Базовые методы обработки данных с использованием Python

Лекция 4. Обработка временных рядов в Python. Работа с датами. Разложение ряда на компоненты

Киреев Василий Сергеевич

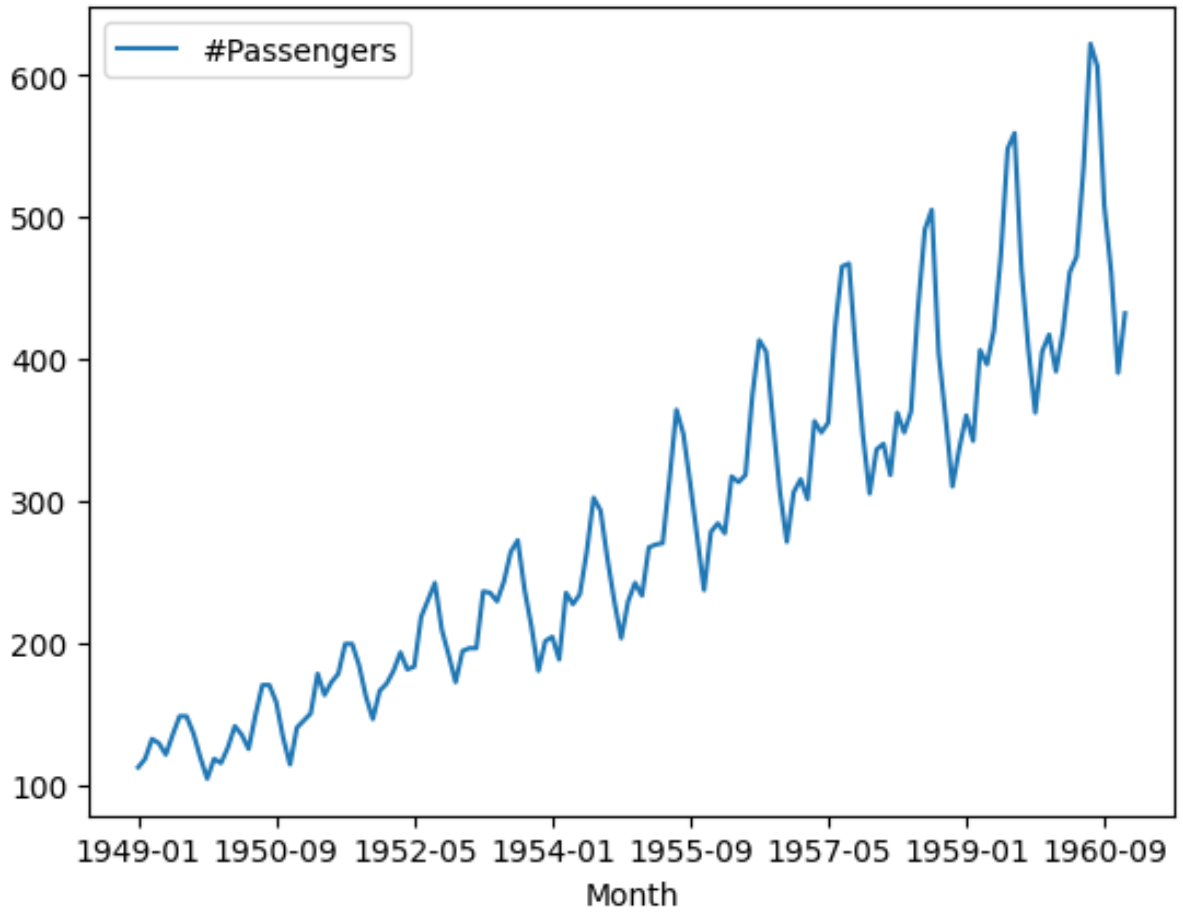
к.т.н., доцент

Москва, 2024

Определение временного ряда

Временной ряд представляет собой набора наблюдений, полученных путем регулярного измерения одной переменной в течение некоторого периода времени. Форма данных для типичного временного ряда - это одна последовательность или список наблюдений, выполненных через равные промежутки времени.

Пример временного ряда



Виды временных рядов. Детерминированность

Детерминированный временной ряд – ряд, в котором нет случайных аспектов или показателей: он может быть выражен формулой. Это значит, что можно проанализировать, как показатели вели себя в прошлом и прогнозировать их поведение в будущем.

