МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

Дисциплина нейронные сети

**ОТЧЕТ**

по лабораторной работе №2

«Нейронные сети глубокого обучения»

Вариант

«Финансовая аналитика»

Обучающийся группы 6131-020402D В.Д. Гижевская

Обучающийся группы 6131-020402D Д.А. Кремнёв

Руководитель А.Н. Жданова

Самара 2022

СОДЕРЖАНИЕ

[1 Описание и анализ предметной области 3](#_Toc122514849)

[1.1 Нейронная сеть 3](#_Toc122514850)

[1.2 Постановка задачи 5](#_Toc122514851)

[2 Проектирование системы 6](#_Toc122514852)

[2.1 Одноступенчатые модели 6](#_Toc122514853)

[2.2 Многошаговые модели 7](#_Toc122514854)

[3 Описание результатов эксперимента 10](#_Toc122514855)

[3.1 Набор данных и окно 10](#_Toc122514856)

[3.2 Одноступенчатые модели 13](#_Toc122514857)

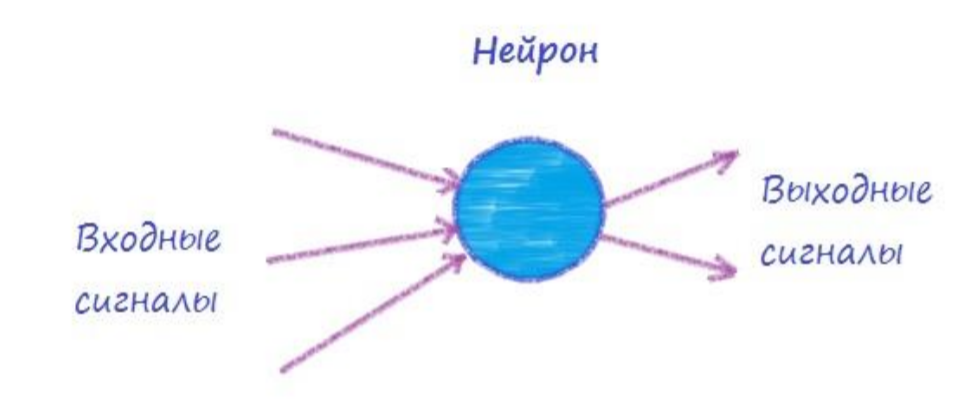
[3.3 Многошаговые модели 17](#_Toc122514858)

[ЗАКЛЮЧЕНИЕ 23](#_Toc122514859)

1. Описание и анализ предметной области
   1. Нейронная сеть

Изучение и использование искусственных нейронных сетей, в принципе, началось уже достаточно давно – в начале 20 века, но по-настоящему широкую известность они получили несколько позже. Связано это, в первую очередь, с тем, что стали появляться продвинутые (для того времени) вычислительные устройства, мощности которых были достаточно велики для работы с искусственными нейронными сетями. По сути, на данный момент можно легко смоделировать нейронную сеть средней сложности на любом персональном компьютере.

Нейронная сеть представляет из себя совокупность нейронов, соединённых друг с другом определённым образом. Рассмотрим один нейрон, его модель представлена на рисунке 1.

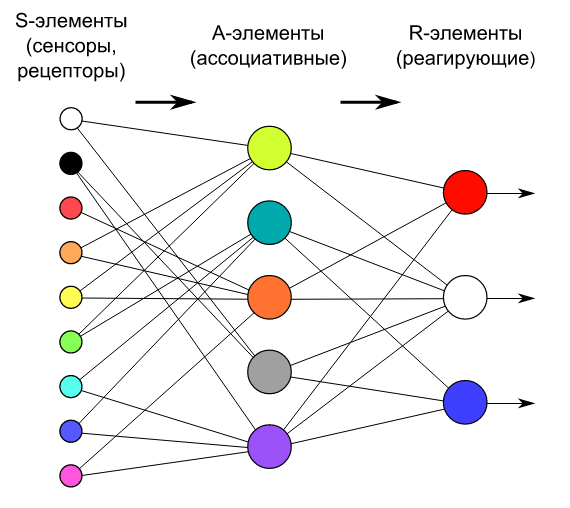
  
Рисунок 1 – Модель нейрона

Нейрон представляет из себя элемент, который вычисляет выходной сигнал (по определённому правилу) из совокупности входных сигналов. То есть основная последовательность действий одного нейрона такая:

* приём сигналов от предыдущих элементов сети;
* комбинирование входных сигналов;
* вычисление выходного сигнала;
* передача выходного сигнала следующим элементам нейронной сети.

Между собой нейроны могут быть соединены абсолютно по-разному, это определяется структурой конкретной сети. Но суть работы нейронной сети остаётся всегда одной и той же. По совокупности поступающих на вход сети сигналов на выходе формируется выходной сигнал (или несколько выходных сигналов). То есть нейронную сеть упрощённо можно представить в виде чёрного ящика, у которого есть входы и выходы. А внутри этого ящика сидит огромное количество нейронов.

Персептрон (Perceptron) – простейший вид нейронных сетей. В основе лежит математическая модель восприятия информации мозгом, состоящая из сенсоров, ассоциативных и реагирующих элементов. Модель персептрона представлена на рисунке 2.

  
Рисунок 2 – Модель персептрона

Разные исследователи по-разному его определяют. В самом общем своём виде (как его описывал Розенблатт) он представляет систему из элементов трёх разных типов: сенсоров, ассоциативных элементов и реагирующих элементов.

Рассмотрим принцип работы персептрона. Первыми в работу включаются S-элементы. Они могут находиться либо в состоянии покоя (сигнал равен 0), либо в состоянии возбуждения (сигнал равен 1). Далее сигналы от S-элементов передаются A-элементам по так называемым S-A связям. Эти связи могут иметь веса, равные только -1, 0 или 1. Затем сигналы от сенсорных элементов, прошедших по S-A связям, попадают в A-элементы, которые ещё называют ассоциативными элементами. Стоит заметить, что одному A-элементу может соответствовать несколько S-элементов. Если сигналы, поступившие на A-элемент, в совокупности превышают некоторый его порог ​θ​, то этот A-элемент возбуждается и выдаёт сигнал, равный 1. В противном случае (сигнал от S-элементов не превысил порога A-элемента), генерируется нулевой сигнал.

Далее сигналы, которые произвели возбуждённые A-элементы, направляются к сумматору (R-элемент). Однако, чтобы добраться до R-элемента, они проходят по A-R связям, у которых тоже есть веса. Здесь они уже могут принимать любые значения (в отличие от S-A связей).

Далее. R-элемент складывает друг с другом взвешенные сигналы от A-элементов и, если превышен определённый порог, генерирует выходной сигнал, равный 1. Это означает, что в общем потоке информации от глаз мы распознали лицо человека. Если порог не превышен, то выход персептрона равен -1. То есть мы не выделили лицо из общего потока информации.

* 1. Постановка задачи

Во время лабораторной работы необходимо разработать программу, с помощью которой можно прогнозировать изменения финансовых временных рядов. Используемый набор данных – цены акций Tesla Inc. в период с 2010 по 2020 год.

