НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”
Факультет прикладної математики
Кафедра прикладної математики

ЛАБОРАТОРНА РОБОТА

з дисципліни “Системи глибинного навчання”

на тему: “ Нейромережеве розпізнавання кібератак”

Керівник:
Терейковський І. А.

Студентки IV курсу, групи КМ-03
Пюстонен С.Р.

Київ – 2023

ЗМІСТ

ВСТУП

Постановка задачі

Завдання: розробка програмного забезпечення для реалізації нейронної мереж і PNN, призначеної для розпізнавання кібератак (**smurf** зокрема**)**, сигнатури яких представлено в базі даних KDD-9.

Описати:

- Характеристику вибірки, що використовується для навчання та тестування НМ (джерело даних, приклади вибірки, їх тип, вхідні, вихідні параметри, процедура нормалізації вхідних параметрів);
- Реалізацію розробленого модуля (алгоритм, скріншот інтерфейсу програми);
- Результати експериментальних досліджень (як проводили навчання, на якому комп'ютері, термін навчання, результати розпізнавання).

2Теоретична частина

PNN, або Probabilistic Neural Network (ймовірнісна нейронна мережа), це тип нейронної мережі, який використовує ймовірнісні методи для розпізнавання та класифікації даних. PNN особливо корисна в задачах класифікації, де важливо враховувати невизначеність та ймовірність прийняття рішення. PNN широко використовується в обробці образів та розпізнаванні образів.

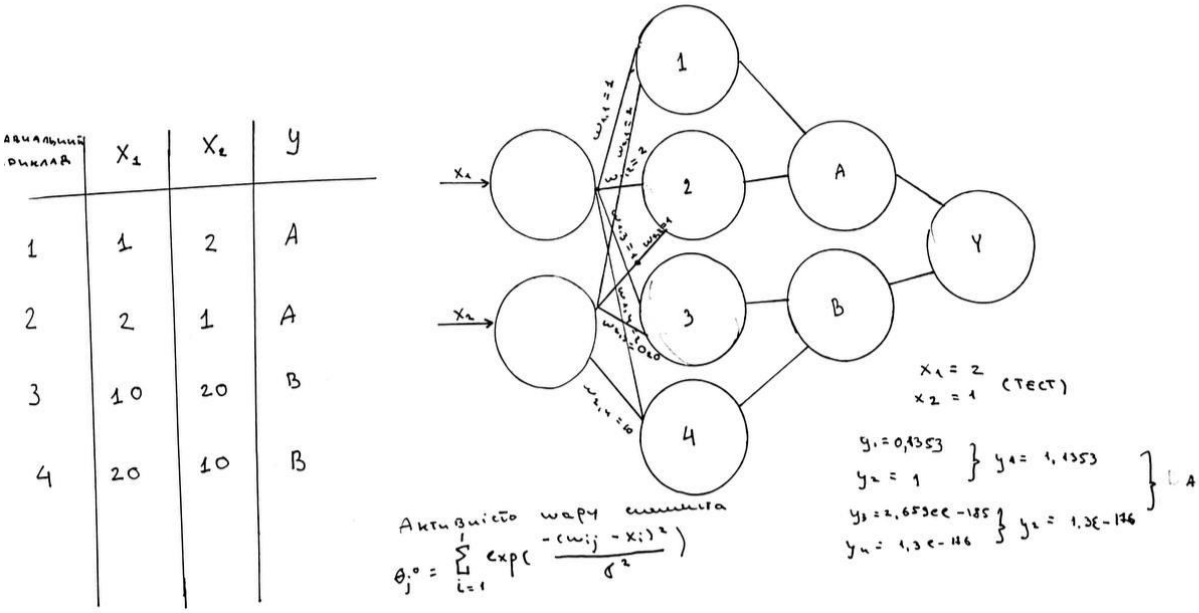

Рисунок 2.1. - Візуалізація лекційного прикладу

Мережі PNN дуже зручно використовувати для класифікації. Вони швидко навчаються, допускають наявність помилкових даних і надають корисні результати навіть на невеликих наборах навчальних даних. Однак мережі PNN вимагають значних ресурсів. Розв'язання деяких проблем потребує сотень або навіть тисяч навчальних зразків, що призводить до витрат часу при класифікації кожного невідомого екземпляра. Проте слід пам'ятати, що якщо мережа реалізована у вигляді апаратних засобів, то обчислення, як правило, виконуються паралельно [1].

Мережева кібератака типу "**Smurf**" - це атака, при якій атакуючий використовує велику кількість комп'ютерів або пристроїв для відправки фальшивих запитань до цільового сервера чи мережі. Атака отримала назву "Smurf" через використання інструменту для генерації запитань, який має назву "Smurf".