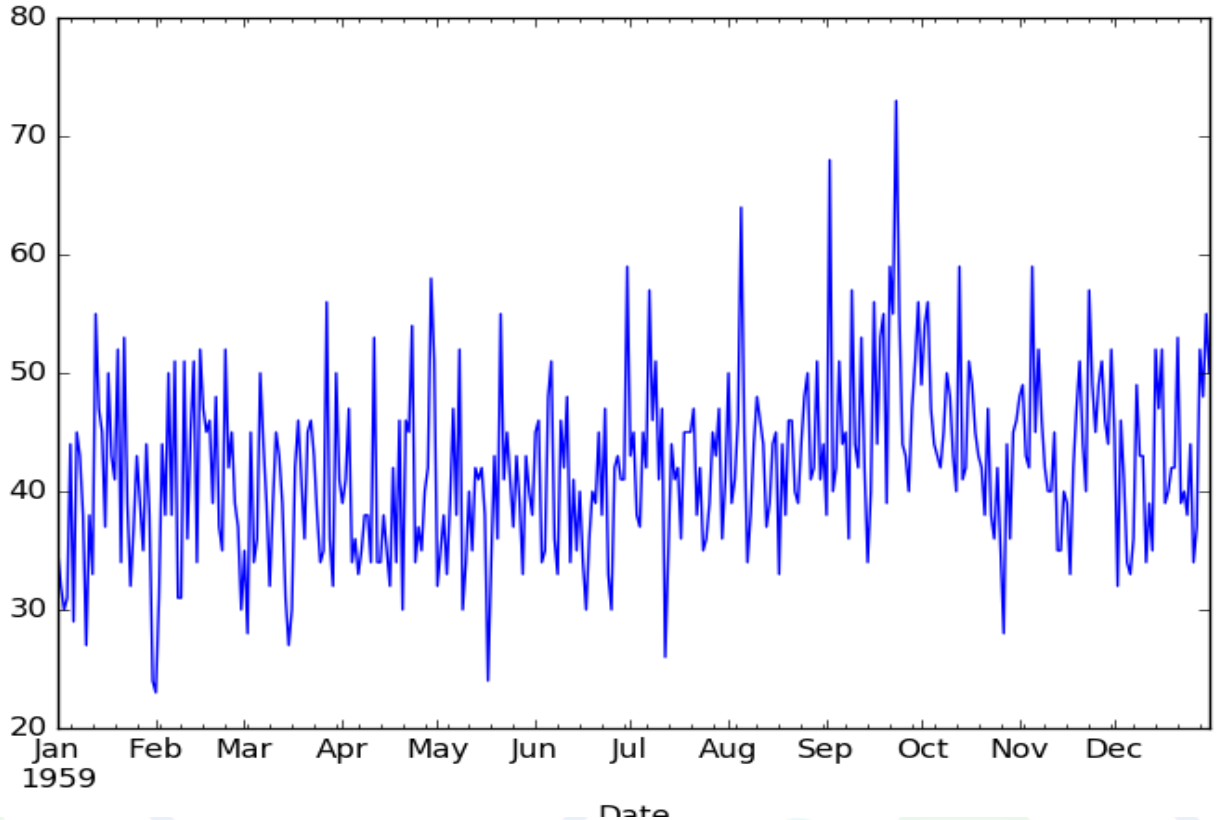
Недетерминированный временной ряд имеет случайный аспект и прогнозирование будущих действий становится сложнее. Природа таких показателей случайна и анализ происходит благодаря средним значениям и дисперсии.

