МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики
Кафедра анализа данных и искусственного интеллекта

Допустить к защите
заведующий кафедрой
д–р тех. наук, доцент
_____ А. В. Коваленко
_____ 2023г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ПРОГНОЗИРОВАНИЯ СТОИМОСТИ АКЦИЙ КОМПАНИИ НА ОСНОВЕ ВРЕМЕННЫХ РЯДОВ

Работу выполнила _____ В. А. Головань
(подпись)

Направление подготовки 09.03.03 Прикладная информатика

Направленность (профиль) Прикладная информатика в экономике

Научный руководитель
канд. пед. наук, доцент _____ В.А. Акиньшина
(подпись)

Нормоконтролер
канд. физ.–мат. наук, доцент _____ Г.В. Калайдина
(подпись)

Краснодар
2023

РЕФЕРАТ

Выпускная квалификационная работа 39 с., 28 рис., 9 источников.

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ПРОГНОЗИРОВАНИЯ СТОИМОСТИ АКЦИЙ КОМПАНИИ НА ОСНОВЕ ВРЕМЕННЫХ РЯДОВ

Объектом исследования являются нейронный сети, временный ряды, телеграмм боты и фондовый рынок, а также основные технологии для работы с ними.

Цель выпускной квалификационной работы – разработка телеграмм бота позволяющего прогнозировать стоимость акций компании на краткосрочный период.

Задачами являются: изучение различных способов прогнозирования с помощью временных рядов и рекуррентных нейронных сетей, построение прогностических моделей и изучение их работоспособности, проведение сравнительного анализа моделей и реальных данных, разработка телеграмм бота для краткосрочного прогнозирования, тестирование разработанного телеграмм бота.

В результате выполнения выпускной квалификационной работы были изучены теоретические основы построения временных рядов и нейронных сетей с долгой краткосрочной памятью, был разработан телеграмм бот, позволяющий получить результаты прогнозирования стоимости курса акций для выбранного временного ряда на основе двух предложенных моделей.

СОДЕРЖАНИЕ

Введение 4
1 Нейронные сети 6
1.1 Классификация нейронных сетей по характеру обучения 9
1.2 Типы нейронных сетей 10
1.2.1 Сверточная нейронная сеть 10
1.2.2 Многослойный перцептрон Розенблатта 13
1.2.3 LSTM 15
2 Временные ряды 18
2.1 Основные понятия прогнозирования временных рядов 18
2.2 Методы для построения моделей временных рядов 22
2.2.1 Регрессионная модель 22
2.2.2 Сезонная ARIMA модель SARIMA 23
3 Телеграм-бот с применением нейронных сетей 25
3.1 Постановка задачи 25
3.2 Программная реализация 26
3.2.1 Построение математической модели и выбор средств реализации 26
3.2.2 Модель прогнозирования LSTM 29
3.2.3 Модель прогнозирования SARIMA 34
Заключение 38
Список использованных источников 39

ВВЕДЕНИЕ

Актуальность работы обусловлена тем, что фондовый рынок отличается своей нестабильностью. Каждый день котировки акций растут или снижаются. Выбрать лучшее время для покупки или продажи очень непросто, так как на фондовый рынок влияют состояние экономики, политика, настроение инвесторов и общее положение дел в мире. Серьезным экономическим спадам в отдельных странах и регионах обычно предшествует падение соответствующих рынков акций**.**

Такие прогнозы позволяют не только оптимизировать прибыль, но и экономические и социальные катастрофы. Накопленная информация часто используется при разработке экономических теорий и финансовых приложений. Обработка исторических временных рядов экономических величин позволила сгладить очередные финансовые кризисы.

Цель работы – разработка телеграмм бота, позволяющего прогнозировать стоимость акций компании на краткосрочный период.

Задачами являются: изучение различных способов прогнозирования с помощью временных рядов и рекуррентных нейронных сетей, построение прогностических моделей и изучение их работоспособности, проведение сравнительного анализа моделей и реальных данных, разработка телеграмм бота для краткосрочного прогнозирования, тестирование разработанного телеграмм бота.

Выпускная квалификационная работа состоит из трех глав, введения, заключения и списка используемой литературы. В первой главе описана теория нейронных сетей, классификация и типы. Во второй главе изучены теоретические аспекты временных рядов, способы работы с ними, особое внимание уделяется модели SARIMA, которая будет использована для построения прогноза. В заключительной, третьей части, представлена непосредственная реализация телеграмм бота с примерами построения

прогноза на разные периоды времени, также описаны используемые методы прогнозирования и проведен сравнительный анализ полученных прогнозов.

1 Нейронные сети

Искусственной нейронной сетью называется математическая модель и ее программная реализация, повторяющие работу головного мозга. Нейросеть имеет связные между собой нейроны, между которых есть синаптическая связь. Нейроны получают на вход набор сигналов и вычисляют по ним с помощью математических функций выходные сигналы. Далее, по аксонам следующему слою передаются нейроны нейросети. В процессе работы значения нейронов могут корректироваться с помощью весовых коэффициентов, подобных биологической нейронной сети, возможно увеличение или уменьшение сигнала, передающегося по данным связям.

Нейроны делятся на три типа:

– входные, заданные извне значения;

– выходные, значения, использующиеся извне;

– внутренние, не относятся ни к первому, ни ко второму классу.

На практике нейронная сеть включает в себя несколько внутренних слоев, количество которых зависит от сложности модели.

В ситуации, когда нейронная сеть содержит большое количество нейронов, вводится термин слоя (Рисунок *1*). Существуют три типа таких слоев:

– входной;

– скрытый;

– выходной.

Первый слой получает необходимые ему данные и работает с ними, затем на следующий этап передается сумма выходных значений и так далее. В конечном итоге, самый последний, выходной слой, нормализует значения с помощью функций активации и показывает результирующий набор.

Синапс – это связь между двумя нейронами. Они обладают всего одним параметром – весовым коэффициентом. Благодаря ему, входная информация изменяется, когда передается от одного нейрона к другому.

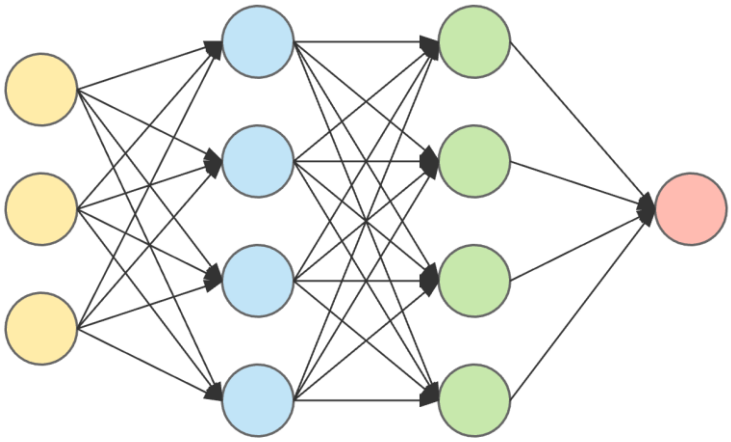

Рисунок 1 – Графическое представление нейронной сети

Нейрон собирает с некоторыми весами значения предков, суммирует их, а дальше к сумме применяет некоторую нелинейную функцию (Рисунок *2*).

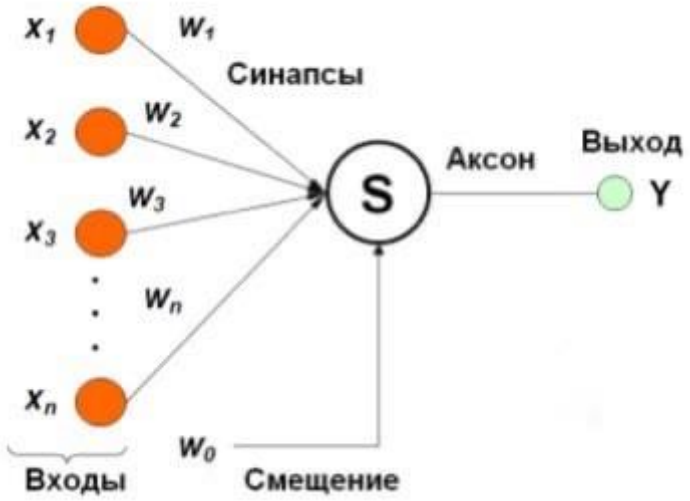

Рисунок 2 – Схема искусственного нейрона

$$S = \sum_{i=1}^{n} x_i \cdot w_i + b,$$

где n – количество входных сигналов нейрона,

b – коэффициент смещения.

