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-19а

Шевченко М. В.

Проверили:

Васяева Т. А.

Шуватова Е. А.

Донецк – 2022

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

Порядок выполнения работы

1. Изучить предметную область. Выбрать данные для анализа <https://www.finam.ru/quotes>
2. Разработать рекуррентную нейросеть для прогнозирования котировок (цена открытия и / или закрытия), для четных вариантов - обязателен минимум один слой GRU, для не четных - обязателен слой LSTM. Номер варианта совпадает с номером в журнале.

Предусмотреть:

* нормирование / денормирование данных;
* обязательное сохранение / считывание обученной НС;
* отдельный этап тестирования для указанного интервала в будущем;
* вывод и расчет ошибки на обучающих, тестовых (валидационных) данных;
* вывод результатов в файл (дата, спрогнозированное значение, реальное значение (если такое имеется для текущего прогноза), ошибка);
* вывод результатов на графиках (отдельно обучающая выборка, отдельно тестовая и обе на одном графике).

1. Провести эксперименты для разного значения глубины (например, от 200 до 4000 баров) показать результаты экспериментов, сделать выводы

**Индивидуальное задание:** Вариант 12, GRU

**Листинг программы**

Импорт

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import GRU, Dense

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

import math

Загрузка данных из .csv файла

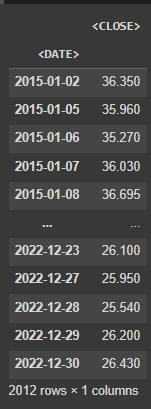
df = pd.read\_csv('INTC.csv')

df['<DATE>'] = pd.to\_datetime(df['<DATE>'], format='%Y%m%d')

df.set\_index('<DATE>', inplace=True)

df = df.filter(['<CLOSE>'])

df



Определение значения глубины

depth = 70

Разделение данных на наборы для обучения и тестирования, нормирование данных

tr\_data\_len = math.ceil(.75 \* len(df))

plt.figure(figsize=(7,3))

plt.plot(df[:tr\_data\_len]['<CLOSE>'], label='Обуч.')

plt.plot(df[tr\_data\_len:]['<CLOSE>'], label='Трен.')

plt.xlabel('Date')

plt.ylabel('Close')

plt.legend()

plt.show()

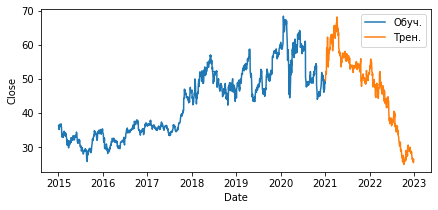
values = df.values

scaler = MinMaxScaler(feature\_range=(0,1))

scaled\_values = scaler.fit\_transform(values)

train = scaled\_values[:tr\_data\_len, :]

test = scaled\_values[tr\_data\_len-depth:, :]



Формирование выборок

x\_train = []

y\_train = []

for i in range(depth, len(train)):

x\_train.append(train[i-depth:i,0])

y\_train.append(train[i, 0])

x\_train, y\_train = np.array(x\_train), np.array(y\_train)

x\_train = np.reshape(x\_train, (x\_train.shape[0], x\_train.shape[1], 1))

x\_test = []

y\_test = []

for i in range(depth, len(test)):

x\_test.append(test[i-depth:i,0])

y\_test.append(test[i, 0])

x\_test, y\_test = np.array(x\_test), np.array(y\_test)

x\_test = np.reshape(x\_test, (x\_test.shape[0], x\_test.shape[1], 1))

Создание и обучение НС

model = Sequential()

model.add(GRU(units=60,input\_shape= (x\_train.shape[1],1)))

model.add(Dense(1))

model.compile(optimizer='adam', loss='mean\_absolute\_error')

model.fit(x\_train, y\_train, batch\_size=1, epochs=3)

Предсказание значений и вывод на график

predict = model.predict(x\_test)

predict = scaler.inverse\_transform(predict)

p\_df = df[-predict.size:]

p\_df['predict'] = predict

plt.figure(figsize=(15,8))

plt.xlabel('Date',fontsize=20)

plt.ylabel('Close',fontsize=20)

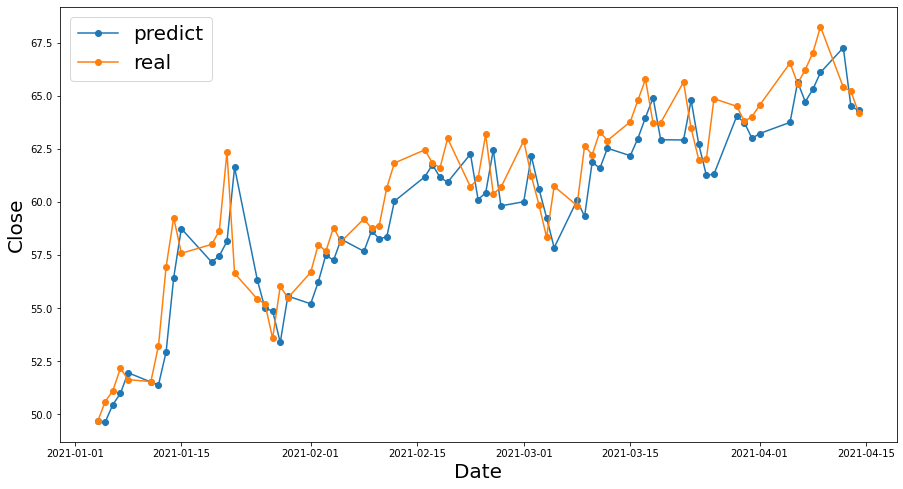
plt.plot(p\_df[:70]['predict'],'-o', label='predict')

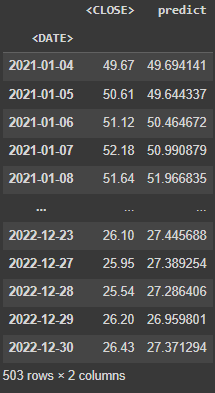
plt.plot(p\_df[:70]['<CLOSE>'],'-o', label='real')

plt.legend(fontsize=20)

plt.show()

p\_df





print(tf.keras.metrics.mean\_absolute\_error(p\_df['<CLOSE>'], p\_df['predict']).numpy())

print(tf.keras.metrics.mean\_squared\_error(p\_df['<CLOSE>'], p\_df['predict']).numpy())

model.save('myModel')

0.9227104

1.4786415