Виды временных рядов. Стационарность

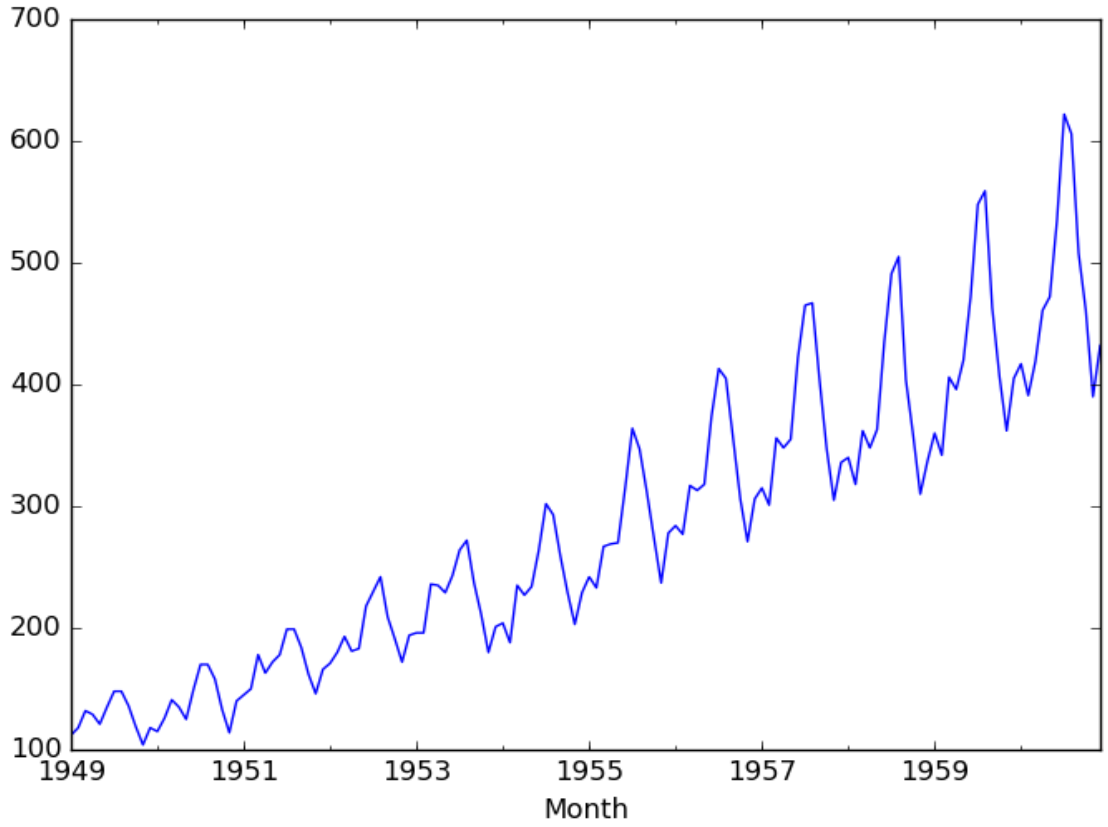
В стационарных временных рядах статистические свойства не зависят от времени, поэтому результат легко предсказать. Большинство статистических методов предполагают, что все временные ряды должны быть стационарными.

В нестационарных временных рядах статистические свойства меняются со временем. Они показывают сезонные эффекты, тренды и другие структуры, которые зависят от временного показателя.

Виды временных рядов. Пример стационарного временного ряда. Рождаемость



Виды временных рядов. Пример нестационарного временного ряда. Число пассажиров АК



Задачи в обработке данных временных рядов

- Создание диапазонов временных меток
- Обработка временных меток данных
- Преобразование строковых данных в временную метку
- Обработка пропущенных значений
- Нарезка данных с использованием временной метки
- Агрегация данных временного ряда/ Передискретизация периодов
- Расчёт скользящего среднего

Pandas. Темпоральные объекты. Временные метки

Timestamp (Временная метка) - это эквивалент pandas для Datetime в Python и в большинстве случаев взаимозаменяем с ним. Она обозначает конкретный момент времени. Индекс, который мы создаем, используя временные метки, имеет тип DatetimeIndex.

```
pd.Timestamp('02-06-2018'),\
pd.Timestamp('02-06-2018').month,\
pd.DatetimeIndex([pd.Timestamp('02-2023'),
      pd.Timestamp('09-2023')])
```

```
(Timestamp('2018-02-06 00:00:00'), 2,
DatetimeIndex(['2023-02-01', '2023-09-01'],
dtype='datetime64[ns]', freq=None))
```

Pandas. Темпоральные объекты. DatetimeIndex

Одним из основных применений DatetimeIndex является использование в качестве индекса для объектов pandas. Класс DatetimeIndex содержит множество оптимизаций, связанных с временными рядами:

- Большой диапазон дат для различных смещений предварительно вычисляется и кэшируется под капотом для того, чтобы сделать генерацию последующих диапазонов дат очень быстрой (нужно просто взять фрагмент).
- Быстрый сдвиг с использованием метода shift на объектах pandas.
- Объединение пересекающихся объектов DatetimeIndex с одинаковой частотой происходит очень быстро (важно для быстрого выравнивания данных).
- Быстрый доступ к полям даты через свойства, такие как год, месяц и т.д.
- Функции регуляризации типа snap и очень быстрая логика asof.