С помощью нейронных сетей решается определенный ряд задач:

– классификации образов. Данный алгоритм получает на вход заранее подготовленные классы и показывает, к какому из них можно отнести полученные значения;

– кластеризации/категоризации. Проводится с помощью разбиения множества входных нейронов на подклассы, количественное соотношение и характеристики которых не известны заранее. В следствие работы алгоритма можно понять к какому классу необходимо отнести полученный на вход сигнал, либо же сказать, что входной сигнал не принадлежит ни к какому из известных классов. Таким образом, обнаруживаются новые, ранее не присутствующие в обучающей выборке значения. Кластеризацию осуществляют, например, нейронные сети Кохонена;

– аппроксимация функций. Предположим, что имеется обучающая выборка $((x^1, y^1), (x^2, y^2) \ldots, (x^N, y^N))$, где (x^K, y^K) – пары данных вход-выход, которая генерируется неизвестной функцией *F(x)*, искаженной шумом. Задача аппроксимации состоит в нахождении оценки неизвестной функции *F(x)*. Аппроксимация функций необходима при решении многочисленных инженерных и научных задач моделирования;

– предсказание/прогноз. Способности к предсказанию нейросетью напрямую зависят от ее работоспособности с точки зрения обобщения и выделения скрытых зависимостей между разными наборами данных. Прогнозирование осуществимо только тогда, когда прошлые изменения в какой-то мере влияют на будущие.

– оптимизация. Данный способ позволяет найти наиболее подходящее решение поставленной задачи. При этом оно должно удовлетворять заданным условиям и иметь максимум или минимум целевой функции.

1.1 Классификация нейронных сетей по характеру обучения

Существует два вида классификации:

– нейронные сети, обучающиеся с помощью учителя;

– самообучающиеся нейронные сети.

При обучении с учителем для каждого входного вектора существует дополнительный целевой вектор, который состоит из требуемых выходных значений. Вместе они образуют пару. Предоставляется необходимая выборка, рассчитываются выходные значения сети и сопоставляются с заданным целевым вектором. Далее веса изменяются алгоритмом для минимизации ошибки обучения Ошибки вычисляются для каждого вектора, после чего происходит многократное подстраивание весов, пока не достигнуто приемлемое значение ошибки на всем наборе обучающих данных. После корректировки нейронная сеть лучше улавливает значения входных параметров. Стоит отметить то, что вся информация нейронной сети о задаче, содержится в наборе примеров. В связи с этим качество обучения сети сильно зависит от количества примеров в выборке, предназначенной для обучения.

Обучение без учителя больше похожа на биологические нейронные сети с точки зрения обучения. В ее использовании нет необходимости предоставлять целевой вектор для выходов и, следовательно, не требует сравнения с предопределенными идеальными ответами. В обучающих данных нет корректировки, так как в первом слое нейронной сети нет весовых коэффициентов, входные векторы поступают в своем первоначальном виде. Алгоритм построен на анализе статистических свойств множества и группировке похожих векторов, что приводит к появлению новых классов.

По архитектуре связей можно выделить следующие классы нейронных сетей:

- однослойные сети прямого распространения;
- многослойные сети прямого распространения;
- рекуррентные сети;
- свёрточные нейронные сети

1.2 Типы нейронных сетей

1.2.1 Сверточная нейронная сеть

Стандартной задачей использования сверточной нейронной сети является классификация изображений, путем приема исходного изображения и определения его в класс, который наилучшим образом характеризует данное изображение. [4]

Суть такой классификации заключается в поиске базовых особенностей, характерных для данного класса, а затем в построении более абстрактных концепций через группы сверточных слоев.

Предположим, есть матрица A $(n_x \times n_y)$ и матрица B $(m_x \times m_y)$, умножив их друг на друга, получим матрицу C = A*B $(n_x - m_x + 1 \times n_y - m_y + 1)$ – свертка. Каждый элемент матрицы C определяется как скалярное произведение матрицы B и некоторой подматрицы A такого же размера.

$$C_{i,j} = \sum_{u=0}^{m_x-1} \sum_{v=0}^{m_y-1} A_{i+u,j+v} B_{u,v},$$

где $B_{u,v}$ – значения матрицы весовых коэффициентов (ядро фильтра).

Входной сигнал изображения подается на вход нейрона только в пределах ограниченной области, затем эта область смещается на один пиксель и подается на соответствующий ей нейрон, причем весовые коэффициенты одинаковы для всей группы сканируемых нейронов. После завершения сканирования всего изображения, данный процесс повторяется, но уже с другими весовыми коэффициентами для данной группы. Так как веса в пределах группы не меняются, то мы имеем окно с набором определенных чисел, которые умножаются на соответствующие входные значения и суммируются. Таким образом получается новый выходной сигнал.

Нейронная сеть «проходится» по изображению, выполняя операцию умножения над каждым элементом первой матрицы на входные данные, полученные в обучающей выборке, после чего получается один выходной пиксель. Таким образом получается новая матрица признаков, которые на выходе является взвешенными суммами признаков на входе (рисунокРисунок *3*).

Размерность набора данный нейросети определяет количество признаков, которые будут соединены для определения нового нейрона на выходном слое. После множественного выполнения данной работы появляется сверточная нейронная сеть

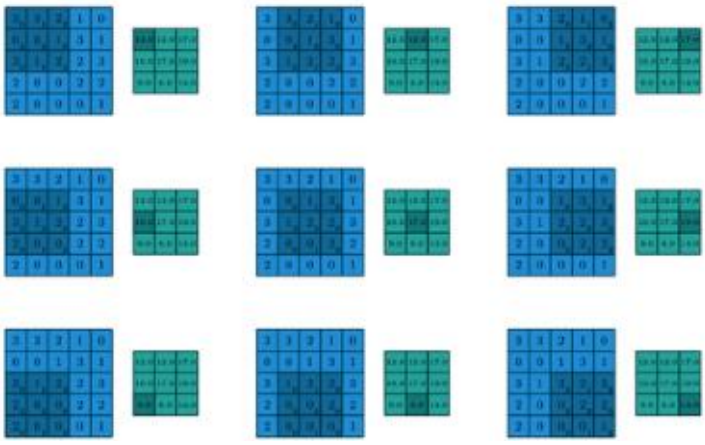

Рисунок 3 – Процесс вычисления матрицы C

В сверточных нейронных сетях часто применяются две техники:

– padding. Прибавляет дополнительные пиксели для того, чтобы у границ всегда была возможность оказаться в центре ядра (рисунокРисунок *4*);

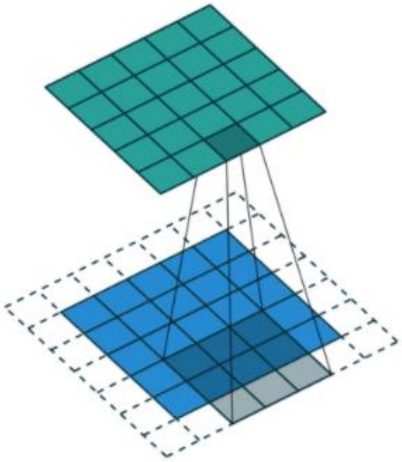

Рисунок 4 – Padding

– striding. Уменьшает исходное изображение, пропуская ячейки, выбранные случайным образом (рисунок Рисунок *5*).

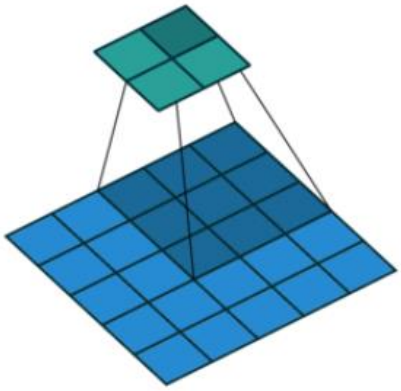

Рисунок 5 – Striding

Коллекция ядер представляет собой фильтр. Каждый фильтр в сверточном слое создает только один выходной канал, «проскальзывая» по соответствующим им входным каналам, создавая обработанную версию

каждого из них. Затем эти версии суммируются для формирования одного канал, а фильтр создает один общий выходной слой.