KDD Cup 1999 Data використовується для вивчення і вдосконалення алгоритмів виявлення вторгнень в комп'ютерні мережі. Цей датасет був підготовлений у зв'язку з конкурсом Knowledge Discovery and Data Mining (KDD Cup) 1999, який спрямовується на розв'язання проблеми виявлення вторгнень у мережах.

ЗПрактична частина

Data Preprocessing

Nr	Features	
	Name	Description
1	duration	duration of connection in seconds
2	protocol_type	connection protocol (tcp, udp, icmp)
3	service	dst port mapped to service (e.g. http, ftp, ..)
4	flag	normal or error status flag of connection
5	src_bytes	number of data bytes from src to dst
6	dst_bytes	bytes from dst to src
7	land	1 if connection is from/to the same host/port; else 0
8	wrong_fragment	number of 'wrong' fragments (values 0,1,3)
9	urgent	number of urgent packets
10	hot	number of 'hot' indicators (bro-ids feature)
11	num_failed_logins	number of failed login attempts
12	logged_in	1 if successfully logged in; else 0
13	num_compromised	number of 'compromised' conditions
14	root_shell	1 if root shell is obtained; else 0
15	su_attempted	1 if 'su root' command attempted; else 0
16	num_root	number of 'root' accesses
17	num_file_creations	number of file creation operations
18	num_shells	number of shell prompts
19	num_access_files	number of operations on access control files
20	num_outbound_cmds	number of outbound commands in an ftp session
21	is_hot_login	1 if login belongs to 'hot' list (e.g. root, adm); else 0
22	is_guest_login	1 if login is 'guest' login (e.g. guest, anonymous); else 0
23	count	number of connections to same host as current connection in past two seconds
24	srv_count	number of connections to same service as current connection in past two seconds
25	serror_rate	% of connections that have 'SYN' errors
26	srv_serror_rate	% of connections that have 'SYN' errors
27	rerror_rate	% of connections that have 'REJ' errors
28	srv_rerror_rate	% of connections that have 'REJ' errors
29	same_srv_rate	% of connections to the same service
30	diff_srv_rate	% of connections to different services
31	srv_diff_host_rate	% of connections to different hosts
32	dst_host_count	count of connections having same dst host
33	dst_host_srv_count	count of connections having same dst host and using same service
34	dst_host_same_srv_rate	% of connections having same dst port and using same service
35	dst_host_diff_srv_rate	% of different services on current host
36	dst_host_same_src_port_rate	% of connections to current host having same src port
37	dst_host_srv_diff_host_rate	% of connections to same service coming from diff. hosts
38	dst_host_serror_rate	% of connections to current host that have an S0 error
39	dst_host_srv_serror_rate	% of connections to current host and specified service that have an S0 error
40	dst_host_rerror_rate	% of connections to current host that have an RST error
41	dst_host_srv_rerror_rate	% of connections to the current host and specified service that have an RST error
42	connection_type	

Рисунок 3.1. – Опис даних датасету [2]

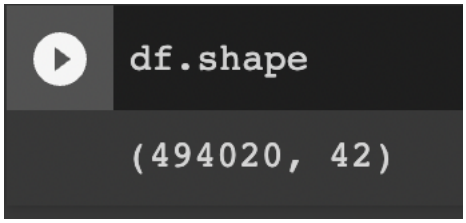

Рисунок 3.2. – Кількість рядків та стовпчиків (відповідо)