Pandas. Темпоральные объекты. Периоды

В отличие от временной метки, которая представляет собой точку во времени, периоды представляют собой период времени. Это может быть месяц, день, год, час и т.д.

Объекты Period сами по себе не очень полезны, пока они не используются в качестве индекса в Dataframe или Series. Индекс, состоящий из периодов, называется PeriodIndex.

```
index_=pd.PeriodIndex([pd.Period('02-2023'),
       pd.Period('09-2023')])
type(index_)
```

```
pandas.core.indexes.period.PeriodIndex
```

Pandas. Темпоральные объекты. Timedeltas

Timedeltas – это разница во времени, выраженная в единицах разницы, например, дни, часы, минуты, секунды. Они могут быть как положительными, так и отрицательными.

```
pd.Timedelta('2 days 2 hours 15 minutes 30 seconds')
```

```
Timedelta('2 days 02:15:30')
```

Pandas. Темпоральные объекты. Признаки даты

Временные характеристики можно разделить на две категории. Первая - моменты времени в периоде, вторая - время, прошедшее с определенного периода. Эти признаки могут быть очень полезны для понимания закономерностей в данных.

Признаки внутри даты

- pandas.Series.dt.year возвращает год времени даты.
- pandas.Series.dt.month возвращает месяц времени даты
- pandas.Series.dt.day возвращает день времени даты
- pandas.Series.dt.hour возвращает час времени даты.

Pandas. Темпоральные объекты. Часовой пояс

Понимание часового пояса очень важно. Возможно, вы находитесь в одном часовом поясе, а ваш клиент - в другом. Pandas обладает функциональностью для работы с различными часовыми поясами. Имеется два типа данных DateTime:

- Наивный DateTime, который не имеет представления о часовом поясе,
- DateTime с учетом часового пояса, который знает часовой пояс.

Задачи в обработке данных временных рядов. Создание диапазонов временных меток

Для создания диапазона дат в pandas можно использовать функцию pandas.date_range()

```
1 import pandas as pd
2
3 date_range = pd.date_range(start ='1/1/2021',
4                            end ='1/08/2021',
5                           freq ='Min')
6 print(date_range)
```

```
DatetimeIndex(['2021-01-01 00:00:00', '2021-01-01 00:01:00',
               '2021-01-01 00:02:00', '2021-01-01 00:03:00',
               '2021-01-01 00:04:00', '2021-01-01 00:05:00',
               '2021-01-01 00:06:00', '2021-01-01 00:07:00',
               '2021-01-01 00:08:00', '2021-01-01 00:09:00',
               ...
               '2021-01-07 23:51:00', '2021-01-07 23:52:00',
               '2021-01-07 23:53:00', '2021-01-07 23:54:00',
               '2021-01-07 23:55:00', '2021-01-07 23:56:00',
               '2021-01-07 23:57:00', '2021-01-07 23:58:00',
               '2021-01-07 23:59:00', '2021-01-08 00:00:00'],
              dtype='datetime64[ns]', length=10081, freq='T')
```

Задачи в обработке данных временных рядов. Преобразование строковых данных в временную метку

Pandas To Datetime (.to_datetime()) преобразует строковое представление даты в фактический формат даты. В дальнейшем это может быть использовано для задействования, например, функции передескретизации периодов временного ряда.

```
1 dt=[str(x) for x in date_range]
2 print(type(dt[0]))
3 pd.to_datetime(dt)
```

```
<class 'str'>
DatetimeIndex(['2021-01-01 00:00:00', '2021-01-01 00:01:00',
               '2021-01-01 00:02:00', '2021-01-01 00:03:00',
               '2021-01-01 00:04:00', '2021-01-01 00:05:00',
               '2021-01-01 00:06:00', '2021-01-01 00:07:00',
               '2021-01-01 00:08:00', '2021-01-01 00:09:00',
               ...
               '2021-01-07 23:51:00', '2021-01-07 23:52:00',
               '2021-01-07 23:53:00', '2021-01-07 23:54:00',
               '2021-01-07 23:55:00', '2021-01-07 23:56:00',
               '2021-01-07 23:57:00', '2021-01-07 23:58:00',
```