Характерной особенностью сверточной нейронной сети является то, что размер входных данных с каждым разом уменьшается, при этом количество каналов возрастает от раза к разу.

1.2.2 Многослойный перцептрон Розенблатта

Перцептрон – нейронная сеть, представленная в виде алгоритма для выполнения двоичной классификации. В свою очередь многослойный перцептрон представляется в виду глубокой искусственной нейросети, включающей в себе несколько перцептронов.

Многослойные персептроны состоят из входного слоя для приема сигнала, выходного слоя, который принимает решение или делает предсказание о входном объекте, а между ними – несколько скрытых слоев, которые являются истинным вычислительным движком (рисунок Рисунок *6*).

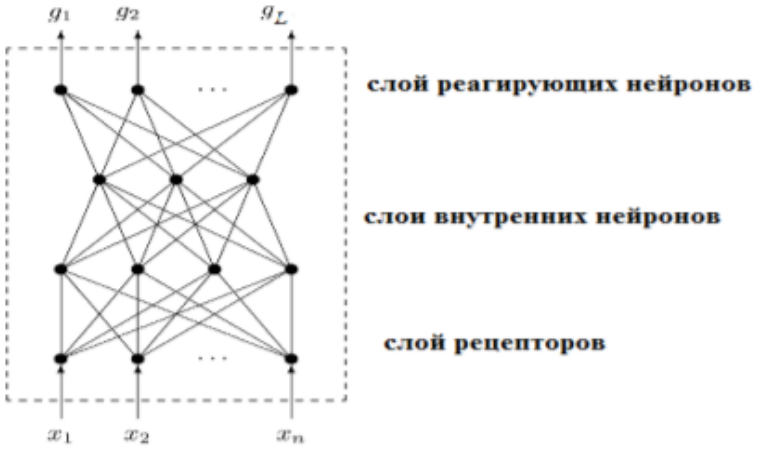

Рисунок 6 – Многослойный перцептрон с двумя внутренними слоями

Структура сети формируется по следующим правилам:

– связь могут иметь нейроны, находящиеся только в соседних слоях;

– связи внутри одного слоя отсутствуют;

– активационные функции одинаковы для всех внутренних нейронов.

Для решения задач классификации с двумя классами в качестве активационной функции используется $\Phi\,(z) = 1$ или $\Phi(z) = -1$. Процедура обучения включает в себя вычисление весовых коэффициентов $(w_0, .., w_n)$. Настройка параметров происходит по обучающим выборкам. На первом этапе происходит преобразование векторов, определяющих признаки для обучающей выборки.

Во время обучения перцептрона нулевое приближение вектора весов $(w_0^{(0)}, \dots, w_n^{(0)})$ определяется случайным образом. Обучающая выборка последовательно подается на вход перцептрона. Если описание $x^{(k)}$ на k-ом шаге классифицируется неправильно, то происходит корректировка по правилу

$$w^{(k+1)} = w^k + x.$$

В случае правильной классификации

$$w^{(k+1)} = w^k.$$

Процедура повторяется до тех пор, пока не достигнуто одно из условий:

– объекты из классов полностью распределены;

– при повторении заданного числа итераций не происходит улучшение разделения.

– достигнут заранее заданный лимит итераций.

Для настройки смещения и весовых коэффициентов используется алгоритм обратного распространения, где погрешность вычисляется различными способами. Перцептрон и подобные ему сети используют движения «вперед-назад». На шаге «вперед» направление сигнала перемещается от входного слоя через скрытые к выходному. При движении

«назад» используется правило дифференцирования сложных функций. Через перцептрон в обратном направлении идут частные производные, погрешности по весовым коэффициентам и смещениям. Если в процессе обучения исчезает погрешность, это означает, что сеть сошлась.

1.2.3 LSTM

LSTM – рекуррентная нейронная сеть с долгой кратковременной памятью. С ее помощью есть возможность выявлять признаки из временных последовательностей, а также обрабатывать многомерные данные.

При обучении данная модель способна схватывать существенные детали прошлого контекста и сохранять их, пока они актуальны.

В базовой версии LSTM состоит из ячеек. Она представляет собой цепочку с повторяющимися нейронами (рисунок Рисунок *7*)

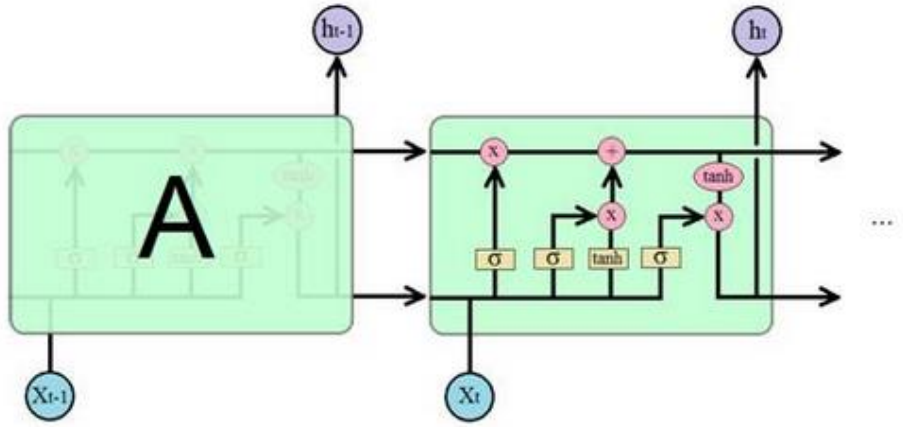

Рисунок 7 – Схема работы LSTM модели

Каждая стрелка является линией вектора. Розовый круг отвечает за математические операции над векторами. Под желтыми ячейками понимаются слои нейронной сети.

σ – сигмовидный слой модели (гейт), ограничивающий поступающие данные. В результате работы на выходе из данного слоя данные представлены в виде нуля или единицы, показывая какая часть каждой единицы проходит на следующий слой. «0» означает «не пропустить ничего», значение «1» – «пропустить все». В LSTM существует три гейта для контроля за состоянием ячейки.

Процесс работы модели состоит из следующих этапов: слой утраты, слой сохранения и новое состояние.

Слой гейта утраты решает какую информацию следует удалить из ячейки, а какую необходимо оставить. Он получает на вход h и x и выдает число от 0 до 1 для каждого номера в состоянии ячейки C.

$$f_t = \sigma(W_t \cdot [h_{t-1}, x_t] + b_f),$$

Второй этап отвечает за сохранение информации в ячейке. Он проходит в два этапа, на первом значения обновляются, после чего слой *tanh* создает вектор новых значений, они суммируются к уже полученному состоянию.

$$i_t = \sigma(W_t \cdot [h_{t-1}, x_t] + b_i),$$

$$\tilde{C}_t = tanh\sigma(W_C \cdot [h_{t-1}, x_t] + b_C)$$

После обновления старого состояния данные обновляются и получаются новые значения ячеек. Умножим старое состояние на f, теряя информацию, которую решили забыть. Затем добавляем i·C. Новые данные зависят от того, каким образом мы обновили их предыдущее состояние.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t.$$

После работы алгоритма получается новое состояние ячейки. После того, как сигмовидный слой указал какие значения следует удалить, пропускаем состояние ячейки через слой tanh (чтобы разместить все значения в интервале [-1, 1]) и умножаем его на выходной сигнал сигмовидного гейта.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o),$$

$$h_t = o_t \cdot tan\, h(C_t).$$

К достоинствам модели LSTM можно отнести способность эффективно строить нелинейные зависимости, точно описывающие исходные данные, параллелизм и адаптивность.

2 Временные ряды

2.1 Основные понятия прогнозирования временных рядов

Под временным (динамическим) рядом рассматривают значение некоторых дынных X в определенные периоды времени t. Уровнями ряда называются отдельные наблюдения, которые обозначаются x_t, t= 1,..,n.

Временной ряд можно представить в виде:

$$x_t = u_t + \gamma_t + c_t + \varepsilon_t, t = 1 \dots n,$$

где u_t – тренд, периодически изменяющаяся часть,

γ_t– сезонность, показывает цикличность каких-либо событий,

c_t – циклическая компонента, отражающая повторяемость процессов в течение длительных периодов времени свыше одного года,

ε_t – случайная компонента, отражающая влияние неподдающихся учету и регистрации случайных факторов.

Временной ряд представляет собой хаотичный процесс X(t) (t – время), а его члены, как правило, зависимы друг от друга и одинаково распределены.

