**Klasifikácia datasetov scikit-learn**

(Zadanie č.3d)

**Mária Matušisková AIS ID: 116248**

**UI (Umelá inteligencia)**

**Cvičiaci: Ing. Adam Novocký (Št 14:00) ZS 2023/2024**

# Zadanie úlohy:

A white text on a black background

Description automatically generated

A screenshot of a graph

Description automatically generated

Table of Contents

[Zadanie úlohy: 2](#_Toc152021833)

[1 Realizácia zadanie, výber datasetu 4](#_Toc152021834)

[2 Dataset Iris 5](#_Toc152021835)

[3 Architektúra neurónovej siete 6](#_Toc152021836)

[4 Trénovacia a testovacia časť 7](#_Toc152021837)

[5 Experimentovanie s architektúrou siete 8](#_Toc152021838)

[6 Výsledky metrík 9](#_Toc152021839)

[7 Výhody a nevýhody použitých architektúr 10](#_Toc152021840)

[Záver 11](#_Toc152021841)

# Realizácia zadania, výber datasetu

Projekt bol realizovaný vo vývojárskom prostredí PyCharm

v programovacom jazyku Python na počítači MacBook Pro (13-inch, M1, 2020). V zadaní sa použil iba jeden súbor *main.py.*

Pri výbere vhodného datasetu pre projekt sa zohľadnilo niekoľko kritérií.

Ako je relevantnosť pre umelú inteligenciu a neurónky. Dataset Iris je široko využívaný, dobre dostupný a ľahký na pochopenie základov ako taká neurónová sieť funguje. Aké princípy využíva na rôzne matematické rovnice a funkcie. Využíva sa najme v oblasti štatistiky a aj strojového učenia. Je tiež veľmi dobre zdokumentovaný a dostupný v mnohých knižniciach.

Boli použité knižnice:

* **keras** – rozhranie (API) pre neurónové siete, trénuje ich a testuje
* **matplotlib** – vizualizácia dát, tvorba grafov
* **sklearn** – implementácia datasetu Iris (môže aj iné), vrátane tried a metód
* **tensorflow** – tvorba a trénovanie neurónových sieti, učí aj rôzne modely

A screen shot of a computer

Description automatically generated

Obrázok 1, import knižníc

# Dataset Iris

Tento dataset poskytuje komplexné informácie o troch rôznych druhoch kvetov (Setosa, Versicolor, Virginica). Tieto dáta obsahujú záznamy o dĺžke a šírke v meraní kališných a okvetných lístkov. Jeho história siaha už od roku 1936, vo vydaní jedného článku od Ronalda Aylmera Fishera s názvom Používanie viacerých meraní v taxonomických problémoch.

Údaje poskytujú morfologické vlastnosti kvetov, meraných v centimetroch, vrátane dĺžky a šírky lístia a tiež okvetných lístkov.

A close-up of a flower

Description automatically generated

Obrázok 2, zo stránky: [Machine Learning HD](https://machinelearninghd.com/iris-dataset-uci-machine-learning-repository-project/)

V kóde sú tieto atribúty rozdelené na SepalLengthCm, SepalWidthCm, PetalLengthCM, PetalWidthCm a príslušné triedy (Labels).

A screen shot of a computer

Description automatically generated

Obrázok 3, inicializácia datasetu Iris

A screen shot of a computer

Description automatically generated

Obrázok 4, výstup databázy 5-tich meraní

A screenshot of a computer

Description automatically generated

Obrázok 5, zobrazenie celej databázy Iris

# Architektúra neurónovej siete

Vybrali sa štyri rôzne architektúry neurónových sietí. Každá je reprezentovaná samostatným modelom.

* model\_one
* model\_two
* model\_three
* model\_four

Skladá sa z niekoľko vrstiev. Základná je vstupná vrstva a výstupná vrstva. Medzi nimi sa nachádzajú skryté vrstvy.

