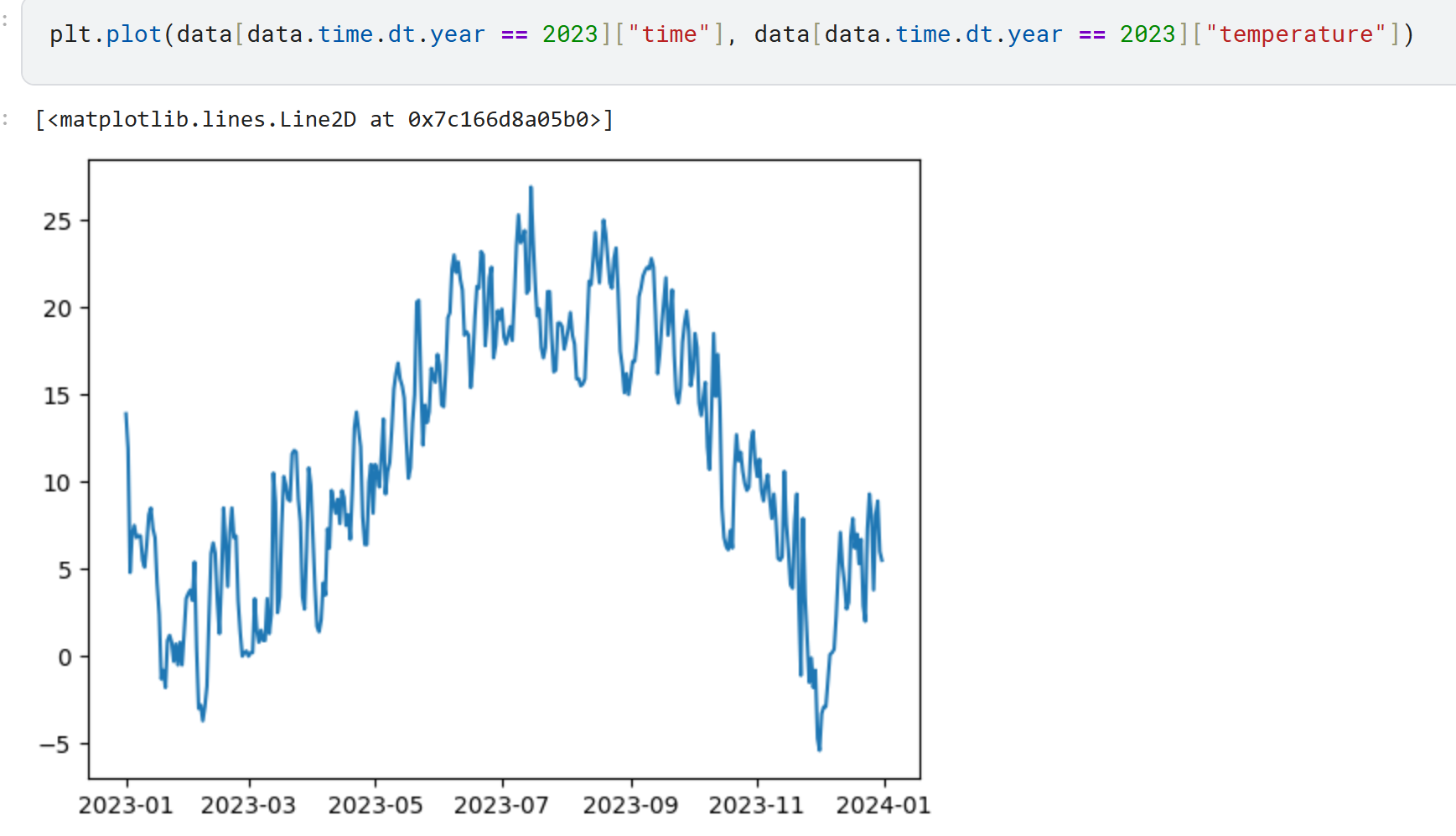
# Определение задачи

## **Цель проекта:** Построение модели для прогнозирования суточной температуры на основе исторических данных о погодных условиях. **Тип задачи:** Регрессия в рамках анализа временных рядов. **Особенности данных:**