Временные ряды обладают следующими сгруппированными признаками:

– по форме представления уровней: ряды абсолютных показателей, относительных показателей, средних величин;

– по количеству показателей, для которых указаны уровни в определенный момент времени: одномерные и многомерные временные ряды;

– по характеру временного параметра: моментные и интервальные временные ряды;

– по расстоянию между датами и интервалами времени: равностоящие, то есть, когда даты начала и окончания периодов следуют друг за другом с одинаковыми интервалами, неполные – интервалы разные.

– по полноте данных: полные и неполные;

– детерминированные и случайные. Первые получаются на основе значений некоторой неслучайной функции, вторые – результат реализации некоторой случайной величины;

– по наличию основной тенденции: стационарные и нестационарные ряды.

Можно выделить следующие этапы анализа и прогнозирования временных рядов:

– графическое представление и анализ составляющих ряда;

– проверка ряда на стационарность;

– сглаживание и фильтрация (удаление низко- или высокочастотных составляющих) временного ряда;

– построение модели прогнозирования;

– прогнозирование временного ряда на основе ранее проведенных исследований. [8]

Важный этап, с которого начинается сам анализ и прогнозирование, заключается в поиске данных и их подготовке.

Анализ временного ряда прежде всего начинается с поиска данных и приведения их к виду, доступному для прогнозирования. Необходимо определить является ли исследуемый временной ряд стационарным или нет.

Временной ряд x_t, t=1,..,n называется строго стационарным, если совместное распределение вероятностей n наблюдений x_1, x_2,…, $x_{n+\tau}$ при любых n, t, τ. То есть у стационарных временных рядов вероятностные характеристики не зависят от момента t. Поэтому математическое ожидание $M(x_t)=a$ и среднеквадратическое отклонение могут быть оценены по значениям x_t по формулам:

$$a = \bar{x}_t = \sum_{t=1}^{n} x_t / n,$$

$$\sigma = \sqrt{\frac{\sum_{t=1}^{n} (x_t - \bar{x}_t)^2}{n}}.$$

Из стационарности временного ряда следует, что такие свойства, как дисперсия, математическое ожидание и ковариация неизменны со временем. [**Ошибка! Источник ссылки не найден.**]

Для того, чтобы проверить временной ряд на стационарность, необходимо провести тест Дикки-Фуллера, который проверяет нулевую гипотезу о наличии единичного корня.

Если из обеих частей уравнения $x_t = p \cdot x_t + \varepsilon_t$, отображающего наш процесс, вычесть x_{t-1}, то получим $x_t - x_{t-1} = (p-1) \cdot x_{t-1} + \varepsilon_t$, где слева от равенства – первые разности. Из этого следует, если $p = 1$, то первые разности дадут стационарный белый шум, то есть временной ряд будет невозможно предсказать. Этот факт и лег в основу теста Дики-Фуллера на наличие единичных корней. [**Ошибка! Источник ссылки не найден.**]

Вводится термин нестационарности временного ряда и единичного корня, где $p = 1$. На этом основана нулевая гипотеза теста Дики-Фуллера.

В альтернативной гипотезе ряд обладает свойством стационарности и $p < 1$.

Для получения статистики, с помощью которой можно было бы проверить нулевую гипотезу, Дики и Фуллер предложили оценить данную авторегрессионную модель и взять из неё обычную t-статистику для гипотезы о том, что $p = 1$.

Если ошибки не коррелируют между собой, то переменная может быть описана авторегрессионным процессом первого порядка. Однако, существуют ограничения на использование такого подхода. Для определения авторегрессионных процессов более высоких порядков необходимо использовать модифицированный критерий Дики-Фуллера, который учитывает стационарность временного ряда. Уравнение для него приобретает следующий вид:

$$\Delta x_t = (p-1)x_{t-1} + \sum_{j=1}^{k} x_j \Delta x_{t-j} + \varepsilon_t,$$

$$\Delta x_t = (p-1)x_{t-1} + \varepsilon_t.$$

Распределения этих критериев асимптотически совпадают с соответствующими обычными распределениями Дики-Фуллера. Роль дополнительной авторегрессионной компоненты сводится к тому, чтобы убрать автокорреляцию из остатков. Процедура проверки гипотез не отличается от описанной выше.

Таким образом, если $p-\text{значение} > 0.05$, то не удается отклонить нулевую гипотезу, следовательно существует единичный корень и ряд является нестационарным.

Если же $p-\text{значение} < 0.05$, то нулевая гипотеза отклоняется, единичного корня не существует и ряд является стационарным.

Следующим этапом прогнозировании временного ряда является сглаживание.

2.2 Методы для построения моделей временных рядов

2.2.1 Регрессионная модель

Прогнозируемая переменная зависит от нескольких независимых переменных.

Выделяют 3 типа регрессионной модели прогнозирования:

– простая линейная регрессия,

– множественная регрессия,

– нелинейная регрессия;

В основу простой линейной регрессии положено предположение, что существует дискретный внешний фактор *X(t)*, оказывающий влияние на исследуемый процесс *Z(t)*, при этом связь между процессом и внешним фактором линейна.

$$Z(t) = \alpha_0 + \alpha_1 X(t) + \varepsilon_t,$$

где α_0 и α_1 – коэффициенты регрессии;

ε_t – ошибка модели.

Недостатком такого способа прогнозирования является то, что для получения значений *Z(t)* в момент времени *t* необходимо иметь значение *X(t)* в тот же момент времени *t*, что редко выполнимо на практике.

В действительности на процесс *Z(t)* оказывают влияние множество дискретных внешних факторов $X_1(t) \dots X_s(t)$, тогда применяется множественная регрессия, которая имеет вид:

$$Z(t) = \alpha_0 + \alpha_1 X_1(t) + \alpha_2 X_2(t) + \cdots + \alpha_s X_s(t) + \varepsilon_t.$$

В основу нелинейной регрессионной модели положено предположение о том, что существует известная функция, описывающая зависимость между исходным процессом *Z(t)* и внешним фактором *X(t)*.

$$Z(t) = F(X(t), A).$$

Основной недостаток регрессионных моделей является наличие заранее известных значений всех факторов.

2.2.2 Сезонная ARIMA модель SARIMA

Сезонная авторегрессионная интегрированная модель SARIMA или Seasonal ARIMA, является расширением ARIMA (p, d, q), которое поддерживает одномерные данные временных рядов с сезонным компонентом.

$$\Delta^d Y_t = \sum_{i=1}^{p} \alpha_i \Delta^d Y_{t-1} + \sum_{j=1}^{q} \beta_j \varepsilon_{t-j} + \varepsilon_t,$$

где ε_t – стационарный временной ряд,

α_i, β_j – параметры модели,

Δ^d – оператор разности d-го порядка (последовательное взятие d раз разностей первого порядка — сначала от временного ряда, затем от полученных разностей первого порядка, затем от второго порядка и т. д.)

Модель SARIMA (p, d, q)(P, D, Q)s является подвидом модели ARIMA, различие состоит в том, что в первой присутствует сезонность. Дополнительно в такой модели вводятся сезонные параметры (P, D, Q, s), позволяющие учесть циклические колебания процесса, где P – порядок сезонной составляющей SAR(P), D – порядок интегрирования сезонной составляющей, Q – порядок сезонной составляющей SMA(Q), s— размерность сезонности (месяц, квартал и т.д.).

Несмотря на свою популярность, модель SARIMA является сложной в реализации, так как дополнительно требуется определять сезонные параметры. В языке Python3 нет отдельного метода для построения данной модели, но можно задать функцию для нахождения параметров перебором. В этом случае необходимо сохранять наилучшую модель для дальнейшего обучения.

Данная модель может не подойти для некоторых проектов, так как использует большое количество ресурсов, в том числе времени, на первоначальных этапах подготовки, сложно настраивается, а также требует переобучения на новых данных.

3 Телеграм-бот с применением нейронных сетей

3.1 Постановка задачи

Для прогнозирования стоимости акций на будущий период времени выбранного предприятия необходим набор данных, взятых прошлый период времени в определенном количестве. Вся выборка данных разбивается на обучающую и контрольную.