1. Проектирование системы
   1. Одноступенчатые модели

Самая простая модель, которую вы можно построить, — это модель, которая предсказывает значение одной функции — 1 временной шаг в будущее, основываясь только на текущих условиях.

**Базовый уровень.**

Перед созданием обучаемой модели было бы неплохо иметь базовый уровень производительности в качестве точки для сравнения с более поздними более сложными моделями.

Эта первая задача состоит в том, чтобы предсказать значение на один шаг вперед, учитывая текущее значение всех признаков.

**Линейная модель.**

Самая простая обучаемая модель, которую можно применить к этой задаче, — это вставить линейное преобразование между входом и выходом. В этом случае результат временного шага зависит только от этого шага.

Одним из преимуществ линейных моделей является то, что их относительно просто интерпретировать. Вы можете вытащить веса слоя и визуализировать вес, назначенный каждому входу.

**Полносвязная модель.**

Прежде чем применять модели, которые фактически работают с несколькими временными шагами, стоит проверить производительность более глубоких и мощных моделей с одним входным шагом.

**Многоступенчатая полносвязная модель.**

Одношаговая модель не имеет контекста для текущих значений входных данных. Он не может видеть, как входные объекты меняются с течением времени. Чтобы решить эту проблему, модели требуется доступ к нескольким временным шагам при прогнозировании.

Основным недостатком этого подхода является то, что результирующая модель может выполняться только на входных окнах именно такой формы.

**Свёрточная нейронная сеть.**

Слой свертки использует несколько временных шагов в качестве входных данных для каждого прогноза.

**Рекуррентная нейронная сеть.**

Рекуррентная нейронная сеть (RNN) — это тип нейронной сети, хорошо подходящий для данных временных рядов. RNN обрабатывают временной ряд шаг за шагом, сохраняя внутреннее состояние от шага к шагу.

**Модели с несколькими выходами.**

До сих пор все модели предсказывали одну выходную характеристику для одного временного шага.

Все эти модели можно преобразовать для прогнозирования нескольких функций, просто изменив количество единиц в выходном слое и настроив окна обучения, чтобы включить все функции.

* 1. Многошаговые модели

Как модели с одним выходом, так и модели с несколькими выходами делали прогнозы с одним временным шагом.

Сейчас мы рассмотрим, как расширить эти модели, чтобы делать прогнозы с несколькими временными шагами.

В многоэтапном прогнозировании модель должна научиться прогнозировать диапазон будущих значений. Таким образом, в отличие от одноступенчатой ​​модели, в которой предсказывается только одна точка будущего, многоступенчатая модель предсказывает последовательность будущих значений.

Есть два грубых подхода к этому:

* Прогнозы одиночного выстрела, когда весь временной ряд прогнозируется сразу.
* Прогнозы авторегрессии, при которых модель делает только одношаговые прогнозы, а ее выходные данные возвращаются в качестве входных данных.

**Базовые показатели.**

Простая базовая линия для этой задачи — повторить последний временной шаг ввода для необходимого количества выходных временных шагов.

Поскольку эта задача состоит в том, чтобы предсказать 24 часа в будущем, учитывая 24 часа в прошлом, другой простой подход состоит в том, чтобы повторить предыдущий день, предполагая, что завтра будет похоже.

**Одноразрядные модели.**

Одним из высокоуровневых подходов к этой проблеме является использование «однократной» модели, в которой модель делает прогноз всей последовательности за один шаг.

* Линейный. Простая линейная модель, основанная на последнем входном временном шаге, работает лучше любой базовой линии, но у нее недостаточно мощности. Модель должна прогнозировать временные шаги на основе одного входного временного шага с линейной проекцией.
* Полносвязная. Добавление между входом и выходом делает линейную модель более мощной, но по-прежнему основано только на одном временном шаге ввода.
* СНН. Сверточная модель делает прогнозы на основе истории фиксированной ширины, что может привести к более высокой производительности, чем плотная модель, поскольку она может видеть, как все меняется с течением времени.
* PHH. Рекуррентная модель может научиться использовать длинную историю входных данных, если она имеет отношение к прогнозам, которые делает модель. Здесь модель будет накапливать внутреннее состояние за 24 часа, прежде чем сделать один прогноз на следующие 24 часа.

**Авторегрессионная модель.**

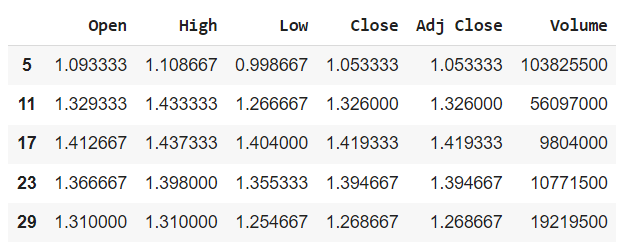
Все вышеперечисленные модели предсказывают всю выходную последовательность за один шаг.

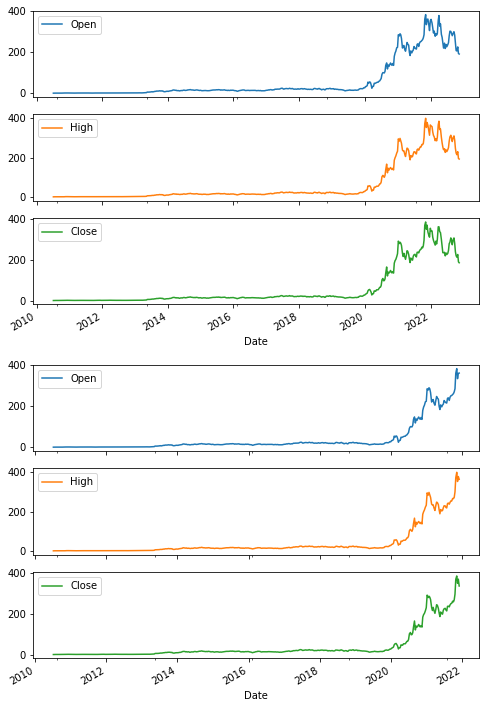
В некоторых случаях для модели может быть полезно разложить этот прогноз на отдельные временные шаги. Затем выходные данные каждой модели могут быть возвращены в себя на каждом шаге, и прогнозы могут быть сделаны в зависимости от предыдущего.

1. Описание результатов эксперимента
   1. Набор данных и окно

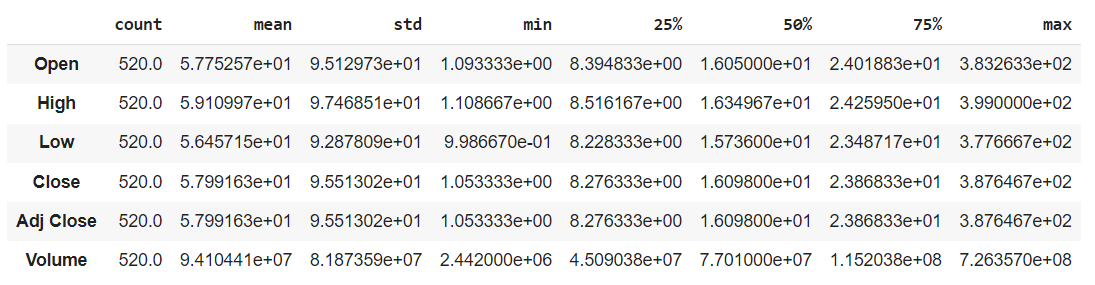
Для эксперимента использовался набор данных цен акций компании Tesla Inc. с 2010 по 2020 год. (рисунок 3).

