04 - Тема 4 - Временные ряды - Основы

Временной ряд (одномерный) — это измерения одной и той же случайной величины в разные моменты времени.

В бизнесе временные ряды ценятся за возможность прогнозировать значение целевой переменной на будущее. Самые очевидные примеры использования временных рядов в бизнесе — финансовое моделирование и прогнозирование спроса или выручки.

Ниже приведено ещё несколько примеров применения временных рядов в бизнесе.

1. Количество посетителей сайта о футболе существенно увеличивается во время матчей. Владельцам сайта может быть интересно узнать, насколько сильно вырастет это число, чтобы запланировать использование дополнительных серверов на время матча. Также важно знать, когда число пользователей уменьшится, чтобы отключить дополнительные серверы и сэкономить деньги.
2. Интернет-магазин товаров для правильного питания каждый день собирает и анализирует отзывы клиентов и оценки, которые показывают, насколько они довольны услугами компании: от -1 (самый недовольный) до +1 (самый довольный).
   - Вдруг оценка начинает постепенно снижаться, и компания встаёт перед выбором: немедленно отреагировать или сэкономить время и ресурсы, поскольку отклонение может быть временным и скоро закончится без дополнительного вмешательства.
   - Прогноз временного ряда показывает, что уровень удовлетворённости вряд ли станет лучше, а в ближайшие несколько дней продолжит падать до неприемлемого уровня. Исходя из этой модели, компания решает привлечь дополнительные ресурсы, чтобы помочь команде обслуживания клиентов уделять им особое внимание и пресечь эту тенденцию.
3. Приложение-навигатор компании по перевозке грузов получает данные с уличных сенсорных устройств, которые каждые 20 минут регистрируют количество транспортных средств, пересекающих определённый перекресток в центре города.
   - Используя эту информацию, модель временного ряда предсказывает, что в следующие 20 минут на перекрестке, вероятно, возникнет пробка. В результате приложение решает перенаправить поездки курьеров, чтобы избежать перегруженного перекрёстка.

Теперь выделим несколько формальных свойств временного ряда:

1. Данные временного ряда **структурированы**, а атрибуты (так иногда называют признаки) **зависимы от времени**. На графике ниже визуализирован временной ряд фактического дохода компании. Данные структурированы (упорядочены и находятся в таблице), а фактический доход — это атрибут, зависящий от времени.

	A	B
1	Год	Фактический доход, млн. долл.
2	1982	581,5
3	1983	581,7
4	1984	590,5
5	1985	620,3
6	1986	699,0
7	1987	781,1
8	1988	891,4
9	1989	992,9
10	1990	1110,6
11	1991	1148,9
12	1992	1301,3
13	1993	1440,4
14	1994	1661,3

Рис. 1: 5548e04e645f52fc74c5e81871ae955c.png

2. Данные временного ряда, в отличие от любых других данных, имеют **определённую последовательность**. На графиках ниже отображены продажи мороженого с января по декабрь.

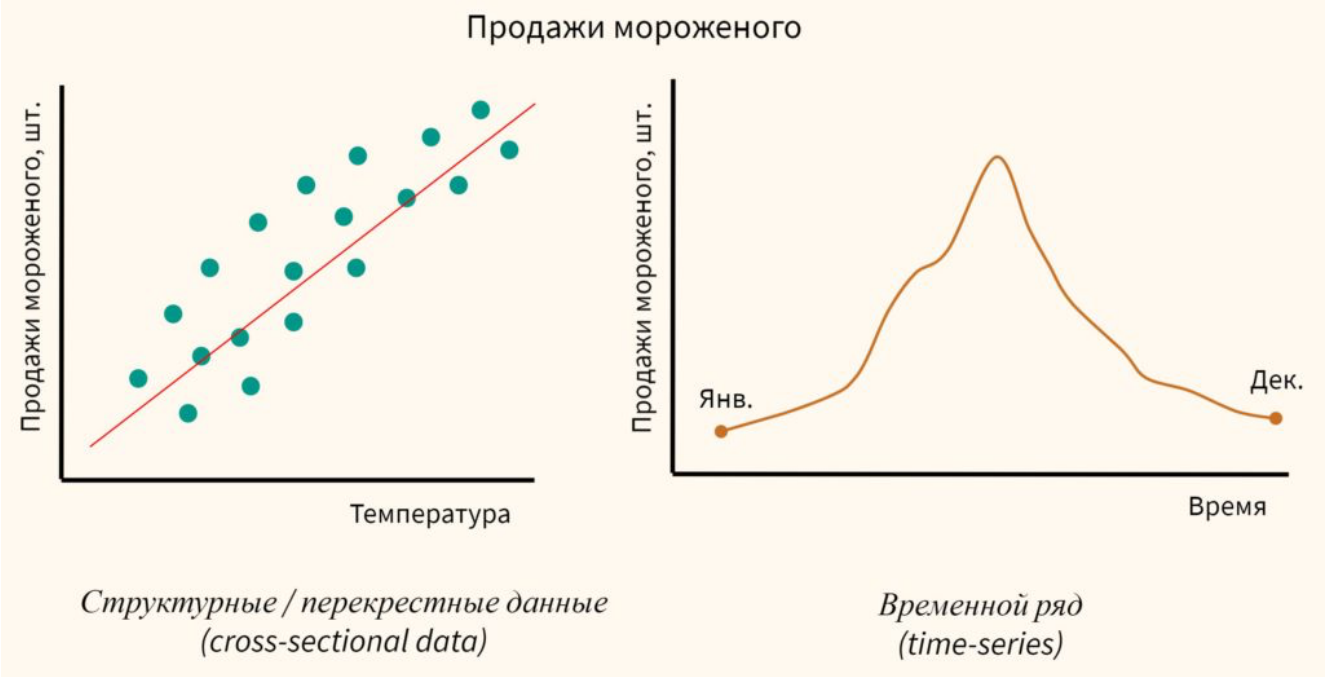

Рис. 2: 9e13803331e38e5a6e5daa1e2673ba34.png

3. В отличие от анализа других данных, в анализе временных рядов важно, чтобы последовательные значения в данных наблюдались **через равные промежутки времени**, например каждый час, неделю, год, каждый понедельник и так далее. На графике ниже визуализирован временной ряд с получением данных за равные промежутки времени (дни).

Итак, теперь мы знаем, что отличительной особенностью временного ряда является наличие временного измерения или номер измерения по порядку, а также что при анализе данных временного ряда учитывается зависимость атрибутов от времени.

Далее познакомимся с неотъемлемыми составляющими временного ряда — трендом, сезонностью и шумом.

Тренд, сезонность, шум. Инструменты для декомпозиции временного ряда

Тренд **Тренд** — это основная тенденция изменения величины со временем.

Если повезёт (как правило нет), тренд будет линейным и предсказать его будет проще.

Ниже на графике с измерениями пульса тренд нелинейный, и здесь можно выделить: * участок старта, когда пульс растёт до более стабильного значения; * более-менее линейный участок основной дистанции; * участок финиша, когда пульс быстро и нелинейно идёт вверх.

Большое преимущество тренда — его можно прогнозировать как функцию времени, не учитывая предыдущие значения временного ряда.

Сезонность и цикличность

Сезонность **Сезонность** задаёт периодические колебания ряда вокруг тренда. Сезонность есть не всегда, но очень часто.

Например, **продажи автомобилей** каждый год немного растут в декабре и падают в январе следующего года.

Затем необходимо оценить цикличность.

