МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №6 з дисципліни

«Методи та технології штучного інтелекту»

«Нейро-нечітке моделювання»

Перевірив:
Шимкович В.М.

Виконав:
студент 3 курсу
групи ІП-11 ФІОТ
Прищепа В.С.

Київ-2023

Лабораторна робота №6

Нейро-нечітке моделювання

Мета роботи: отримання і закріплення знань про методи моделювання та принципи функціонування нейронечітких систем, а також формування практичних навичок з конструювання нейронечітких мереж.

Завдання:

1. Сформулювати завдання в галузі обчислювальної техніки, для вирішення якої було б обґрунтовано застосування гібридної нейронечіткої мережі.
2. Сформувати вибірку для навчання гібридної нейронної мережі.
3. Згенерувати і візуалізувати структуру гібридної нейронної мережі.
4. Навчити гібридну нейронну мережу, при цьому задати і обґрунтувати параметри її навчання.
5. Побудувати систему нечіткого виводу для отриманої гібридної нейронної мережі.
6. Виконати перевірку адекватності побудованої нечіткої моделі гібридної мережі.
7. Оформіть звіт по лабораторній роботі.

Хід роботи:

В якості завдання було обрано курс кріптовалюти Etherium за період з 11-14-2018 по сьогоднішній день, дані мають наступний вигляд:

Date	# Open	# High	# Low	# Close	# Adj Close	# Volume
11/14/2018	206.533997	207.044998	174.084	181.397003	181.397003	2595330000
11/15/2018	181.899002	184.251007	170.188995	180.806	180.806	2638410000
11/16/2018	180.865005	181.350006	173.126007	175.177002	175.177002	2015330000
11/17/2018	175.360001	175.850998	172.869003	174.001007	174.001007	1832800000
11/18/2018	174.175003	179.151993	174.175003	177.067001	177.067001	1810920000
11/19/2018	177.179001	177.179001	147.850998	149.175003	149.175003	2745160000
11/20/2018	148.811005	151.253006	126.360001	130.339005	130.339005	3134410000
11/21/2018	131.141998	138.889999	125.758003	136.701004	136.701004	2685930000
11/22/2018	136.811005	137.740005	126.706001	126.706001	126.706001	1792150000
11/23/2018	126.418999	127.028	119.558998	123.295998	123.295998	1998010000
11/24/2018	123.304001	126.788002	110.824997	113.494003	113.494003	1800960000
11/25/2018	113.125999	118.880997	101.769997	116.449997	116.449997	2466750000

Для подальших дій підготуємо дані та розділимо їх на тренувальний та тестувальний набори:

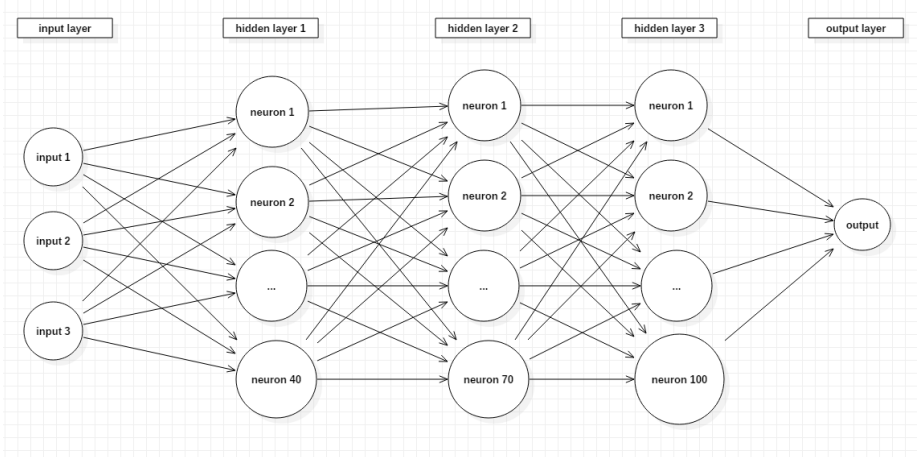
```
import pandas as pd
import torch
import torch.nn as nn
import torch.nn.functional as f
import torch.optim as optim
```

```
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

def prepdata(fn):
    csvfile = pd.read_csv(fn)
    csvfile = csvfile[csvfile['Currency'] == 'Etherium']
    x = csvfile[['Open', 'High', 'Low']]
    y = csvfile['Close']
    x = torch.tensor(x.values).float()
    y = torch.tensor(y.values).float()
    xtrn, xtst, ytrn, ytst = train_test_split(x, y, test_size=0.15, shuffle=False)
    return xtrn, xtst, ytrn.unsqueeze(1), ytst.unsqueeze(1), csvfile

xtrain, xtest, ytrain, ytest, csvf = prepdata('ETH-BTC-USD.csv')
```

