Практическая работа №5.Анализ временных рядов

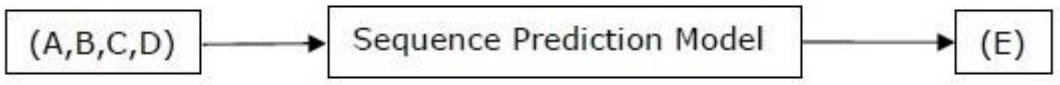
Данные временного ряда означают данные, которые находятся в последовательности определенных временных интервалов. Если мы хотим построить предсказание последовательности в машинном обучении, то нам приходится иметь дело с последовательными данными и временем.

Основная концепция анализа последовательностей или анализа временных рядов

Анализ последовательности или анализ временных рядов предназначен для прогнозирования следующего в заданной входной последовательности на основе ранее наблюдавшегося. Предсказание может быть обо всем, что может быть следующим: символ, число, погода на следующий день, следующий термин в речи и т. Д. Анализ последовательности может быть очень полезен в таких приложениях, как анализ фондового рынка, прогнозирование погоды и рекомендации по продукту.

Пример

Рассмотрим следующий пример, чтобы понять прогноз последовательности. Здесь **A, B, C, D** — заданные значения, и вы должны предсказать значение **E,** используя модель прогнозирования последовательности.



Установка полезных пакетов

Для анализа данных временных рядов с использованием Python нам необходимо установить следующие пакеты:

Панды

Pandas — это библиотека с открытым исходным кодом, имеющая лицензию BSD, которая обеспечивает высокую производительность, простоту использования структуры данных и инструменты анализа данных для Python. Вы можете установить Pandas с помощью следующей команды

```
pip install pandas
```

Если вы используете Anaconda и хотите установить с помощью менеджера пакетов **conda** , вы можете использовать следующую команду:

```
conda install -c anaconda pandas
```

hmmlearn

Это BSD-библиотека с открытым исходным кодом, которая состоит из простых алгоритмов и моделей для изучения скрытых марковских моделей (HMM) в Python. Вы можете установить его с помощью следующей команды

```
pip install hmmlearn
```

Если вы используете Anaconda и хотите установить с помощью менеджера пакетов **conda** , вы можете использовать следующую команду:

```
conda install -c omnia hmmlearn
```

PyStruct

Это структурированная библиотека обучения и прогнозирования. Алгоритмы обучения, реализованные в PyStruct, имеют такие имена, как условные случайные поля (CRF), марковские случайные сети с максимальным запасом (M3N) или машины опорных векторов структур. Вы можете установить его с помощью следующей команды —

```
pip install pystruct
```

CVXOPT

Используется для выпуклой оптимизации на основе языка программирования Python. Это также бесплатный программный пакет. Вы можете установить его с помощью следующей команды —

```
pip install cvxopt
```

Если вы используете Anaconda и хотите установить с помощью менеджера пакетов **conda** , вы можете использовать следующую команду:

```
conda install -c anaconda cvdoxt
```

Панды: обработка, нарезка и извлечение статистики из данных временных рядов

Панды — очень полезный инструмент, если вам приходится работать с данными временных рядов. С помощью панд вы можете выполнить следующее —

- Создайте диапазон дат с помощью пакета **pd.date_range**
- Индекс панды с датами с помощью пакета **pd.Series**
- Выполните повторную выборку с помощью пакета **ts.resample**
- Изменить частоту

Пример

В следующем примере показано, как обрабатывать и разрезать данные временных рядов с помощью Pandas. Обратите внимание, что здесь мы используем данные по месячным колебаниям в Арктике, которые можно загрузить с сайта month.ao.index.b50.current.ascii и которые мы можем преобразовать в текстовый формат.

Обработка данных временных рядов

Для обработки данных временных рядов вам необходимо выполнить следующие шаги:

Первый шаг включает в себя импорт следующих пакетов

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Затем определите функцию, которая будет считывать данные из входного файла, как показано в приведенном ниже коде

```
def read_data(input_file):
   input_data = np.loadtxt(input_file, delimiter = None)
```

Теперь преобразуйте эти данные во временные ряды. Для этого создайте диапазон дат нашего временного ряда. В этом примере мы сохраняем один месяц как частоту данных. Наш файл содержит данные, которые начинаются с января 1950 года.

```
dates = pd.date_range('1950-01', periods = input_data.shape[0], freq = 'M')
```

На этом этапе мы создаем данные временных рядов с помощью Pandas Series, как показано ниже

```
output = pd.Series(input_data[:, index], index = dates)
return output
if __name__=='__main__':
```

Введите путь к входному файлу, как показано здесь —

```
input_file = "/Users/admin/AO.txt"
```