В этом эксперименте будут рассматриваться только ежедневные прогнозы. Эволюция некоторых функций с течением времени приведена на рисунке 4.

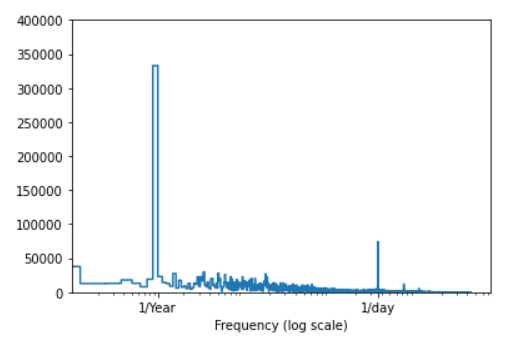
  
Рисунок 3 – Данные

  
Рисунок 4 – Эволюция некоторых функций с течением времени

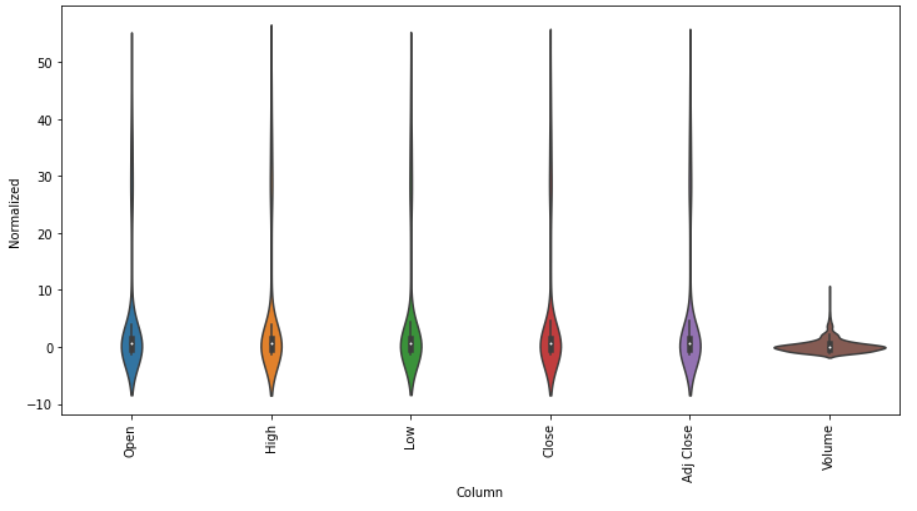
Посмотрим на статистику набора данных (рисунок 5).

  
Рисунок 5 – Статистика набора данных

Определим, какие частоты важны, извлекая признаки с помощью быстрого преобразования Фурье. Чтобы проверить предположения, вот tf.signal.rfft изменения цены закрытия торгов (рисунок 6).

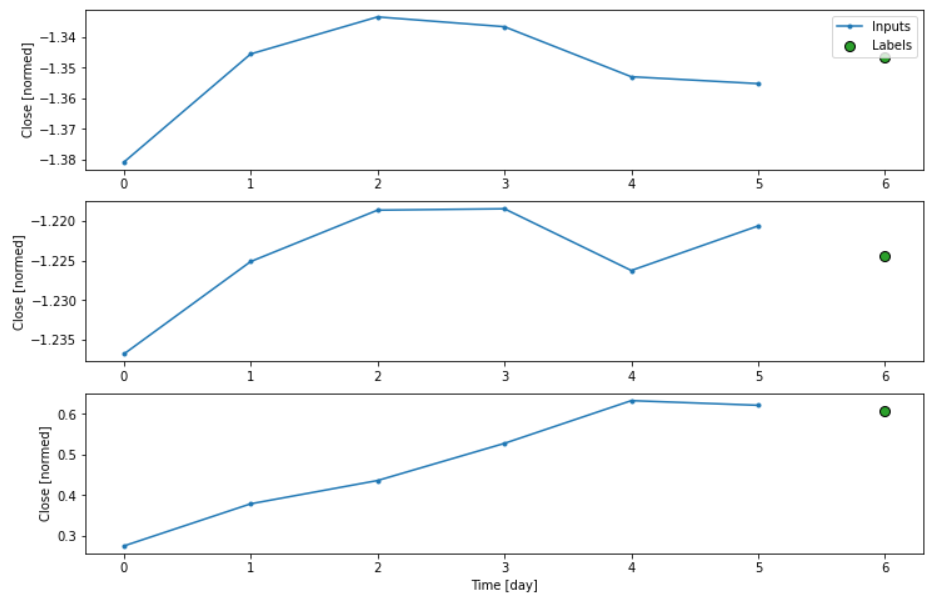
  
Рисунок 6 – Быстрое преобразование Фурье изменения цены закрытия торгов

Теперь посмотрим на распределение функций (рисунок 7).

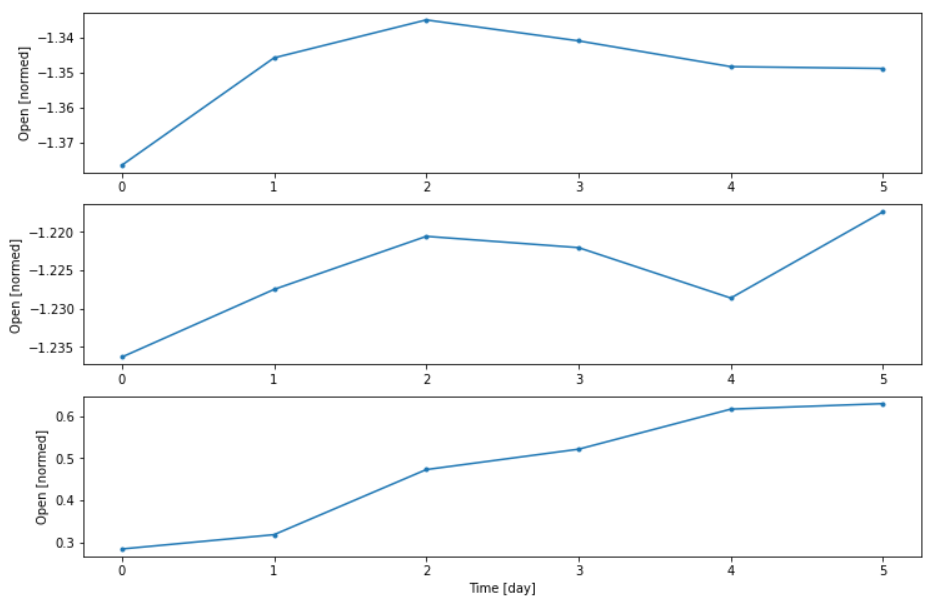
  
Рисунок 7 – Распределение функций

У некоторых функций действительно есть длинные хвосты, но нет явных ошибок, таких как значение отклонения цены от предыдущей больше чем в несколько раз.