Задачи в обработке данных временных рядов. Обработка пропущенных значений

Отсутствующие данные возникают, когда в наборе данных нет данных, сохраненных для интересующей переменной. В зависимости от объема отсутствующие данные могут нанести ущерб результатам любого анализа данных или надежности моделей машинного обучения.

```
time_index = pd.date_range("1/01/2021", periods=6, freq="W")

df = pd.DataFrame(index=time_index);
df["Sales"] = [5.0,4.0,np.nan,np.nan,1.0,np.nan];

df1= df.interpolate();
print(df1)
```

```
            Sales
2021-01-03    5.0
2021-01-10    4.0
2021-01-17    3.0
2021-01-24    2.0
2021-01-31    1.0
2021-02-07    1.0
```

Задачи в обработке данных временных рядов. Обработка пропущенных значений

- df.interpolate() - интерполяция пропущенных значений
- df.ffill() - заполнение пропущенных значений - использование значения следующей строки для заполнения пропущенного значения
- df.bfill() - заполнение пропущенных значений - использование значения предыдущей строки для заполнения пропущенного значения

Передискретизация. Метод resample

В pandas реализована простая функциональность для выполнения операций передискретизации при преобразовании частоты (например, преобразование секундных данных в 5-минутные). Это очень часто встречается в финансовых приложениях, но не ограничивается ими.

resample() представляет собой группировку по времени, за которой следует метод редукции для каждой из ее групп. Некоторые продвинутые стратегии см. в примерах книги рецептов.Метод resample() может быть использован непосредственно из объектов DataFrameGroupBy

Задачи в обработке данных временных рядов. Передискретизация периодов

Передискретизация периодов включает в себя изменение частоты (гранулярности) наблюдений временного ряда.

Существует два типа передискретизации:

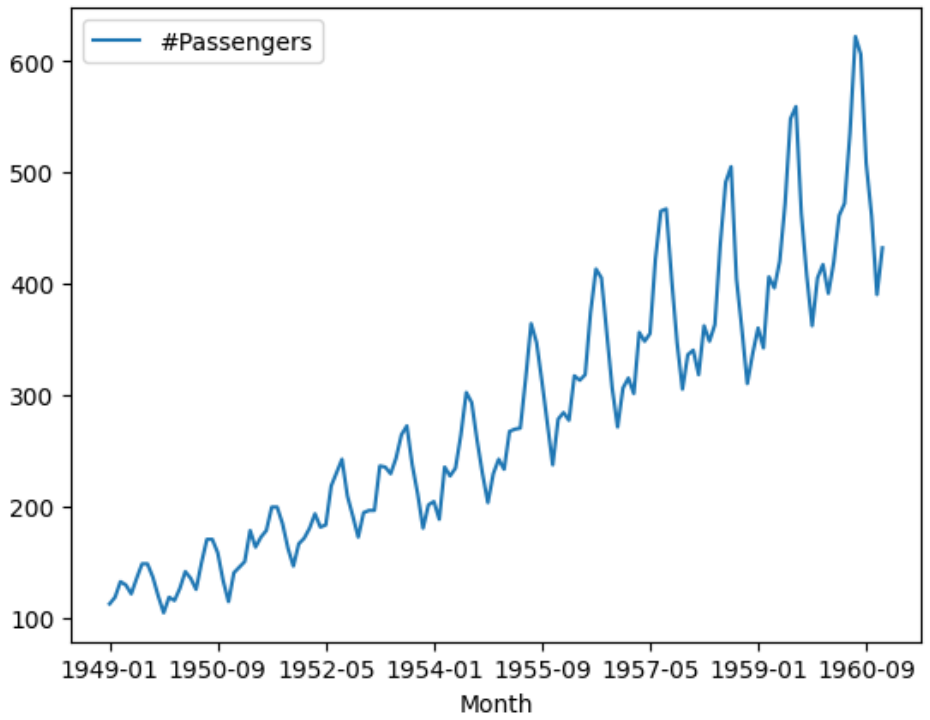
- Повышающая дискретизация: увеличение частоты выборок, например, с минут до секунд.
- Понижающая дискретизация : уменьшение частоты выборок, например, с дней до месяцев.