## Многомерный временной ряд с 15 признаками (температура, влажность, направление ветра и др.).

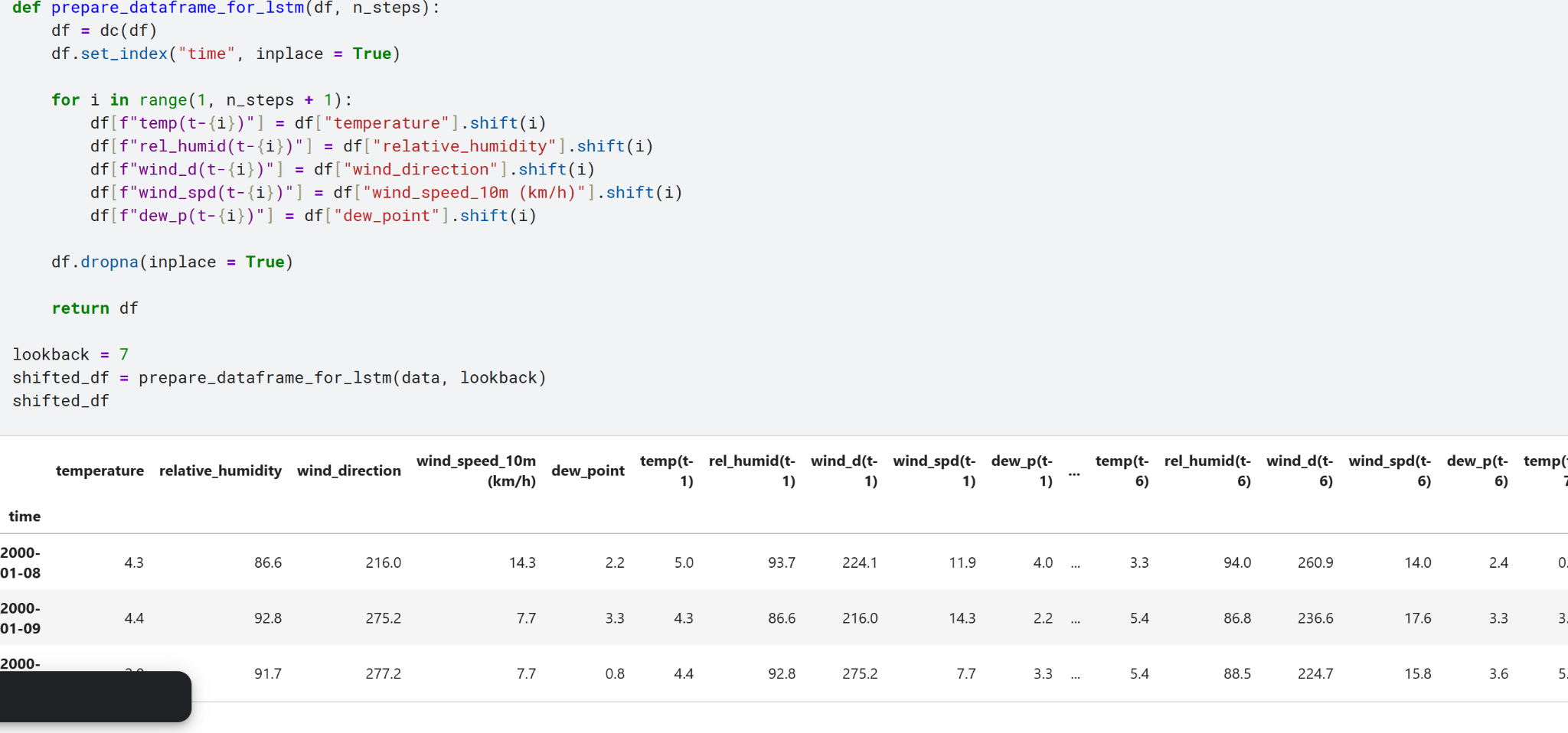
## Данные охватывают период с 2000 по 2024 год с ежедневной частотой.