Построим график, который позволяет легко визуализировать разделенное окно. График выравнивает входные данные, метки и (более поздние) прогнозы на основе времени, к которому относится элемент. (рисунок 8).

  
Рисунок 8 – График

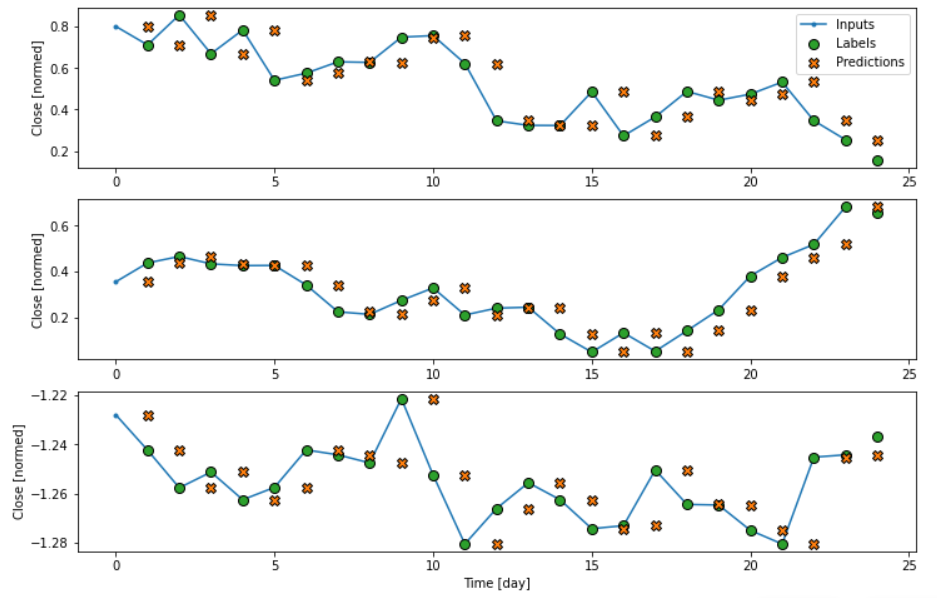
Построим другие столбцы, но пример конфигурации окна w2 имеет метки только для столбца Open (цена открытия торгов USD).

  
Рисунок 9 – График (метки только для столбца Open)

* 1. Одноступенчатые модели

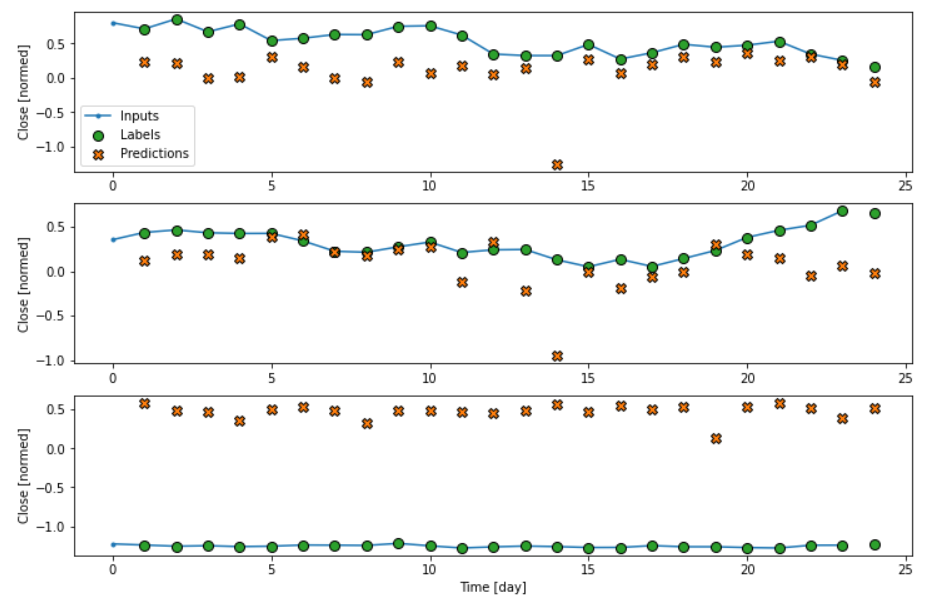
Начнём с построения моделей для прогнозирования значения Close (цены закрытия USD) на один день вперед. Настроим объект WindowGenerator для создания этих одношаговых пар (input, label). Объект window создает tf.data.Dataset из обучающих, проверочных и тестовых наборов, что позволяет легко перебирать пакеты данных.

Построим прогнозы базовой модели (рисунок 10).

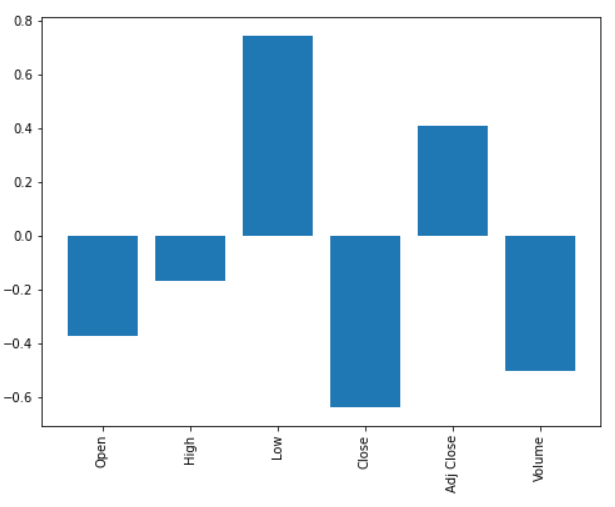
  
Рисунок 10 – Прогнозы базовой модели

Построив прогнозы базовой модели, можно заметить, что это просто метки, сдвинутые вправо на один день.

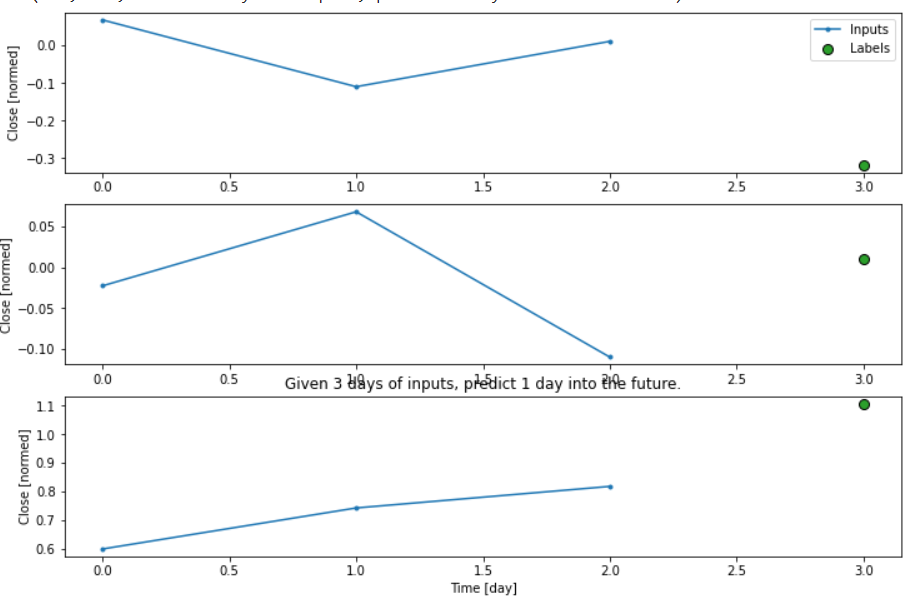
График примерных прогнозов для wide\_window изображен на рисунке 11. Можем заметить, что многих случаях прогноз явно лучше, чем просто возврат входной цены, но в некоторых случаях он хуже.