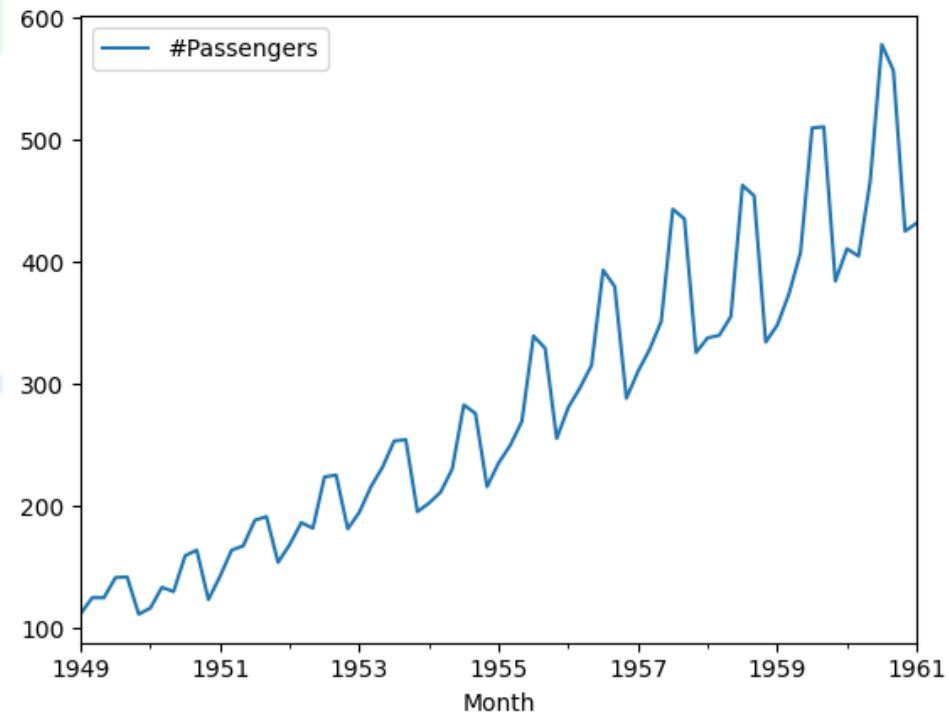
Передискретизация может быть использована для улучшения качества моделей машинного обучения с учителем

```
1 ts.resample('2M').mean()
```

Month	#Passengers
1949-01-31	112.0
1949-03-31	125.0
1949-05-31	125.0
1949-07-31	141.5
1949-09-30	142.0
...	...
1960-05-31	466.5
1960-07-31	578.5
1960-09-30	557.0

Задачи в обработке данных временных рядов. Передискретизация периодов. Пример





Типовая гранулярность в Pandas

D	периодичность в календарный день
W	еженедельная периодичность
M	периодичность на конец месяца
H	почасовая частота
T, min	минутная частота
S	Секундная частота

Задачи в обработке данных временных рядов. Нарезка данных с использованием временной метки

Одной из самых мощных и удобных функций pandas при обработке временных рядов является индексация на основе времени - использование дат и времени для интуитивно понятной организации данных и доступа к ним.

При индексации на основе времени можно использовать строки в формате даты/времени для выбора данных в нашем фрейме данных с помощью средства доступа loc.

Индексация работает аналогично стандартной индексации на основе меток с помощью loc, но с несколькими дополнительными функциями.

```
1 ts.loc['1949-01-01':'1949-06-22']
```

Month	#Passengers
1949-01-01	112
1949-02-01	118
1949-03-01	132
1949-04-01	129
1949-05-01	121
1949-06-01	135

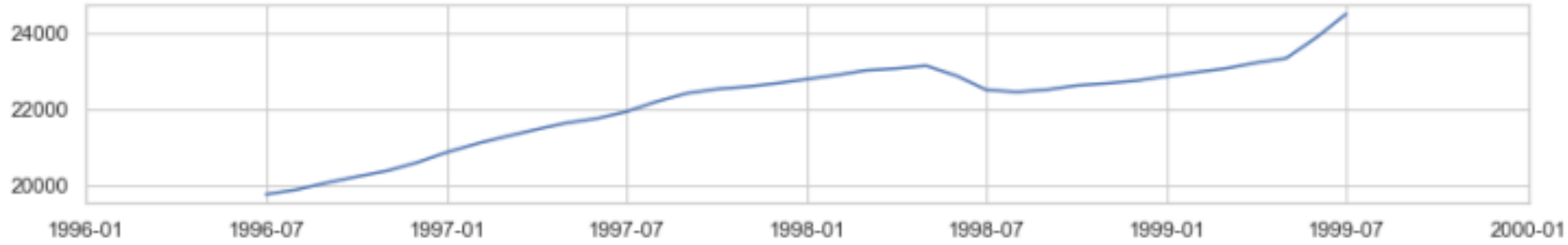
Типовая модель временного ряда

При анализе временного ряда выделяют три составляющие: тренд, сезонность и шум.