Skrytá vrstva

Výstupná vrstva

Vstupná vrstva

Každý model začína vstupnou vrstvou s 16 neurónmi, ktorá zodpovedá štyrom vstupným atribútom datasetu Iris. Aktivačná funkcia tejto vrstvy je nastavená na relu (Recified Linear Unit).

Skryté vrstvy obsahuj rôzny počet neurónov s aktivačnými funkciami relu.

V prípade vybraného datasetu výstupnou vrstvou budú vždy 3 neuróny. Pretože zodpovedajú počtu tried (setosa, versicolor, virginica). Aktivačná funkcia je nastavená na softmax, čo je vhodné pre viacnásobnú klasifikáciu.

Kompiláciu modelu využívajú modely stratovej funkciou kategoriálnej krížovej entropie (categorical\_crossentropy). Tiež používajú optimalizátor Adam s rýchlosťou učenia sa 0,001. Metrikou pri trénovaní je presnosť (accuracy).

Trénuje sa po 50 epochách, pretože dataset má dokopy 150 dát. Z toho 50 je pre jeden druh. Vývoj chyby a presnosti je sledovaný na trénovanej a validačnej sade.

Pre každú jednu sieť sú vytvorené grafy, ktoré vizualizujú vývoj chyby a presnosti počas jednotlivých epoch trénovania. Slúžia pre lepšie pochopenie danej problematiky.

# Trénovacia a testovacia časť

V tejto fáze implementácie sú dáta rozdelené pomocou funkcie *train\_test\_split* na:

* **testovací set –** 10% dát, vyhodnotí konečný výkon modelu po jeho úplnom natrénovaní, podľa neho sa potom počítajú aj metriky a porovnávanie predikcii s reálnymi hodnotami
* **trénovací set –** 80% dát, slúži na adaptáciu váh modelu počas učenia
* **validačný set –** 10% dát, poskytuje možnosť monitorovať výkon modelu na dátach, ktoré neboli použité počas trénovania, čím zabráni pretrénovaniu

A screen shot of a computer program

Description automatically generated

Obrázok 6, rozdelenie dát na sady

# Experimentovanie s architektúrou siete

Pre riešenie úlohy klasifikácie na datasete sa realizovala séria experimentácii s architektúrou neurónových sieti. Každý model od model\_one po model\_four bol vytvorený s rôznymi konfiguráciami vrstiev a počtom neurónov a cieľom preskúmať ich vplyv na výkonnosť modelu.

Experimentovalo sa aj s rôznym počtom skrytých vrstiev. Tieto sledovania sú určené na optimalizáciu a schopnosť sa adaptovať na komplexné vzory v dátach (kvetov Iris).

## Popis architektúr modelov

**model\_one** **–** základná architektúra s jednou skrytou vrstvou

A screen shot of a computer program

Description automatically generated

Obrázok 7, vytvorenie neurónovej siete model\_one

**model\_two –** rozšírená architektúra s viacerými skrytými vrstvami a zvýšeným počtom neurónov

A screenshot of a computer program

Description automatically generated

Obrázok 8, vytvorenie neurónovej siete model\_two

**model\_three –** architektúra s vyšším počtom neurónov v prvých dvoch vrstvách

A computer screen shot of a program

Description automatically generated

Obrázok 9, vytvorenie neurónovej siete model\_three

**model\_four –** architektúra s menším počtom neurónov v tretej skrytej vrstve

A screen shot of a computer program

Description automatically generated

Obrázok 10, vytvorenie neurónovej siete model\_four

## Vyhodnotenie a porovnanie

Každý model bol skompilovaný s optimalizačným algoritmom Adam a stratovou funkciou categorical\_crossentropy. Po trénovaní sa analyzovali vývoje presnosti a chyby v priebehu epoch pre trénovanie a validačné dáta.

Modely boli testované na testovacej vzorke a následne sa vyhodnotili metriky. Výsledky boli vizualizované v grafoch.