  
Рисунок 11 – График примерных прогнозов для wide\_window

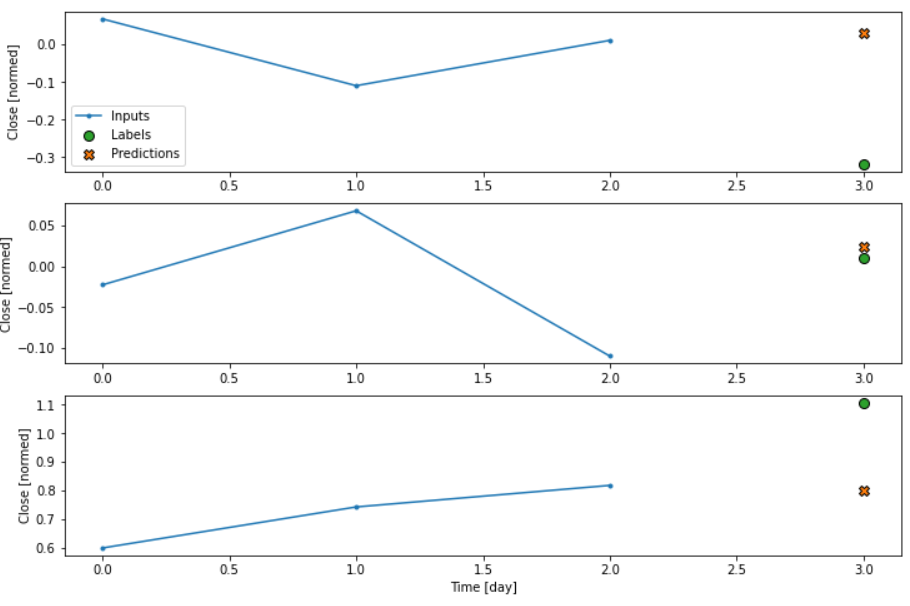
Одним из преимуществ линейных моделей является то, что их относительно просто интерпретировать. Можно вытащить веса слоя и визуализировать вес, назначенный каждому входу (рисунок 12).

  
Рисунок 11 – График визуализации весов, назначенных каждому входу

Теперь создадим WindowGenerator, который будет создавать пакеты трехдневных входных данных и однодневных меток (рисунок 12).

  
Рисунок 12 – График разреза в четыре дня

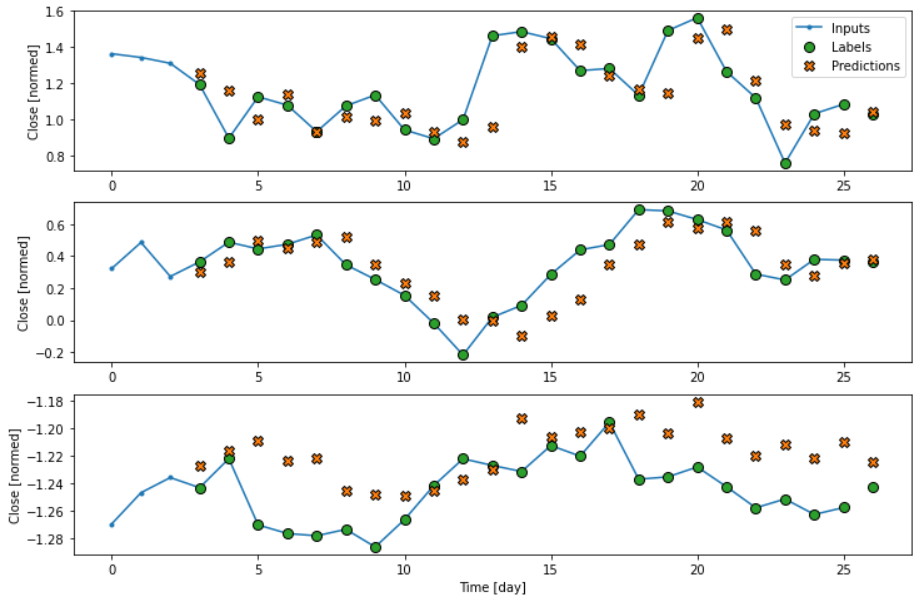
Также можно обучить dense модель в окне с несколькими входными шагами, добавив tf.keras.layers.Flatten в качестве первого слоя модели (рисунок 13).

  
Рисунок 13 – График разреза в четыре дня с предсказанным значением

Основным недостатком этого подхода является то, что результирующая модель может выполняться только на входных окнах именно такой формы.

Для того, чтобы решить эту проблему, воспользуемся свёрточными нейронными сетями.

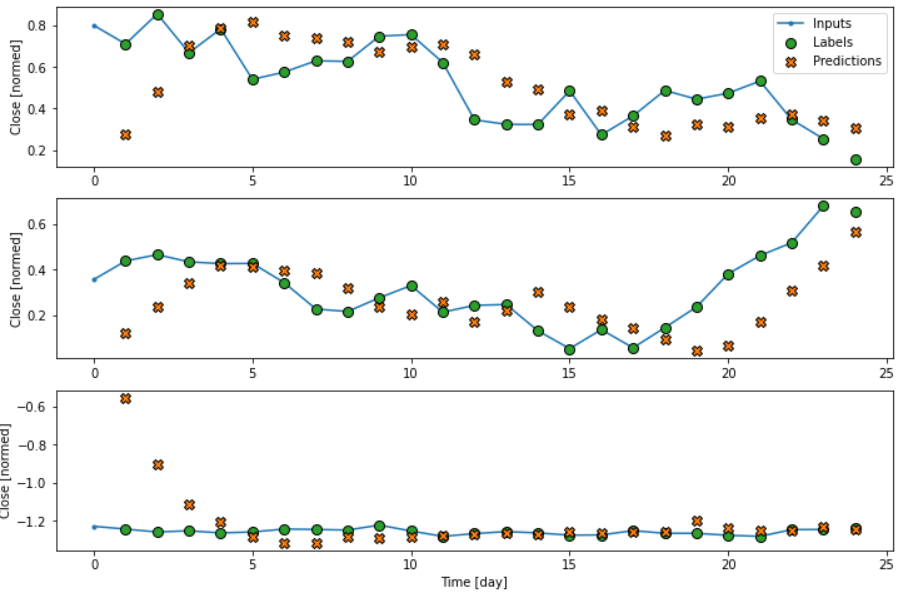
Построим прогнозы модели в более широком окне (рисунок 14). Каждый прогноз здесь основан на 3 предыдущих временных шагах.