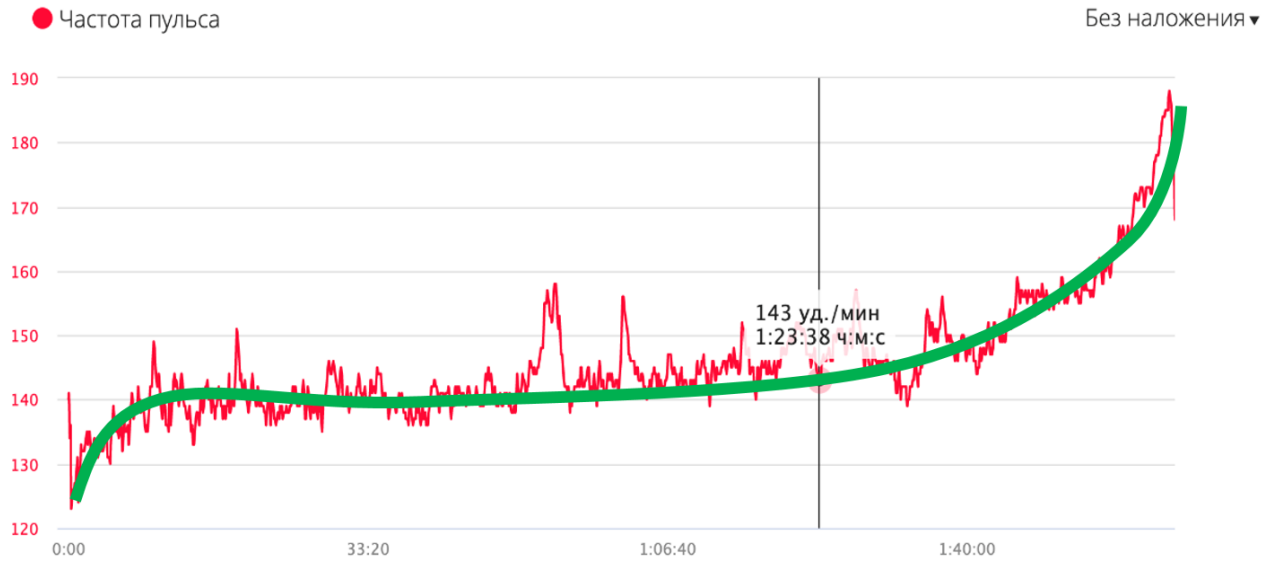

Рис. 3: 874c2f52f13ce7f65185506402cfe29e.png

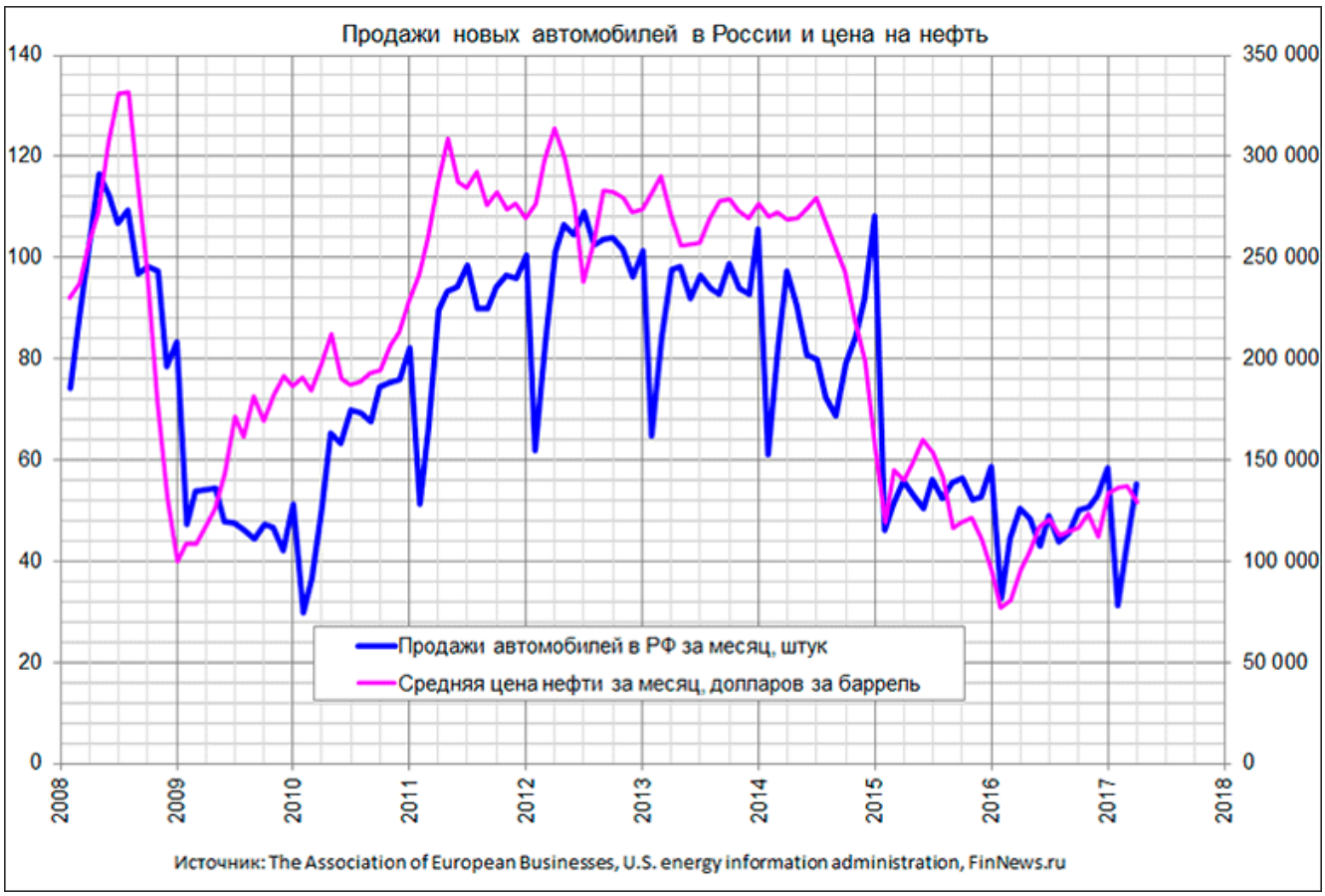

Рис. 4: 0ec343a789625778da6bbf714f4ddf6f.png

Цикличность **Цикличность** — это колебания временного ряда относительно тренда.

Отличием цикличности от сезонности является то, что сезонность возникает из периода в период (каждый декабрь, каждые выходные и т. д.), а цикличность проявляется на более длительных дистанциях и может слегка меняться от цикла к циклу.

На графике ниже показан цикл изменения объёма валового национального продукта в зависимости от времени. Такая цикличность скорее зависит не от сезона/квартала, а от внешних факторов.

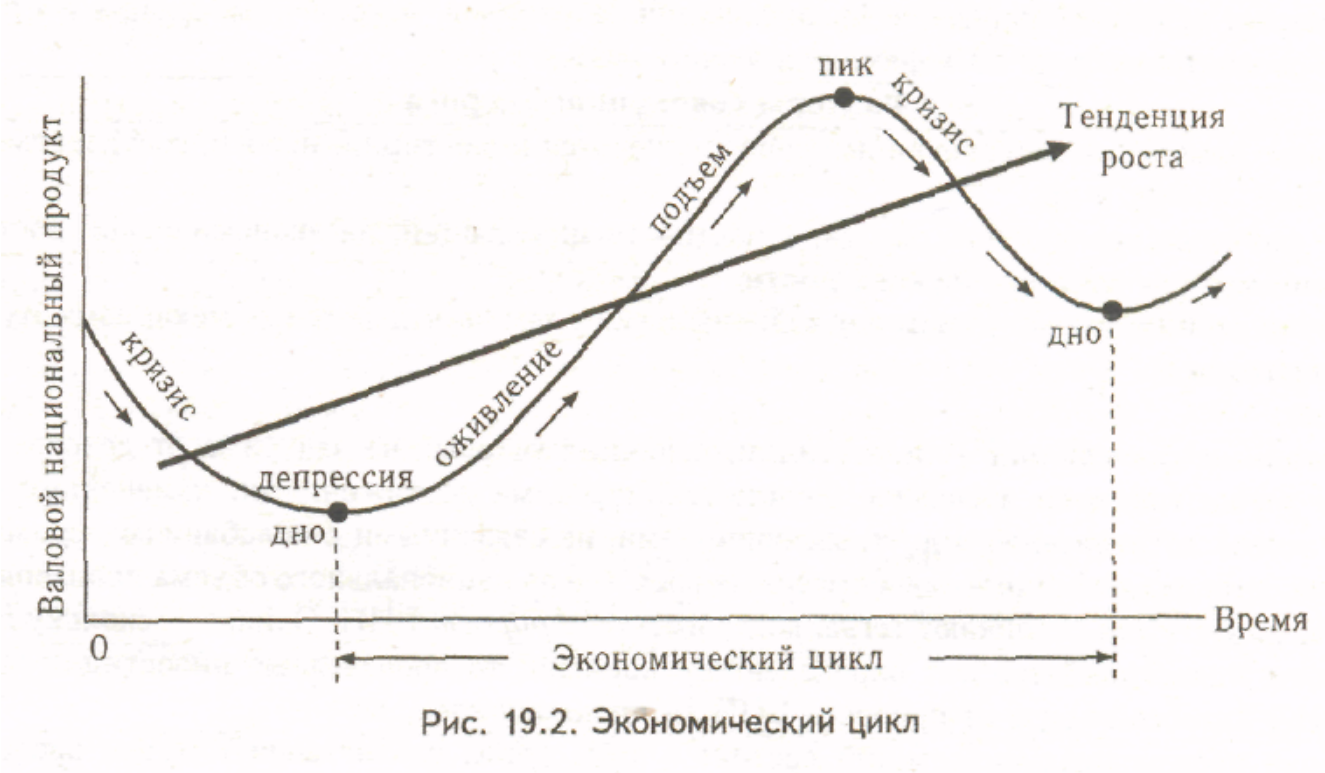

Рис. 19.2. Экономический цикл

Рис. 5: d8cc5133447559e64beee95e43aef213.png

Проверка на шум **Белый шум** — это значения, которые являются независимыми друг от друга и одинаково распределены в районе нуля на протяжении всего временного интервала.

Как только мы получили белый шум в остатке ряда, дальше прогнозировать бессмысленно.

Обратите внимание на схему ниже и сравните все компоненты. Нерегулярность отвечает за шум.

Варианты сложности временного ряда

1. **Простой.** Самый простой вариант предсказания данных для временного ряда:
   - Выявили линейный тренд — остался белый шум.
2. **Более сложный вариант**:
   - Убрали тренд — осталась сезонность.
   - Убрали сезонность — остался белый шум.
3. **Самый сложный вариант**:
   - Убрали тренд и сезонность — остался всё ещё не белый шум.

Что делать в самом сложном случае? Зависит от задачи: возможно, для магазина достаточно учесть нестандартное поведение в праздничные дни, а для средств в банкомате — еженедельную инкассацию.

Давайте сведём основную информацию в небольшой список: - Тренд. Описывает чистое влияние долговременных факторов, изменяется плавно. *Пример: рост численности населения.* - Цикличность. Состоит из циклов, меняющихся по длительности и амплитуде, описывает периоды подъёма и спада. *Пример: циклы в экономике, связанные с изменением спроса и предложения или с переменами в финансовой и налоговой политике.* - Сезонность. Представляет собой последовательность почти повторяющихся циклов. *Пример: объёмы продаж цветов накануне 8 марта или авиабилетов в сезон отпусков летом.* - Шум (случайная компонента). Останется после вычитания всех вышеперечисленных компонентов. Не несёт никакого глубокого смысла (в идеале).

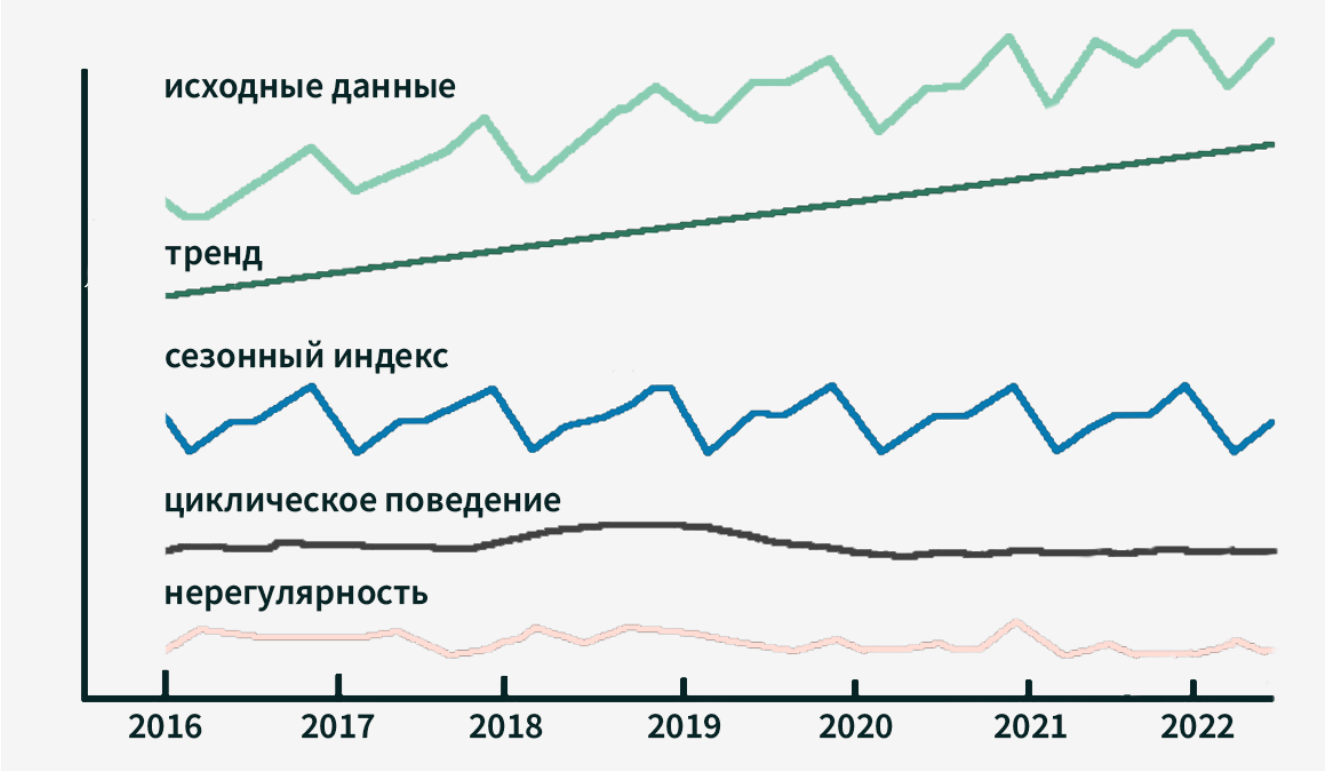

Рис. 6: 54f23d034963709cb866beeb32138acf.png

Инструменты для декомпозиции временного ряда

Разделить ряд на компоненты в *Python* можно с помощью библиотеки statsmodels. Если вы не устанавливали её ранее, это можно сделать стандартным способом (pip install statsmodels) или следуя рекомендациям в официальной документации.