Рассмотрим следующий ряд данных (рисунок Рисунок *8*):

Date	Open	High	Low	Close	Adj Close	Volume
2015-01-02 00:00:00-05:00	18.020000	18.440001	17.780001	18.370001	18.370001	2384500
2015-01-05 00:00:00-05:00	18.389999	18.430000	17.290001	17.410000	17.410000	2430300
2015-01-06 00:00:00-05:00	17.370001	17.870001	17.180000	17.530001	17.530001	1904500
2015-01-07 00:00:00-05:00	17.790001	17.870001	17.330000	17.370001	17.370001	1471300
2015-01-08 00:00:00-05:00	17.680000	18.930000	17.680000	18.420000	18.420000	3764200
...	...	...	...	...	...	...
2023-05-23 00:00:00-04:00	18.940001	18.940001	18.940001	18.940001	18.940001	0
2023-05-24 00:00:00-04:00	18.940001	18.940001	18.940001	18.940001	18.940001	0
2023-05-25 00:00:00-04:00	18.940001	18.940001	18.940001	18.940001	18.940001	0
2023-05-26 00:00:00-04:00	18.940001	18.940001	18.940001	18.940001	18.940001	0
2023-05-30 00:00:00-04:00	18.940001	18.940001	18.940001	18.940001	18.940001	0

2116 rows × 6 columns

Рисунок 8 – Таблица с данными по компании Яндекс

Рассматривается стоимость акций компании Яндекс, данные взяты с сайта Yahoo.finance в период с 02.01.2015 по 31.05.2023. Набор данных представляет собой файл, содержащий 2116 строк и 6 столбцов. Столбец Date содержит информацию о днях, Open показывает цену акции на момент открытия фондовой биржи, Close – закрытия. High и Low отображают максимальный и минимальный уровень цены во время торгов, Adj Close – скорректированная стоимость на момент закрытия, Volume показывает

количество сделок в данном временном периоде. В процессы работы будет прогнозироваться стоимость акций на дальнейший период времени, что поможет людям, работающим с биржей строить свою инвестиционную стратегию, отталкиваясь от показателей, показанных в результате прогноза.

Для работы нейронной сети понадобятся колонка Close.

Результаты прогнозов должны быть представлены в виде графика, который отправляется пользователю в чат.

В данной работе применяется нейронная сеть с долгой краткосрочной памятью LSTM, а также модель SARIMA. Выбор предоставляется посредством кнопок. Пользователь может выбрать на какой срок делать прогноз: следующий день или ближайшие 10. Исходя из полученных данных, человек может выстроить свою стратегию на бирже.

3.2 Программная реализация

3.2.1 Построение математической модели и выбор средств реализации

Для решения поставленной задачи было принято решение реализовать 2 модели – SARIMA и нейронная сеть LSTM. Разработка алгоритма и программная реализация поведения финансового временного ряда будет выполнена на языке Python, так как он является одним из самых популярных языков и включает в с набор библиотек для работы с машинным обучением, таких как Keras, NumPy, Pandas и Tensorflow.

Все поступающие для обработки нейронной сетью данные преобразовываются в NumPy-массив, который вводится в библиотеке NumPy. Для данной структуры в этом модуле реализованы специальные операции, рассмотрим некоторые из них:

– yf.download () – позволяет получать данные с сайта Yahoo.finance без скачивания файла на локальный компьютер,

– filter () – метод, позволяющий отобрать колонки, которые будут присутствовать в новом массиве данных,

– Sequential() – функция, возвращающая незаполненный объект нейронной сети, то есть некоторый каркас модели из определённого количества слоёв,

– model.add() – метод для добавления слоев в нейронную сеть;

Рассмотрим также некоторые методы библиотеки Keras, которые используются при реализации нейронной сети:

– Sequential() – функция, возвращающая незаполненный объект нейронной сети, то есть некоторый каркас модели из определённого количества слоёв,

– model.add() – метод для добавления слоев в нейронную сеть,

– model.summary() – графически отображает информацию о нейронной сети;

Разработка интерфейсной части приложения велась с использованием библиотека aiogram, предназначенной для создания Телеграмм-ботов.

Работа телеграмм бота построена на взаимодействии хэндлеров, которые отлавливают введенные пользователем значения:

– @bot.message_handler () – обработчик события, который реагирует на определённые команды, либо формат посылаемых данных,

– @bot.callback_query_handler () – работа с кнопками, с помощью которых пользователь может выбрать из предлагаемых вариантов;

Ниже представлена схема нейронной сети, в которой указывается количество выходных нейронов в каждом слое (рисунок Рисунок *9*, рисунок Рисунок *10*).

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 49, 100)           41200

 dropout (Dropout)           (None, 49, 100)           0

 lstm_1 (LSTM)               (None, 49, 100)           80400

 lstm_2 (LSTM)               (None, 100)               80400

 dropout_1 (Dropout)         (None, 100)               0

 dense (Dense)               (None, 10)                1010

=================================================================
Total params: 203,010
Trainable params: 203,010
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 9 – Сводное представление модели

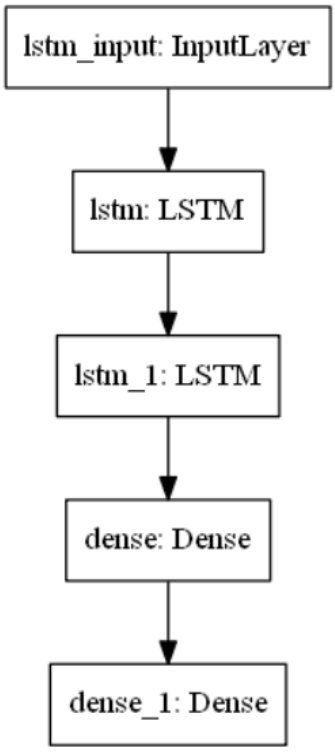

Рисунок 10 – Схема слоев нейронной сети

Для обучения нейросети 60% данных отдавались на обучающую выборку, остальные 40% на тестовую. В процессе подбирались наиболее подходящие параметры модели, опытным путем было установлено, что сочетание функции потерь mean_squared_error, функции оптимизации adam и обучение на 10 эпохах показывают наилучший результат.

3.2.2 Модель прогнозирования LSTM

Работа с ботом начинается со слова «Прогноз». Далее пользователя просят загрузить файл с данными (рисунок Рисунок *11*).

Рисунок 11 – Загрузка файла

Необходимо загрузить файл с данными формата csv, структура файла может быть любая. Например, если брать данные с сайта Yahoo.finance, то стандартный файл содержит 6 столбцов, а количество строк зависит от выбранного периода времени. Чем больше период – тем больше строк будет в файле.

Затем необходимо выбрать каким методом будет осуществляться прогнозирование. Дается два варианта: LSTM и SARIMA, пользователь выбирает понравившийся ему вариант для прогнозирования.

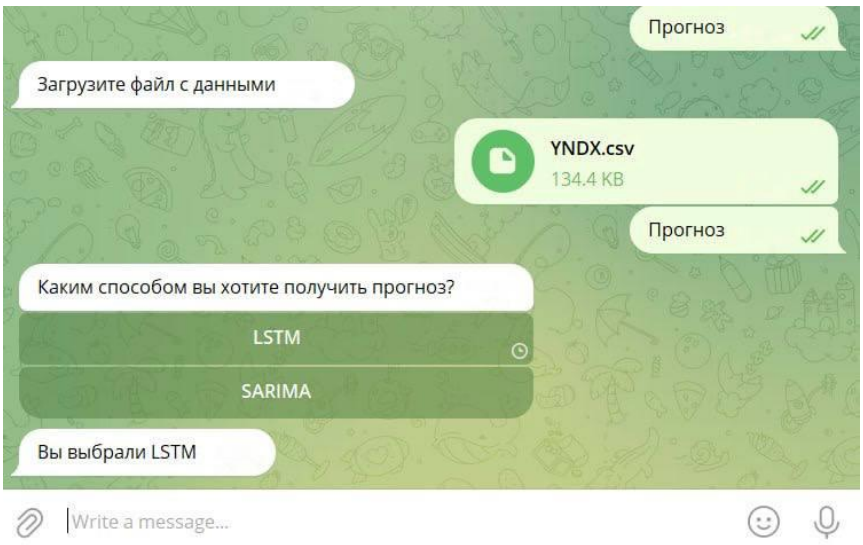

Рисунок 12 – Выбор нейронной сети

Выше показан (рисунок Рисунок *12*) выбор LSTM. Приходит сообщение с информацией о сделанном выборе.

Следующим шагом необходимо выбрать период для прогнозирования стоимости акций. В данном случае представим, что пользователь выбирать ближайшие 10 дней (рисунок Рисунок *13*).