  
Рисунок 14 – Прогнозы модели, основанные на 3 предыдущих временных шагах

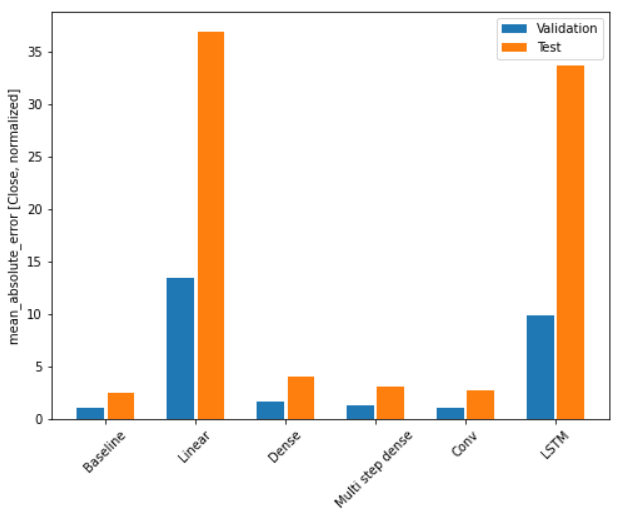
Далее воспользуемся рекуррентной нейронной сетью. В этом эксперименте мы будем использовать слой RNN под названием Long Short-Term Memory (tf.keras.layers.LSTM).

Если False по умолчанию, слой возвращает только выходные данные последнего временного шага, давая модели время, чтобы прогреть свое внутреннее состояние, прежде чем делать один прогноз.

Если True, слой возвращает результат для каждого входа. Это полезно для укладка слоев RNN и обучение модели на нескольких временных шагах одновременно.

  
Рисунок 15 – Прогнозы модели с помощью РНС

На рисунке 16 изображено представление.

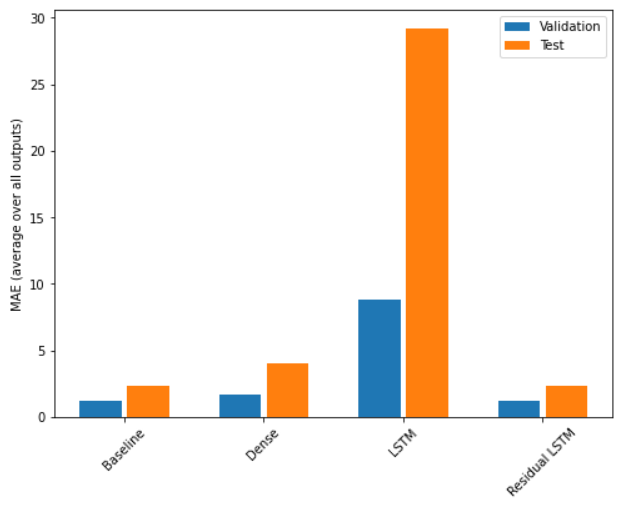
  
Рисунок 16 – Представление

Можем сделать вывод, что с таким набором данных лучше всего себя показали полносвязные и свёрточные сети. Вероятнее всего это произошло из-за нехватки данных.

До сих пор все модели предсказывали одну выходную характеристику Close (цена закрытия USD) для одного временного шага.

Все эти модели можно преобразовать для прогнозирования нескольких функций, просто изменив количество единиц в выходном слое и настроив окна обучения, чтобы включить все функции в labels (example\_labels).

Таким образом мы получаем представление, изображённое на рисунке 17.

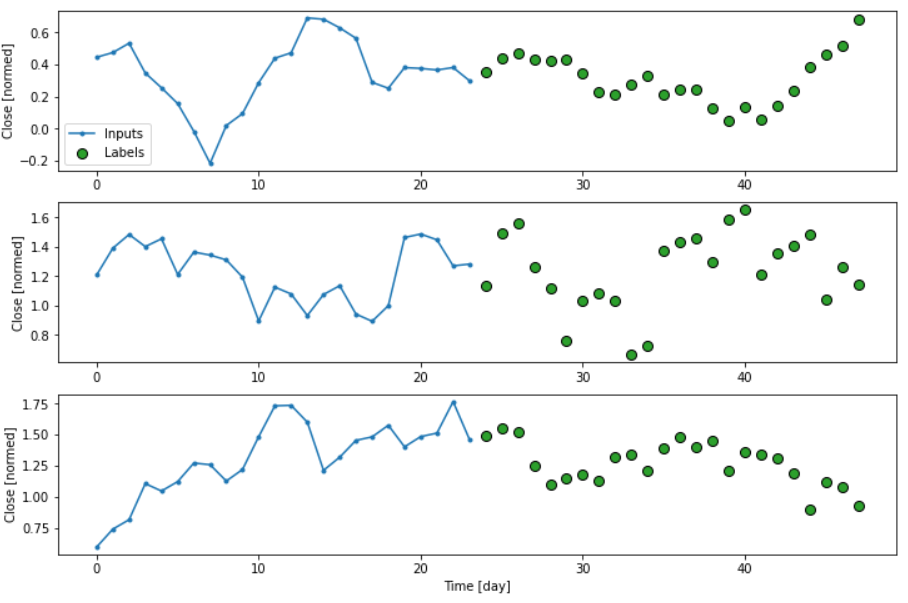
  
Рисунок 17 – Представление

* 1. Многошаговые модели

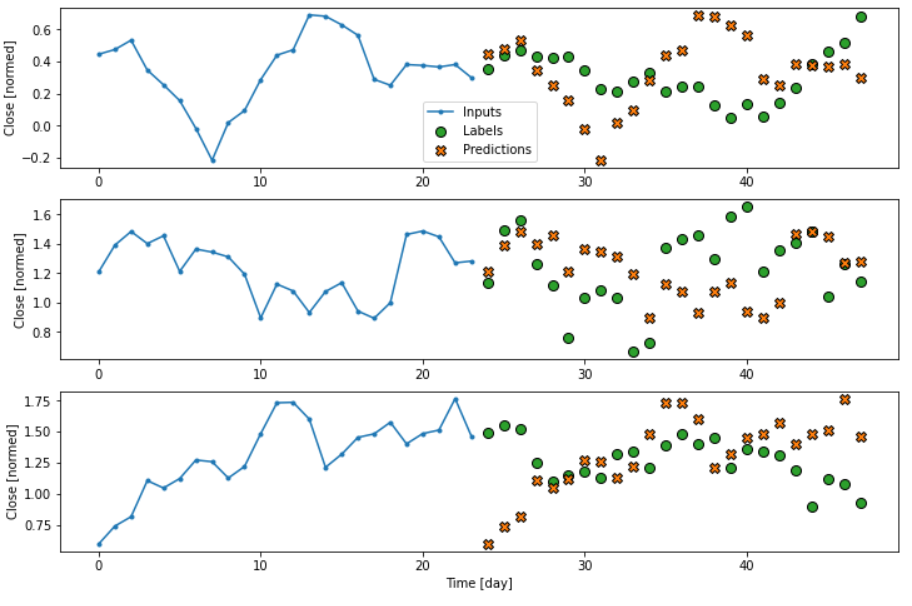
Как модели с одним выходом, так и модели с несколькими выходами делали прогнозы с одним временным шагом, на один день ъ вперед.

Сейчас мы рассмотрим, как расширить эти модели, чтобы делать прогнозы с несколькими временными шагами. То есть все модели будут предсказывать все функции на всех выходных временных шагах.