В датасете `AirPassengers.zip` представлены данные о количестве авиапассажиров в 1949-1960 годах. Рассмотрим некоторый временной ряд df из данного датасета.

```
df = pd.read_csv("AirPassengers.csv", index_col='Month', parse_dates=['Month'])
```

Теперь рассмотрим код, в котором демонстрируется, как применять декомпозицию к временному ряду df.

Декомпозиция выполняется методом seasonal_decompose(), который принимает на вход временной ряд с одним признаком. Индексом ряда должна быть дата или время. Именно поэтому, считывая датасет, мы указываем индексом столбец month и приводим его к формату даты. Также, если вы знаете, что в вашем временном ряду присутствует период, его вы также можете передать в качестве параметра в `seasonal_decompose()`. Более подробно о необязательных параметрах можно узнать в документации.

```
_# импортируем библиотеку
from statsmodels.tsa.seasonal import seasonal_decompose
# производим декомпозицию временного ряда
decomposition = seasonal_decompose(df)
fig = decomposition.plot()
plt.show()
```

В результате выполнения кода мы увидим примерно следующий результат:

Как можно увидеть, по исходному ряду был получен тренд, выявлена некоторая сезонность и шум.

Экспоненциальное сглаживание

Одним из интересных способов анализа временного ряда является экспоненциальное сглаживание. Давайте разберёмся, что это такое и чем оно отличается от других методов прогнозирования.

Экспоненциальное сглаживание — это метод прогнозирования временных рядов для одномерных данных с трендом или сезонным компонентом. Оно также известно как метод простого экспоненциального сглаживания, или метод Брауна.

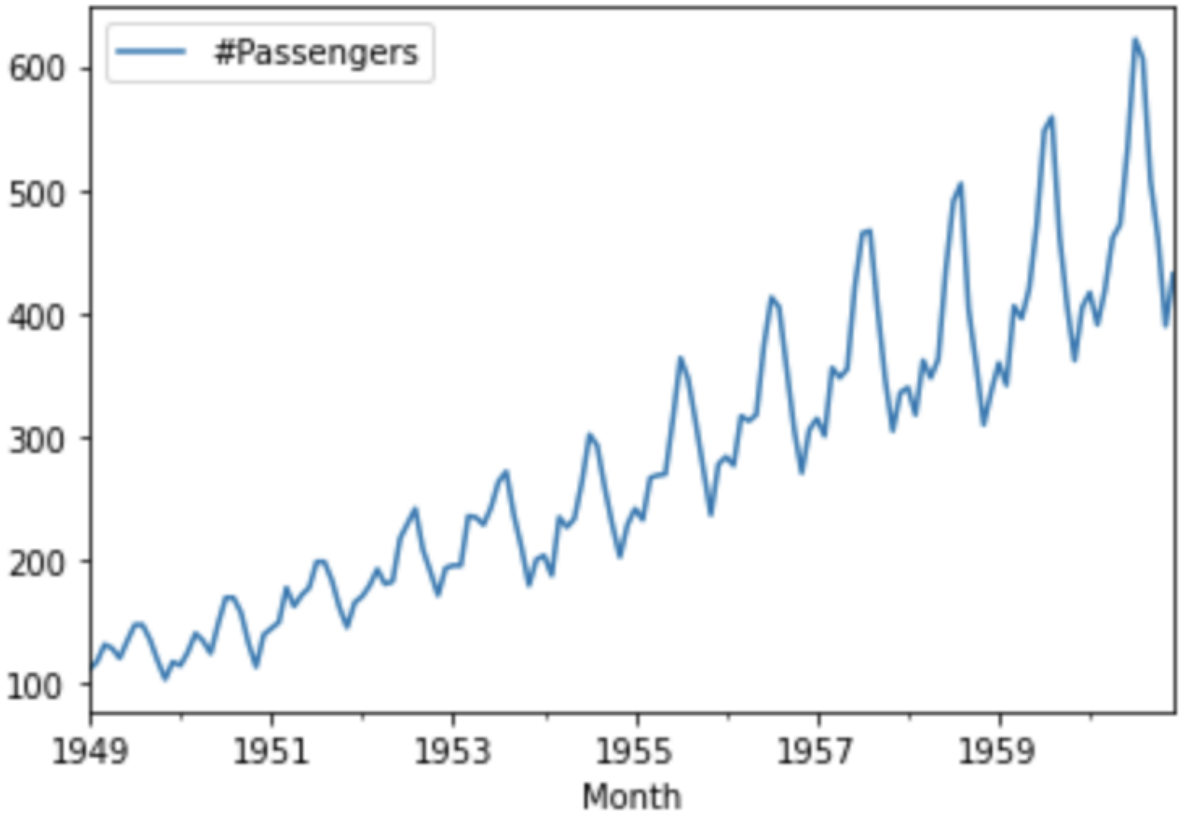

Рис. 7: 2f96a260a431f7cafc0f077479a594da.png

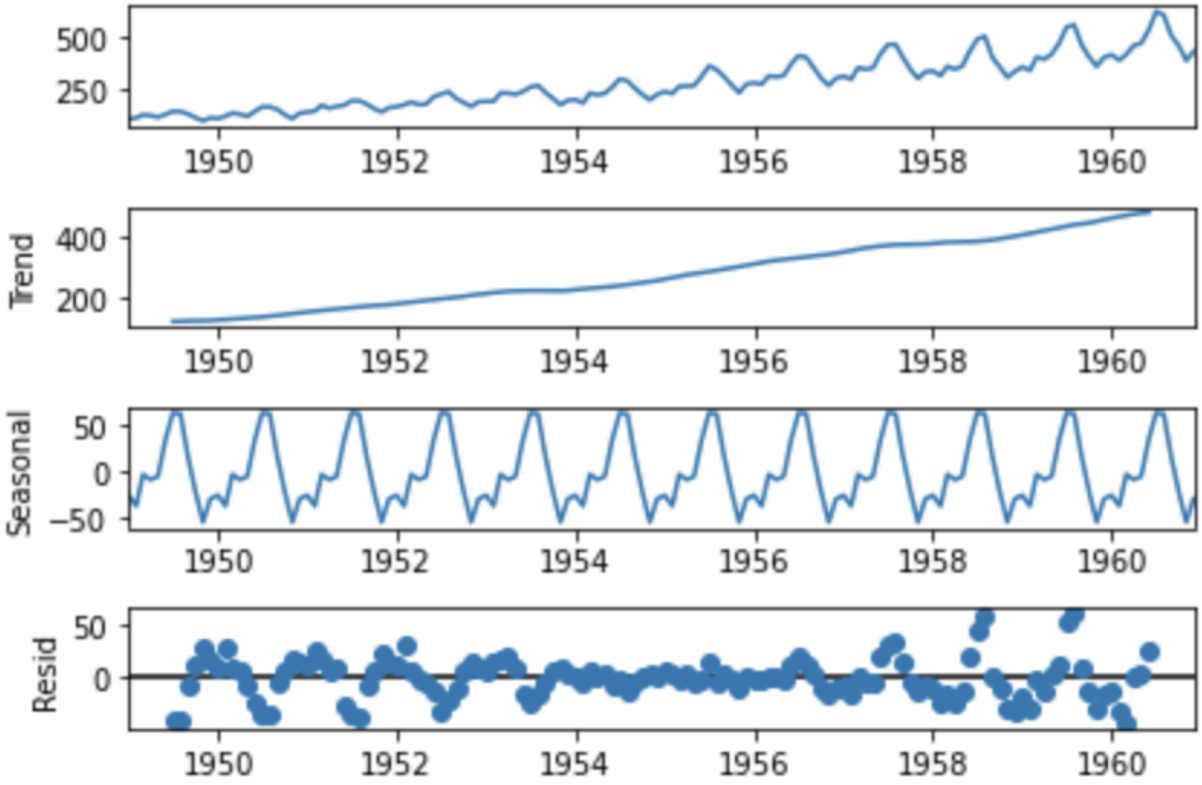

Рис. 8: 8cc41353f89ae3da12bfa60c046111a0.png

Формула для получения экспоненциального сглаживания выглядит так:

$$S_0 = X_0$$
$$S_t = \alpha \cdot X_{t-1} + (1 - \alpha) \cdot S_{t-1}$$

где:

S_t — сглаженное значение в момент времени;

X_t — фактическое наблюдение в момент времени;

α — коэффициент сглаживания, который выбирается априори.

Обратите внимание, что формулу необязательно заучивать: главное — чтобы вы понимали принцип работы алгоритма.

В случае с этой формулой каждый новый элемент временного ряда рассчитывается от предыдущего и от сглаженного исходного.

Рассмотрим пример.

Нам известны значения температуры за прошедший месяц, и мы хотим предсказать погоду на следующий день. Мы можем предположить, что в этом случае погода завтра в большей степени будет зависеть от погоды вчера и сегодня, чем от погоды 30 дней назад. Если мы хотим учитывать удалённость значений от текущего момента, то экспоненциальное сглаживание пригодится тут как нельзя кстати.

Проще говоря, под **экспоненциальным сглаживанием** понимается взвешенная линейная сумма наблюдений, при этом веса для наблюдений экспоненциально уменьшаются для более старых наблюдений. Тем самым мы не обращаем особого внимания на поведение в прошлом, а недавнему поведению присваиваем больший вес. Если быть точнее, наблюдения взвешиваются с геометрически уменьшающимся коэффициентом.

Так, если значения температуры за последние пять дней были `data = np.array([15, 20, 25, 30, 25, 27])` (в формуле это будет ряд X_t, при этом $X_0 = 15$, $X_1 = 20$ и т. д.), а **коэффициент сглаживания** α будет равен 0.7, то, подставив значения в формулу, получим значения сглаженного экспоненциального ряда: `[15.0, 18.5, 23.05, 27.915, 25.8745, 26.66235]`.

```
def exp_smth(x_t, a, s_t_1):
    return a * x_t + (1 - a) * s_t_1

data = np.array([15, 20, 25, 30, 25, 27])
a = 0.7
s = np.zeros(6)
s[0] = data[0]

for i in range(1, len(data)):
    s[i] = exp_smth(data[i], a, s[i-1])

print(s)
```

Так как по формуле значение для следующего дня рассчитывается от значения для текущего, мы можем продолжить получать следующие значения для экспоненциально сглаженного ряда, таким образом совершая прогноз (день за днём). В нашем примере для шестого дня мы получили прогнозируемое значение температуры в 26.6 градусов (26.66235, если быть точнее):

Коэффициент экспоненциального сглаживания подбирается интуитивно. Чем выше коэффициент, тем меньше внимания мы обращаем на старые данные. Если коэффициент близок к 0, данным в далёком прошлом будет уделено больше внимания. Так, при коэффициенте, равном 0.1, значения экспоненциально сглаженного ряда будут выглядеть так (сравните с предыдущим графиком):

В примере выше сглаженный ряд мы рассчитывали «вручную» по формуле, но делать это каждый раз нет необходимости, так как эта возможность уже встроена в библиотеку statsmodels. Для совершения предсказания методом простого экспоненциального сглаживания воспользуемся методом SimpleExpSmoothing из statsmodels.tsa.api.