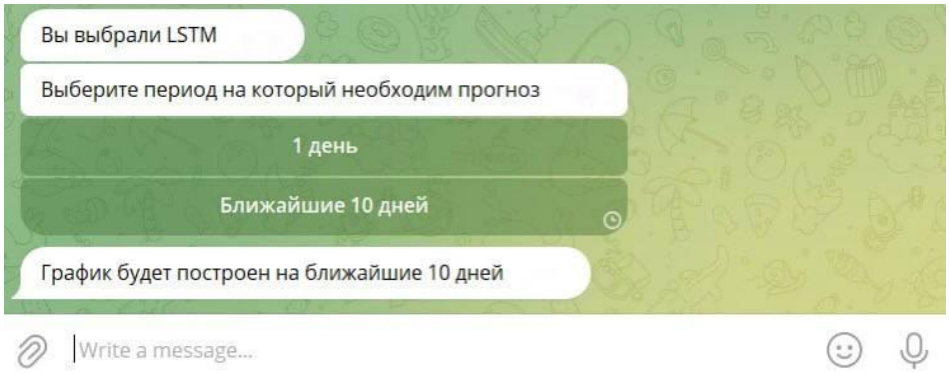

Рисунок 13 – Выбор периода для прогнозирования

В результате работы телеграмм бота пользователю в ответном сообщении отправляется файл формата .jpg с графиком цены на акции компании (рисунок Рисунок *14*).

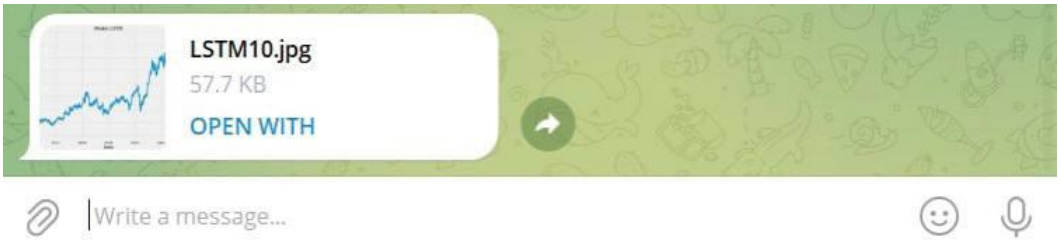

Рисунок 14 – Результат работы бота и нейронной сети

Для подробного изучения построенного прогноза, нужно нажать пользователю на кнопку OPEN WITH. Если мы нажнем на сообщение (рисунок Рисунок *14*), то открывается картинка на весь экран для удобства просмотра (рисунок Рисунок *15*).

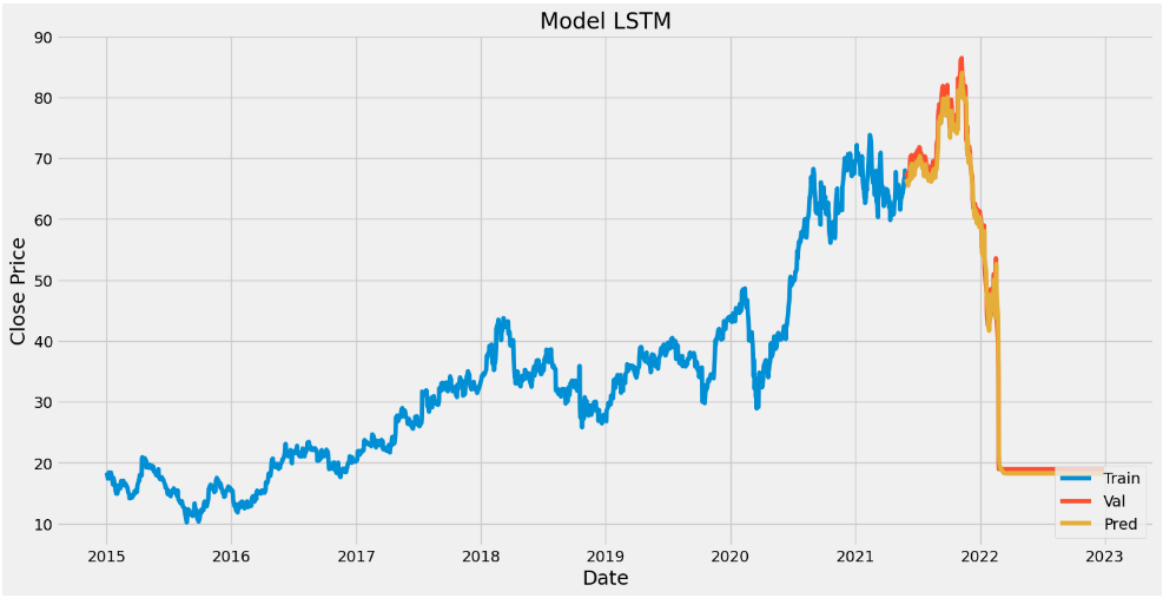

Рисунок 15 – Открывшийся график прогнозной цены.

Для того, чтобы проверить достоверность результатов и качество обучения нейронной сети, построим график реальных данных (рисунок Рисунок *16*). Можно увидеть, что нейронная сеть обучена достаточно хорошо, чтобы делать прогнозы с небольшой погрешностью обучения.

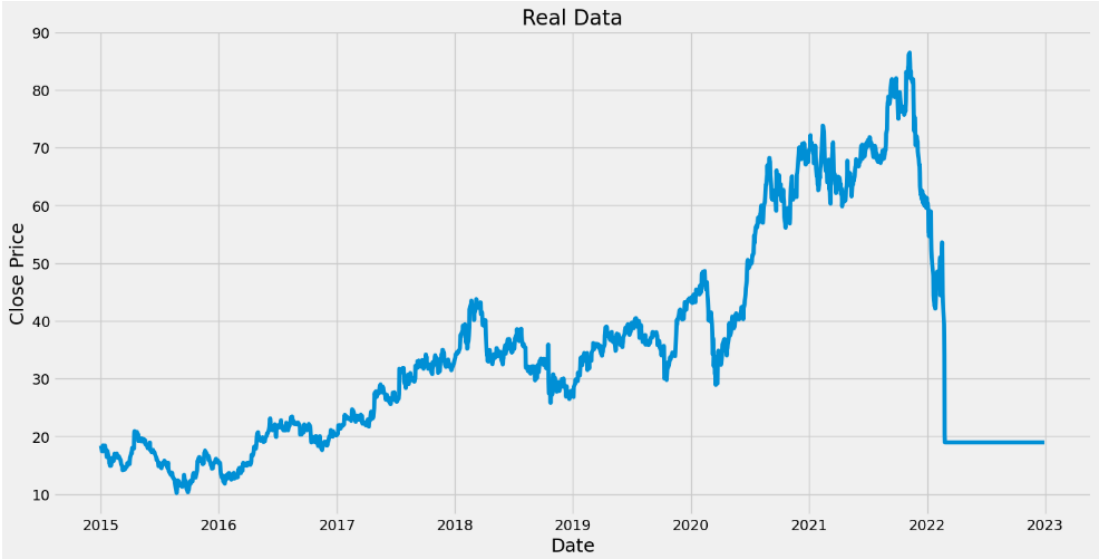

Рисунок 16 – График реальной цены за выбранный период

Теперь покажем результат работы в случае, когда пользователь выбирает прогнозирование на один день. Все также загружается файл с данными и предоставляется возможность выбрать способ прогнозирования (Рисунок 17 – Загрузка файла и выбор нейронной сети

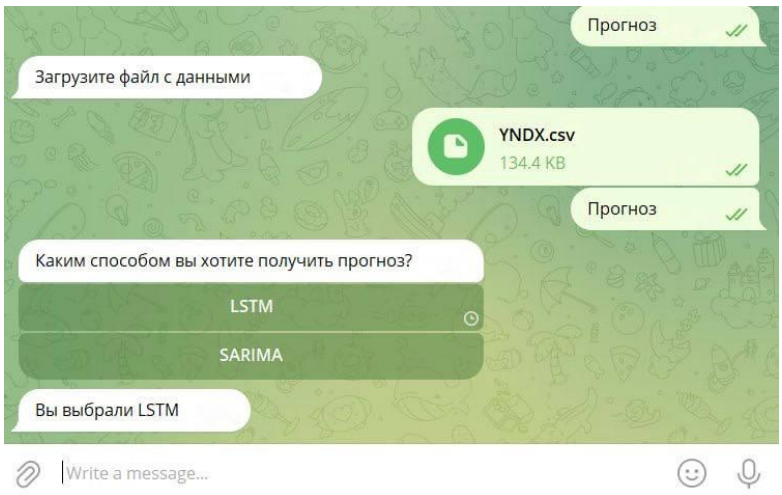

Рисунок 17 – Загрузка файла и выбор нейронной сети

Предположим, пользователь выбирает прогнозирование на следующий день (Рисунок 18).