## Основной целевой переменной является температура (temperature).



## **📊 Подготовка данных**

1. **Загрузка и предобработка:**
   * Преобразование временных меток в формат datetime.
   * Агрегация данных по дням с усреднением значений.
   * Удаление шума и округление значений до одного знака после запятой.
2. **Инженерия признаков:**
   * Создание оконных признаков для будущих моделей: каждая строка содержит данные за текущий день и предыдущие 7 дней.
   * Использование 5 ключевых признаков: temperature, relative\_humidity, wind\_direction, wind\_speed\_10m (km/h), dew\_point.
3. **Нормализация:**
   * Применение MinMaxScaler для масштабирования данных в диапазон [-1, 1].
4. **Разделение данных:**
   * Разделение на тренировочные (95%) и тестовые (5%) наборы.
   * Преобразование данных в тензоры PyTorch.



## **🤖 Модели для обучения**

### **Описание модели LSTM**

Модель **LSTM (Long Short-Term Memory)** — это разновидность рекуррентных нейронных сетей (RNN), предназначенная для работы с временными рядами и последовательными данными. Она решает проблему исчезающего градиента, свойственную классическим RNN, за счет механизмов забывания и запоминания информации в долгосрочной перспективе.

#### **Архитектура модели**

Представленная модель **LSTM** включает в себя:

* **Входной слой** с параметром input\_size, определяющим количество входных признаков.
* **LSTM-слой**, состоящий из num\_stacked\_layers слоев и hidden\_size скрытых нейронов. Он принимает входные последовательности и передает их через многослойную LSTM.
* **Полносвязный слой (fc)**, который преобразует выходное состояние последнего временного шага в конечный прогноз.

Во время прямого прохода:

1. **Инициализируются нулевые скрытые состояния (h0, c0)**, необходимые для обработки последовательности.
2. **Проход через LSTM-слой**, который обрабатывает входную последовательность и возвращает выходные состояния.
3. **Используется только последнее состояние (out[:, -1, :])**, так как задача регрессии предполагает прогноз на основе последнего временного шага.
4. **Выходной слой (fc) предсказывает целевое значение** (например, температуру, осадки и др.).