Попробуйте запустить код ниже:

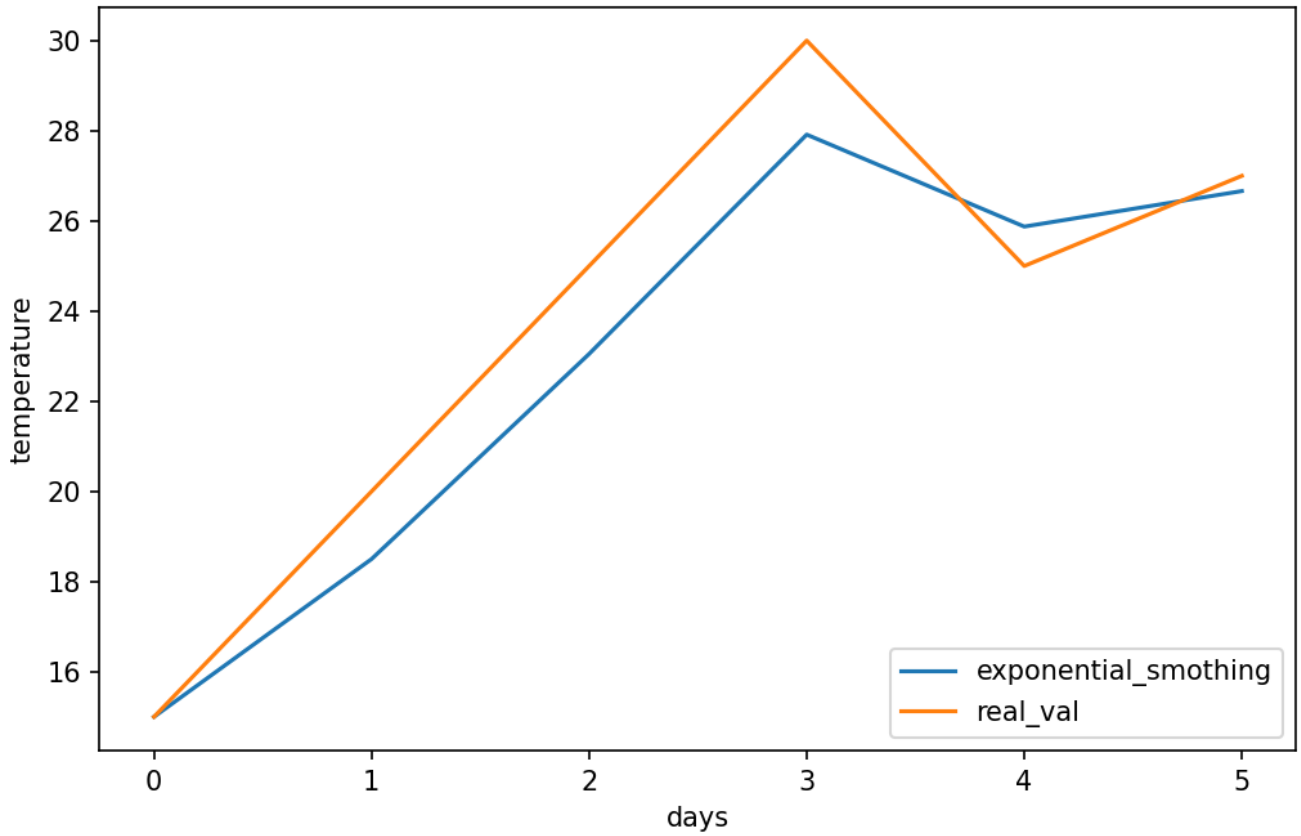

Рис. 9: e6b53fb397dc9da6183a4fcd4cc41cf7.png

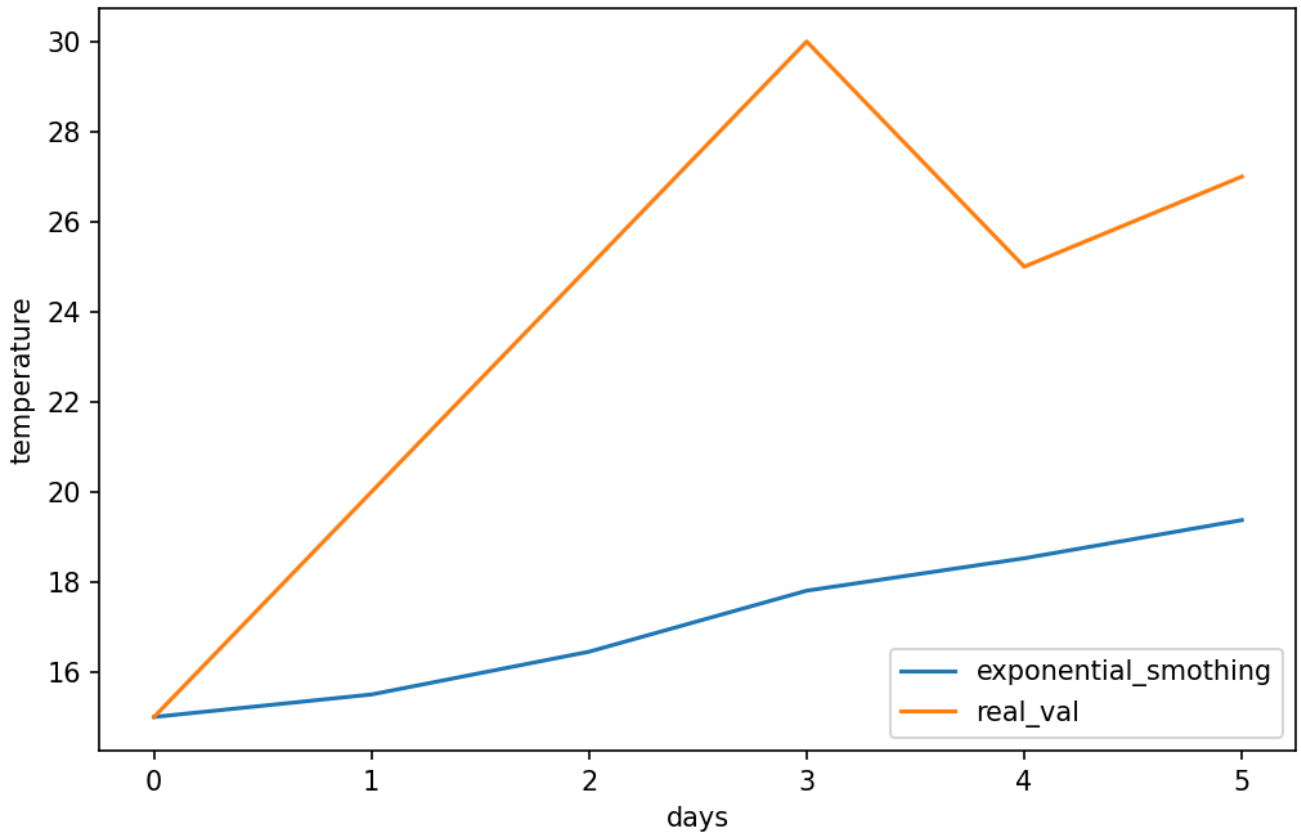

Рис. 10: e150353c03ae87a4c6cf373e201d3e77.png

```
from statsmodels.tsa.api import SimpleExpSmoothing

ses = SimpleExpSmoothing(data)

alpha = 0.7

model = ses.fit(smoothing_level = alpha, optimized = False)

forecast = model.forecast(1)

print(forecast)
```

Простое **экспоненциальное сглаживание** используется в задачах **сглаживания и краткосрочного прогнозирования** временных рядов.

Далее познакомимся с ещё одной важной характеристикой временного ряда, а пока предлагаем вам ознакомиться с дополнительной литературой по изученной теме и выполнить несколько заданий.

Дополнительные ссылки: - Статья про типы экспоненциального сглаживания - Экспоненциальное сглаживание. Распознавание образов: метод k-го ближайшего соседа

Стационарность

Прежде чем погрузиться в формальные определения стационарности и связанные с ней концепции, разберёмся, почему они важны.

Стационарность означает, что сам временной ряд может меняться с течением времени, однако статистические свойства генерирующего его процесса не меняются.

Почему это важно? Потому что стационарные процессы легче анализировать, а ещё их можно предсказывать, поскольку предсказуем способ их изменения.

Говоря простым языком, **стационарный процесс (стационарный временной ряд)** — это процесс, который не меняет свои основные характеристики со временем (обратите внимание на графики стационарного и нестационарного временного ряда выше). Это значит, что при сдвиге во времени не меняются математическое ожидание, дисперсия и совместное распределение вероятности.

Примером стационарного процесса является маятник без трения, который колеблется назад и вперёд. Из-за отсутствия трения его амплитуда и частота остаются неизменными.

- В нашем же случае временной ряд будет стационарным, если у него отсутствуют тренд и сезонность, а математическое ожидание и дисперсия при этом остаются постоянными на протяжении всего периода времени.
- У нестационарного временного ряда статистики (математическое ожидание и дисперсия) будут изменяться со временем, а сам ряд будет иметь сезонность и/или тренд.

Так как нестационарный ряд анализировать труднее, в анализе временных рядов принято приводить любой временной ряд к стационарности. Это можно сделать путём выявления и устранения тренда и сезонности.

Существует несколько методов проверки временного ряда на стационарность:

1. Визуально оценить по графику данных, есть ли какие-либо очевидные тенденции или сезонность. Например, на графике ниже нет ни выраженного тренда, ни сезонности.
2. Просмотреть сводную статистику для данных по сезонам, чтобы понять, есть ли очевидные и существенные различия.
3. Использовать статистические тесты, чтобы проверить, выполняются ли ожидания стационарности. О статистических тестах пойдёт речь ниже.