Для многоступенчатой ​​модели обучающие данные снова состоят из ежедневных выборок. Однако здесь модели научатся предсказывать будущее на 24 дня, учитывая 24 дня из пршлого. Объект Window, который генерирует эти срезы из набора данных представлен на рисунке 18.

  
Рисунок 18 – График значений отклонений цены за промежуток 48 дней

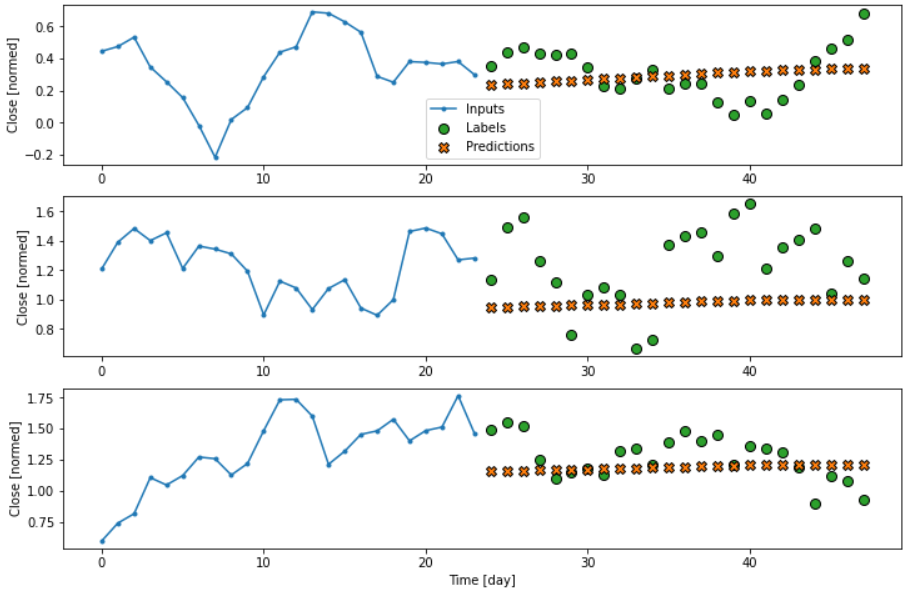
Простая базовая линия для этой задачи — повторить последний временной шаг ввода для необходимого количества выходных временных шагов. График представлен на рисунке 19.

  
Рисунок 19 – График модели базовой линии

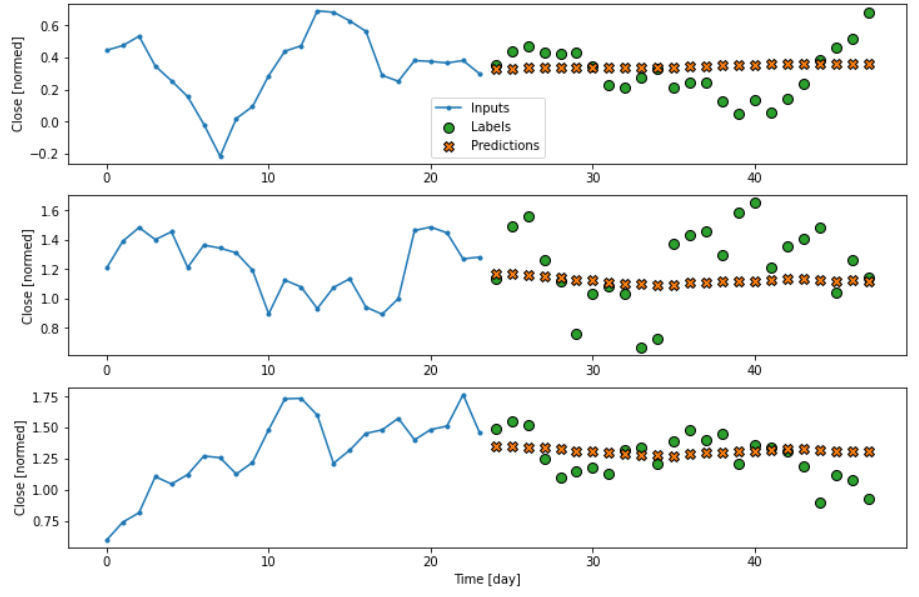
Поскольку эта задача состоит в том, чтобы предсказать 24 дня в будущем, учитывая 24 дня в прошлом, другой простой подход состоит в том, чтобы повторить предыдущий промежуток, предполагая, что в последующем будет похоже (рисунок 19).

Одним из высокоуровневых подходов к этой проблеме является использование «однократной» модели, в которой модель делает прогноз всей последовательности за один шаг.

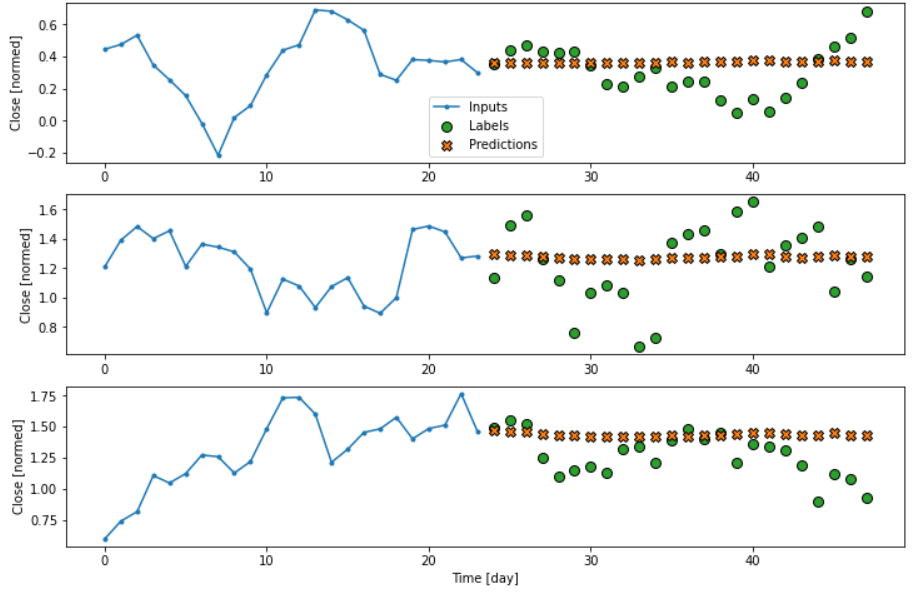
Простая линейная модель, основанная на последнем входном временном шаге, работает лучше любой базовой линии, но у нее недостаточно мощности. Модель должна прогнозировать временные шаги OUTPUT\_STEPS на основе одного входного временного шага с линейной проекцией. Он может зафиксировать только низкоразмерный фрагмент поведения, вероятно, основанный в основном на восходящем тренде (рисунок 20).