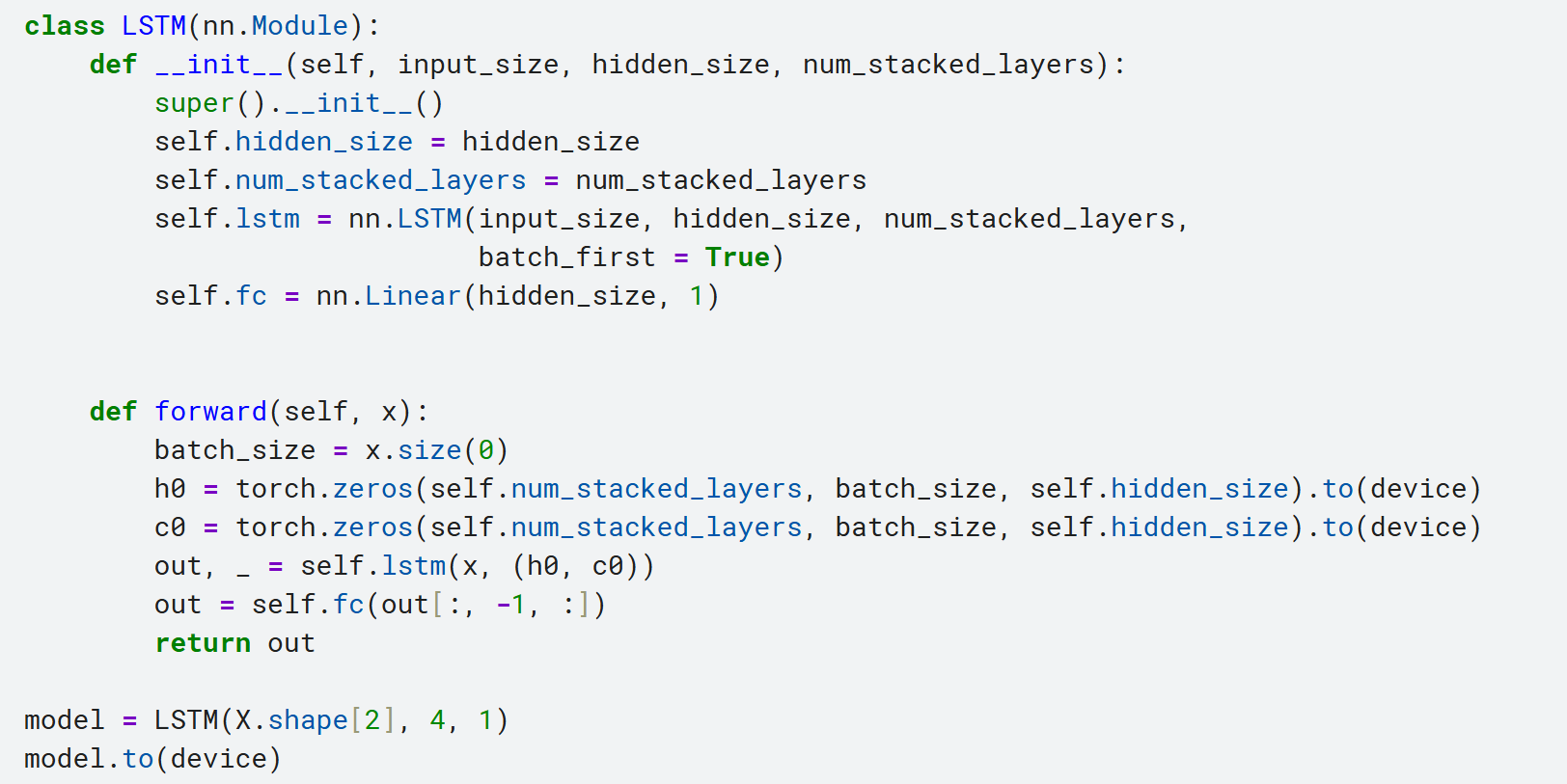
#### **Плюсы и минусы LSTM**

✅ **Плюсы**:

* Подходит для задач временных рядов, так как эффективно учитывает долгосрочные зависимости.
* Устойчивость к исчезающему градиенту благодаря механизмам input, forget и output gate.
* Хорошо работает при наличии нелинейных зависимостей в данных.

❌ **Минусы**:

* Длительное время обучения по сравнению с простыми методами, такими как линейная регрессия.
* Требует большого количества данных для корректного обучения.
* Сложнее интерпретировать по сравнению с традиционными статистическими методами..



### **Описание модели Transformer**

Модель **Transformer** — это архитектура, основанная на механизме **self-attention**, которая изначально была разработана для задач обработки естественного языка, но нашла широкое применение в прогнозировании временных рядов, обработке изображений и других областях.

#### **Архитектура модели**

Представленная **Transformer-модель** включает в себя следующие компоненты:

1. **Линейное преобразование входных данных (encoder)**
   * Исходные признаки (размерность input\_dim=5) проецируются в пространство размерностью d\_model=64.
   * Это необходимо, так как Transformer работает с фиксированной размерностью эмбеддингов.
2. **Позиционное кодирование (PositionalEncoding)**
   * В отличие от LSTM, Transformer не имеет встроенного механизма обработки последовательности, поэтому используется **синусоидальное позиционное кодирование**, которое позволяет учитывать порядок элементов во входной последовательности.
3. **Многоголовое внимание (Multi-Head Attention)**
   * Внутри nn.TransformerEncoderLayer используется механизм **self-attention**, который позволяет модели учитывать зависимости между различными временными шагами.
   * Используется nhead=4 (количество голов в механизме внимания).
4. **Каскад энкодеров (nn.TransformerEncoder)**
   * Включает num\_layers=2 слоев, каждый из которых состоит из механизма **self-attention** и полносвязных слоев с нормализацией.
   * Позволяет модели захватывать сложные паттерны в данных.
5. **Выходной линейный слой (decoder)**
   * После прохода через трансформер используется только **последний временной шаг** для предсказания целевого значения.