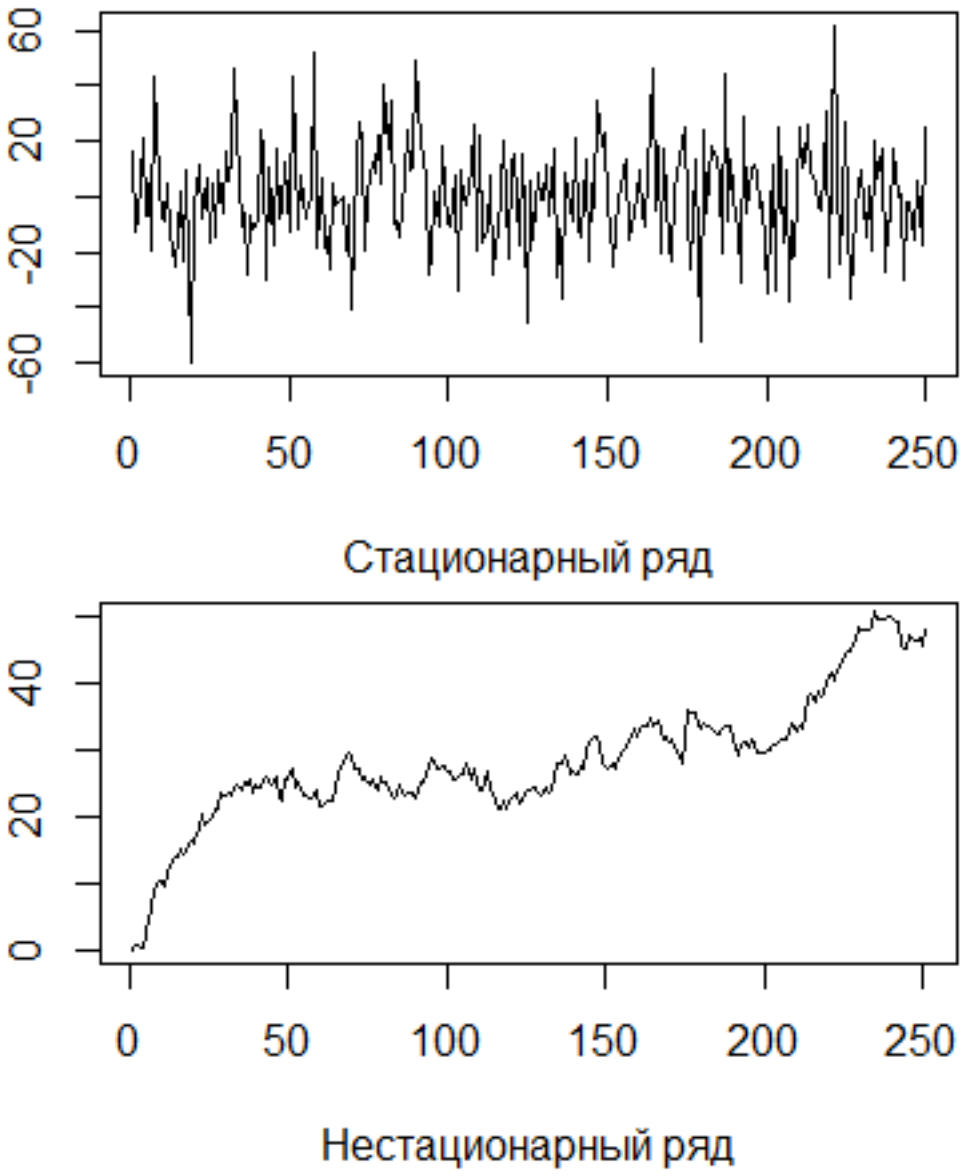

Рис. 11: 781405e88d359e59511584394bb5a5eb.png

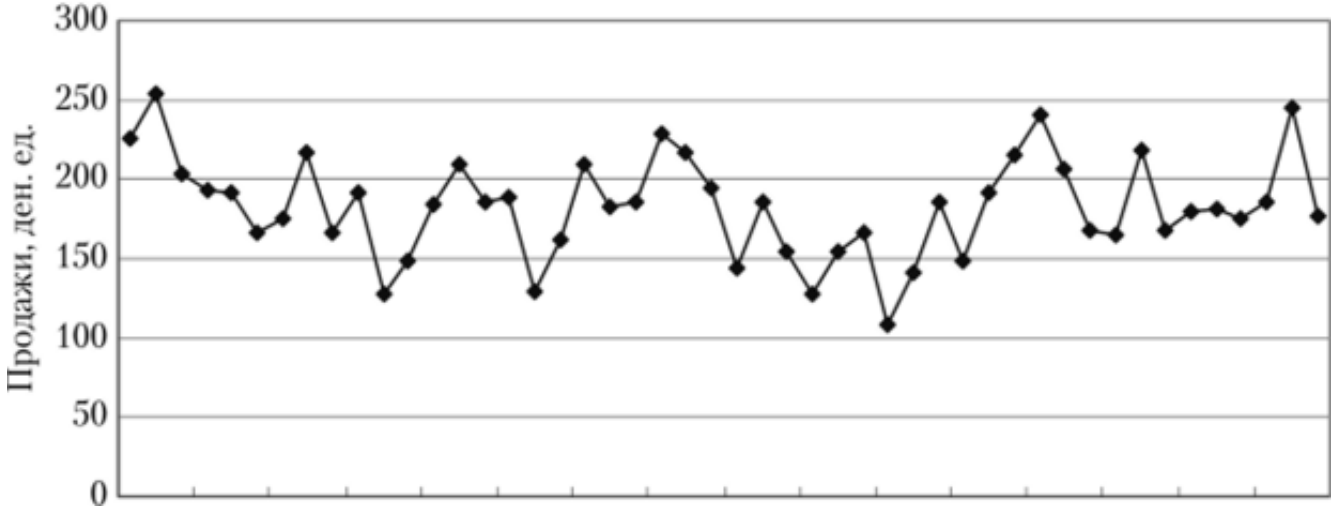

Рис. 12: 65bc1e84aee7257f103b44948b11a3b6.png

Статистические тесты на python Одним из наиболее распространённых тестов на проверку временного ряда на стационарность является расширенный тест Дики — Фуллера. В тесте формулируется две гипотезы:

- нулевая гипотеза *(H_0)*: временной ряд нестационарный, то есть имеет некоторый тренд и сезонную компоненту;
- альтернативная гипотеза *(H_1)*: временной ряд стационарный, то есть не имеет тренда и сезонной компоненты, и данные скорее случайны.

В результате проведения теста мы получим несколько значений: значение статистики из теста и критические значения разных уровней значимости (1 %, 5 %, 10 %). Уровень значимости означает допустимую для данной задачи вероятность ошибки, то есть чем ниже уровень значимости, тем ниже вероятность ошибочного результата теста.

- Если значение статистики ниже критического значения выбранного уровня значимости, отвергаем гипотезу *H0* и принимаем гипотезу *H1* (ряд стационарный).
- Если значение статистики выше критического значения выбранного уровня значимости, принимаем гипотезу *H0* (ряд нестационарный).

Рассмотрим пример проведения теста Дики — Фуллера на данных временного ряда *Daily Total Female Births* (`daily-total-female-births.zip`). В данных содержится количественная информация о девочках, рождённых за несколько месяцев. Для проведения теста будем использовать функцию adfuller() из пакета `statsmodels`.

```
import pandas as pd
from statsmodels.tsa.stattools import adfuller

df = pd.read_csv(
    "daily-total-female-births.csv",
    header=0,
    index_col=0
).squeeze("columns")

result = adfuller(df.values)

print(f"ADF Statistic: {result[0]}")
print(f"p-value: {result[1]}")
print("Critical Values:")

for key, value in result[4].items():
    print(f"\t{key}: {value:.3f}")
```

Что делать, если ряд нестационарный? Если тест на стационарность показал, что ряд нестационарный и в нём присутствуют тренд и сезонность, необходимо избавиться от них.

Обычно для этого достаточно взять разность рядов. Разность выполняется путём дифференцирования ряда, для этого вычисляется разность между двумя соседними наблюдениями ряда. Если полученная первая разность ряда окажется стационарной, то этот ряд называется **интегрированным рядом первого порядка**.

Для определения **порядка интегрированного ряда** необходимо сделать следующее:

1. Получить новый ряд посредством взятия разности (применяем к необходимому датафрейму):

```
df_diff_1 = df.diff().dropna()
```

2. Провести для нового ряда тест на стационарность (например, тест Дики — Фуллера):

```
test1 = adfuller(df_diff_1)
print ('adf: ', test1[0])
print ('p-value: ', test1[1])
print ('Critical values: ', test1[4])
if test1[0] > test1[4]['5%']:
    print ('ряд нестационарен')
else:
    print ('ряд стационарен')
```

Если полученный ряд нестационарен, можно провести эту процедуру ещё раз, то есть ещё раз дифференцировать разность ряда, полученную на предыдущем этапе.

Если после двукратного дифференцирования результат окажется стационарным временным рядом, то исходный временной ряд будет называться интегрированным рядом второго порядка, и так далее.

Запомните этот момент: далее нам пригодится порядок разности, приводящий ряд к стационарности.

Автокорреляция

Раньше вы уже встречались с понятием «корреляция» на примере линейной регрессии.

Корреляция — это статистическая взаимосвязь, присутствующая между двумя и более величинами, то есть некоторая линейная зависимость (например, между признаками).

Что же такое автокорреляция временного ряда?

Автокорреляция — это статистическая взаимосвязь между последовательностями значений одного временного ряда, взятыми со сдвигом. Другими словами, автокорреляция говорит нам о том, насколько значение во временном ряду похоже на предыдущее значение.

Таким образом, автокорреляция — это корреляция ряда с самим собой (отсюда приставка «авто»), но со сдвигом во времени. Она помогает выявлять тенденции в данных и оценивать влияние ранее наблюдаемых значений на текущее наблюдение.

У автокорреляции много применений, но в первую очередь её используют для обработки сигналов, прогнозирования погоды и анализа рынка ценных бумаг. Иногда автокорреляция позволяет обнаружить скрытые тенденции.

Научиться находить автокорреляцию в *Python* довольно просто — далее мы разберём, когда и как можно применять эту функцию, а когда она может оказаться неэффективной. Но сначала рассмотрим несколько определений — чтобы построить график автокорреляции на *Python*, необязательно знать эти термины, однако их понимание позволит вам значительно лучше интерпретировать результаты.

- **Лаг** — это предыдущее наблюдение (например, лаг в шесть дней относительно сегодняшнего дня указывает на значение чего-либо, полученное шесть дней назад).
- **Положительная корреляция** — это отношение, при котором увеличение одного значения предсказывает увеличение другого.
- **Отрицательная корреляция** — это отношение, при котором увеличение одного значения предсказывает уменьшение другого.
- **Доверительный интервал** — это рассчитанный диапазон значений, в котором, вероятно, будет содержаться неизвестное (предсказанное) значение для наших данных.
- **Уровень достоверности** — это вероятность того, что доверительный интервал будет содержать наблюдаемое значение (фактическое значение для предсказания).

Рассмотрим формулу автокорреляции. Мы уже знаем, что простой линейный коэффициент корреляции определяется по формуле:

$$r_{xy} = \frac{\sum(x_j - \bar{x}) \cdot (y_j - \bar{y})}{\sqrt{\sum(x_j - \bar{x})^2 (y_j - \bar{y})^2}}$$

Как мы уже сказали, автокорреляция — это корреляция ряда с самим собой, сдвинутым во времени, а значит, в формуле автокорреляции вместо x и y будет сам временной ряд и значения этого сдвинутого временного ряда:

$$r_1 = \frac{\sum_{t=2}^{n}(x_t - \bar{x_1}) \cdot (x_{t-1} - \bar{x_2})}{\sqrt{\sum_{t=2}^{n}(x_t - \bar{x_1})^2 \cdot \sum_{t=2}^{n}(x_{t-1} - \bar{x_2})^2}},$$

где $\bar{x_1} = \frac{\sum_{t=2}^{n} x_t}{n-1}$; $\bar{x_2} = \frac{\sum_{t=2}^{n} x_{t-1}}{n-1}$.

Так будет рассчитываться коэффициент автокорреляции первого порядка (он рассчитывает зависимость между уровнями ряда $t-1$ и t).

График автокорреляций разного порядка называется **коррелограмма**. Его довольно просто построить с помощью метода plot_acf из пакета statsmodels.graphics.tsaplots. Методу необходимо передать всё тот же временной ряд с индексом-датой.

