Министерство образования и науки Российской Федерации

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Факультет прикладной математики и компьютерных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ

Задание 2. Обучение нейросетевых моделей анализа последовательностей

Выполнила студентка группы 932001:

Ерыгина Виктория

Томск – 2022

**Цель** работы – получить навыки создания нейронных сетей для анализа сигналов с помощью моделей рекуррентных сетей и ячеек LSTM (GRU).

**Задачи** – Построить три нейросетевые модели:

a) Регрессор, использующий слой RNN в качестве скрытого слоя

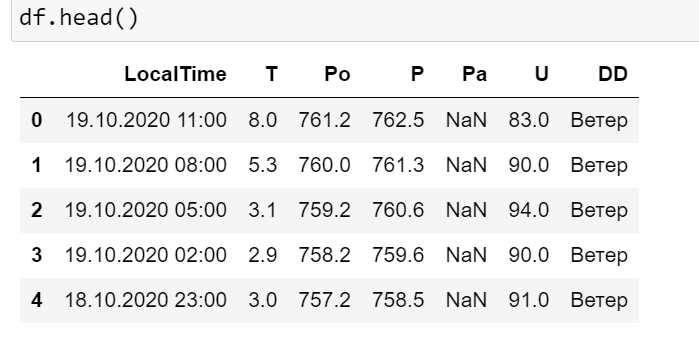
b) Регрессор, ячейку LSTM (или GRU) в скрытом слое

c) Регрессор – стек двух слоёв с LSTM (или GRU)

для прогноза средней температуры в городе на следующий день, используя информацию за предыдущие дни.

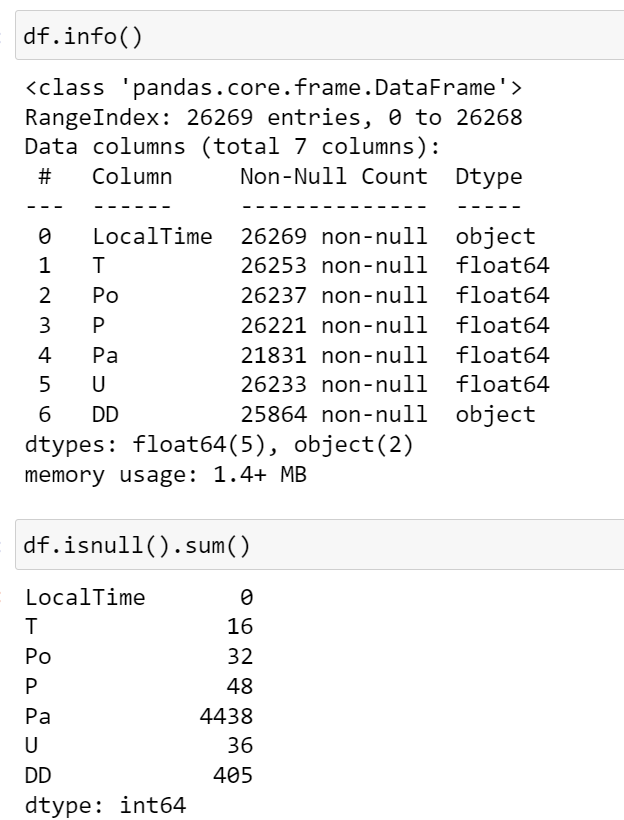
**Описание данных**

В ходе выполнения лабораторной работы использовалась выборка «Погода в Калининграде». Датасет содержит 26269 записей и 7 признаков. Выведем первые 5 записей с помощью метода head():



Листинг 1. Вывод первых 5 записей

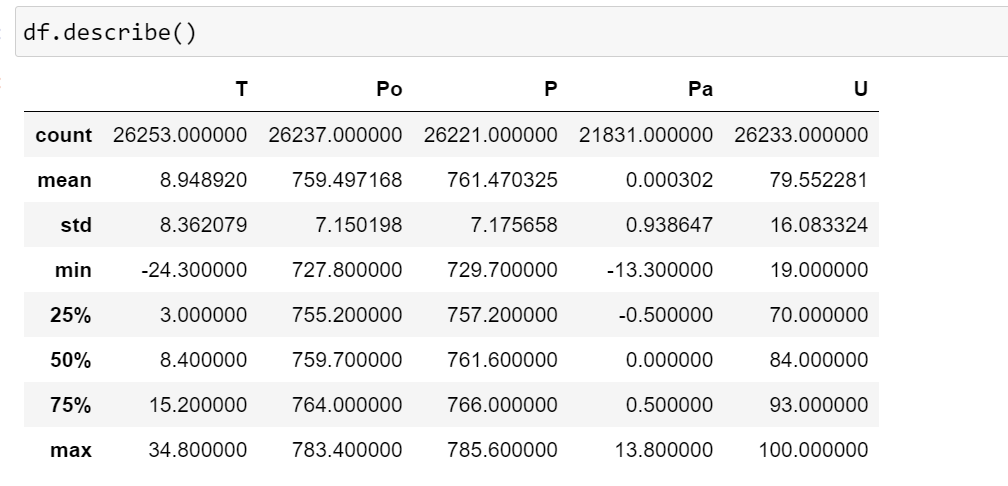
Получим типы данных наших признаков с помощью метода info(). Перед обучением модели нам нужно проанализировать и обработать данные, избавиться от пропущенных значений и типов object.



Листинг 2. Получение типов данных и суммы пропущенных значений

1. LocalTime – время наблюдения. Объектный тип данных. Нет пропущенных значений
2. T – температура в момент наблюдения. Числовой тип данных (float). Имеется 16 пропущенных значений
3. Po – давление на уровне моря. Числовой тип данных (float). Имеется 32 пропущенных значения.
4. P – давление в мм ртутного столба. Числовой тип данных (float). Имеется 48 пропущенных значений.
5. Pa – давление в паскалях. Числовой тип данных (float). Имеется 4438 пропущенных значений.
6. U – относительная влажность воздуха. Числовой тип данных (float). Имеется 36 пропущенных значений.
7. DD – объектный тип данных. Вариация значений: Ветер, Штиль. Имеется 405 пропущенных значений.

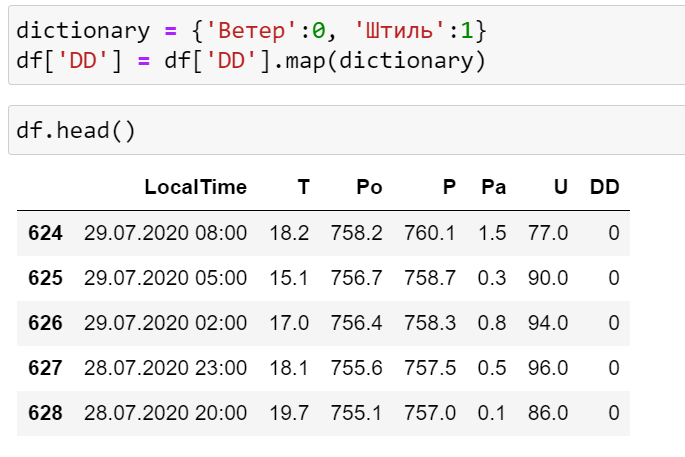
Получаем описание данных с помощью метода describe():



Листинг 3. Описание данных

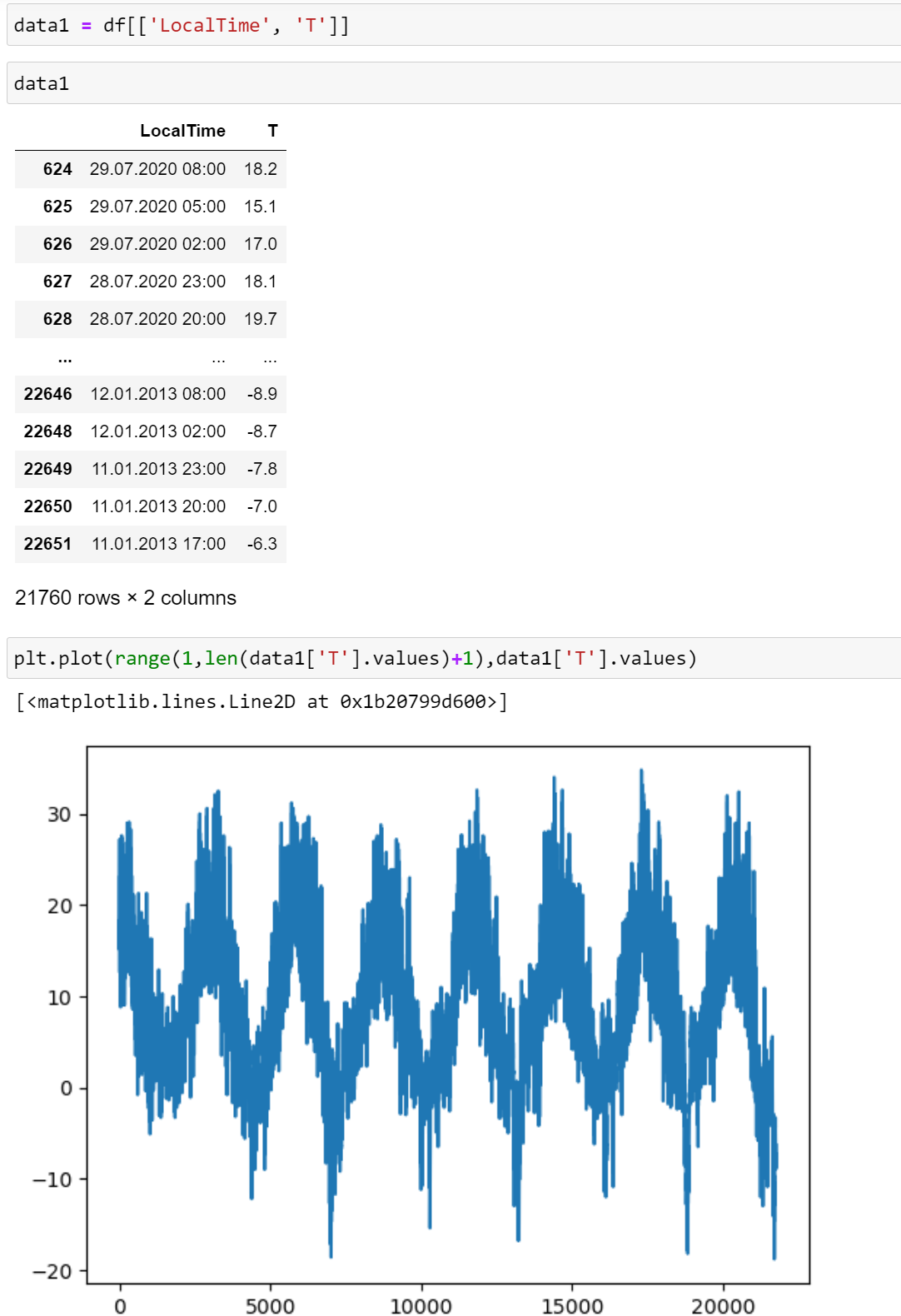
* count (количество данных для каждого столбца).
* mean (среднее значение) показывает среднее арифметическое каждого признака.
* std (стандартное отклонение) позволяет оценить разброс данных выборки.
* min (минимальное значение).
* 25% (первый квантиль).
* 50% (второй квантиль, медиана).
* 75% (третий квантиль).
* max (максимальное значение).