Далі реалізуємо гібридну модель:

```
class HybrNN(nn.Module):
    def __init__(self):
        super(HybrNN, self).__init__()
        self.fc1 = nn.Linear(3, 40)
        self.fc2 = nn.Linear(40, 70)
        self.fc3 = nn.Linear(70, 100)
        self.fc4 = nn.Linear(100, 1)
    def forward(self, x):
        x = f.relu(self.fc1(x))
        x = f.relu(self.fc2(x))
        x = f.relu(self.fc3(x))
        x = self.fc4(x)
        return x

model = HybrNN()
```

Маємо наступну нейрону мережу:



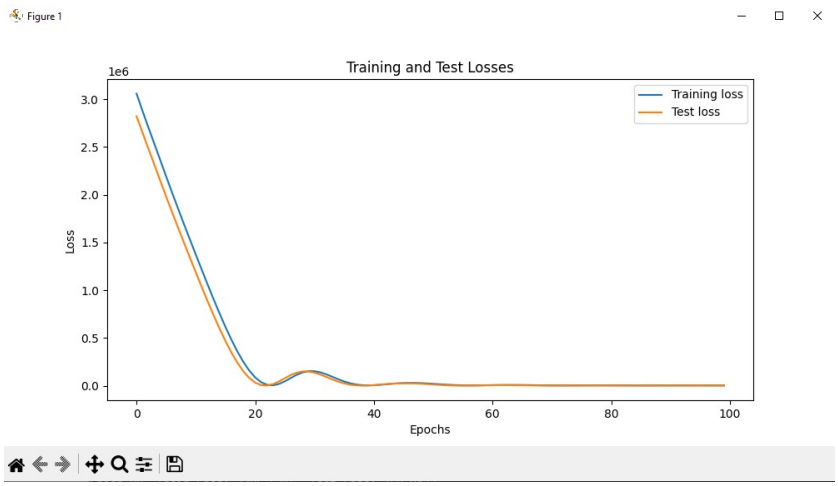
Далі натренуємо модель:

```
def trainmodel(mdl, xtrn, ytrn, xtst, ytst, epochs=100):
    crit = nn.MSELoss()
    optimize = optim.Adam(mdl.parameters(), lr=0.001)
    trnloss = []
    tstloss = []
    print("Model Training")
    for epoch in range(epochs):
        mdl.train()
        optimize.zero_grad()
        outs = mdl(xtrn)
        loss = crit(outs, ytrn)
        loss.backward()
        optimize.step()
        trnloss.append(loss.item())
        mdl.eval()
        with torch.no_grad():
            test_outputs = mdl(xtst)
            test_loss = crit(test_outputs, ytst)
            tstloss.append(test_loss.item())
            print(f'Epoch {epoch}, Train Loss: {loss.item():.4f}, Test Loss:
{test_loss.item():.4f}')
    return trnloss, tstloss

trainlosses, testlosses = trainmodel(model, xtrain, ytrain, xtest, ytest)

def printlosses(trnloss, tstloss):
    plt.figure(figsize=(10, 5))
    plt.plot(trnloss, label='Training loss')
    plt.plot(tstloss, label='Test loss')
    plt.title('Training and Test Losses')
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend()
    plt.show()

printlosses(trainlosses, testlosses)
```