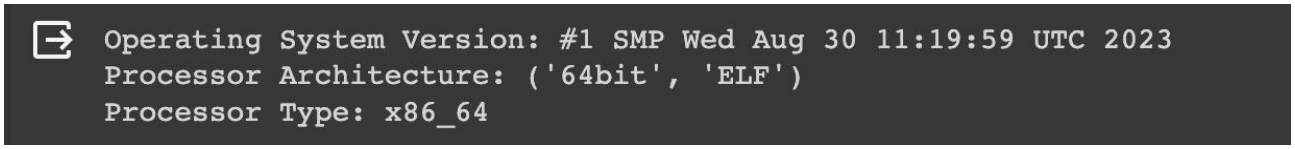

Рисунок 3.3. – Дані про комп'ютер що виконує програму

```
[182] df.dtypes

duration                        int64
protocol_type                  object
service                        object
flag                           object
src_bytes                       int64
dst_bytes                       int64
land                            int64
wrong_fragment                  int64
urgent                          int64
hot                             int64
num_failed_logins               int64
logged_in                       int64
lnum_compromised                int64
lroot_shell                     int64
lsu_attempted                   int64
lnum_root                       int64
lnum_file_creations             int64
lnum_shells                     int64
lnum_access_files               int64
lnum_outbound_cmds              int64
is_host_login                   int64
is_guest_login                  int64
count                           int64
srv_count                       int64
serror_rate                   float64
srv_serror_rate               float64
rerror_rate                   float64
srv_rerror_rate               float64
same_srv_rate                 float64
diff_srv_rate                 float64
srv_diff_host_rate            float64
   dst_host_count                   int64
   dst_host_srv_count               int64
   dst_host_same_srv_rate         float64
   dst_host_diff_srv_rate         float64
   dst_host_same_src_port_rate    float64
   dst_host_srv_diff_host_rate    float64
   dst_host_serror_rate           float64
   dst_host_srv_serror_rate       float64
   dst_host_rerror_rate           float64
   dst_host_srv_rerror_rate       float64
   label                           object
   dtype: object
```

Рисунок 3.4. – Тип колонок датасету

```
✓ 0 сек.  [187] null_values = df.isnull().sum()
          print(null_values)

duration                        0
protocol_type                   0
service                         0
flag                            0
src_bytes                       0
dst_bytes                       0
land                            0
wrong_fragment                  0
urgent                          0
hot                             0
num_failed_logins               0
logged_in                       0
lnum_compromised                0
lroot_shell                     0
lsu_attempted                   0
lnum_root                       0
lnum_file_creations             0
lnum_shells                     0
lnum_access_files               0
lnum_outbound_cmds              0
is_host_login                   0
is_guest_login                  0
count                           0
srv_count                       0
serror_rate                     0
srv_serror_rate                 0
rerror_rate                     0
srv_rerror_rate                 0
same_srv_rate                   0
diff_srv_rate                   0
srv_diff_host_rate              0
```

Рисунок 3.5. – Перевірка на пропущені значення

```
[183] df.label.value_counts()

0
сек.

smurf              280790
neptune            107201
normal              97277
back                 2203
satan                1589
ipsweep              1247
portsweep            1040
warezclient          1020
teardrop              979
pod                   264
nmap                  231
guess_passwd           53
buffer_overflow        30
land                   21
warezmaster            20
imap                   12
rootkit                10
loadmodule              9
ftp_write               8
multihop                7
phf                     4
perl                    3
spy                     2
Name: label, dtype: int64
```

Рисунок 3.6. – Кількість значень кожного класу

Закодуємо мітки за допомогою LabelEncoder (sklearn.preprocessing). Одночасно застосуємо тип даних float до всіх колонок (окрім таргету).

```
df = pd.read_csv("kddcup99.csv")
df.head()
```

	duration	protocol_type	service	flag	src_bytes	ds
0	0	tcp	http	SF	181	
1	0	tcp	http	SF	239	
2	0	tcp	http	SF	235	
3	0	tcp	http	SF	219	
4	0	tcp	http	SF	217	

5 rows × 42 columns

→

```
label_encoder = LabelEncoder()
for column in df.select_dtypes(include=["object"]).colum
    if column != "label":
        df[column] = label_encoder.fit_transform(df[colu

df.iloc[:, :-1] = df.iloc[:, :-1].astype(float)
df.head()
```

```
<ipython-input-184-08cb937bd6a2>:6: DeprecationWarning:
  df.iloc[:, :-1] = df.iloc[:, :-1].astype(float)
```

	duration	protocol_type	service	flag	src_bytes	dst
0	0.0	1.0	22.0	9.0	181.0	
1	0.0	1.0	22.0	9.0	239.0	
2	0.0	1.0	22.0	9.0	235.0	
3	0.0	1.0	22.0	9.0	219.0	
4	0.0	1.0	22.0	9.0	217.0	

5 rows × 42 columns

Рисунок 3.7. – Результат кодування та зміну типу даних

Використаємо мінімаксну нормалізацію за допомогою minmax_scale (sklearn.preprocessing). Ідея полягає в тому, що значення кожної ознаки масштабуються так, щоб вони потрапляли в діапазон від 0 до 1.

Рисунок 3.8. – Результат нормалізації

Розділяємо дані на тренувальний та тестовий набори (80%/20%). Після отримання тестового набору, вибираємо по 50 екземплярів з кожного класу (за умови, що класів більше 50) і об'єднуємо їх в новий тестовий набір. Необхідна міра через неможливість машини опрацювати більші об’єми.

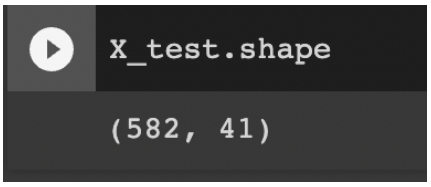

Рисунок 3.9. – Об’єми тестового набору

PNN

Створючи додаткову колонку у датасеті, будуємо процес роботи PNN.