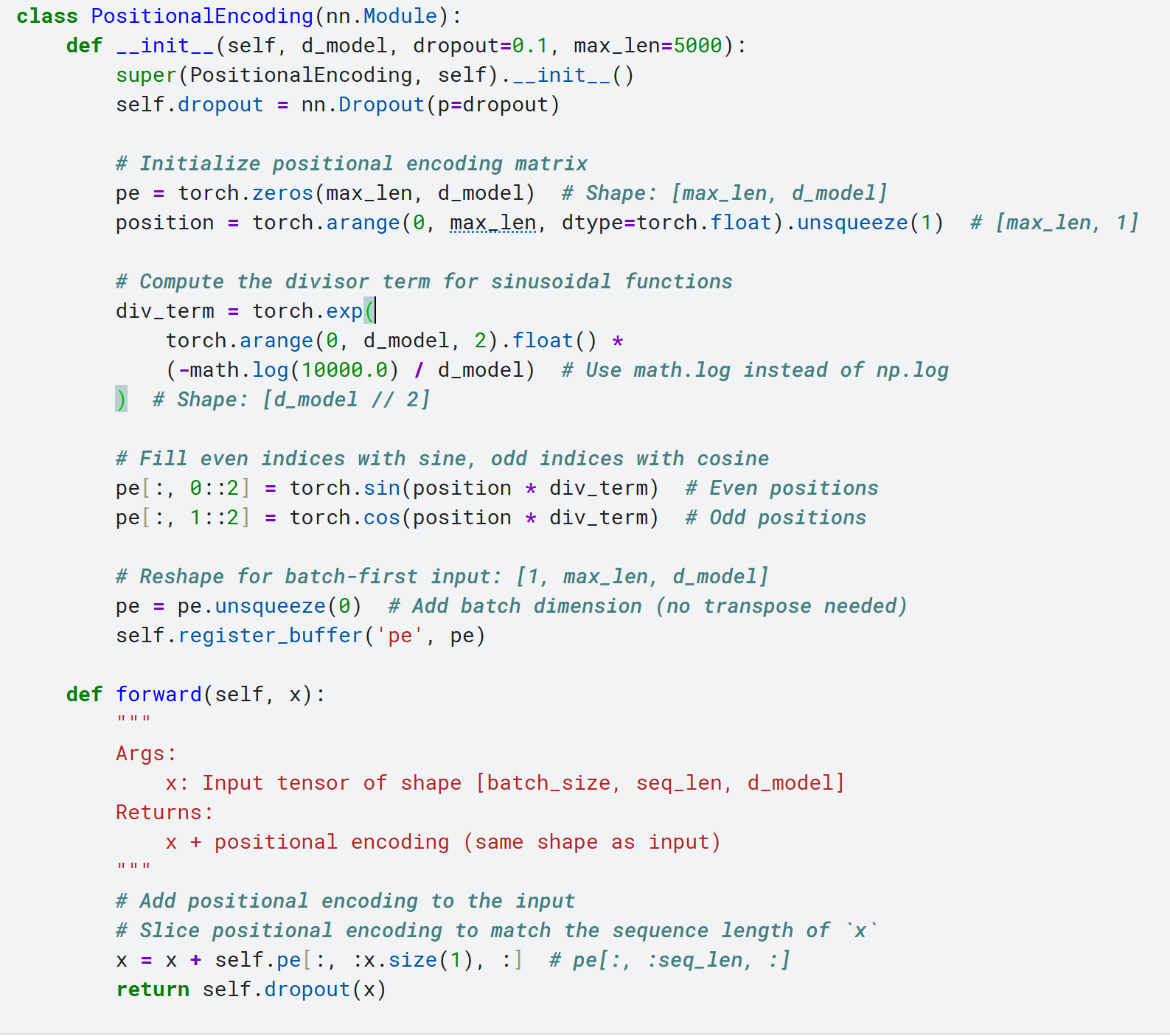
#### **Плюсы и минусы Transformer**

✅ **Плюсы**:

* Эффективно моделирует **долгосрочные зависимости** благодаря self-attention.
* Лучше параллелизуется по сравнению с LSTM (не требует последовательной обработки).
* Хорошо подходит для **многомерных временных рядов**, так как может одновременно учитывать зависимости между различными признаками.