Воспользуемся датасетом `AirPassengers.zip` из примера сезонной декомпозиции, в котором представлены данные о количестве авиапассажиров в 1949–1960 годах.

```
from statsmodels.graphics.tsaplots import plot_acf
df = pd.read_csv("AirPassengers.csv", index_col='Month', parse_dates=['Month'])
plot_acf(df)
```

Коррелограмма из данного примера выглядит примерно так:

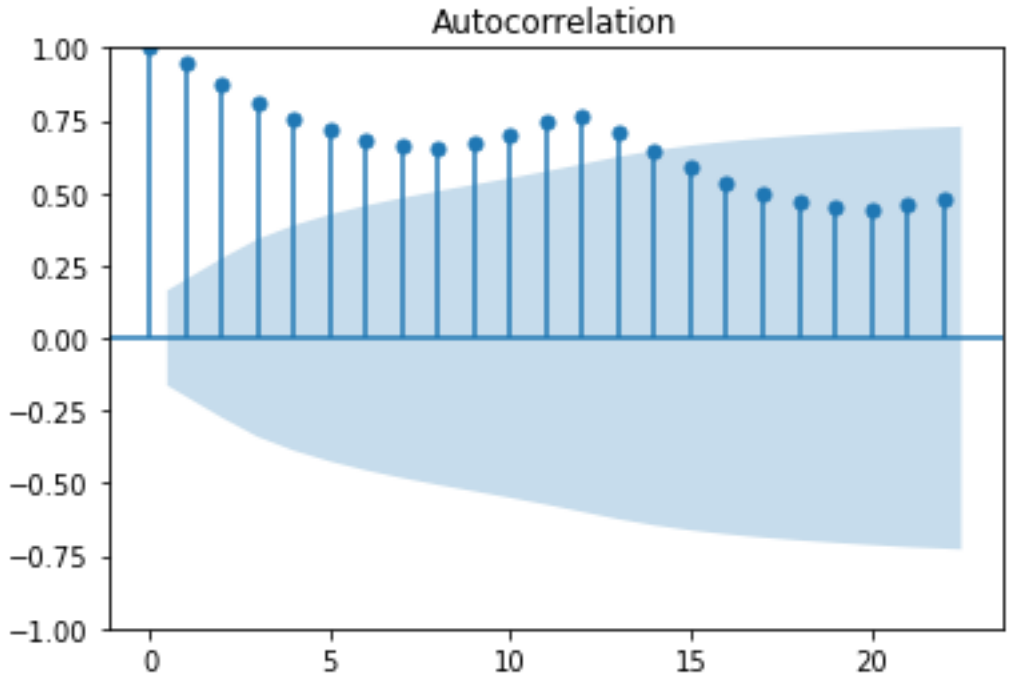

Рис. 13: ba8a20f7a6a11d771bc9007bba5081bd.png

На оси x коррелограммы расположен лаг (порядок), при котором вычисляется автокорреляция. Ось y показывает значение корреляции (от -1 до 1). Чем ближе значение корреляции к 1 или -1, тем выше зависимость, например:

- пик при лаге, равном 1, показывает сильную зависимость между значениям ряда и предыдущим значением;
- пик при лаге, равном 2, показывает сильную корреляцию между каждым значением и значением в более ранний момент на расстоянии 2 от данного.

Также можно встретить коррелограммы, похожие на гистограммы, но их смысл от этого не меняется.

Для совершения предсказаний по данным временного ряда статистическими моделями, необходимо, чтобы во временном ряду присутствовала зависимость, а коррелограмма — хороший способ визуально определить наличие такой зависимости. Значения на коррелограмме будут близки к 0 в случае, если данные ряда не зависят от себя в прошлом. Если скрытая зависимость всё-таки имеется, то одно или несколько значений будут значительно отличаться.

Как читать коррелограмму?

1. Если максимальное значение коррелограммы (не считая значения в нуле) оказывается выраженным для лага, равного k (на рисунке выше $k = 3$, то временной ряд содержит циклическую компоненту с периодом k. То есть данные являются зависимыми/схожими с данными, находящимися на расстоянии k дней/недель и т. д.
2. Если максимальное значение на коррелограмме находится в $k = 1$, то ряд содержит только тенденцию (тренд).
3. А если все значения на графике автокорреляции колеблются в районе 0, то ряд не содержит циклической компоненты и тренда либо содержит нелинейный тренд, который не видно на коррелограмме (так как нелинейный тренд не может быть выражен линейным коэффициентом корреляции).

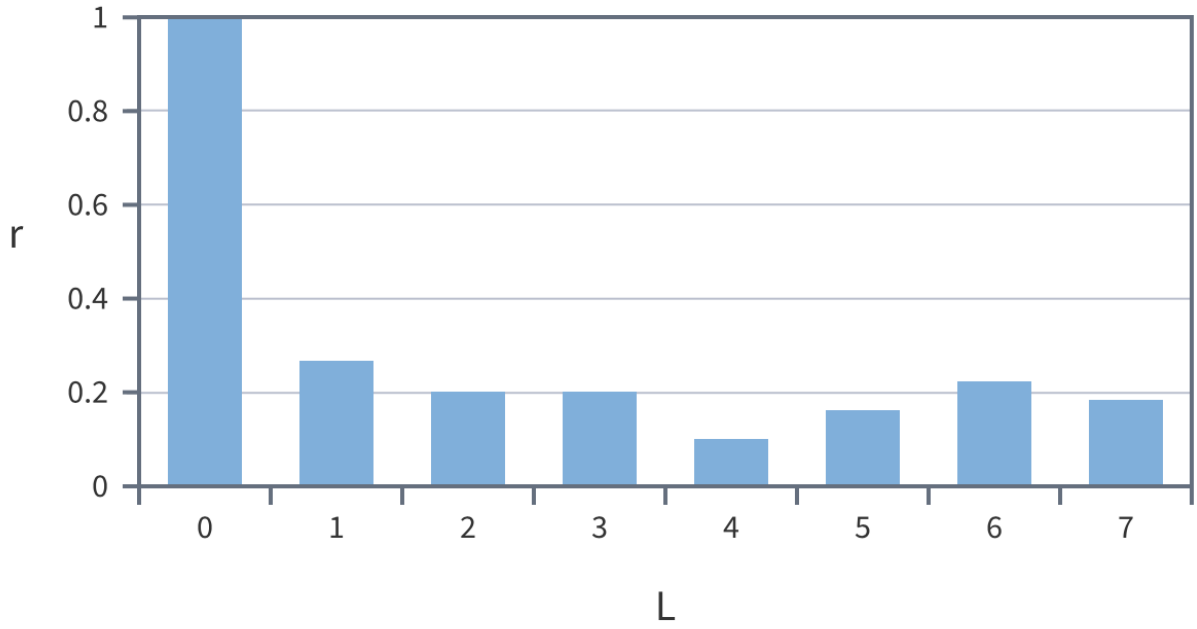

Рис. 14: faf6d7249adaf052522e0174ef525002.png

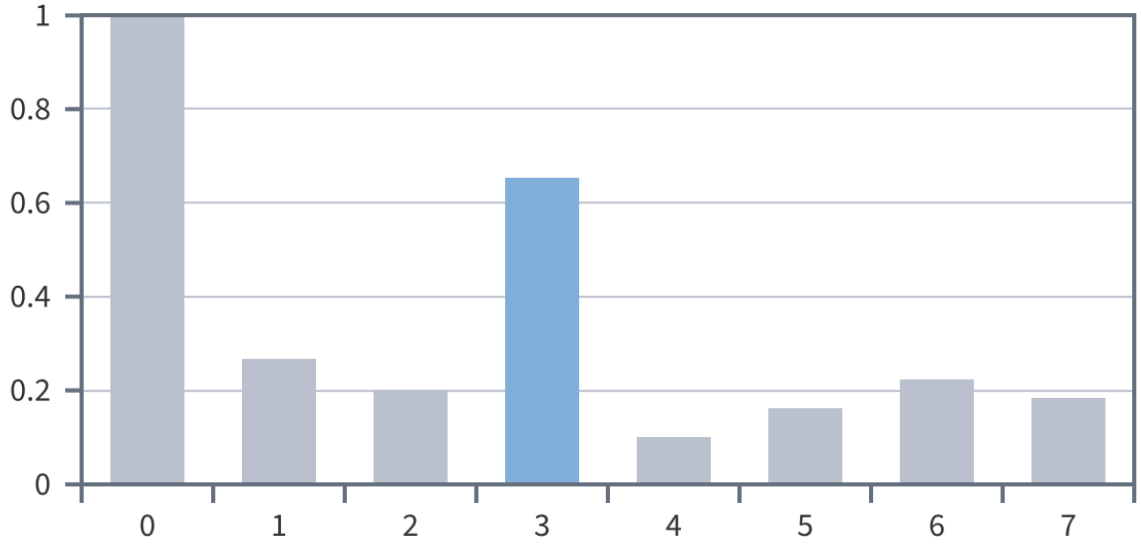

Рис. 15: 4992413078f52eae75a199004b047e41.png

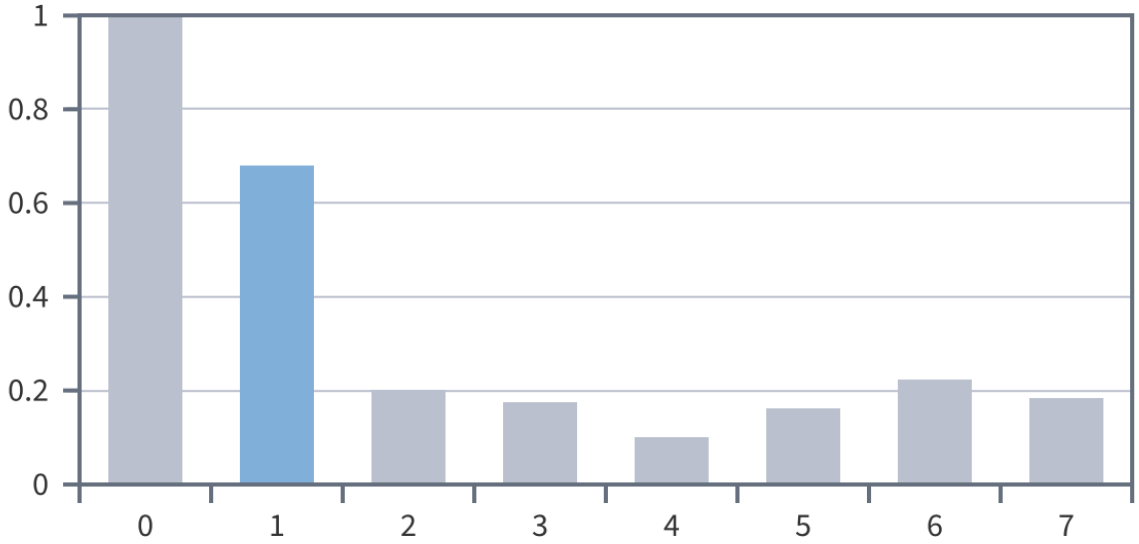

Рис. 16: da2166f81a668ba2cc41ce7d007f39a1.png

Частичная автокорреляция Для определения сезонного периода используется частичная автокорреляция. Она похожа на классическую автокорреляцию, однако дополнительно избавляется от линейной зависимости между сдвинутыми рядами. График частичной автокорреляции читается так же, как и коррелограмма.