  
Рисунок 20 – График линейной модели

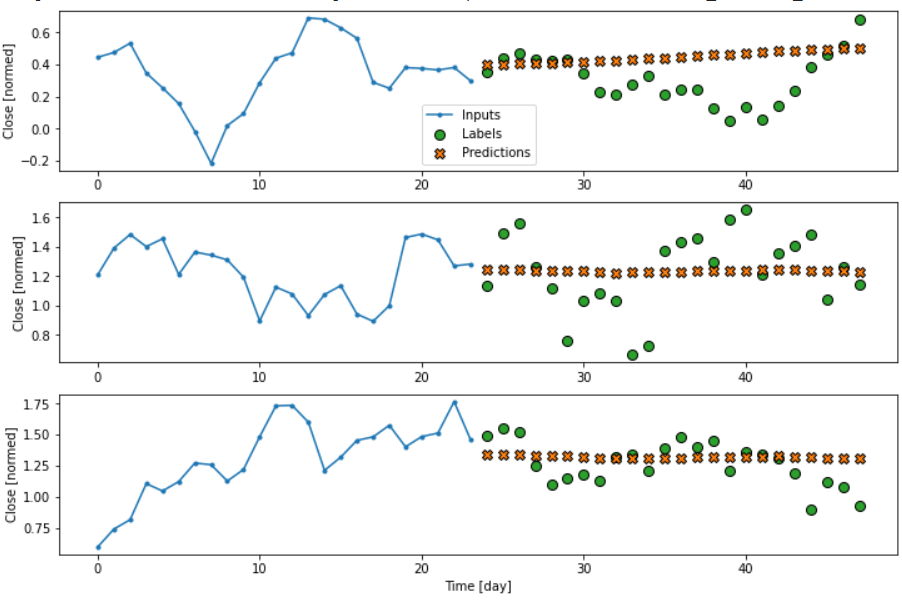
Добавление tf.keras.layers.Dense между входом и выходом делает линейную модель более мощной, но по-прежнему основано только на одном временном шаге ввода (рисунок 21).

  
Рисунок 21 – График модифицированной модели

Сверточная модель делает прогнозы на основе истории фиксированной ширины, что может привести к более высокой производительности, чем плотная модель, поскольку она может видеть, как все меняется с течением времени (рисунок 22).

  
Рисунок 22 – График сверочной модели

Рекуррентная модель может научиться использовать длинную историю входных данных, если она имеет отношение к прогнозам, которые делает модель. Здесь модель будет накапливать внутреннее состояние за 24 дня, прежде чем сделать один прогноз на следующие 24 дня (рисунок 23).

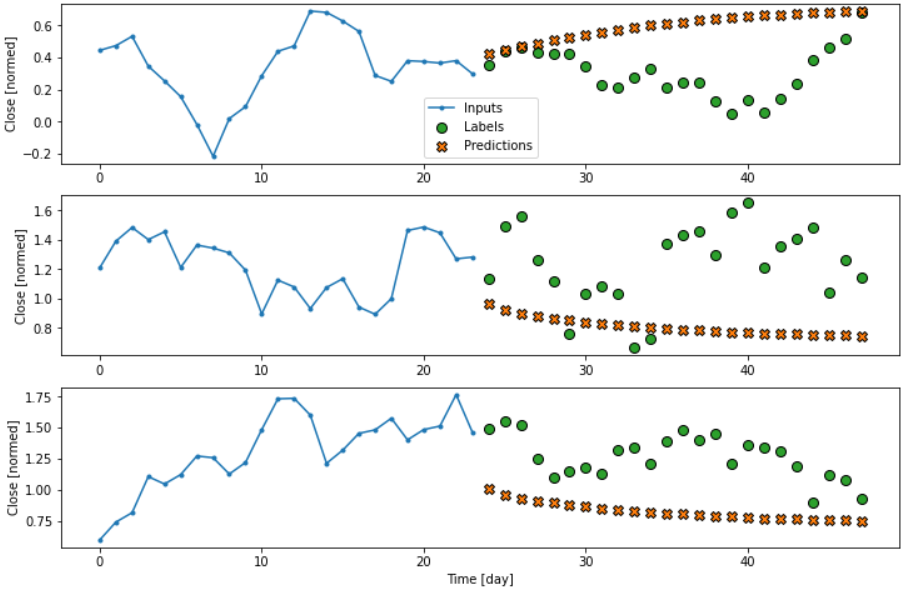
  
Рисунок 23 – График рекуррентной модели

Все вышеперечисленные модели предсказывают всю выходную последовательность за один шаг.

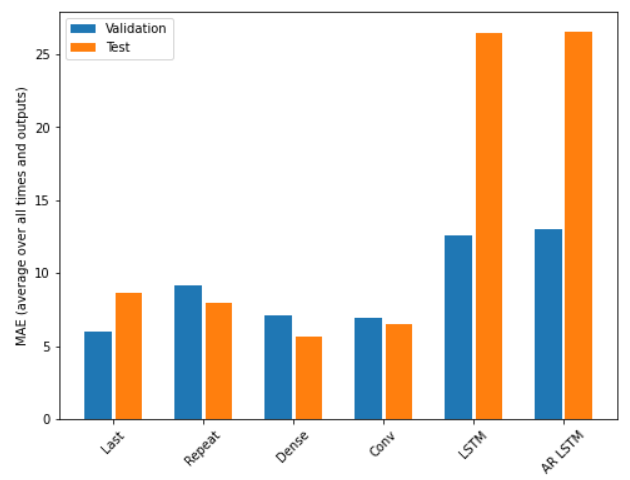
В некоторых случаях для модели может быть полезно разложить этот прогноз на отдельные временные шаги. Затем выходные данные каждой модели могут быть возвращены в себя на каждом шаге, и прогнозы могут быть сделаны в зависимости от предыдущего.

Одним из явных преимуществ модели этого типа является то, что ее можно настроить для получения выходных данных различной длины.

Построим авторегрессионную модель RNN (рисунок 24).

  
Рисунок 24 – График авто регрессионной рекуррентной модели

Таким образом мы получаем представление, изображённое на рисунке 25.

  
Рисунок 25 – Представление

Можно заметить, что хуже всего справились рекуррентные модели, что может быть связано с недостатком данных.

Лучше всего показали себя полносвязная модель и свёрточная модель.

Код программы можно посмотреть по ссылке https://colab.research .google.com/drive/1ZrrMxynLacyzABd\_XTLlkcY4QxlVMGi6?usp=sharing#scrollTo=H7MpUfQkO20u

ЗАКЛЮЧЕНИЕ

В ходе работы было изучено применение нейронных сетей для задач прогнозирования финансовых временных рядов, а также определена наиболее эффективная модель прогнозирования финансового временного ряда.

Была написана программа для прогнозирования значений финансовых показателей на языке программирования Python с использованием библиотеки tensorflow. В ходе написания программы решались проблемы подготовки данных, которые были связаны с определением окон, нормализацией данных и представлением данных для задачи машинного обучения.

Метрики для моделей с несколькими выходами показывают среднюю производительность по всем выходным функциям. Эти характеристики аналогичны, но также усреднены по временным шагам вывода.

Выигрыш, достигнутый при переходе от рекуррентной модели к сверточной и полносвязной моделям, составляет всего несколько процентов а авторегрессионная модель показала себя явно хуже. Таким образом, эти более сложные подходы могут быть бесполезными для решения этой проблемы в случае недостаточности обучающих данных, что невозможно было узнать, не попробовав, и эти модели могут быть полезны для решения проблемы с использованием других наборов данных.