❌ **Минусы**:

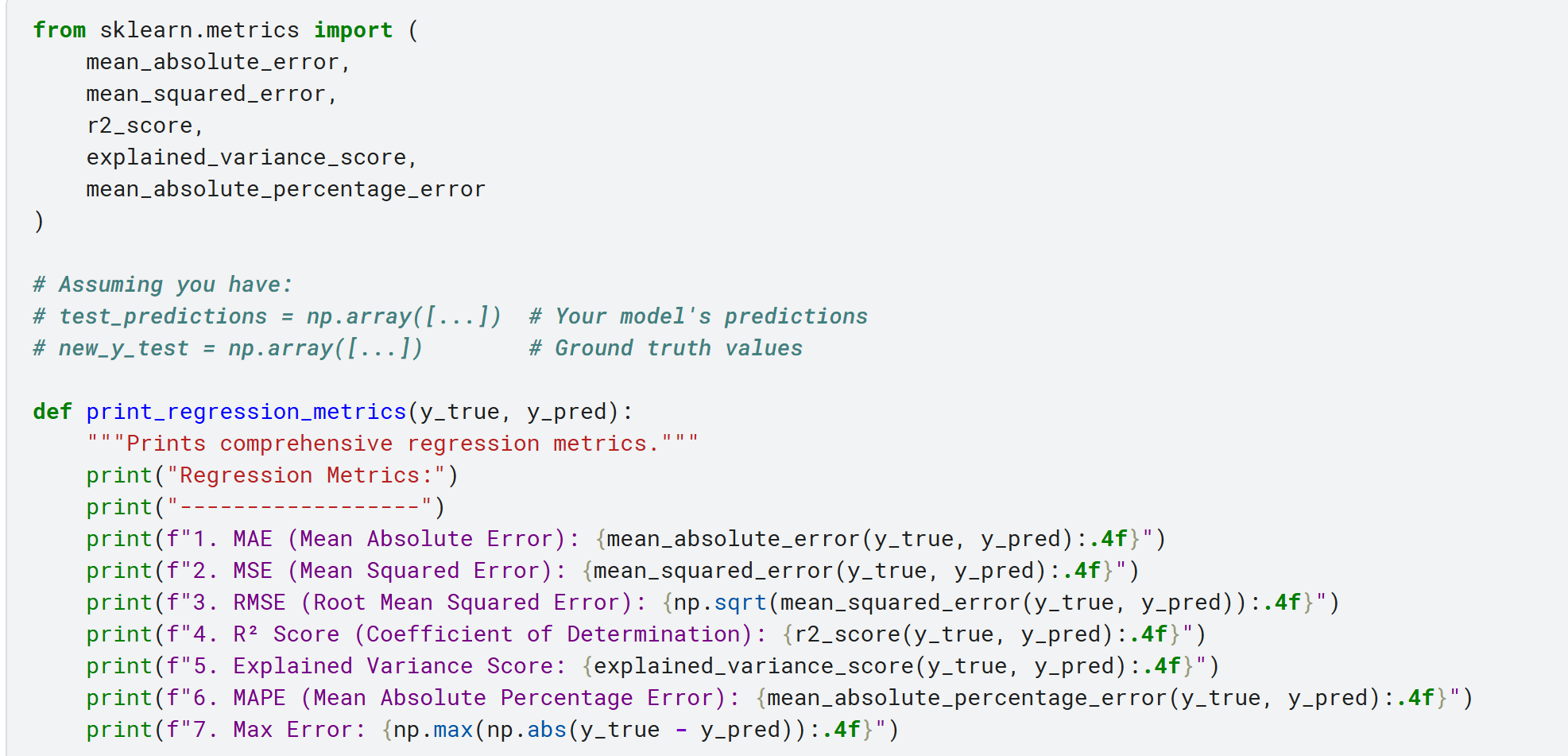
* **Высокая вычислительная сложность**, особенно для длинных последовательностей (квадратичная сложность по отношению к длине последовательности).
* Требует большого количества данных для хорошей генерализации.
* Временные паттерны могут быть хуже уловлены без дополнительной обработки (например, в чистом виде Transformer не использует рекуррентные механизмы).