Частичная автокорреляция строится с помощью метода plot_pacf из пакета statsmodels.graphics.tsaplots.

Если вы хотите узнать больше про частичную автокорреляцию, рекомендуем обратиться к этой статье.

Автокорреляция остатков Помимо анализа временного ряда на наличие или отсутствие взаимозависимости, иногда смотрят автокорреляцию **остатков** модели. В хорошей модели остатки (ошибки) должны иметь случайный характер — это означает, что модель уловила все существующие зависимости.

Рассмотрим, как определять автокорреляцию остатков по графикам остатков (ошибок предсказания):

На графике выше в большинстве случаев после положительных остатков следуют положительные, а после отрицательных — отрицательные. Это пример положительной автокорреляции.

А в этом случае после положительных остатков чаще всего следуют отрицательные и наоборот, на графике — отрицательная автокорреляция.

Причины автокорреляции остатков: - Если в остатках имеется автокорреляция (наличие зависимости), это значит, что какая-то зависимость осталась незамеченной для вашей модели — возможно, какие-то важные признаки не были учтены. - На появление автокорреляции в остатках может повлиять предварительное сглаживание данных, так как вы искусственно сглаживаете значения (накладывая соседние друг на друга, добавляя зависимость).

Остатки будут случайными, если автокорреляции нет. Статистически, а не только визуально проверить её наличие или отсутствие можно с помощью **теста Дарбина — Уотсона**. Пример применения теста на автокорреляцию остатков приведён в статье.

Если в остатках присутствует автокорреляция, скорее всего, предсказания будут далеки от реальных значений и лучше доработать модель, исправив перечисленные выше недочёты.

Далее посмотрим как автокорреляция используется для предсказания.

Если необходимо глубже изучить теорию по автокорреляции, то можно разобрать эту статью.

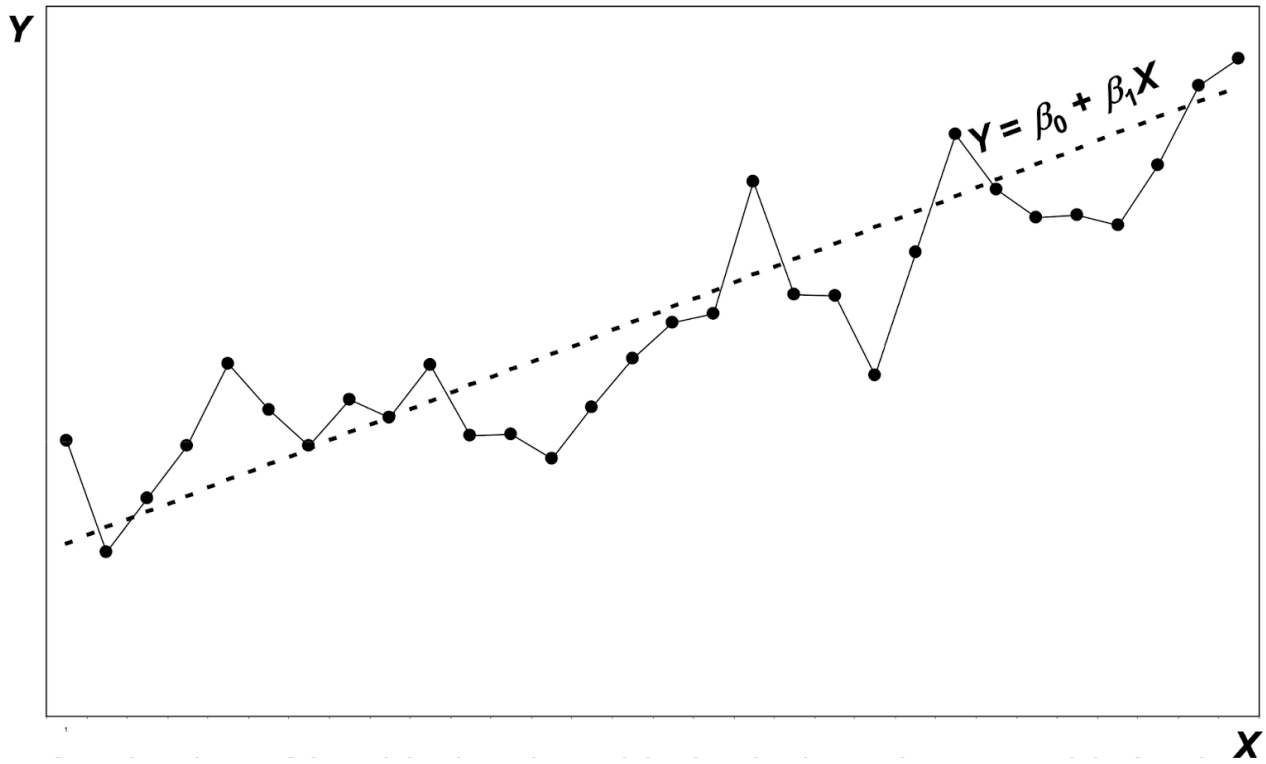

Рис. 17: 087ca754edb2c64ae2ad7a5249dacaae.png

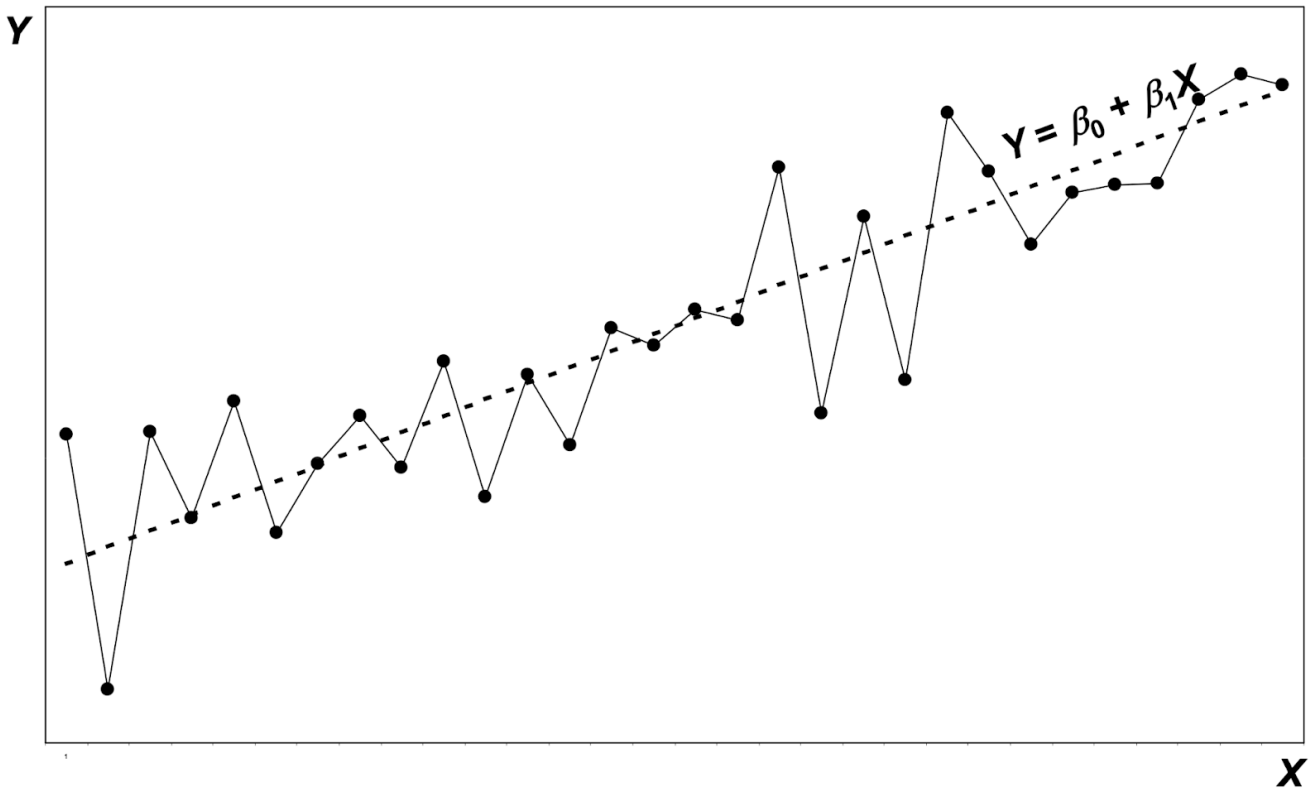

Рис. 18: 6d4f04a4a110a215d8c86badc71de716.png

Авторегрессия

Модель авторегрессии использует методы регрессии и полагается на уже известную нам автокорреляцию.

Авторегрессионная модель — это модель временных рядов, которая описывает, как прошлые значения временного ряда влияют на его текущее значение. Как можно понять из значений частей слова, авторегрессия представляет собой линейную регрессию на себя.

В контексте прогнозирования временных рядов авторегрессионное моделирование будет означать создание модели, в которой переменная Y будет зависеть от предыдущих значений Y с заранее определённой постоянной задержкой во времени. Временной лаг может быть ежедневным (или два, три, четыре дня и т. д.), еженедельным, ежемесячным и т. п.

Модели *AR (autoregressive models)* можно использовать для моделирования всего, что имеет некоторую степень автокорреляции, то есть имеет корреляцию между наблюдениями на соседних временных шагах. Наиболее распространённый вариант применения этого типа моделирования — цены на фондовом рынке, где сегодняшняя цена (t) сильно коррелирует с ценой вчера ($t-1$).

$$Y_t = \beta_0 + \beta_1 \times Y_{t-1} + error_t$$

В приведённой выше формуле берётся значение последнего временного лага (лаг = 1). Если выбрать лаг, равный неделе, то Y_{t-1} будет представлять значение Y за последнюю неделю, а Y_t — за текущую. Коэффициенты β_1 и β_0 — это настраиваемые коэффициенты (как в линейной регрессии), которые мы получим после обучения модели, а $error_t$ — это ошибка, которую мы, скорее всего, не сможем предсказать, но будем иметь в виду, что итоговое значение включает в себя предсказание и некоторую ошибку.

Модель, в которой для расчёта следующего значения используется только предыдущее, называется **моделью первого порядка**, или **AR(1)**.