Результат тренування моделі:

```
Model Training
Epoch 0, Train Loss: 3056869.5000, Test Loss: 2820034.7500
Epoch 1, Train Loss: 2876487.0000, Test Loss: 2648961.5000
Epoch 2, Train Loss: 2701802.2500, Test Loss: 2480095.7500
Epoch 3, Train Loss: 2531473.0000, Test Loss: 2312828.5000
Epoch 4, Train Loss: 2360090.0000, Test Loss: 2144054.0000
Epoch 5, Train Loss: 2188356.0000, Test Loss: 1977139.2500
Epoch 6, Train Loss: 2018738.5000, Test Loss: 1814376.5000
Epoch 7, Train Loss: 1853163.0000, Test Loss: 1653287.8750
Epoch 8, Train Loss: 1689409.1250, Test Loss: 1496282.2500
Epoch 9, Train Loss: 1529613.2500, Test Loss: 1338823.2500
Epoch 10, Train Loss: 1369487.8750, Test Loss: 1183951.0000
Epoch 11, Train Loss: 1211757.7500, Test Loss: 1030589.8750
Epoch 12, Train Loss: 1055623.3750, Test Loss: 879875.5000
Epoch 13, Train Loss: 902319.9375, Test Loss: 734853.9375
Epoch 14, Train Loss: 754595.9375, Test Loss: 595865.2500
Epoch 15, Train Loss: 612867.3750, Test Loss: 465169.7812
Epoch 16, Train Loss: 479523.0625, Test Loss: 345401.1562
Epoch 17, Train Loss: 357239.5312, Test Loss: 239655.5312
Epoch 18, Train Loss: 249124.1406, Test Loss: 150567.1875
Epoch 19, Train Loss: 157956.6562, Test Loss: 80676.2188
Epoch 20, Train Loss: 86338.5391, Test Loss: 32252.4434
Epoch 21, Train Loss: 36446.4805, Test Loss: 6025.6025
Epoch 22, Train Loss: 9137.2461, Test Loss: 1533.8215
Epoch 23, Train Loss: 3984.6621, Test Loss: 16025.7422
Epoch 24, Train Loss: 18267.6543, Test Loss: 44204.3750
Epoch 25, Train Loss: 46484.7070, Test Loss: 78821.9922
Epoch 26, Train Loss: 81235.3125, Test Loss: 111885.1719
Epoch 27, Train Loss: 114455.9375, Test Loss: 136850.8125
Epoch 28, Train Loss: 139537.9844, Test Loss: 149596.9844
Epoch 29, Train Loss: 152314.7500, Test Loss: 149065.7969
Epoch 30, Train Loss: 151717.6875, Test Loss: 136852.5000
Epoch 31, Train Loss: 139367.3594, Test Loss: 116240.3438
Epoch 32, Train Loss: 118575.5078, Test Loss: 91215.1172
Epoch 33, Train Loss: 93370.8828, Test Loss: 65588.0547
Epoch 34, Train Loss: 67605.0078, Test Loss: 42429.3320
Epoch 35, Train Loss: 44375.6602, Test Loss: 23794.7461
Epoch 36, Train Loss: 25757.2598, Test Loss: 10694.0469
Epoch 37, Train Loss: 12759.7646, Test Loss: 3210.8518
Epoch 38, Train Loss: 5451.9058, Test Loss: 706.3940
Epoch 39, Train Loss: 3179.4146, Test Loss: 2093.3958
Epoch 40, Train Loss: 4830.6196, Test Loss: 6063.1099
Epoch 41, Train Loss: 9071.3574, Test Loss: 11288.2354
Epoch 42, Train Loss: 14550.6748, Test Loss: 16594.3750
Epoch 43, Train Loss: 20074.4902, Test Loss: 21074.1348
Epoch 44, Train Loss: 24721.4531, Test Loss: 24134.5254
Epoch 45, Train Loss: 27890.4062, Test Loss: 25492.4277
Epoch 46, Train Loss: 29295.6680, Test Loss: 25135.9316
Epoch 47, Train Loss: 28927.7402, Test Loss: 23267.5234
Epoch 48, Train Loss: 26995.2266, Test Loss: 20240.1113
Epoch 49, Train Loss: 23859.9062, Test Loss: 16492.7148
Epoch 50, Train Loss: 19971.2832, Test Loss: 12489.7627
Epoch 51, Train Loss: 15805.1084, Test Loss: 8667.1787
Epoch 52, Train Loss: 11808.3262, Test Loss: 5387.2095
Epoch 53, Train Loss: 8353.2871, Test Loss: 2904.8118
Epoch 54, Train Loss: 5703.4663, Test Loss: 1348.2344
Epoch 55, Train Loss: 3993.6948, Test Loss: 715.4093
Epoch 56, Train Loss: 3226.3809, Test Loss: 886.8015
Epoch 57, Train Loss: 3284.3481, Test Loss: 1653.1920
Epoch 58, Train Loss: 3958.9060, Test Loss: 2754.3096
