Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Umelá inteligencia

Klasifikácia datasetov scikit-learn

Artúr Kozubov

Meno cvičiaceho: Ing. Ivan Kapustík

Čas cvičení: Št 18:00

Dátum vytvorenia: 22. 11 . 2023

## Zadanie úlohy (d) Problém 3. Klasifikácia datasetov scikit-learn

Nasej úlohou bolo vytvoriť sofistikovanú neurónovú sieť, ktorá bude schopná klasifikovať dáta zo známeho datasetu  
dostupného v knižnici scikit-learn.

## Implementačné prostredie

Program je vytvorenej v Python 3.10.11 a na správne fungovanie sa využíva nasledujúci knižnici:

* argparse
* tensorflow
* sklearn
* matplotlib.pyplot
* seaborn

### Vyber datasetu

Vybral som make\_blobs dataset z knižnice sklearn a vytvoril som 3 architektúry neurónových sietí.

Výber tohto súboru údajov vychádza z niekoľkých kľúčových dôvodov:

* Kontext úlohy:
  + Tento súbor údajov modeluje vlastnosti buniek, ktoré charakterizujú rakovinu prsníka. Rakovina prsníka je jedným z najčastejších typov rakoviny u žien a jej včasné odhalenie zohráva kľúčovú úlohu pri liečbe a prežívaní  
    pacientov.
* Binárna klasifikácia:
  + Dataset poskytuje možnosť vykonať binárnu klasifikáciu (benígny alebo malígny nádor), čo umožňuje  
    vytvoriť model na určenie povahy nádoru na základe jeho vlastností.
* Dobre definované charakteristiky:
  + Súbor údajov obsahuje dobre definované charakteristiky buniek, ako je polomer, textúra, obvod, plocha a ďalšie parametre, ktoré môžu byť dôležité pri určovaní typu nádoru.
* Výskumný záujem:
  + Analýza a vytváranie modelov na takomto súbore údajov umožňuje hlbšie pochopiť, ktoré bunkové charakteristiky môžu súvisieť s tým, či je nádor malígny alebo benígny.

## Priebeh programu

Program sa spustí, a pomocou operátora „--architecture“ je mozne nastaviť architecture neurnovej siete:

use:  
 python main.py [--architecture <1-3>](1)

Program bude vypisovať aktuálni hodnoty, ako:

* Aktuálnu epochu (strata, presnosť pre kazdy epoch)
* A report áž na konci

A po, ukončeniu zobrazí metriky, aj graficky:

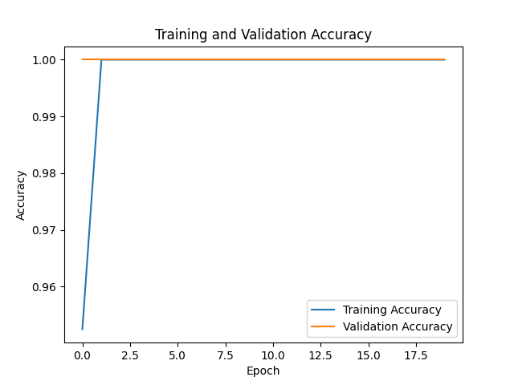
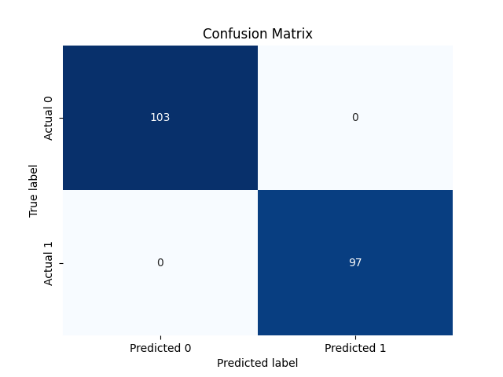
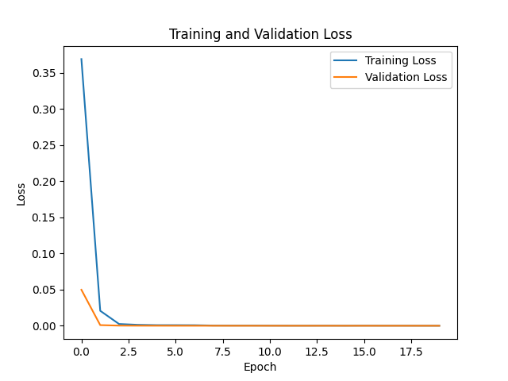
Classification Report:  
 precision recall f1-score support

0 1.00 1.00 1.00 103

1 1.00 1.00 1.00 97  
  
 accuracy 1.00 200

macro avg 1.00 1.00 1.00 200

weighted avg 1.00 1.00 1.00 200



* Presnosť testu:
  + Vypočíta sa pre každú architektúru po vyškolení a vyhodnotení na testovacej množine.
* Správa o klasifikácii:
  + Uvádza presnosť, odvolanie, skóre F1 a podporu pre obe triedy.
* Matica zmätočnosti:
  + Visualize pravdivé pozitívne, falošne pozitívne, pravdivé negatívne a falošne negatívne predpovede modelu.

