**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ   
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра АСУ

Отчет

о лабораторной работе № 4

по дисциплине «Интеллектуальный анализ данных»

# на тему: «нейронные сети для прогнозирования временных рядов»

Выполнил:

студент группы ИСТ-19б

Карпович В. Д.

Проверили:

Шуватова Е.А.

Васяева Т.А.

Донецк – 2022

**Цель работы:** изучение возможности нейронных сетей применительно к прогнозированию временных рядов; разработка архитектуры и обучение нейронной сети для прогнозирования временного ряда.

**Листинг:**

import numpy as np

import pandas as pd

import tensorflow as tf

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

dataframe = pd.read\_csv("tothemoon.csv")

print(dataframe)

dates = pd.to\_datetime(df["<DATE>"], format='%d/%m/%y').dt.date

plt.ylabel("Свеча закрылась")

plt.plot(dates, dataframe['<CLOSE>'])

plt.show()

def convert(data, window\_size = 10):

    n = data.shape[0] - window\_size

    X = np.array([data[i : i + window\_size, :] for i in range(n)])

    Y = data[window\_size:]

    return X, Y

scaler = MinMaxScaler()

close = scaler.fit\_transform(dataframe["<CLOSE>"].values.reshape(-1, 1))

train\_offset = 190

window\_size = 10

close\_train = close[:train\_offset]

close\_test = close[train\_offset:]

X\_train, Y\_train = convert(close\_train, window\_size)

X\_test, Y\_test = convert(close\_test, window\_size)

model = tf.keras.models.Sequential([

    tf.keras.layers.GRU(units = 64),

    tf.keras.layers.Dense(units = 1)

])

model.compile(

    loss = tf.keras.losses.MeanSquaredError(),

    optimizer = tf.keras.optimizers.Adam(),

    metrics = [tf.keras.metrics.MeanAbsoluteError()]

)

history = model.fit(X\_train, Y\_train, epochs = 311, batch\_size = 16)

predict = model.predict(X\_test)

predict = scaler.inverse\_transform(predict)

print(predict)

y\_original = scaler.inverse\_transform(Y\_test)

start = train\_offset - window\_size

plt.plot(dates[start:], dataframe["<CLOSE>"].values[start:], '-o', label = 'Source')

plt.plot(dates[train\_offset + window\_size:], predict, '-o', label = 'Predict')

plt.legend()

plt.show()

**Результат:** можно делать деньги