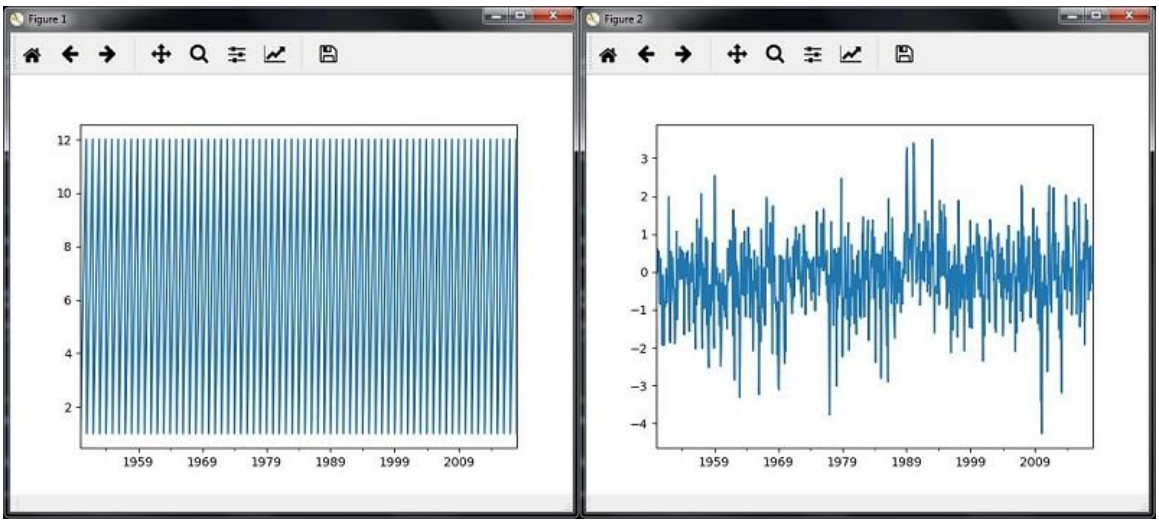
Теперь преобразуйте столбец в формат временных рядов, как показано здесь —

```
timeseries = read_data(input_file)
```

Наконец, нанесите на график и визуализируйте данные, используя показанные команды —

```
plt.figure()
timeseries.plot()
plt.show()
```

Вы увидите графики, как показано на следующих изображениях



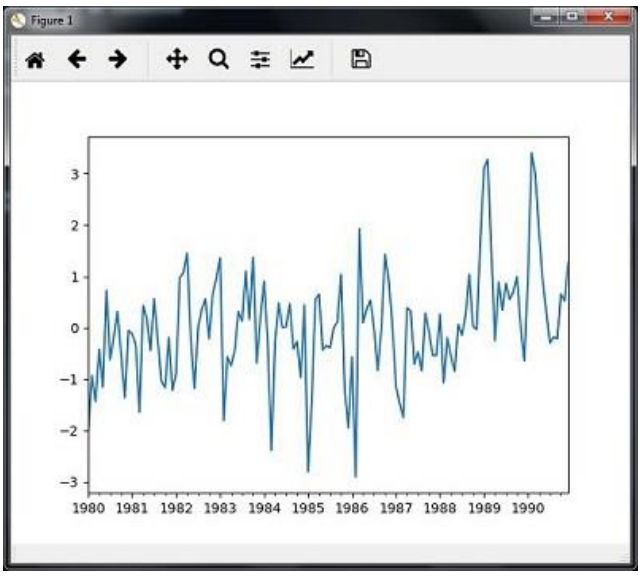
Нарезка данных временного ряда

Разрезание включает в себя получение только некоторой части данных временного ряда. В качестве примера мы разбиваем данные только с 1980 по 1990 год. Обратите внимание на следующий код, который выполняет эту задачу:

```
timeseries['1980':'1990'].plot()
<matplotlib.axes._subplots.AxesSubplot at 0xa0e4b00>

plt.show()
```

Когда вы запускаете код для нарезки данных временных рядов, вы можете увидеть следующий график, как показано на рисунке



Извлечение статистики из данных временных рядов

Вам придется извлечь некоторую статистику из заданных данных, в случаях, когда вам нужно сделать какой-то важный вывод. Среднее, дисперсия, корреляция, максимальное значение и минимальное значение являются одними из таких статистических данных.