Для того чтобы подготовить данные к дальнейшей работе, для начала удалим все строки с пропущенными данными с помощью метода dropna(). Далее преобразуем категориальные значения у признака DD. Для этого создадим словарь, с помощью которого заменим значение «Ветер» на 0 и «Штиль» на 1:



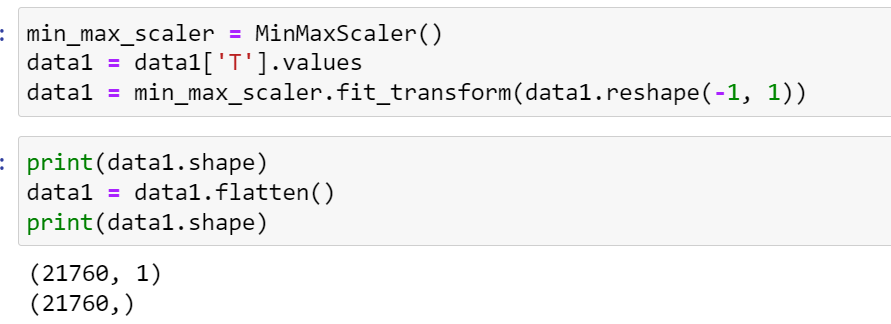
Листинг 4. Создание словаря

Рассмотрим изменение температуры в зависимости от времени. Для этого будем использовать признаки LocalTime и T. Нарисуем график, в котором будет отображено изменение температуры, при этом более высокие значение будут располагаться выше по оси Oy:



Листинг 5. Вывод графика зависимости температуры от времени

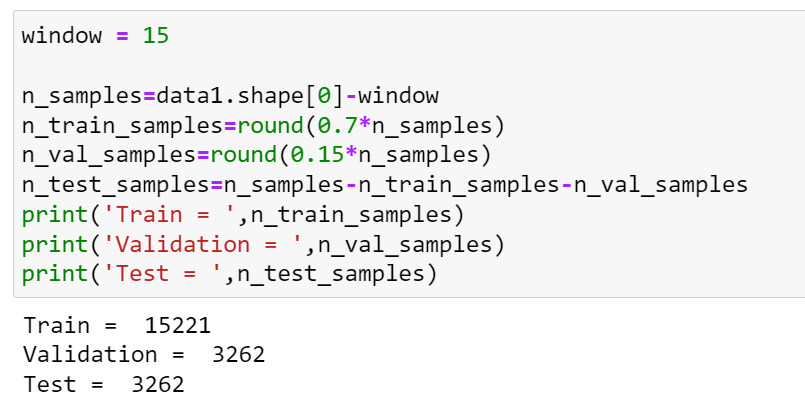
Для корректной работы нейросети нужно предварительно провести нормализацию данных. Для этого используем инструмент для масштабирования MinMaxScaler(). Все значения будут находиться в диапазоне от 0 до 1. Получаем правильный размер с помощью метода flatten():



Листинг 6. Нормализация данных

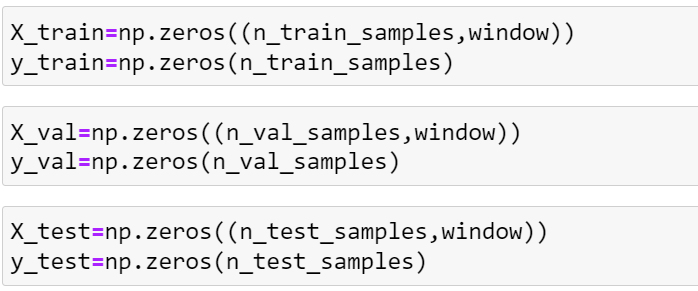
Далее определяем размер окна window = 15. Нам необходимо это сделать, чтобы составить прогноз температуры на следующий день по последним 15 записям.

Вычислим количество образцов, которые можно создать, чтобы распределить их для обучающей, тестовой и валидационной выборки. При этом 70% будут использовать для обучающей, 15% для валидационной и оставшиеся 15% для тестовой. Выведем получившееся количество образцов для каждого набора



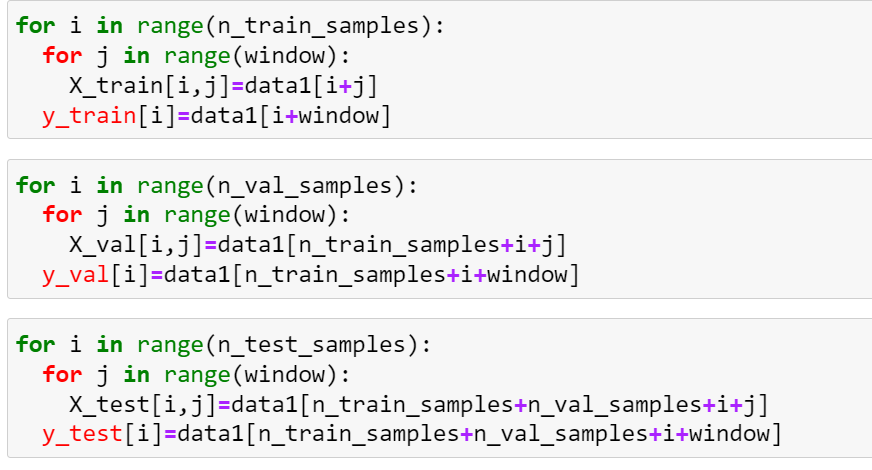
Листинг 7. Определение количества образцов для обучающего, тестового и валидационного наборов

Создаем нулевые массивы, в которые будем сохранять данные обучающего, тестового и валидационного наборов.



Листинг 8. Создания нулевых массивов

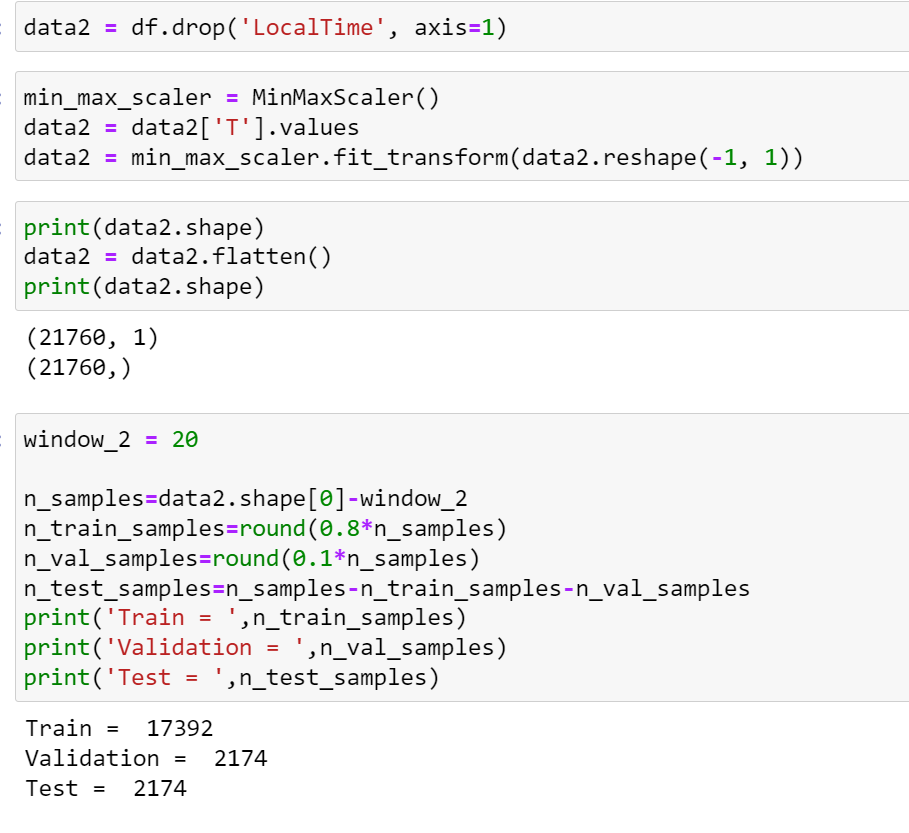
Далее создадим последовательные окна данных, где X будет содержать предыдущие значения, а y целевую переменную:



Листинг 9.

Для каждого регрессора будем создавать два примера, которые нужно будем сравнить по метрикам работы: R2, MAE, MSE. Для этого заранее определим данные для 2 примера. Удалим столбец LocalTime, нормализуем значения с помощью инструмента масштабирования MinMaxScaler().

Проделаем те же действия, но теперь размер окна window = 20 (прогноз на следующей день будет осуществляться по последним 20 записям). Для обучающего набора определим 80% образцов, для валидационного и тестового по 10%.



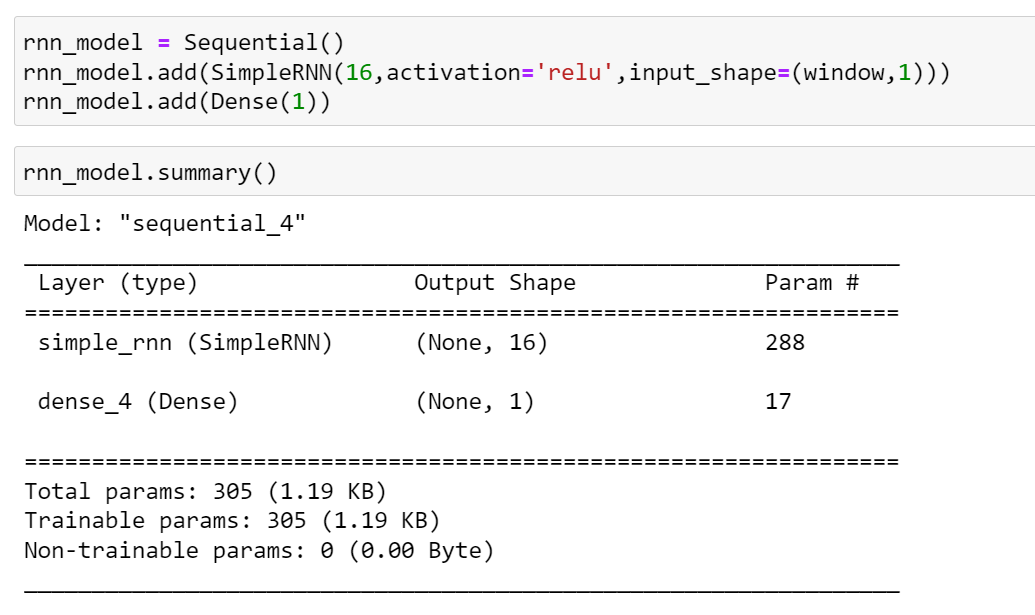
Листинг 10. Создание данных для примера 2

**Регрессор, использующий слой RNN в качестве скрытого слоя**

Пример 1

Для построения нейронной сети используется библиотека TensorFlow.