У функції predict_pnn, для кожного рядка у тестовому наборі, обчислюється ймовірність належності до кожного класу за допомогою PNN. Для цього використовується експоненціальна функція від невідстані між рядком у тестовому наборі та кожним рядком у вхідних даних.

Потім для кожного класу вираховується середнє значення цих ймовірностей, і визначається клас, для якого середнє значення найбільше. Навчання на вищезазначеному наборі даних тривало 2 хвилини 17 секунд.

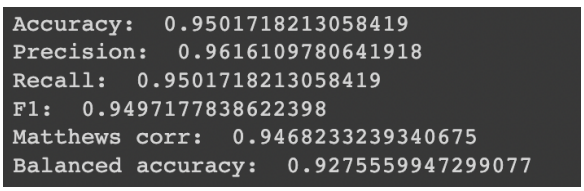

Рисунок 3.10. – Основні класифікаційні метрики

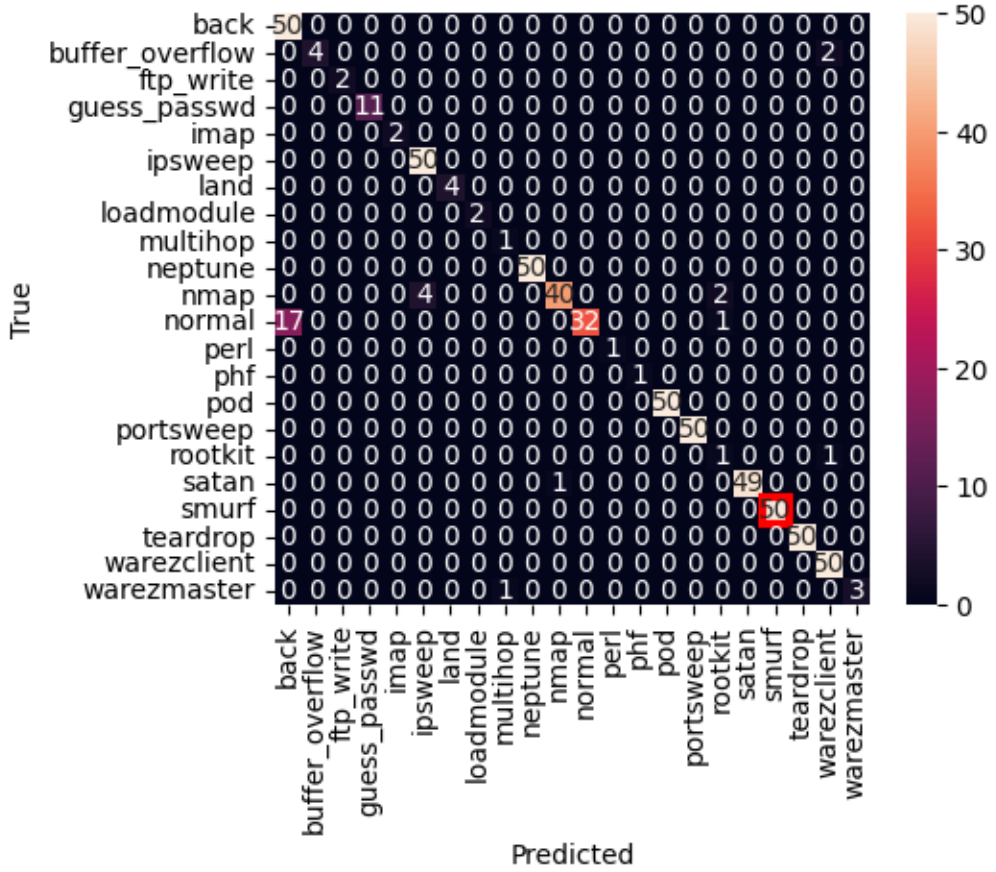

Рисунок 3.11. – Матриця помилок (smurf атака обведена червоним)

```
Accuracy for class smurf: 1.0000
```

Рисунок 3.12. – Точність класифікації smurf атаки на тестовому наборі

ВИСНОВКИ

Програма досить непогано працює (за аналізом класифікаційних метрик). Трошки кращих результатів можна досягнути для певних метрик за зміни параметрів/гіперпараметрів (залежить від задачі). Розпізнавання кібератаки типу smurf працює з accuracy рівною 1.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Навчальний посібник «Основні концепції нейроних мереж» Роберт Каллан – стор. 158-164.
2. The 41 features provided by the KDD Cup '99 datasets. (URL: https://www.researchgate.net/figure/The-41-features-provided-by-the-KDD-Cup-99-datasets_tbl1_263274883)

ДОДАТКИ

https://colab.research.google.com/drive/1gavrz_d_dZAIuAJpz1EIxGpeXoEJExbA?usp=sharing