Вы можете использовать функцию **mean ()** для нахождения среднего значения, как показано здесь:

```
timeseries.mean()
```

Вы можете использовать функцию **max ()** , чтобы найти максимум, как показано здесь

```
timeseries.max()
```

Вы можете использовать функцию min (), чтобы найти минимум, как показано здесь

```
timeseries.min()
```

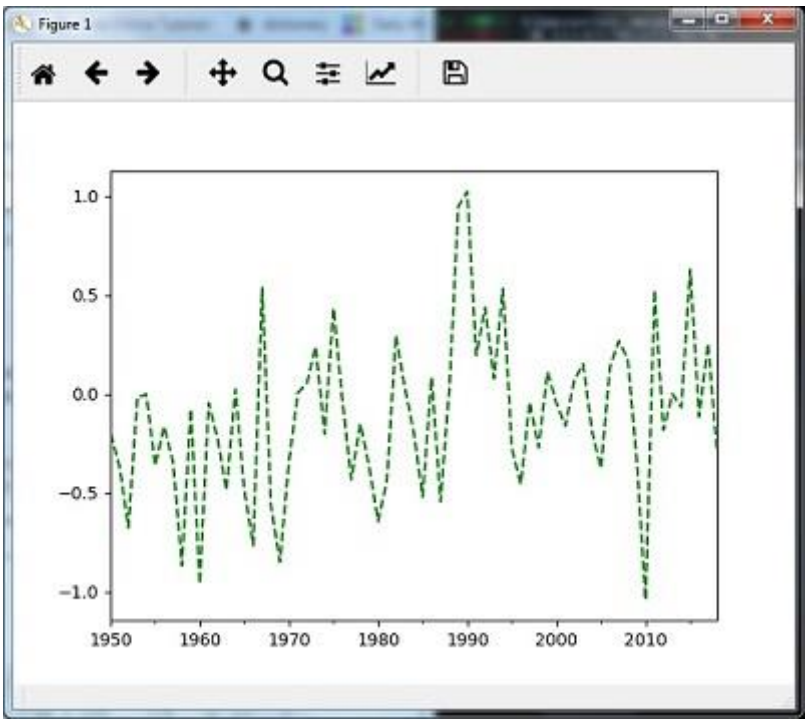
Если вы хотите рассчитать всю статистику за раз, вы можете использовать функцию description **(),** как показано здесь

```
timeseries.describe()
```

Вы можете использовать следующий код для повторной выборки данных с помощью метода mean (), который является методом по умолчанию —

```
timeseries_mm = timeseries.resample("A").mean()
timeseries_mm.plot(style = 'g--')
plt.show()
```

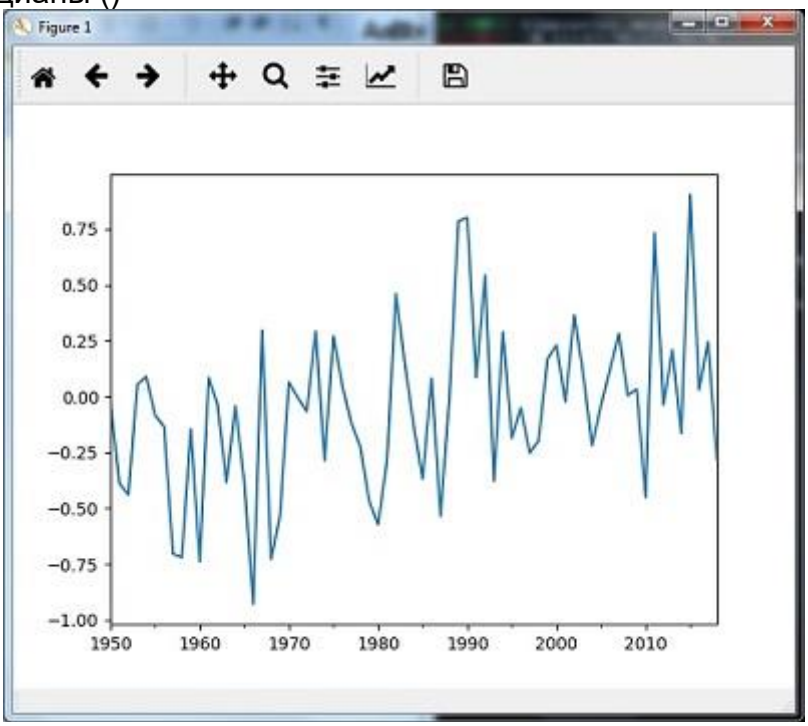
Затем вы можете наблюдать следующий график как результат передискретизации, используя mean ()



Вы можете использовать следующий код для повторной выборки данных, используя метод **median ()** —

```
timeseries_mm = timeseries.resample("A").median()
timeseries_mm.plot()
plt.show()
```

Затем вы можете наблюдать следующий график как результат повторной выборки с помощью медианы () —



Вы можете использовать следующий код для расчета скользящего (скользящего) среднего значения

```
timeseries.rolling(window = 12, center = False).mean().plot(style = '-g')
plt.show()
```

Затем вы можете наблюдать следующий график как результат скользящего (скользящего) среднего значения