Epoch 59, Train Loss: 4988.9175, Test Loss: 3922.3345
Epoch 60, Train Loss: 6104.7856, Test Loss: 4922.7510
Epoch 61, Train Loss: 7069.9199, Test Loss: 5586.0020
```

```
Epoch 41, Train Loss: 9071.3574, Test Loss: 11288.2354
Epoch 42, Train Loss: 14550.6748, Test Loss: 16594.3750
Epoch 43, Train Loss: 20074.4902, Test Loss: 21074.1348
Epoch 44, Train Loss: 24721.4531, Test Loss: 24134.5254
Epoch 45, Train Loss: 27890.4062, Test Loss: 25492.4277
Epoch 46, Train Loss: 29295.6680, Test Loss: 25135.9316
Epoch 47, Train Loss: 28927.7402, Test Loss: 23267.5234
Epoch 48, Train Loss: 26995.2266, Test Loss: 20240.1113
Epoch 49, Train Loss: 23859.9062, Test Loss: 16492.7148
Epoch 50, Train Loss: 19971.2832, Test Loss: 12489.7627
Epoch 51, Train Loss: 15805.1084, Test Loss: 8667.1787
Epoch 52, Train Loss: 11808.3262, Test Loss: 5387.2095
Epoch 53, Train Loss: 8353.2871, Test Loss: 2904.8118
Epoch 54, Train Loss: 5703.4663, Test Loss: 1348.2344
Epoch 55, Train Loss: 3993.6948, Test Loss: 715.4093
Epoch 56, Train Loss: 3226.3809, Test Loss: 886.8015
Epoch 57, Train Loss: 3284.3481, Test Loss: 1653.1920
Epoch 58, Train Loss: 3958.9060, Test Loss: 2754.3096
Epoch 59, Train Loss: 4988.9175, Test Loss: 3922.3345
Epoch 60, Train Loss: 6104.7856, Test Loss: 4922.7510
Epoch 61, Train Loss: 7069.9199, Test Loss: 5586.0020
Epoch 62, Train Loss: 7712.6772, Test Loss: 5824.7964
Epoch 63, Train Loss: 7944.0439, Test Loss: 5635.6182
Epoch 64, Train Loss: 7759.3804, Test Loss: 5085.8330
Epoch 65, Train Loss: 7225.0078, Test Loss: 4290.2993
Epoch 66, Train Loss: 6455.0581, Test Loss: 3384.6892
Epoch 67, Train Loss: 5584.3633, Test Loss: 2499.6689
Epoch 68, Train Loss: 4742.4463, Test Loss: 1740.5953
Epoch 69, Train Loss: 4033.1162, Test Loss: 1175.3000
Epoch 70, Train Loss: 3522.1963, Test Loss: 830.3708
Epoch 71, Train Loss: 3233.8386, Test Loss: 694.8030
Epoch 72, Train Loss: 3154.3608, Test Loss: 728.9217
Epoch 73, Train Loss: 3241.3848, Test Loss: 876.0697
Epoch 74, Train Loss: 3435.7361, Test Loss: 1074.6344
Epoch 75, Train Loss: 3673.6731, Test Loss: 1268.5082
Epoch 76, Train Loss: 3897.4968, Test Loss: 1414.6940
Epoch 77, Train Loss: 4063.2771, Test Loss: 1487.5221
Epoch 78, Train Loss: 4145.0757, Test Loss: 1479.4397
Epoch 79, Train Loss: 4135.7266, Test Loss: 1398.8917
Epoch 80, Train Loss: 4044.6211, Test Loss: 1266.1017
Epoch 81, Train Loss: 3893.3779, Test Loss: 1107.7299
Epoch 82, Train Loss: 3710.3550, Test Loss: 951.3360
Epoch 83, Train Loss: 3524.9585, Test Loss: 820.5522
Epoch 84, Train Loss: 3362.6965, Test Loss: 731.6027
Epoch 85, Train Loss: 3241.5593, Test Loss: 691.5378
Epoch 86, Train Loss: 3170.1741, Test Loss: 698.2592
Epoch 87, Train Loss: 3147.7642, Test Loss: 742.0942
Epoch 88, Train Loss: 3165.7019, Test Loss: 808.4735
Epoch 89, Train Loss: 3210.1787, Test Loss: 881.1058
Epoch 90, Train Loss: 3265.3931, Test Loss: 945.0208
Epoch 91, Train Loss: 3316.6223, Test Loss: 988.9593
Epoch 92, Train Loss: 3352.6399, Test Loss: 1006.7614
Epoch 93, Train Loss: 3367.1475, Test Loss: 997.6481
Epoch 94, Train Loss: 3359.0581, Test Loss: 965.4991
Epoch 95, Train Loss: 3331.8108, Test Loss: 917.4036
Epoch 96, Train Loss: 3291.9456, Test Loss: 861.8942
Epoch 97, Train Loss: 3247.3428, Test Loss: 807.2160
Epoch 98, Train Loss: 3205.5215, Test Loss: 759.9472
Epoch 99, Train Loss: 3172.2903, Test Loss: 724.1808
```