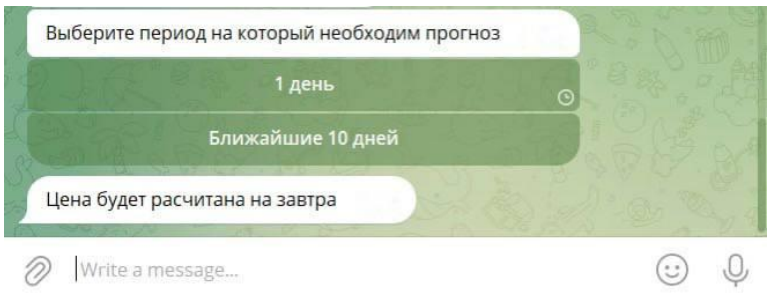

Рисунок 18 – выбор периода для прогнозирования с помощью LSTM

В конечном итоге, мы опять получим график, но на этот раз на нем будет отображен другой интервал прогнозной цены (Рисунок 19).

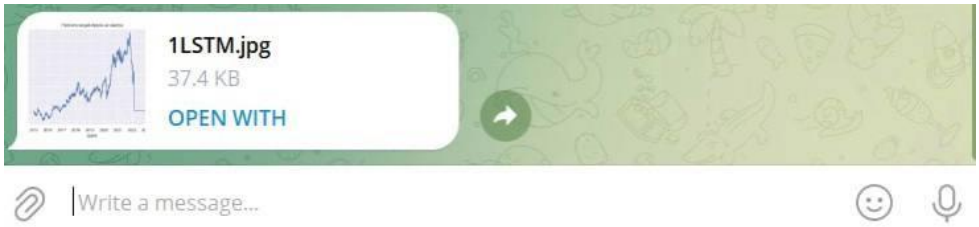

Рисунок 19 – Результат работы бота

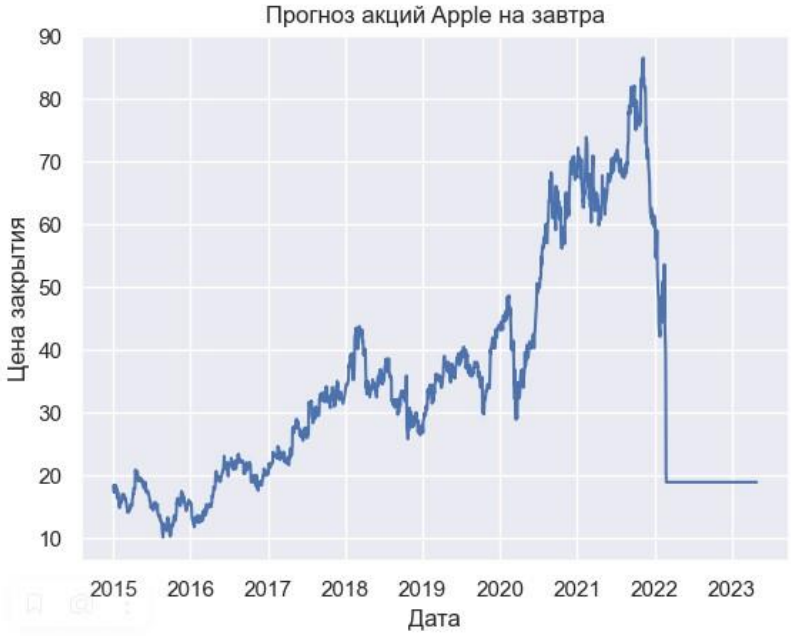

Рисунок 20 – Итоговый результат работы LSTM

3.2.3 Модель прогнозирования SARIMA

Рассмотрим вариант, при котором пользователь выбирает другие варианты для прогнозирования. В данном случае результат будет представлен в виде одного числа, что означает цену акции на следующий день.

Для работы нам так же понадобится опять загрузить файл с данными (рРисунок 21).

Рисунок 21 – Загрузка файла с данными

Затем, как и в первом примере пользователю дается на выбор два способа прогнозирования. Сделаем выбор в пользу модели SARIMA и получим подтверждающее сообщение (Рисунок 22).

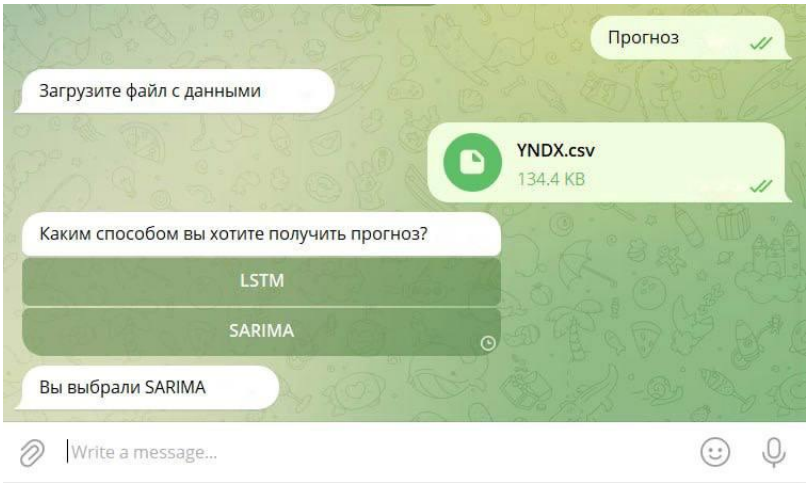

Рисунок 22 – Выбор способа прогнозирования

Далее, выбираем период, на который необходимо сделать прогноз. В этом случае представим, что пользователю необходимы результаты на завтрашний день (рисунок 23).

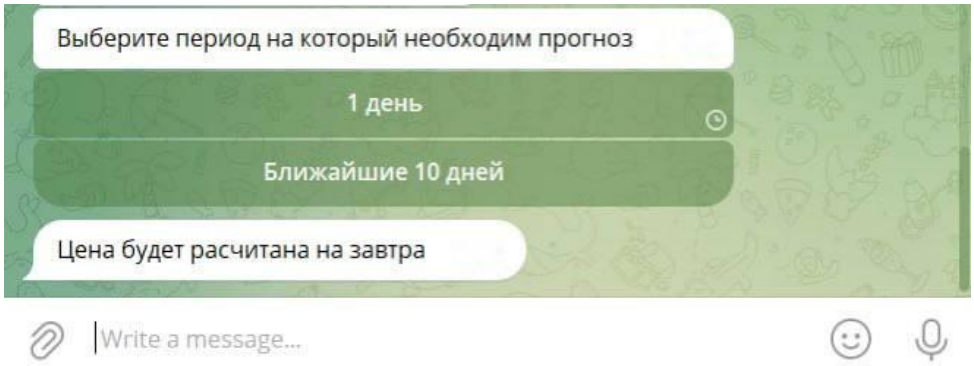

Рисунок 23 – Выбор периода для прогнозирования

В итоге получаем итоговый расчет в виде цены на завтрашний день (рисунок 24) сравним с ценой, которая была в действительности (рисунок 25).

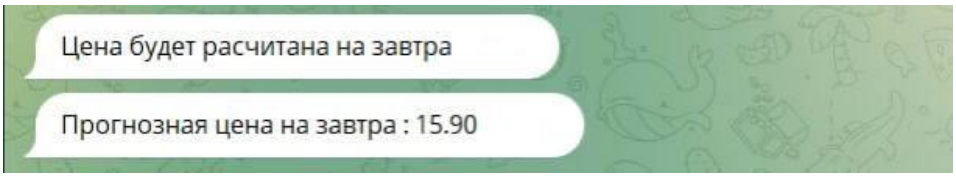

Рисунок 24 – Рассчитанная цена на один день

Date	Open	High	Low	Close*
Jun 07, 2023	23.94	24.05	17.83	18.94
Jun 06, 2023	18.94	18.94	18.94	18.94
Jun 05, 2023	18.94	18.94	18.94	18.94
Jun 02, 2023	18.94	18.94	18.94	18.94
Jun 01, 2023	18.94	18.94	18.94	18.94
May 31, 2023	18.94	18.94	18.94	18.94
May 30, 2023	18.94	18.94	18.94	18.94
May 26, 2023	18.94	18.94	18.94	18.94

Рисунок 25 – Фактическая цена на предсказанный день

Теперь покажем работу бота и модели SARIMA в случае прогноза на 10 дней. Привычным способом загружаем и выбираем модель прогнозирования (рисунок 26).

