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

Деркач К. И.

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

**Листинг**

import tensorflow as tf

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

import math

from tensorflow.keras.metrics import mean\_absolute\_error, mean\_squared\_error

from tensorflow.keras.models import Sequential

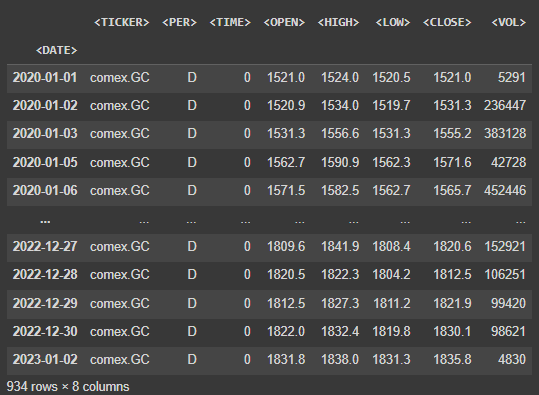
from tensorflow.keras.layers import GRU, Dense

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

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

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

df



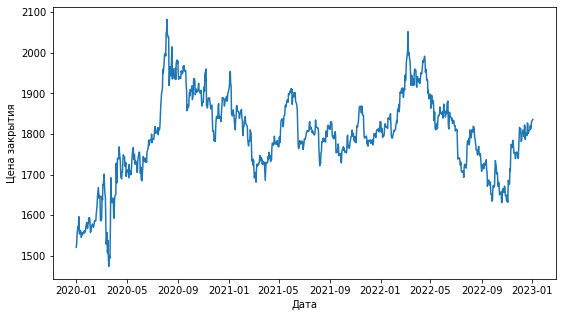
plt.figure(figsize=(9,5))

plt.plot(df['<CLOSE>'])

plt.xlabel('Дата')

plt.ylabel('Цена закрытия')

plt.show()



days = 60

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

data = dff.values

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

scaled\_data = scaler.fit\_transform(data)

train\_size = math.ceil(.8 \* len(data))

train = scaled\_data[:train\_size, :]

test = scaled\_data[train\_size-days:, :]

print(scaled\_data.shape,train.shape,test.shape)

(934, 1) (748, 1) (246, 1)

x\_train = []

y\_train = []

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

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

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

x\_test = []

y\_test = []

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

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

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

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

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

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

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

model = Sequential()

model.add(GRU(units=50, return\_sequences=True, input\_shape= (x\_train.shape[1],1)))

model.add(GRU(units=25))

model.add(Dense(1))

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

model.fit(x\_train, y\_train, batch\_size=10, epochs=10)

pred = model.predict(x\_test)

pred = scaler.inverse\_transform(pred)

val = dff[-pred.size:]

val['pred'] = pred

plt.figure(figsize=(12,6))

plt.xlabel('Дата',fontsize=20)

plt.ylabel('Цена закрытия')

plt.plot(val['pred'][:50],'-o', label='предсказания')

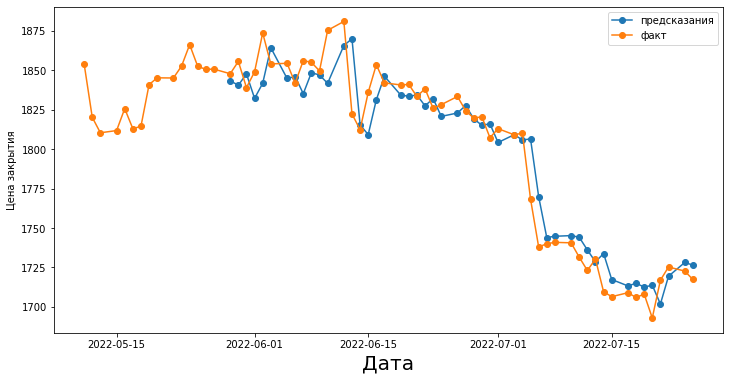
plt.plot(dff['<CLOSE>'][-pred.size-15:-pred.size+50],'-o', label='факт')

plt.legend()

plt.show()

model.save('model')

model.summary()



print(val.tail(3))

print('Среднее значение ошибки:', mean\_absolute\_error(val['<CLOSE>'], val['pred']).numpy())