy: 0.3114754
Epoch 7	Loss: 3.0114877,	Accuracy: 0.4586777,	Val_loss: 2.2792959, Val_accuracy: 0.6885246
Epoch 8	Loss: 2.5231826,	Accuracy: 0.7314050,	Val_loss: 3.9252729, Val_accuracy: 0.6885246
Epoch 9	Loss: 2.8211939,	Accuracy: 0.7355372,	Val_loss: 2.9782467, Val_accuracy: 0.6885246
Epoch 10	Loss: 1.7631516,	Accuracy: 0.7396694,	Val_loss: 1.1858230, Val_accuracy: 0.6721311
Epoch 11	Loss: 0.8358786,	Accuracy: 0.7024794,	Val_loss: 1.0549974, Val_accuracy: 0.5737705
Epoch 12	Loss: 0.7619929,	Accuracy: 0.7066116,	Val_loss: 0.9309990, Val_accuracy: 0.6393443
Epoch 13	Loss: 0.7057591,	Accuracy: 0.7438017,	Val_loss: 1.0355244, Val_accuracy: 0.6557377
Epoch 14	Loss: 0.6535415,	Accuracy: 0.7644628,	Val_loss: 0.8666046, Val_accuracy: 0.6393443
Epoch 15	Loss: 0.6311843,	Accuracy: 0.7396694,	Val_loss: 0.8695702, Val_accuracy: 0.6229508

Slika 4.11: Rezultati vrednovanja modela

Testing results	
Loss: 0.6032947301864624,	Accuracy: 0.7213114500045776

Slika 4.12: Rezultati testiranja modela

5. Zaključak

U ovom radu obrađena je implementacija web sustava za dubinsku analizu podataka korištenjem modernih knjižnica za strojno učenje. Web sustav približava dubinsku analizu podataka temeljenu na dubokim neuronskim mrežama korisnicima koji žele eksperimentirati bez instalacije i učenja visokosloženih alata. Omogućuje usporedbu rezultata za različite arhitekture neuronskih mreža čime može pomoći prilikom učenja o takvim temama.

Web sustav mogao bi se poboljšati automatiziranjem postupka odabira opcija za analizu tako da sustav sam preporuči koje opcije odabrati ovisno o cilju analize i skupu podataka. Također mogle bi se dodati neke dodatne opcije kao što su grupiranje kao cilj analize, vizualizacija rezultata, više metoda za filtriranje podataka.

Izrada ovog web sustava omogućila mi je upoznavanje s najvažnijim dijelovima dubinske analize podataka kao i savladavanje korištenja modernih alata i knjižnica za strojno učenje i izradu web aplikacija.

LITERATURA

- [1] TensorFlow Playground, 10.06.2021., <https://playground.tensorflow.org/>
- [2] Umjetna neuronska mreža, 10.06.2021., <https://hackernoon.com/log-analytics-with-deep-learning-and-machine-learning-20a1891ff70e>
- [3] How to create a train/test split for your Machine Learning model, 10.06.2021., <https://www.machinecurve.com/index.php/2020/11/16/how-to-easily-create-a-train-test-split-for-your-machine-learning-r>
- [4] Artificial neural network, Wikipedia, 04.06.2021., https://en.wikipedia.org/wiki/Artificial_neural_network
- [5] Dalbelo Bašić B. i Šnajder J. Čupić M. Umjetne neuronske mreže, Službeni materijali s predmeta Umjetna inteligencija, 04.06.2021., [https://www.fer.unizg.hr/_download/repository/UI_12_UmjetneNeuronskeMreze\[1\].pdf](https://www.fer.unizg.hr/_download/repository/UI_12_UmjetneNeuronskeMreze[1].pdf)
- [6] Vue.js Introduction, 02.06.2021., <https://vuejs.org/v2/guide/>
- [7] Scikit-learn: machine learning in Python, 03.06.2021., <https://scikit-learn.org/stable/>
- [8] Keras: the Python deep learning API, 03.06.2021., <https://keras.io/>
- [9] Keras tutorial: The Ulitimate Beginner's Guide to Deep Learning in Python, 03.06.2021., <https://elitedatascience.com/keras-tutorial-deep-learning-in-python>
- [10] Deep Learning in Keras - Building a Deep Learning Model, 03.06.2021., <https://stackabuse.com/deep-learning-in-keras-building-a-deep-learning-model>

- [11] Dalbelo Bašić B. i Šnajder J., Strojno učenje, 04.06.2021., https://www.fer.unizg.hr/_download/repository/StrojnoUcenje.pdf
- [12] Deep learning, Wikipedia, 04.06.2021., https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks
- [13] Developing a Single Page App with Flask and Vue.js, 04.06.2021., <https://testdriven.io/blog/developing-a-single-page-app-with-flask-and-vuejs/#alert-component>
- [14] FLask, User's Guide, 04.06.2021., <https://flask.palletsprojects.com/en/2.0.x/>
- [15] SQLAlchemy - The Python SQL Toolkit and Object Relational Mapper, 04.06.2021., <https://www.sqlalchemy.org/>

Web sustav za dubinsku analizu podataka temeljen na dubokim neuronskim mrežama

Sažetak

U ovom radu obrađena je implementacija web sustava za dubinsku analizu podatka temeljenog na dubokim neuronskim mrežama. Sustav omogućuje učitavanje datoteke sa skupom podataka, pripremu podataka, definiranje arhitekture duboke neuronske mreže i prikaz rezultata. Web sustav približava dubinsku analizu podataka temeljenu na dubokim neuronskim mrežama korisnicima koji žele eksperimentirati bez instalacije i učenja visokosloženih alata.

Ključne riječi: Dubinska analiza podataka, neuronske mreže, obrada podataka, strojno učenje, Keras, Scikit-learn .

Web System for Data Mining Based on Deep Neural Networks

Abstract

The theme of this thesis was developing a web system for data mining based on deep neural networks. The system supports uploading a file with a data set, data preparation, defining the architecture of a deep neural network and displaying the results. The web system makes data mining based on deep neural networks more approachable to users who want to experiment without installing and learning highly sophisticated tools.

Keywords: data mining, machine learning, neural networks, data processing, Keras, Scikit-learn.