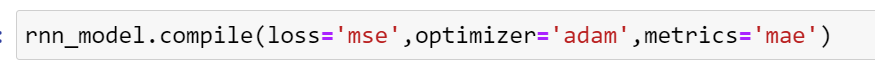
Создадим модель с одним скрытым слоем RNN, который содержит 10 нейронов, использует функцию активации Relu и формой входных данных (window, 1), где window – определенный нами заранее размер данный и 1 это количество признаков. Добавляем полносвязный слой Dense с 1 выходом.



Листинг 11. Построение модели

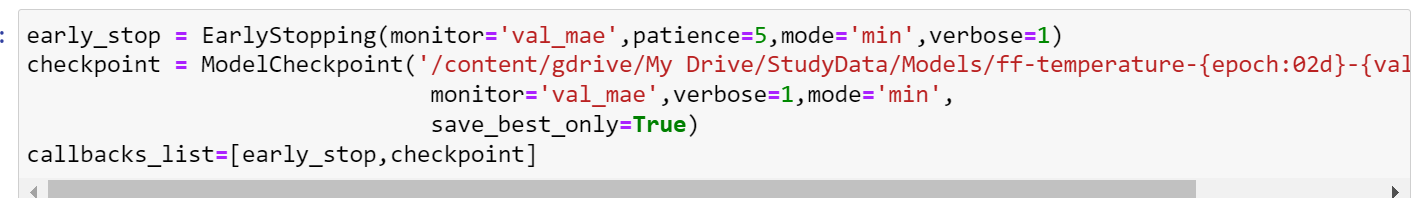
Модель сконструирована, настроим процесс ее обучения вызовом метода compile.

1. Optimizer – объект, который определяет процедуру обучения.
2. Loss – функция потерь (функция, которая минимизируется в процессе обучения).
3. Metrics используются для мониторинга обучения.



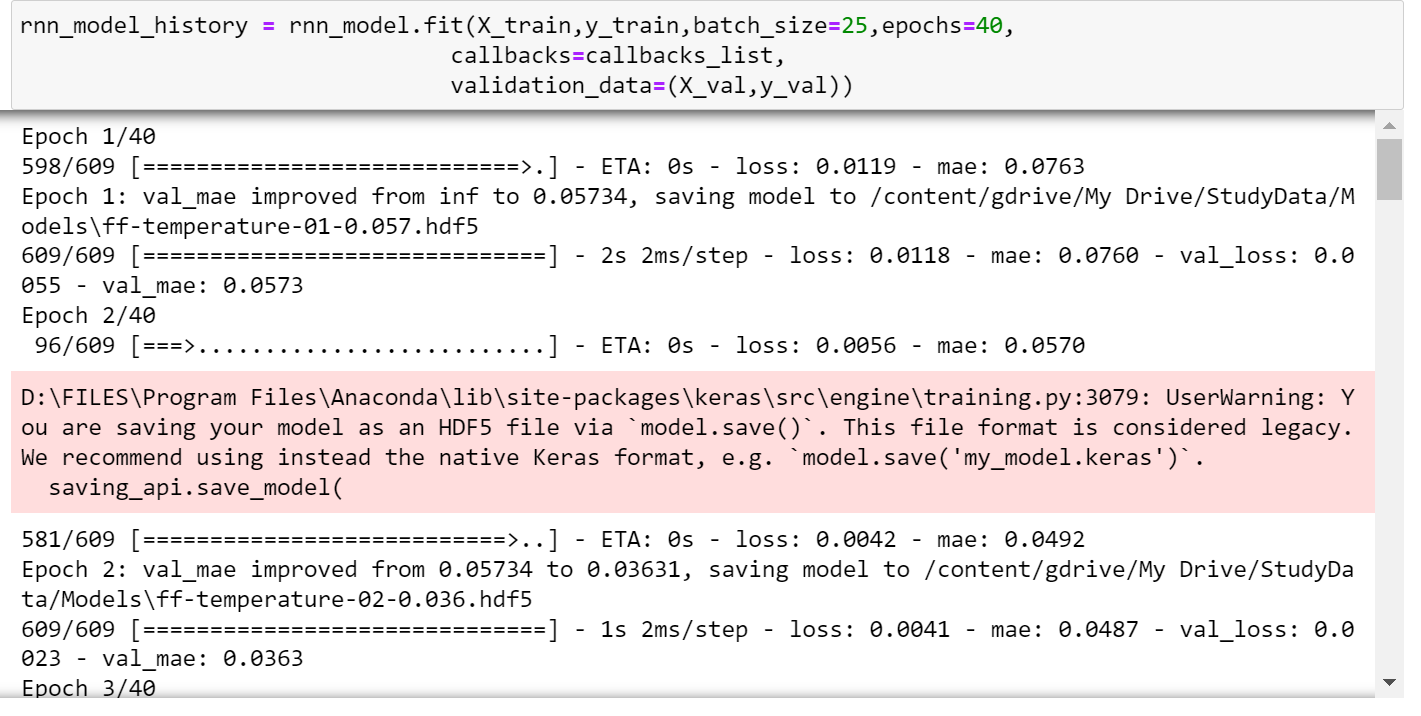
Листинг 12. Использование метода compile

Определим ранний выход, который может быть вызван в процессе обучения модели:



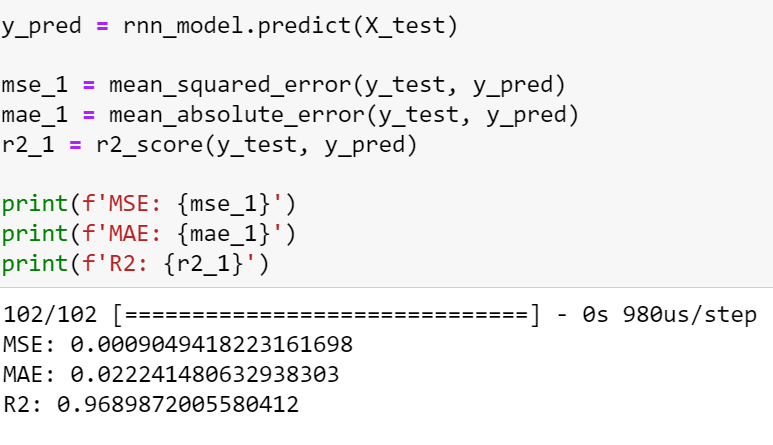
Листинг 13. Определение остановки

Далее обучаем модель на 40 эпохах и 25 количествам образцов, которые модель обрабатывает за одну итерацию:

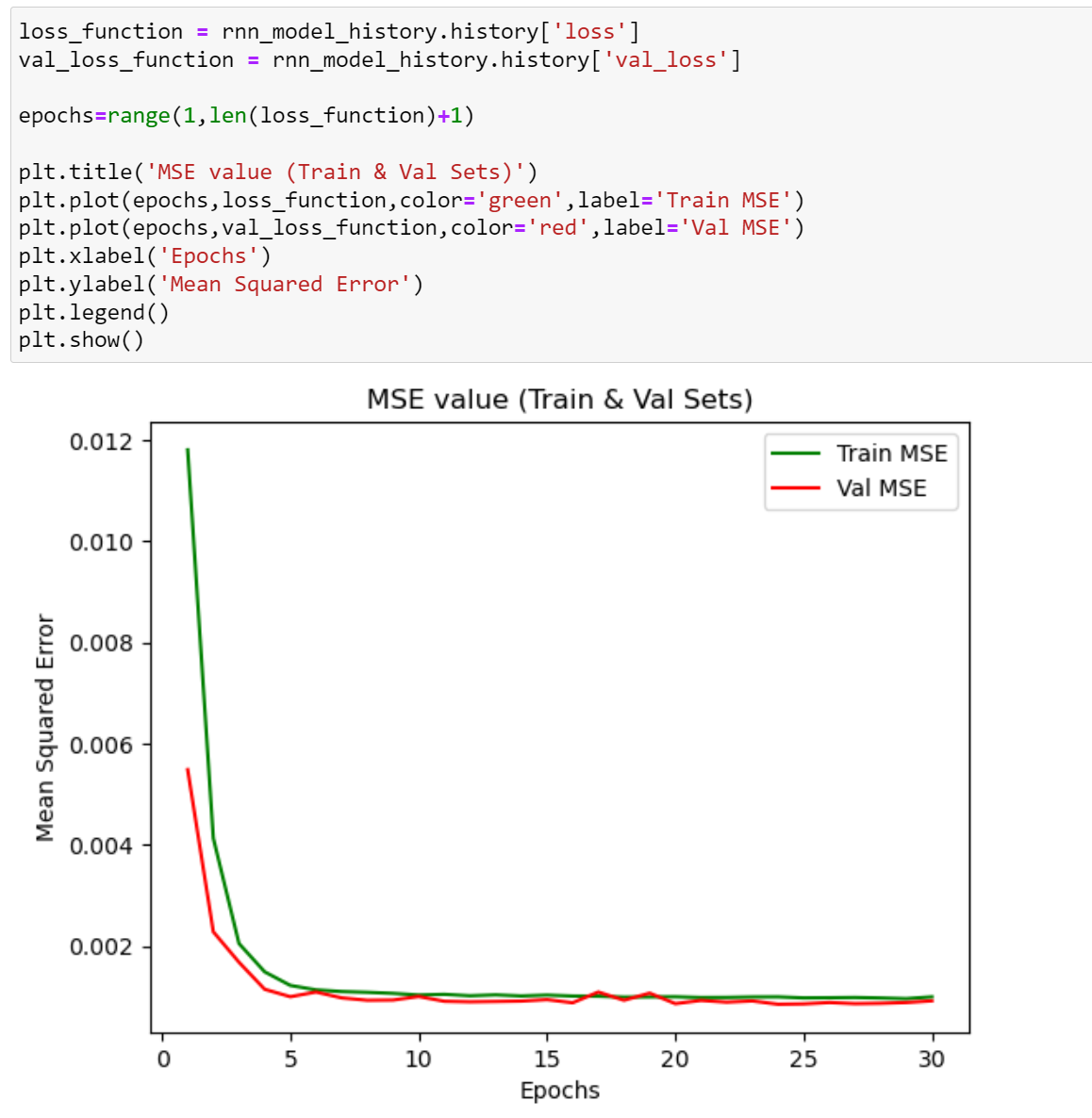


Листинг 14. Обучение модели

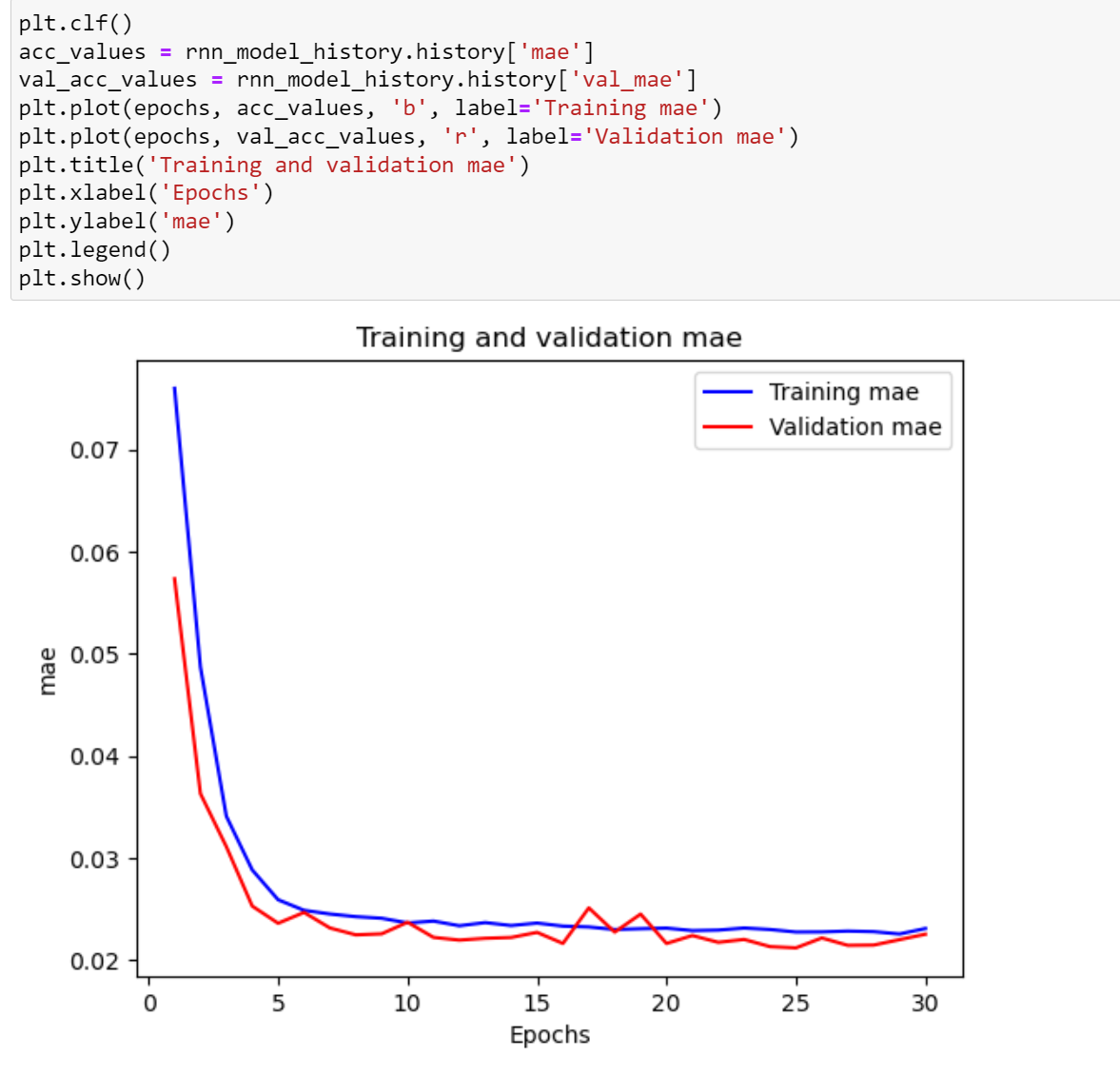
Найдем метрики работы: R2, MAE, MSE, которые нам пригодятся для сравнения моделей. И построим графики.



Листинг 15. Вычисление метрик работы

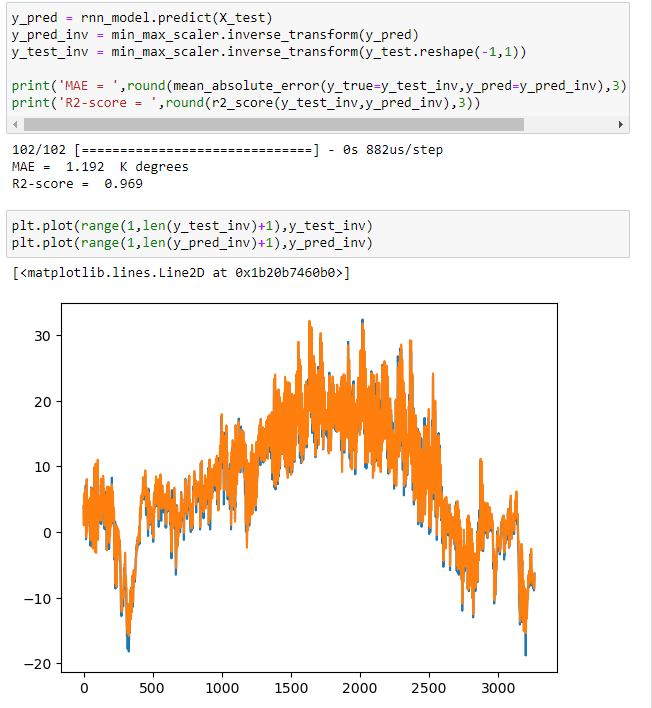


Листинг 16. Построение графика



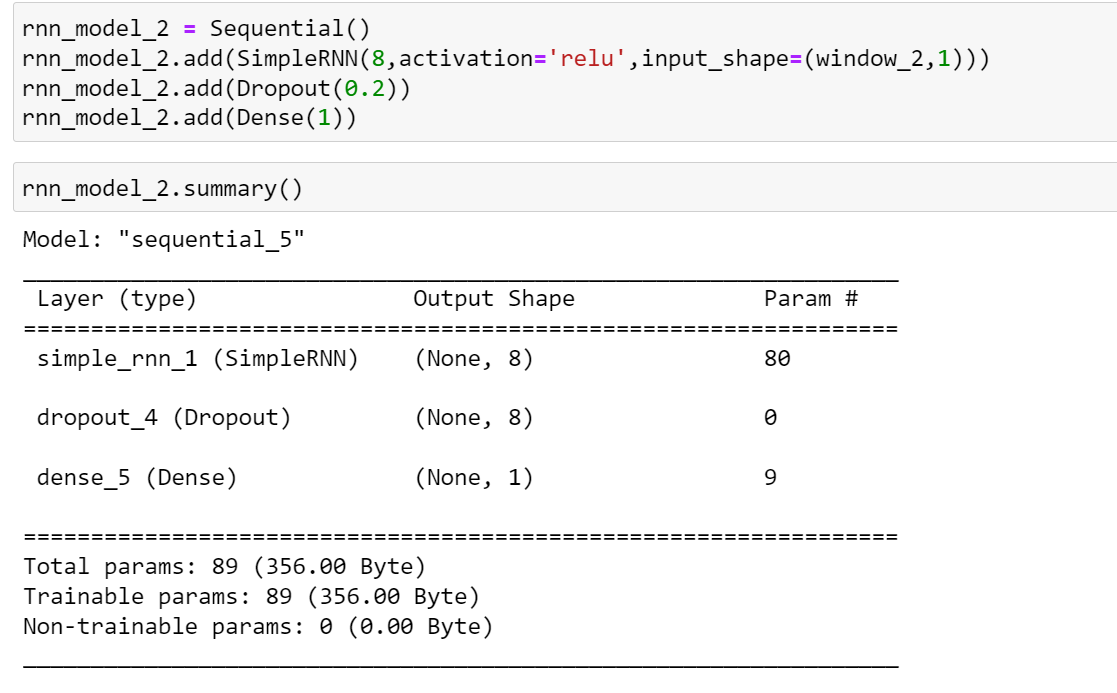
Листинг 17. Построение графика

Оценим результат прогнозирования модели с тестовыми данными и построим график, где выделенное синим – в тестовом наборе, оранжевым – предсказанное:



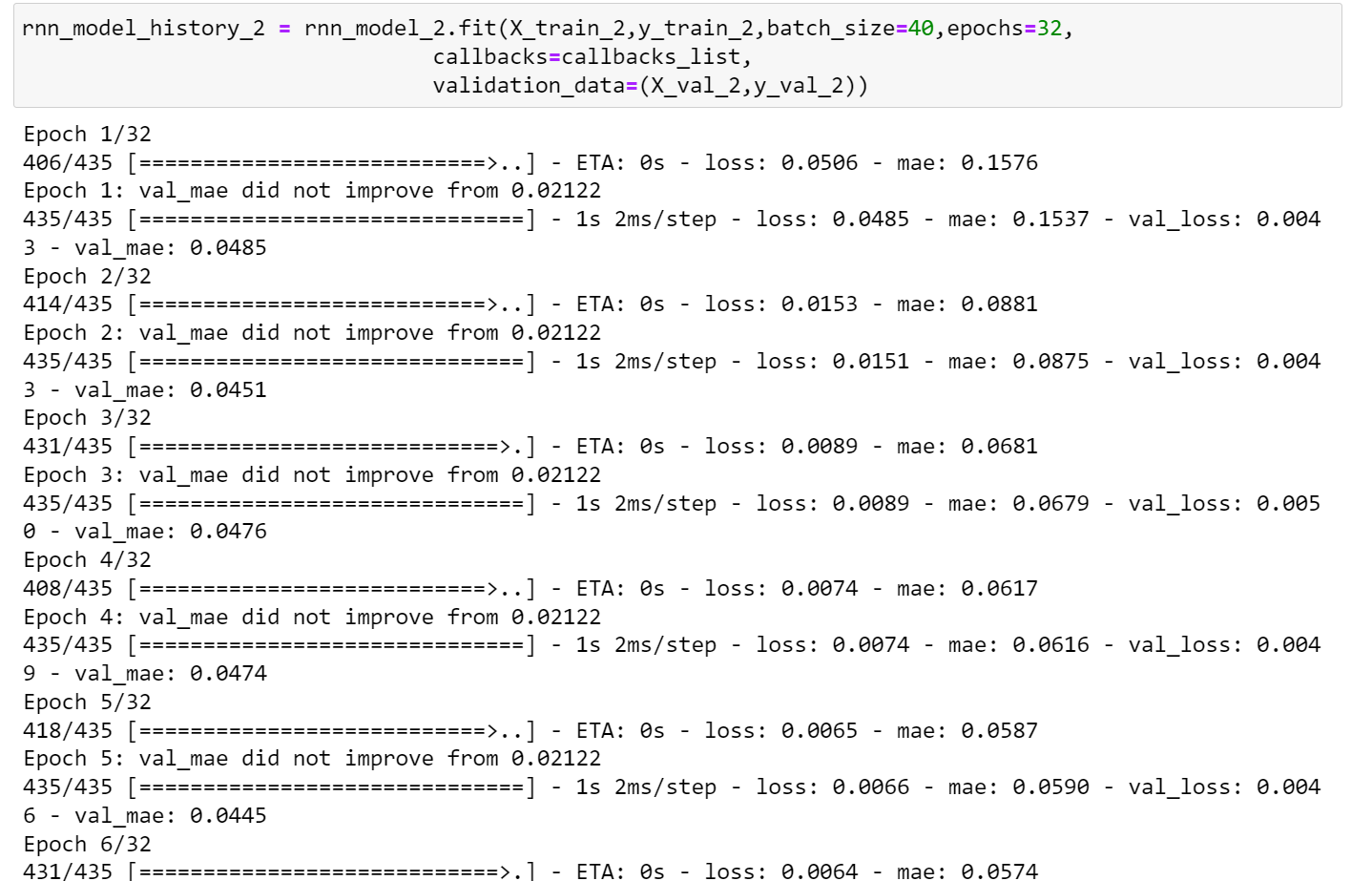
Пример 2

Проделаем такую же работу с данными для примера 2. Создадим модель с одним скрытым слоем RNN, который содержит 8 нейронов, использует функцию активации Relu и формой входных данных (window\_2, 4), где window\_2 – определенный нами заранее размер данных для второго примера и 4 это количество признаков. Добавляем полносвязный слой Dense с 1 выходом. Dropout – используется для прореживания.



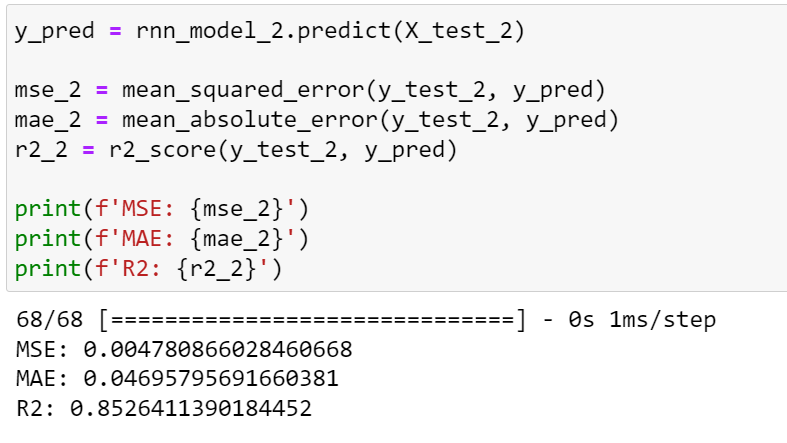
Листинг 18. Создание модели для примера 2

Далее обучаем модель на 32 эпохах и 40 количеством образцов, которые модель обрабатывает за одну итерацию:

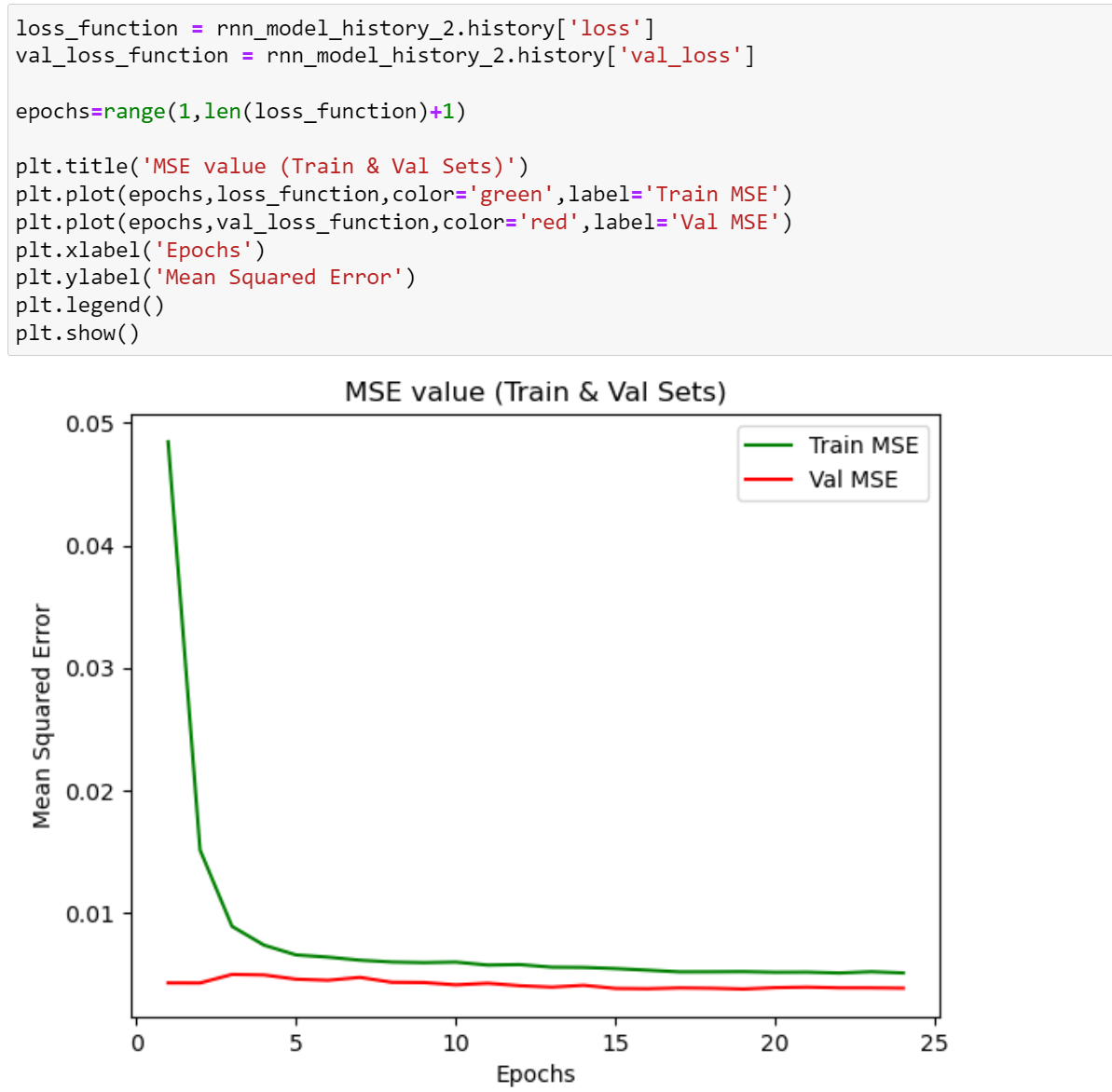


Листинг 19. Обучение модели 2

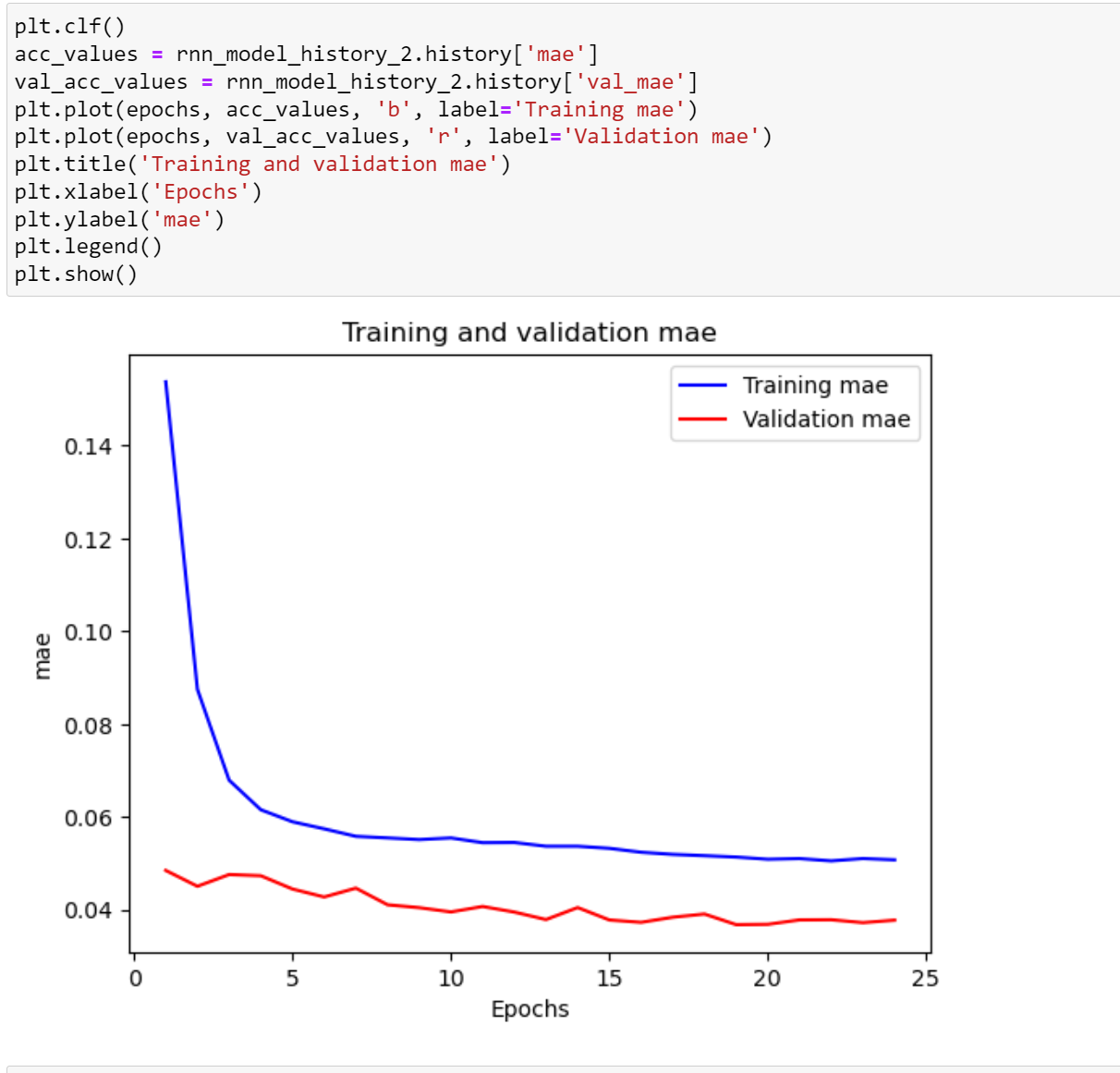
Выводим метрики работы:



Листинг 20. Вычисление метрик работы



Листинг 21. Построение графика



Листинг 22. Построение графика

Проведем сравнение двух моделей:



Листинг 23. Сравнение моделей

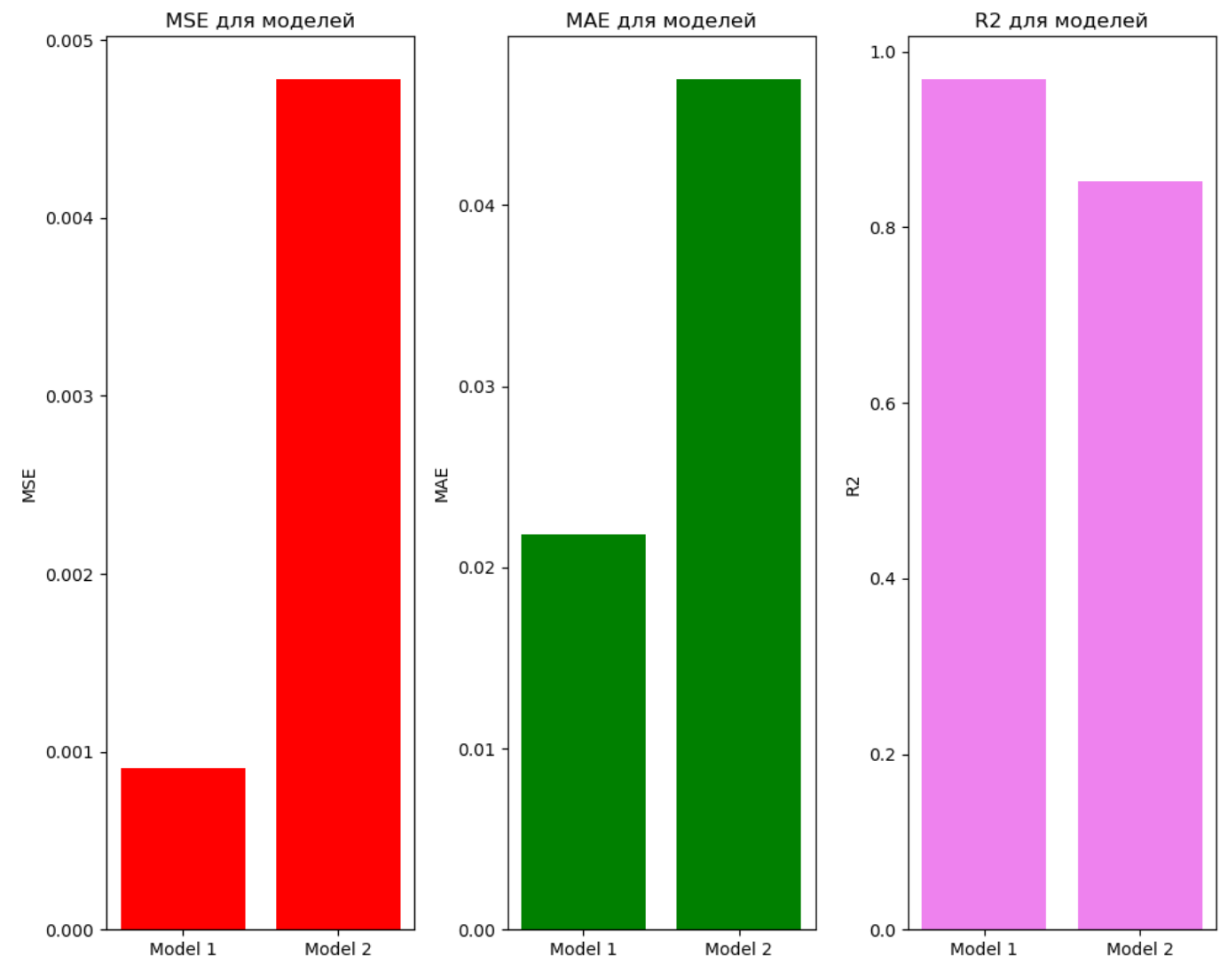


Рис 1. Сравнение двух моделей

**MSE –** среднеквадратическая ошибка, с помощью мы оцениваем качество модели относительно ошибок. Чем ниже MSE, тем лучше модель соответствует набору данных, поэтому точность прогнозирования **первой** модели лучше.

**MAE** – средняя абсолютная ошибка. Чем ниже MAE, тем лучше модель соответствует набору данных, поэтому точность прогнозирования **первой** модели лучше.

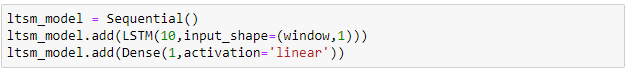
**R2** – значение **первой** модели превосходит, поэтому она более эффективна для прогнозирования.

Можно сделать вывод о том, что **первая** модель работает лучше.

**Регрессор, ячейку LTSM в скрытом слое**

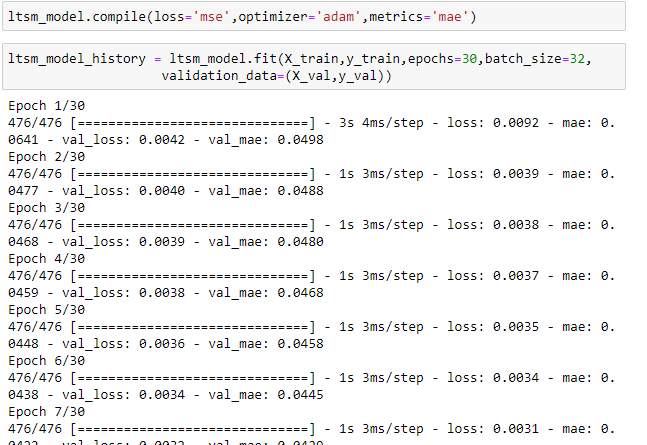
Пример 1

Создадим модель с одним скрытым слоем LTSM, который содержит 10 нейронов и использует форму входных данных (window, 1), где window – определенный нами заранее размер данных для первого примера и 1 это количество признаков. Добавляем полносвязный слой Dense с 1 выходом и линейной функцией активации.



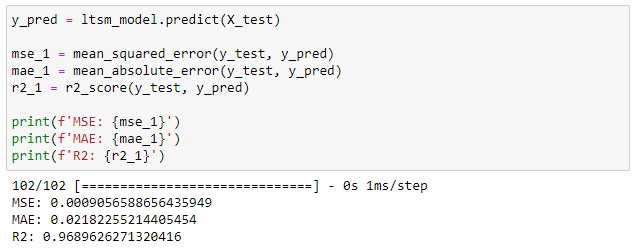
Листинг 24. Создание модели для примера 1

Модель сконструирована, настроим процесс ее обучения вызовом метода compile и обучим модель на 30 эпохах и 32 образцами обучения, которая модель обрабатывает за одну итерацию.

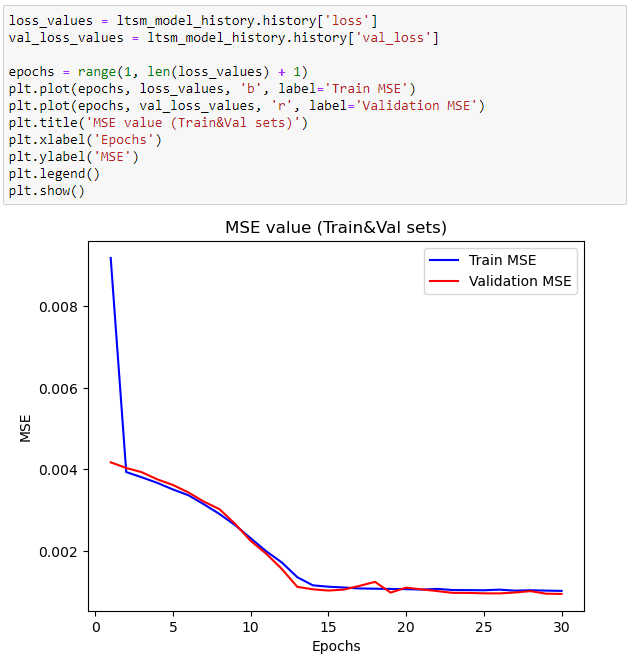


Листинг 25. Обучение модели

Найдем метрики работы: R2, MAE, MSE, которые нам пригодятся для сравнения моделей. И построим графики.



Листинг 26. Вычисление метрик работы

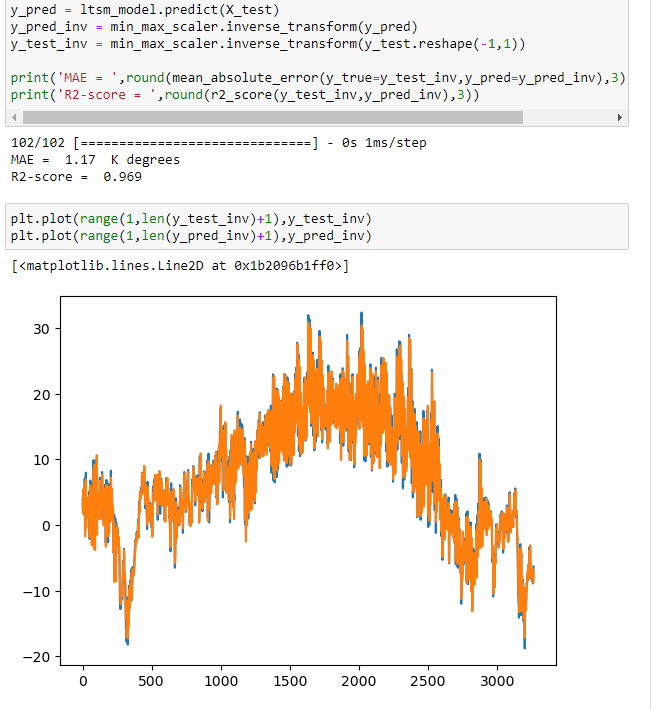


Листинг 27. Построение графика



Листинг 28. Построение графика

Оценим результат прогнозирования модели с тестовыми данными и построим график, где выделенное синим – в тестовом наборе, оранжевым – предсказанное:

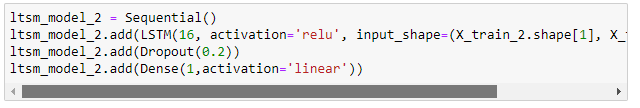


Листинг 29. Прогнозирование модели

Видим, что уровень прогнозирования модели 96,9%.