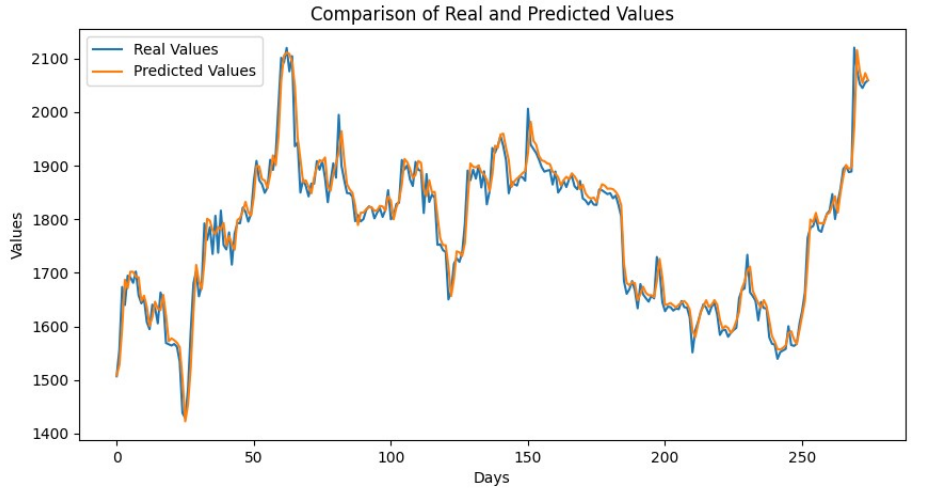


Графік похибки.

Тестування моделі:

```
def predictreal(mdl, xtst, ytst):
    mdl.eval()
    with torch.no_grad():
        predictions = mdl(xtst)
    plt.figure(figsize=(10, 5))
    plt.plot(ytst.numpy(), label='Real Values')
    plt.plot(predictions.numpy(), label='Predicted Values')
    plt.title('Comparison of Real and Predicted Values')
    plt.xlabel('Days')
    plt.ylabel('Values')
    plt.legend()
    plt.show()

predictreal(model, xtest, ytest)
last_row = csvf.iloc[-1][['Open', 'High', 'Low']].astype(float).values
last_row_tensor = torch.tensor(last_row).float()
model.eval()
with torch.no_grad():
    predicted_currency = model(last_row_tensor.unsqueeze(0))
print(f'Prognosed tomorrow`s currency: {predicted_currency.item()}')
```

Результат:



Графік порівняння передбачених і реальних результатів

```
Prognosed tomorrow`s currency: 2060.0234375
Press any key to continue . . .
```

Спрогнозований результат (перевірка адекватності).

Висновок: під час виконання лабораторної роботи я натренував гібридну модель для передбачення курсу кріптовалюти. Модель адекватна і передбачає курс валюти доволі точно. Код і результати виконання наведені вище.