## Architektúry

### Prvá

* Dve skryté vrstvy
  + Vrstva 1: 128 neurónov, ReLU aktivácia, 30% výpadok
  + Vrstva 2: 64 neurónov, aktivácia ReLU, 30 % výpadok
* Výstupná vrstva: 1 neurón, Sigmoid aktivácia (pre binárnu klasifikáciu)

Táto architektúra má menej vrstiev a neurónov, čo môže viesť k nedostatočnému naučeniu modelu. Má tendenciu k jednoduchším modelom a nemusí byť schopná zvládnuť učenie zložitých vzťahov v údajoch. Môže však aj mať tendenciu stabilnejšie zovšeobecňovať nové údaje, čím sa vyhne nadmernému učeniu.

model = tf.keras.Sequential([  
 tf.keras.layers.Dense(128, input\_shape=(X\_train.shape[1],), activation='relu'),  
 tf.keras.layers.Dropout(0.3),  
 tf.keras.layers.Dense(64, activation='relu'),  
 tf.keras.layers.Dropout(0.3),  
 tf.keras.layers.Dense(1, activation='sigmoid')  
])

### Druha

* Tri skryté vrstvy
  + Vrstva 1: 256 neurónov s lineárnou aktiváciou a 40% výpadkom
  + Vrstva 2: 128 neurónov s lineárnou aktiváciou a 40 % výpadkom
  + Vrstva 3: 64 neurónov s lineárnou aktiváciou a 40 % výpadkom
* Výstupná vrstva: Zachovaný jeden neurón so sigmoid aktivačnou funkciou.

Použitie lineárnej aktivácie vo všetkých skrytých vrstvách môže obmedziť schopnosť modelu získať komplexné nelineárne závislosti v údajoch. To môže viesť k strate schopnosti zovšeobecňovania a zhoršeniu výkonnosti modelu na nových  
údajoch.

model = tf.keras.Sequential([  
 tf.keras.layers.Dense(256, input\_shape=(X\_train.shape[1],), activation='linear'),  
 tf.keras.layers.Dropout(0.4),  
 tf.keras.layers.Dense(128, activation='linear'),  
 tf.keras.layers.Dropout(0.4),  
 tf.keras.layers.Dense(64, activation='linear'),  
 tf.keras.layers.Dropout(0.4),  
 tf.keras.layers.Dense(1, activation='sigmoid')  
])

### Tretia

* Tri skryté vrstvy
  + Vrstva 1: 64 neurónov, ReLU aktivácia, 20% výpadok
  + Vrstva 2: 128 neurónov, ReLU aktivácia, 30 % výpadok
  + Vrstva 3: 256 neurónov, ReLU aktivácia, 30% výpadok
* Výstupná vrstva: 1 neurón, Sigmoid aktivácia (pre binárnu klasifikáciu)

Táto architektúra predstavuje zložitejší model s rôznymi kombináciami vrstiev, aktivačných funkcií a vypadávania. Takéto siete majú väčší potenciál naučiť sa zložitejšie závislosti v údajoch, ale môžu trpieť nadmerným učením na trénovaných  
údajoch.

model = tf.keras.Sequential([  
 tf.keras.layers.Dense(64, input\_shape=(X\_train.shape[1],), activation='relu'),  
 tf.keras.layers.Dropout(0.2),  
 tf.keras.layers.Dense(128, activation='relu'),  
 tf.keras.layers.Dropout(0.3),  
 tf.keras.layers.Dense(256, activation='relu'),  
 tf.keras.layers.Dropout(0.3),  
 tf.keras.layers.Dense(1, activation='sigmoid')  
])

Tieto architektúry sa líšia počtom skrytých vrstiev, počtom neurónov v každej vrstve, použitými aktivačnými funkciami  
(ReLU, sigmoid, linear) a mierou vypadúvania použitou na regularizáciu.

## Pomer testovacích a školiacich údajov

Na rozdelenie súboru údajov na tréningovú a testovaciu množinu bol zvolený pomer 80:20:

# Split the data into training and testing sets  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(data, labels, test\_size=0.2, random\_state=42)

Tento pomer zabezpečuje významnú časť údajov na trénovanie modelu a zároveň zachováva značný samostatný súbor na vyhodnotenie výkonnosti modelu. Pomáha predchádzať nadmernému prispôsobeniu tým, že poskytuje dostatok údajov na  
zovšeobecnenie.