- Тренд
- Сезонность
- Остаток
- Цикл

Типовая модель временного ряда. Тренд

Данные временного ряда показывают тенденцию, когда их значение изменяется с течением времени, причем увеличение значения показывает положительную тенденцию, а уменьшение – отрицательную.

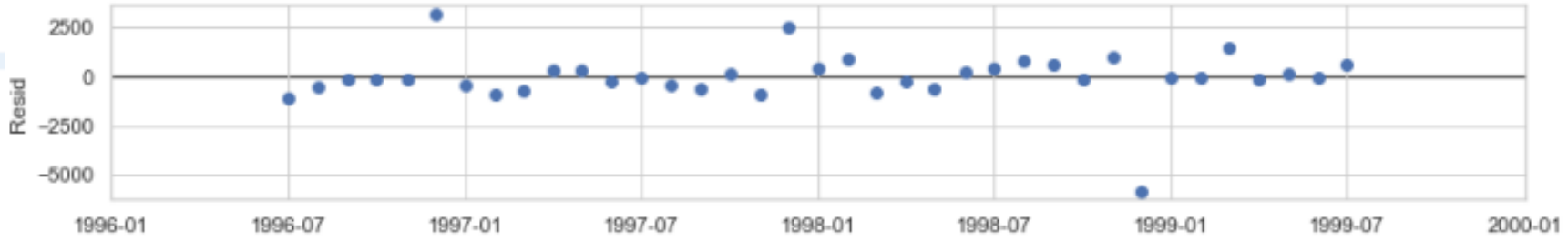


Типовая модель временного ряда. Сезонность

Сезонность - это свойство временных рядов, которое проявляется в периодических закономерностях, повторяющихся с постоянной частотой.

Типовая модель временного ряда. Остаток

После извлечения из данных тренда и сезонности остается то, что мы называем остатком (ошибкой) или остатком. Это помогает выявлять аномалии во временных рядах.



Типовая модель временного ряда. Цикл

Данные временных рядов называются циклическими, если в них присутствуют тенденции без установленных повторений или сезонности.

Понятие скользящего среднего

Скользящее среднее представляет собой среднее арифметическое значение наблюдений в определенном окне или периоде, который "скользит" вдоль временного ряда. Скользящее среднее помогает сгладить краткосрочные колебания и шумы, выявлять тренды и улавливать долгосрочные закономерности в данных.

$$SMA_t = \frac{1}{n}\sum_{i=0}^{n-1} p_{t-1} = \frac{p_t + p_{t-1} + \ldots + p_{t-i} + \ldots + p_{t-n+2} + p_{t-n+1}}{n}$$

Задачи в обработке данных временных рядов. Расчет скользящего среднего

Для вычисления скользящего среднего в Pandas реализован метод rolling(),который создает скользящее окно для проведения различных операций на окне данных.