A screen shot of a computer program

Description automatically generated

Obrázok 11, funkcia na skompilovanie, vypísanie a testovanie dát

A screen shot of a computer program

Description automatically generated

Obrázok 12, testovacia funkcia

A computer screen with text and images

Description automatically generated

Obrázok 13, funkcie na vytvorenie výsledkov testov v grafe

# Výsledky metrík

Vizualizácia grafov jednotlivých modelov.

## Graf trénovacej a validačnej presnosti

Tento graf zobrazuje presnosť modelu počas trénovania a overovania v priebehu epoch. Presnosť indikuje percento správne klasifikovaných príkladov. Čím sú krivky bližšie, tým lepšie model generalizuje na nové dáta.

A graph of a graph

Description automatically generated

Obrázok 14, model\_one, test presnosti

A graph of a graph

Description automatically generated with medium confidence

Obrázok 15, model\_two, test presnosti

A graph with green and orange lines

Description automatically generated

Obrázok 16, model\_three, test presnosti

A graph with green and orange lines

Description automatically generated

Obrázok 17, model\_four, test presnosti

## Graf trénovacej a validačnej chyby

Grafické spracovanie chybovosti entropie (loss) modelov počas trénovania a overovania v priebehu epoch. Trénovacia časť môže identifikovať, ako rýchlo model konverguje, zatiaľ čo časť overovania pomáha odhaliť prípadné preučenie alebo nedoučenie.

A graph of loss of a model

Description automatically generated

Obrázok 18, model\_one, test chybovosti

A graph of loss of a model

Description automatically generated

Obrázok 19, model\_two, test chybovosti

A graph of loss of a model

Description automatically generated

Obrázok 20, model\_three, test chybovosti

A graph of loss of a model

Description automatically generated

Obrázok 21, model\_four, test chybovosti

## Vyhodnotenie na testovacích dátach

Po trénovaní sa modely otestovali na testovacích dátach a získala sa presnosť a strata. Taktiež sa vykonala analýza výsledkov pomocou klasifikačnej správy a matice. Táto analýza ukáže presnejšie štatistiky.

A screenshot of a computer

Description automatically generated

Obrázok 22, ukázanie vrstiev vo výpise modelu one

A screenshot of a computer

Description automatically generated

Obrázok 23, vypísanie výsledkov testovania modelu one

# Výhody a nevýhody použitých architektúr

Každá z navrhnutých architektúr neurónových sietí má svoje výhody a nevýhody, ktoré môžu byť zohľadnené pri rozhodovaný o ich použití.

* model\_one
  + výhody – jednoduchá architektúra s nízkou náročnosťou na zdroje
  + nevýhody – obmedzená schopnosť zložitejšie vzory v dátach
* model\_two
  + výhody – zvýšený počet neurónov a vrstiev pre hlbšiu reprezentáciu dát
  + nevýhody – zvýšená náročnosť na zdroje, môže vykazovať znaky preučenia
* model\_three
  + výhody – zmena počtu neurónov v rôznych vrstvách pre flexibilitu
  + nevýhody – zvýšená komplexita, ktorá môže viesť ku nadmernému trénovaniu
* model\_four
  + výhody – redukcia počtu neurónov pre efektívnejšie učenie
  + nevýhody – zvýšená komplexita, ktorá môže viesť k nadmernému trénovaniu

# Záver

Vo všeobecnosti platí, že čím väčšia a hlbšia architektúra je, tak môže mať lepšiu schopnosť naučiť sa komplexnejšie vzory. Avšak môže dochádzať ku nadmernému trénovaniu.

Naopak, jednoduchšie architektúry môžu byť rýchlejšie a menej náročné na zdroje. Ale nemusia zvládať náročnejšie úlohy.

Pri výbere architektúry je dôležité zvážiť kompromis medzi výkonom a náročnosťou na zdroje vzhľadom na konkrétnu úlohu a dataset.