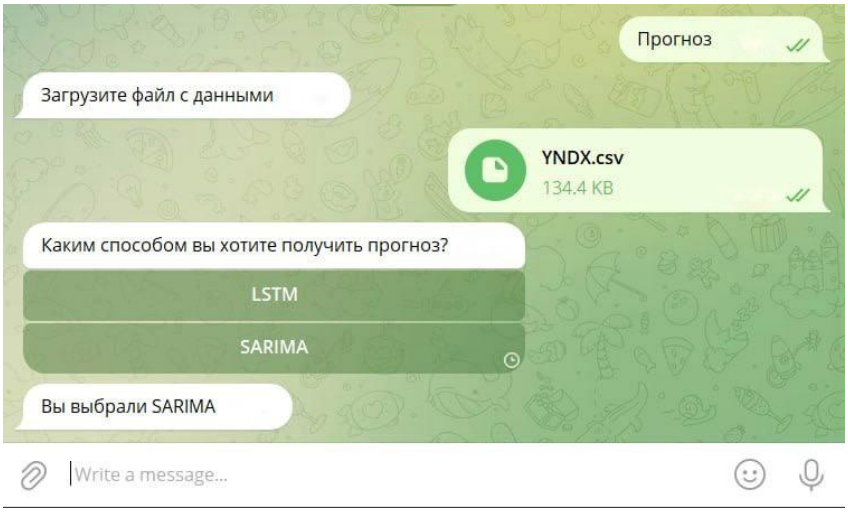

Рисунок 26 – Выбор модели SARIMA

Затем пользователь может выбрать на какой срок ему необходим прогноз, отталкиваясь от своей стратегии (рисунок 27).

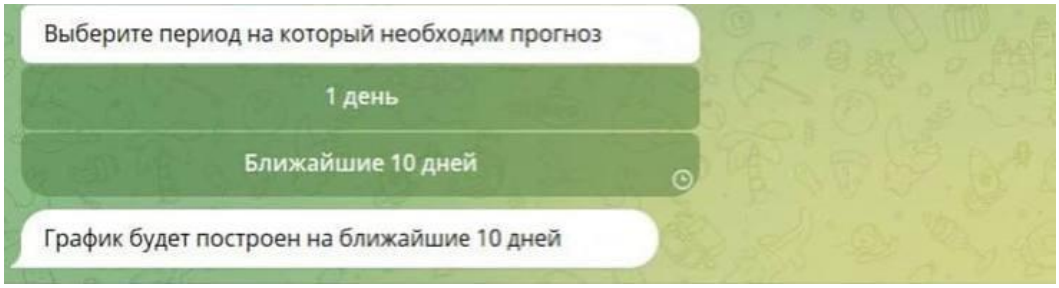

Рисунок 27 – Выбор десятидневного прогноза

После работы алгоритма пользователю в чат присылается сообщение, в котором расписана цена акций на каждый из 10 дней (рисунок 28).

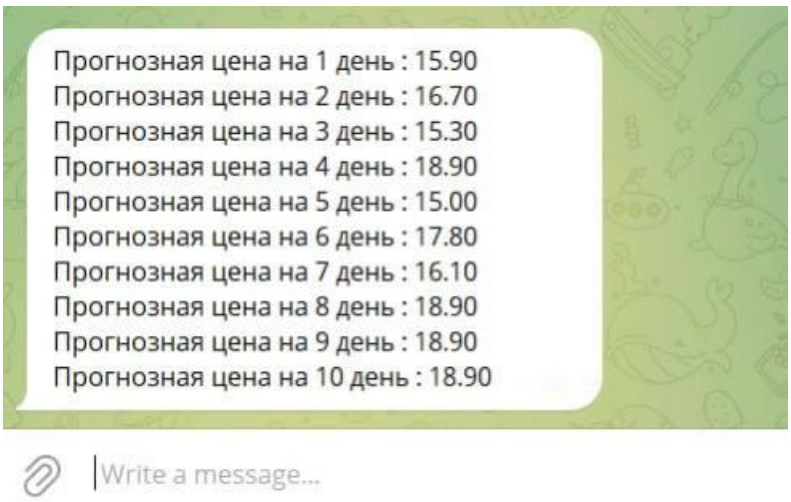

Рисунок 28 – Модель SARIMA с периодом прогнозирования 10 дней

Исходя из полученных результатов, можно сделать вывод, что наиболее хорошо справляется со своей задачей нейронная сеть с долгой краткосрочной памятью, она лучше улавливает колебания временного ряда, его резкие спады и подъемы, а также не требует дополнительных действий по нормализации.

ЗАКЛЮЧЕНИЕ

В условиях мировых кризисов и экономической нестабильности, данные разработки могут оказать важную помощь для людей, которые каким-либо образом связаны с торговлей на бирже.

В результате выполнения выпускной квалификационной работы были выполнены все поставленные задачи и цели, в частности, изучена теоретическая основа построения временных рядов, нейронных сетей, было выбрано программное средство для разработки телеграмм бота. В его основу легла библиотека Aiogram для создания программного кода, построены математические и информационные модели нейронной сети, на их основе разработан телеграмм бот, позволяющий получить результаты прогнозирования стоимости курса акций для выбранного временного ряда на основе нейронной сети с долгой краткосрочной памятью и модели SARIMA. Предоставляемый результат оформлен в виде графика, либо простого текстового сообщения, а также проведен сравнительный анализ результатов прогнозирования.

Предложенная разработка может быть использована для специалистов компаний, участвующих в торгах на бирже, таких как биржевые маклеры, комиссионеры, брокеры, дилеры. Данное приложение может послужить основой для дальнейшей разработки программного обеспечения в финансовой сфере для прогнозирования финансовой устойчивости разного рода предприятий. Исходя из представленных результатов, можно сделать вывод, что сеть LSTM справилась лучше, уловив значения цен, их рост и падение.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Как работает нейронная сеть: обучение, функции активации и потери. // Neurohive : – 2018. – URL: https://neurohive.io/ru/osnovy-data-science/osnovy-nejronnyh-setej-algoritmy-obuchenie-funkcii-aktivacii-i-poteri/ (дата обращения 18.02.2023).

2. Коэльо, Л.П. Построение систем машинного обучения на языке Python / Л. П. Коэлью, В. Ричарт. – Москва: ДМК Пресс, 2016. – 302 с. – ISBN 978-5-97060-330-7.

3. Эпоха, батч, итерация – в чем разница? // Neurohive : – 2018. – URL: https://neurohive.io/ru/osnovy-data-science/jepoha-razmer-batcha-iteracija (дата обращения 03.04.2023).

4. Грас, Дж. DataScience. Наука о данных с нуля / Дж. Грас – Санкт-Петербург: БХВ-Петербург, 2017. – 336с. – ISBN 978-5-9775-3758-2.

5. Халафян, А.А. STATISTICA 6. Статистический анализ данных. / А. А. Халафян – Москва: ООО “Бином-Пресс”, 2007. – 512 с. – ISBN 978-5-9518-0215-6.

6. Элбон, К. Машинное обучение с использованием Python. Сборник рецептов. / К. Элбон – Санкт-Петербург: БХВ-Петербург, 2019. – 384 с. – ISBN 978-5-9775-4056-8.

7. Открытый курс машинного обучения. Тема 9. Анализ временных рядов с помощью Python / 2021. URL: https://habr.com/ru/company/ods/blog/327242/ (дата обращения: 05.04.2023).

8. Временной ряд. / 2021. URL: https://blog.skillfactory.ru/glossary/vremennoj-ryad-2/ (дата обращения: 26.04.2023).

9. LSTM — нейронная сеть с долгой краткосрочной памятью. / 2021. URL: https://neurohive.io/ru/osnovy-data-science/lstm-nejronnaja-set/ (дата обращения: 01.05.2023).