Давайте разберёмся с концепцией модели *AR* на примере следующего графика:

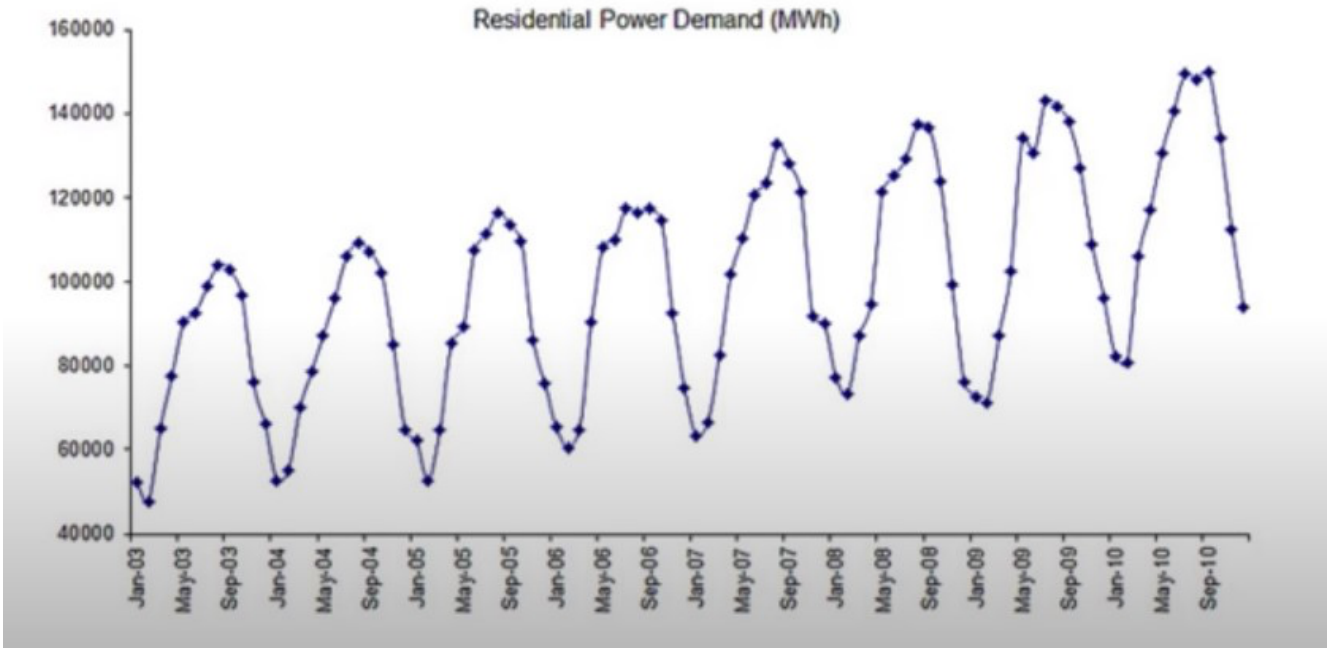

Рис. 19: b25c43b2abeb812315039bd6a73e445f.png

Здесь представлена динамика потребления электроэнергии в жилых домах в разные месяцы с 2003 по 2010 год.

Задача — спрогнозировать потребление на ближайшие месяцы, используя имеющиеся данные.

При временном лаге в один месяц модель *AR(1)*, или модель *AR* первого порядка, будет выглядеть следующим образом:

$$Y_t = \beta_0 + \beta_1 \times Y_{t-1} + error_t$$

Модель *AR* второго порядка будет рассчитывать значение переменной в любое конкретное время в зависимости от значений последних двух задержек. Таким образом, модель *AR(2)* будет выглядеть следующим образом:

$$Y_t = \beta_0 + \beta_1 \times Y_{t-1} + \beta_2 \times Y_{t-2} + error_t$$

1 и 2 в *AR(1)* и *AR(2)* — это параметр, который обозначается как *p*. Таким образом, обобщённая формула для *AR*-модели с параметром *p* будет выглядеть следующим образом:

$$Y_t = \beta_0 + \beta_1 \times Y_{t-1} + ... + \beta_p \times Y_{t-p} + error_t$$

Одним из основных методов оценивания вектора неизвестных параметров $\beta = (\beta_0, \beta_1, ..., \beta_p)$ является метод наименьших квадратов (МНК). Если ошибки модели имеют нормальное распределение, то данный метод также эквивалентен условному методу максимального правдоподобия. Так как коэффициенты настраиваются в процессе обучения модели, вам необходимо лишь верно определить параметр модели *p*.

Как выбирать p? Выше мы познакомились с автокорреляцией и частичной автокорреляцией. Для определения значения p будем использовать график частичной автокорреляции — будем обращать внимание на последний лаг, сильно отличный от нуля, при условии, что ряд стационарный.

Давайте научимся определять p. Перед нами график частичной автокорреляции (*pacf*). Нам необходимо найти последний лаг, отличный от нуля. В данном случае такими лагами являются лаги 1, 2, 3, 4. Остальные лаги колеблются в районе нуля. Поэтому на этом графике выберем *p=4*.

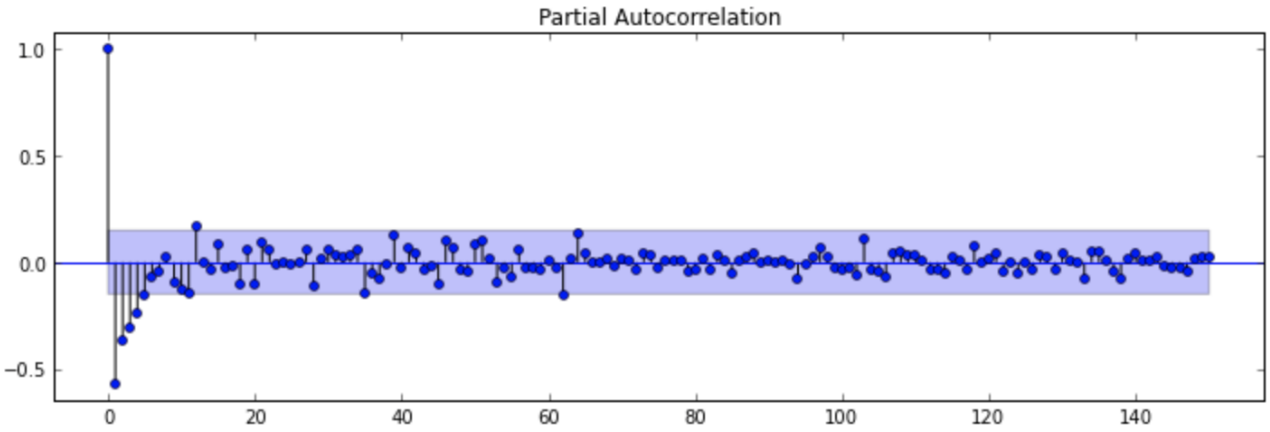

Рис. 20: 3d3c568515bad3dc4b27c2c138e08515.png

Обратите внимание на следующий график:

Здесь *p* будет равно 1, так как сильнее всего от нуля отличен первый лаг. Но если у вас возникают сомнения, вы можете проверить модель с разными параметрами и выбрать лучшую.

AR-моделирование на python Для загрузки класса ar_model.AutoReg, который применяется для обучения одномерной авторегрессионной модели порядка *p*, используется пакет `statsmodels.tsa`.

Ниже приведены некоторые из ключевых шагов, которые необходимо выполнить для обучения *AR*-модели:

1. Отобразить временной ряд.
2. Проверить ряд на стационарность (модель *AR* можно применять только к стационарному временному ряду).
3. Выбрать параметр *p* (порядок модели *AR*).
4. Обучить модель.

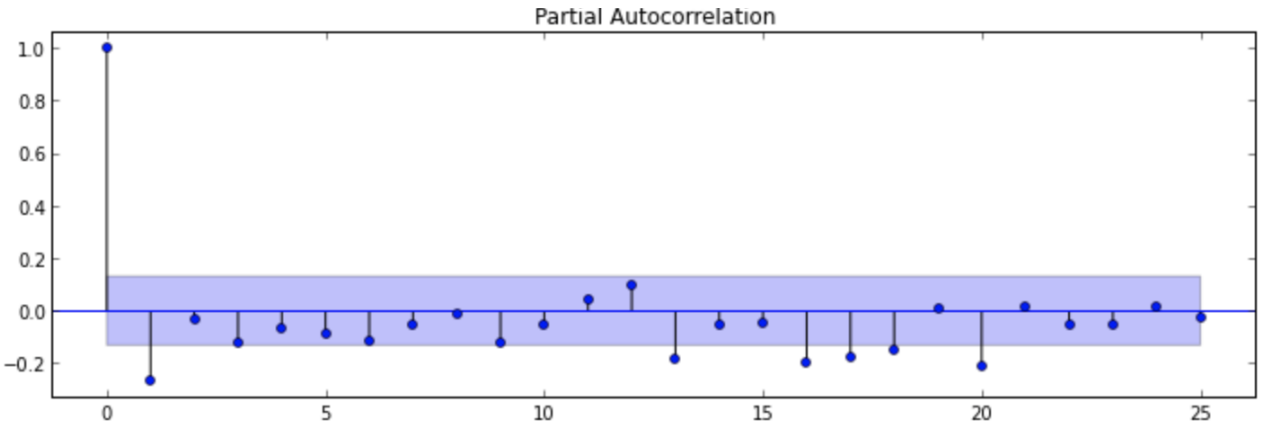

Рис. 21: 11490c5435cf693775e31cfe1f08f585.png

Все эти шаги вы будете выполнять в итоговой практике, а пока давайте перейдём к самому инструменту моделирования.

Применим авторегрессионную модель к датасету по производству возобновляемой энергии в Германии. Набор данных включает общую информацию о потреблении электроэнергии в стране, а также о производстве ветровой и солнечной энергии за 2006–2017 годы.

Признаки в данных:

- Date — дата (в формате гггг-мм-дд);
- Consumption — потребление электроэнергии (ГВтч);
- Wind — производство ветровой энергии (ГВтч);
- Solar — производство солнечной энергии (ГВтч);
- Wind+Solar — сумма производства ветровой и солнечной энергии (ГВтч).

В качестве примера построим предсказание объёма потребления электроэнергии на три месяца (возьмём последние 100 дней).

Импортируем необходимые библиотеки и загрузим датасет с данными.

```
import pandas as pd
import numpy as np
from statsmodels.tsa.ar_model import AutoReg

url='https://raw.githubusercontent.com/jenfly/opsd/master/opsd_germany_daily.csv'
df = pd.read_csv(url, sep=",")
```

Проведём тест на стационарность.

```
from statsmodels.tsa.stattools import adfuller

result = adfuller(df['Consumption'].values)

if result[0] > result[4]['5%']:
    print ('Ряд нестационарен')
else:
    print ('Ряд стационарен')
```

При использовании статистических моделей временные ряды нельзя делить на обучающую и тестовую выборки случайным образом. Так как нам важно сохранять последовательность, разделим данные на обучающую и тестовую выборки упорядоченно, то есть возьмём в качестве тестовой выборки последние 100 значений. Остальные данные будем использовать для обучения.

```
train_data = df['Consumption'][:len(df)-100]
test_data = df['Consumption'][len(df)-100:]
```

Инициализируем и обучим модель. Возьмём lags=8 (AR(8)) — на практике же будем определять это значение по графику частичной автокорреляции:

```
ar_model = AutoReg(train_data, lags=8).fit()
print(ar_model.summary())
```

Сделаем предсказание. Метод `predict` требует два обязательных параметра — метки начала и окончания предсказания. Метка начала предсказания будет равна количеству данных в обучающей выборке, так как нас интересует прогноз со следующего дня. Метка окончания в нашем случае будет равна `len(train_data)+100`, что эквивалентно `len(df)-1`, так как `len(df) = len(train_data) + len(test_data)`.

```
pred = ar_model.predict(start=len(train_data), end=(len(df)-1), dynamic=False)
```

Существуют и альтернативные AR-методы прогнозирования временных рядов: - MA (скользящее среднее), - ARMA (авторегрессионное скользящее среднее), - ARIMA (авторегрессионное интегрированное скользящее среднее), - SARIMA (сезонное авторегрессионное интегрированное скользящее среднее), - VAR (векторная авторегрессия), - VARMA (скользящее среднее векторной авторегрессии), - SES (простое экспоненциальное сглаживание) - etc