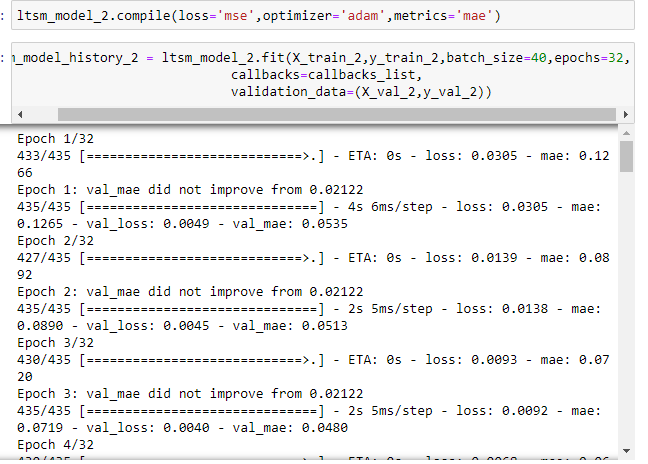
Пример 2

Проделаем такую же работу с данными для примера 2. Создадим модель с одним скрытым слоем LTSM, который содержит 16 нейронов, использует функцию активации Relu и формой входных данных (X\_train\_2.shape[1], X\_train\_2.shape[2]), которые определены в обучающей выборке для второго примера. Добавляем полносвязный слой Dense с 1 выходом и линейной функцией активации. Dropout – используется для прореживания.



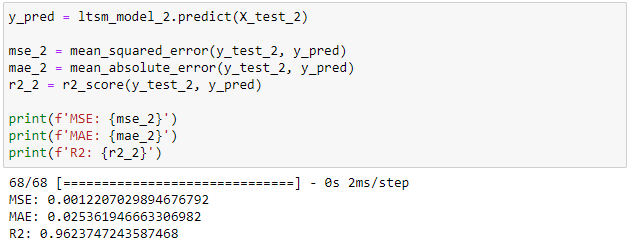
Листинг 30. Построение модели

Модель сконструирована, настроим процесс ее обучения вызовом метода compile и обучим модель на 32 эпохах и 40 образцах обучения, которые модель обрабатывает за одну итерацию.

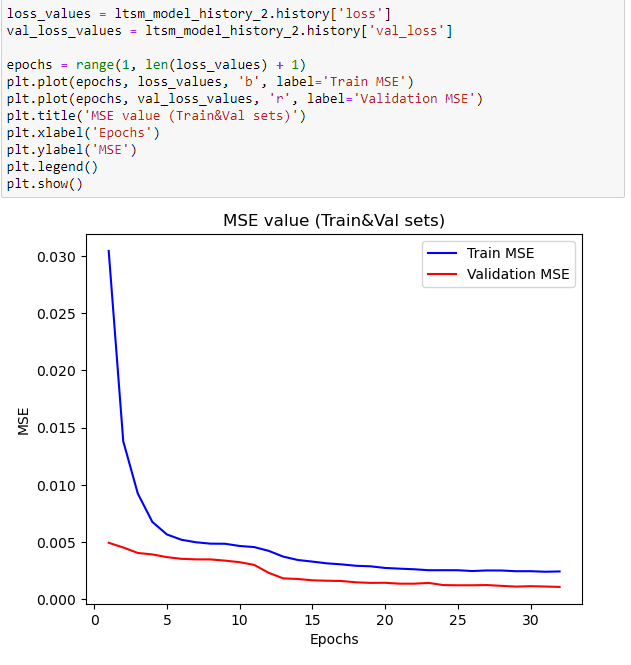


Листинг 31. Компиляция и обучение модели

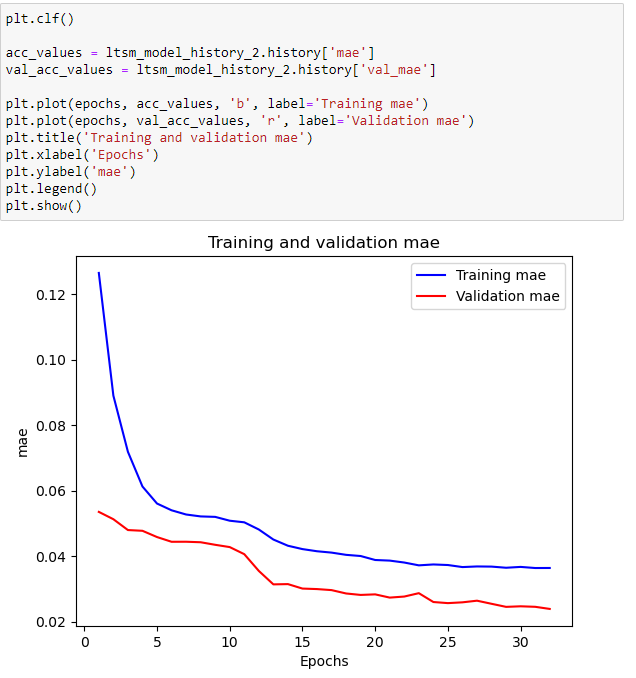
Найдем метрики работы: R2, MAE, MSE, которые нам пригодятся для сравнения моделей. И построим графики.



Листинг 32. Вычисление метрик работы

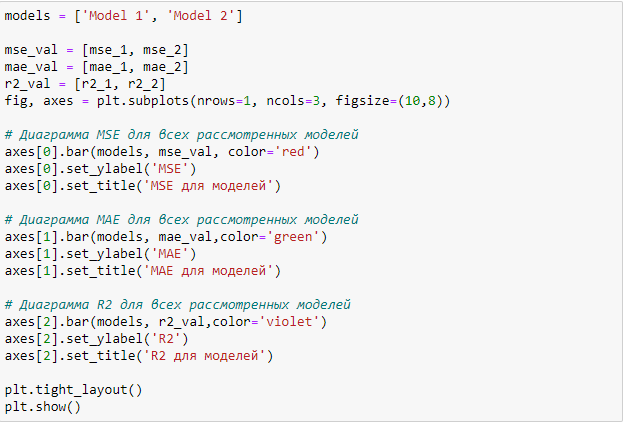
****

Листинг 33. Построение графика



Листинг 34. Построение графика

Сравним получившиеся модели.



Листинг 35. Сравнение моделей

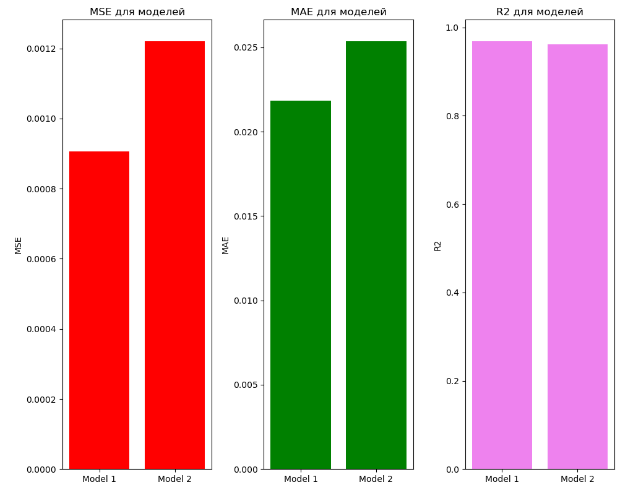


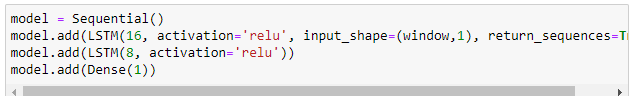
Рис 2. Сравнение моделей

На диаграмме видно, что MSE и MAE **первой** модели ниже, поэтому ее точность прогнозирования больше. Значения R2 обеих моделей высокие. Значение R2 первой модели ~ 0.9687, а у второй модели ~ 0.9624, так как значение у первой модели по этой метрике больше, то она лучше объясняет вариацию в данных.

**Регрессор – стек двух слоев с LTSM**

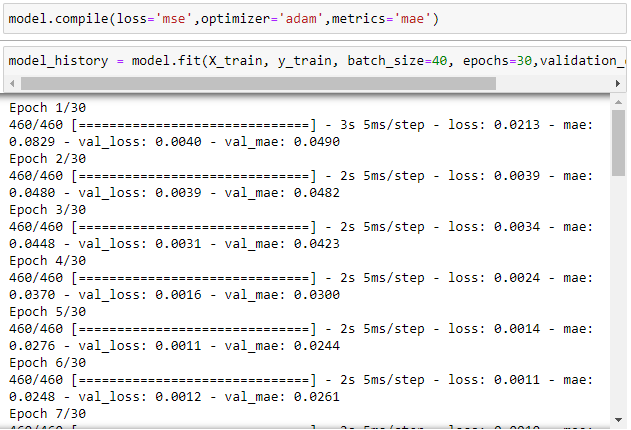
Пример 1

Создадим модель с двумя слоями LTSM. Первый слой содержит 16 нейронов, использует функцию активации Relu и форму входных данных (window, 1), где window – определенный нами заранее размер данный и 1 это количество признаков. Второй слой LTSM содержит 8 нейронов и использует функцию активации Relu. Добавляем полносвязный слой Dense с 1 выходом.



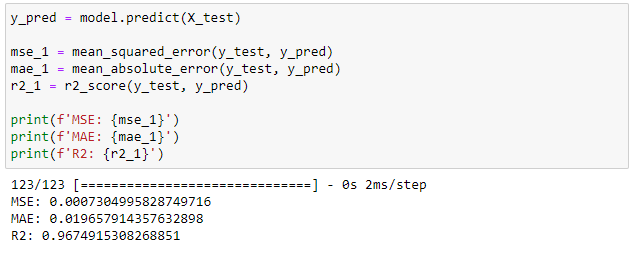
Листинг 36. Создание модели

Модель сконструирована, настроим процесс ее обучения вызовом метода compile и обучим модель на 30 эпохах и 40 образцах обучения, которые модель обрабатывает за одну итерацию.