### **Оценка качества моделей**

Для оценки точности предсказаний используемых моделей применяется набор метрик регрессии. Код ниже позволяет вычислить основные показатели, оценивающие, насколько хорошо модель предсказывает значения временных рядов:



### **Описание метрик**

1. **MAE (Mean Absolute Error)** – средняя абсолютная ошибка предсказаний. Показывает среднее отклонение прогнозов от фактических значений.
2. **MSE (Mean Squared Error)** – среднеквадратическая ошибка, более чувствительна к выбросам из-за квадратичного масштаба.
3. **RMSE (Root Mean Squared Error)** – корень из MSE, который сохраняет единицы измерения целевой переменной.
4. **R² Score (Коэффициент детерминации)** – показатель, отражающий долю объясненной дисперсии. Чем ближе к 1, тем лучше модель объясняет данные.
5. **Explained Variance Score** – аналог R², но с меньшей чувствительностью к смещению модели.
6. **MAPE (Mean Absolute Percentage Error)** – средний абсолютный процент ошибки, удобен для понимания относительной точности модели.
7. **Max Error** – максимальная ошибка между предсказанным и фактическим значением, полезна для оценки худшего случая.

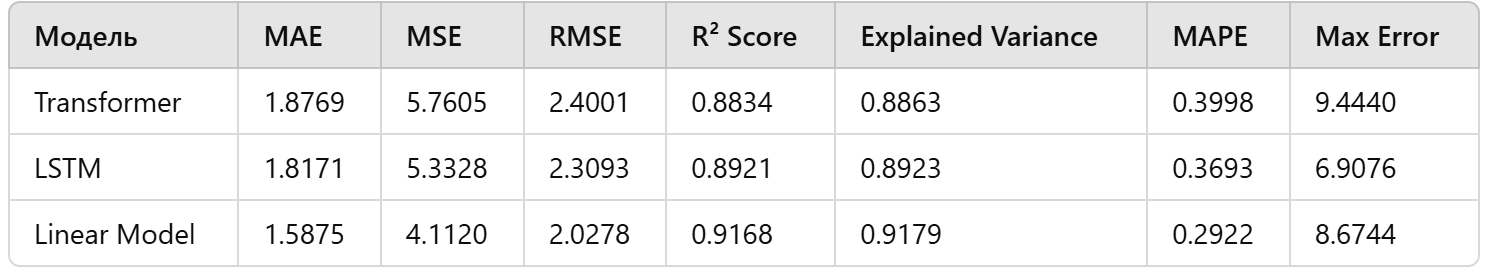
**Отчет по результатам моделей**

### **1. Введение**

В ходе эксперимента были протестированы три модели для задачи прогнозирования:

* **Transformer**
* **LSTM**
* **Линейная регрессия**

Модели оценивались по стандартным метрикам регрессии, включая MAE, MSE, RMSE, R², Explained Variance Score, MAPE и Max Error.



### **3. Анализ**

1. **Линейная регрессия показала лучшие результаты** по большинству метрик, включая наименьший MAE (1.5875) и MSE (4.1120), а также наибольший R² (0.9168).
2. **LSTM немного лучше Transformer'а**, особенно по R² (0.8921 против 0.8834) и MAPE (0.3693 против 0.3998).
3. **Нейросетевые модели пока уступают линейной**, что может быть связано с недостатком данных или чрезмерной сложностью моделей для данной задачи.
4. **Максимальная ошибка у Transformer'а самая высокая** (9.4440), что может говорить о нестабильности предсказаний на отдельных примерах.

### **4. Выводы и рекомендации**

* **Для данной задачи линейная регрессия оказывается наиболее эффективной.** Это может говорить о том, что временные зависимости не слишком сложны, и более сложные модели не дают значительного выигрыша.
* **Нейросетевые модели могут показать лучшие результаты при увеличении объема данных.** Если будет доступно больше данных, стоит попробовать дообучить LSTM/Transformer.
* **Стоит проверить простую стратегию копирования предыдущего дня.** Если она дает результаты, сопоставимые с лучшей моделью, значит, сложные модели могут быть избыточны.

## В целом, несмотря на лучшие результаты линейной модели, нейросетевые подходы остаются перспективными при наличии большего объема данных и более сложных зависимостей.