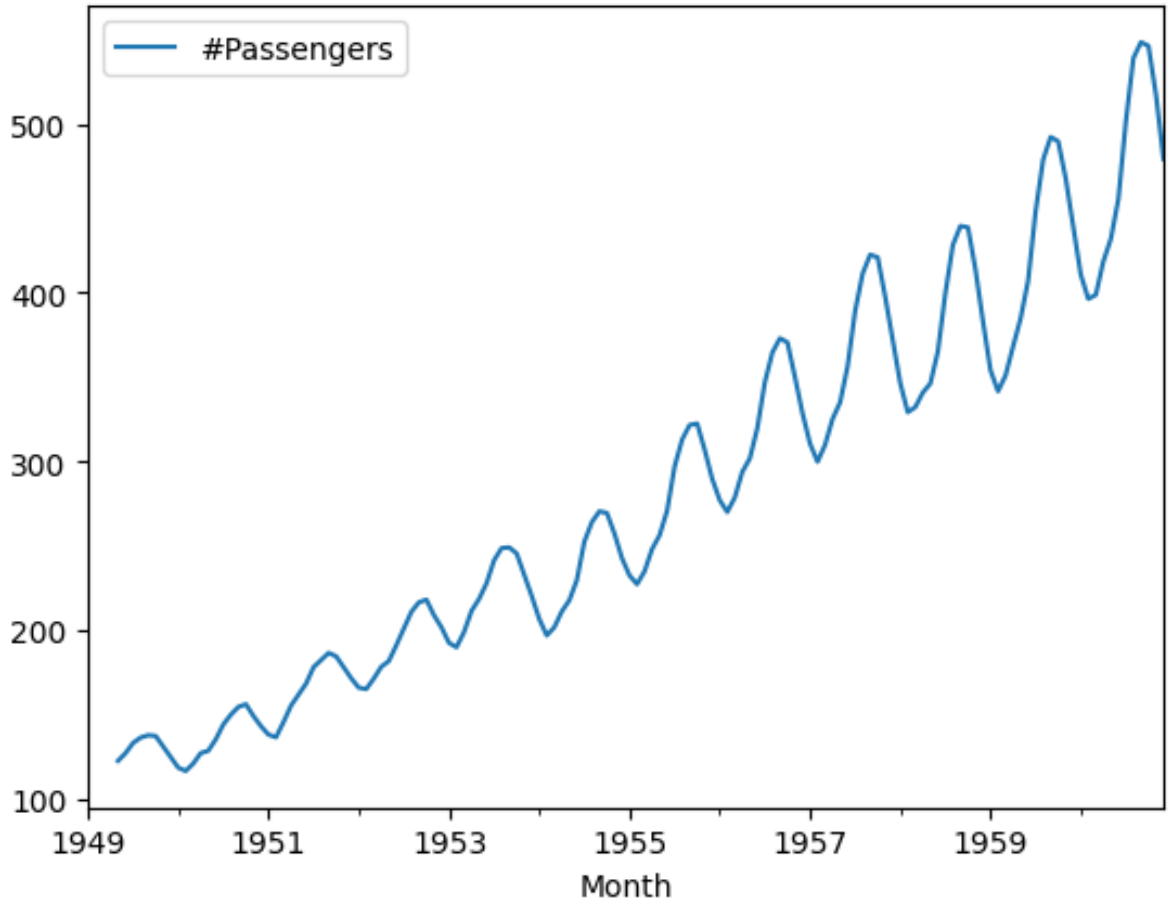
Скользящее окно имеет определенный размер и перемещается по временному ряду или другому набору данных с одним шагом за раз. Для каждого положения окна вычисляется статистика, такая как среднее, медиана, сумма и т. д.

При использовании метода rolling(), нужно указать размер окна с помощью параметра window. Затем можно применять различные функции к этому скользящему окну, например mean.

```
1 ts.rolling(window=3).mean()
```

Month	#Passengers
1949-01-01	NaN
1949-02-01	NaN
1949-03-01	120.666667
1949-04-01	126.333333
1949-05-01	127.333333
...	...

Задачи в обработке данных временных рядов. Расчет скользящего среднего. Пример сглаженного временного ряда



Pandas. Метод asfreq

Функция asfreq обеспечивает другую технику повторной выборки. Она возвращает значение в конце заданного интервала. Например, asfreq("W") возвращает значение в последний день каждой недели. Для использования функции asfreq необходимо задать столбец date в качестве индекса фрейма данных.

```
df=pd.DataFrame()
df['date']=pd.date_range('2021-01-01','2021-
01-20',freq='D')
df['sales']=[1000]*df.shape[0]
df.set_index("date").asfreq("W").head()
```

2021-01-03	1000
2021-01-10	1000
2021-01-17	1000

Временные ряды. Компоненты

Временные ряды. Компоненты

- Анализ автокорреляций
- Partial Autocorrelation Functions (PACF)
- Анализ
- Seasonality Analysis
- Decomposition
- Spectrum Analysis
- Seasonal and Trend decomposition using Loess (STL)
- Rolling correlation
- Cross-correlation Analysis
- Метод Бокса-Дженкинса
- Granger Causality Analysis

Временные ряды. Метрики качества

ЭМИТ
Институт экономики, математики и ИТ

$$WMAPE = \frac{\sum_{t=1}^{n}(w_t|A_t - F_t|)}{\sum_{t=1}^{n}(w_t|A_t|)}$$

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}|\frac{A_t - F_t}{A_t}|$$

$$WAPE = \frac{\sum_{t=1}^{n}|A_t - F_t|}{\sum_{t=1}^{n}|A_t|}$$

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|A_t - F_t|$$

Временные ряды. Библиотеки Python

ЭМИТ
Институт экономики, математики и ИТ

Statsmodels	Pmdarima	Prophet	tslearn	ARCH	GluonTS
PyFlux	Sktime	PyCaret	Darts	Kats	AutoTS
	Scikit-learn	TensorFlow	Keras	PyTorch	

Спасибо за внимание!