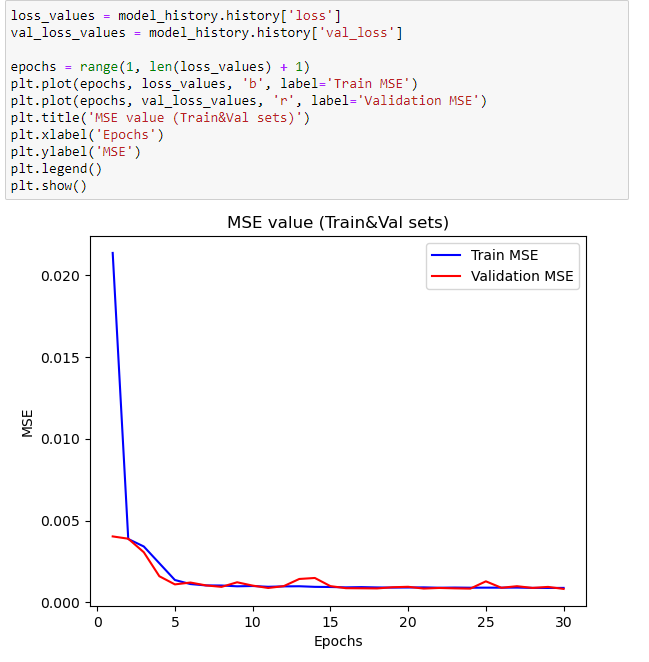


Листинг 37. Обучение модели

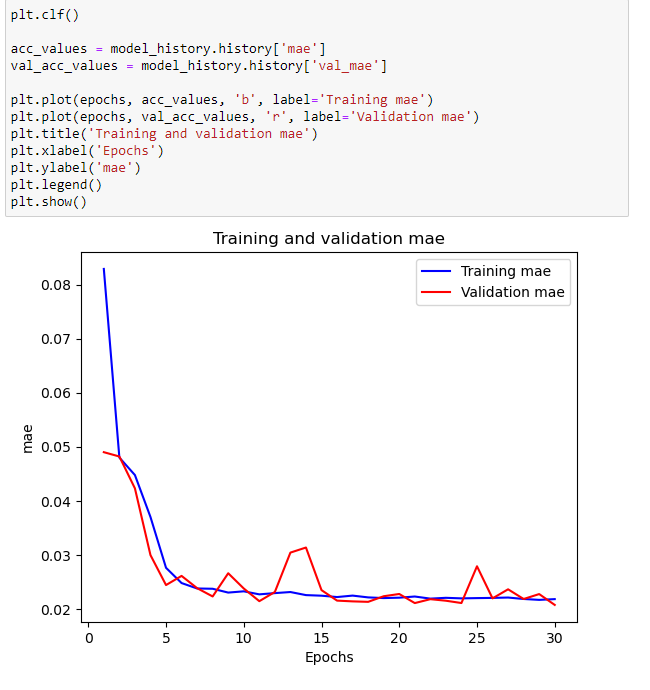
Найдем метрики работы и построим графики.

****

Листинг 38. Вычисление метрик работы



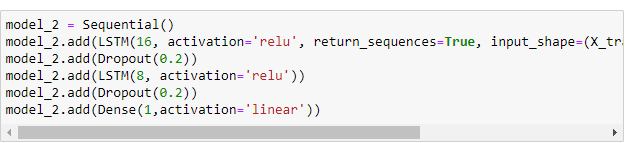
Листинг 39. Построение графика



Листинг 40. Построение графика

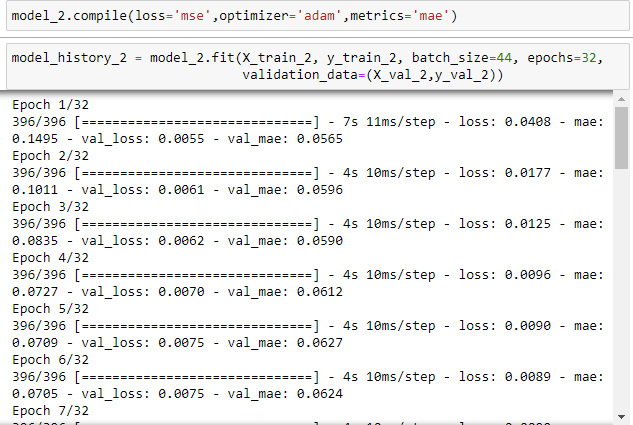
Пример 2

Проделаем то же самое с данными, подготовленными для второго примера, но теперь добавим прореживание с помощью Dropout.



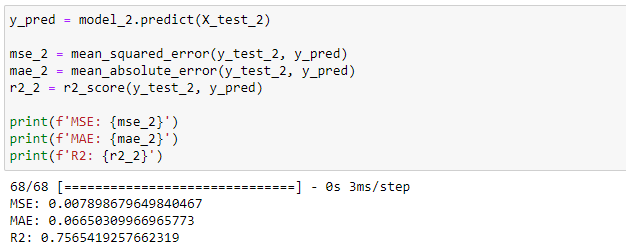
Листинг 41. Создание модели

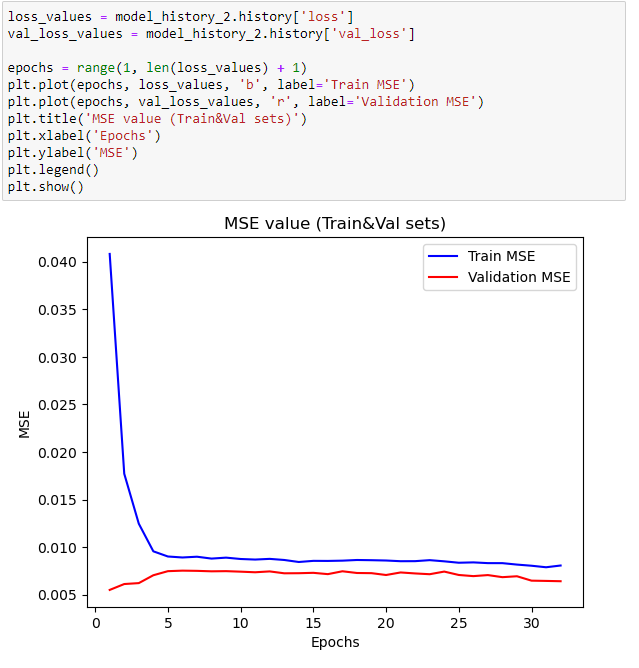
Обучать модель будем на 32 эпохах и 44 образцах обучения, которые модель обрабатывает за одну итерацию.



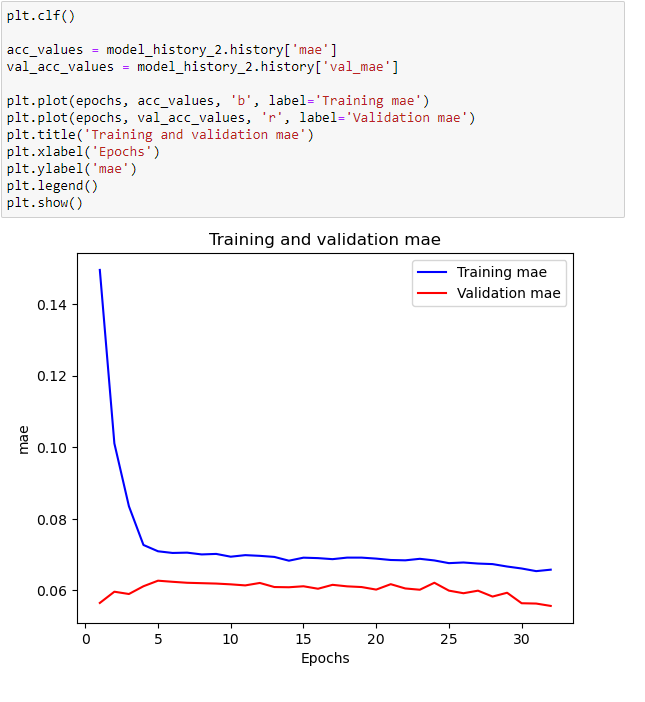
Листинг 42. Обучение модели

Вычислим метрики работы и нарисуем графики:

  
Листинг 43. Вычисление метрик работы

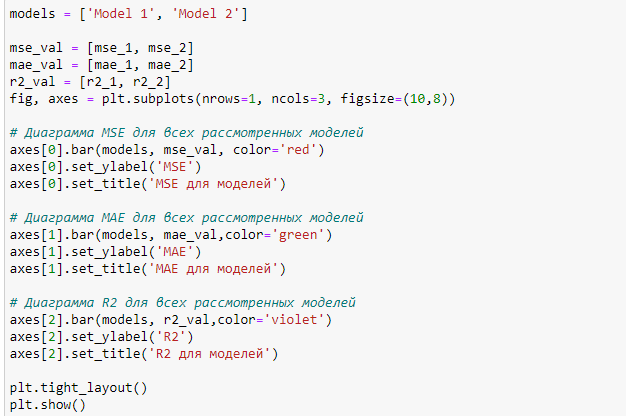


Листинг 44. Построение графика



Листинг 45. Построение графика

Проведем сравнение между двумя полученными моделями.



Листинг 46. Построение сравнительных диаграмм

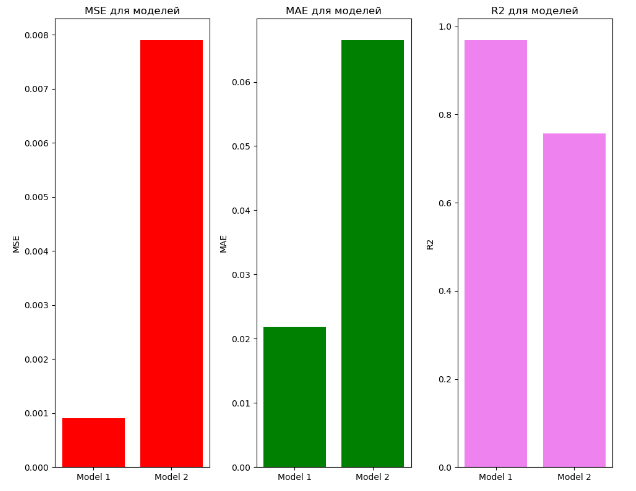


Рис 3. Сравнительные диаграммы

На диаграмме видно, что MSE и MAE **первой** модели ниже, поэтому ее точность прогнозирования больше. Значения R2 **первой** модели значительно больше, чем у второй, поэтому она лучше объясняет вариацию в данных.

**Вывод**

В ходе выполнения лабораторной работы было построено три регрессора: регрессор, использующий слой RNN в качестве скрытого слоя; регрессор, использующий ячейку LTSM в скрытом слое; регрессор – стек двух слоев с LTSM. Для всех заданных регрессоров построено по две модели и в ходе эксперимента каждая **первая** модель оказалась лучше для прогнозирования.